

# **A Mathematical Theory of the Unknown**

Journey Beyond the Frontiers of Human Understanding

**R. A. García Leiva**

(This book is 84% complete)

Copyright © 2025 R. A. García Leiva  
PUBLISHED BY THE AUTHOR



To my wife Justi, my son Daniel,  
and my two daughters Teresa and Lucía.



# Contents

|          |                      |           |
|----------|----------------------|-----------|
| <b>1</b> | <b>Introduction</b>  | <b>25</b> |
| 1.1      | Scientific Knowledge | 26        |
| 1.2      | Entities             | 27        |
| 1.3      | Representations      | 28        |
| 1.4      | Descriptions         | 31        |
| 1.5      | Miscoding            | 33        |
| 1.6      | Inaccuracy           | 37        |
| 1.7      | Surfeit              | 39        |
| 1.8      | Nescience            | 41        |
| 1.9      | Perfect Knowledge    | 43        |
| 1.10     | Unknown Unknown      | 45        |

|          |  |            |
|----------|--|------------|
| <b>2</b> | <b>Fundamental Elements .....</b>      | <b>51</b>  |
| 2.1      | Entities                               | 52         |
| 2.2      | Representations                        | 54         |
| 2.3      | Invalid Representations                | 60         |
| 2.4      | Joint Representations                  | 62         |
| 2.5      | Descriptions                           | 64         |
| 2.6      | Descriptions for Joint Representations | 69         |
| 2.7      | Conditional Descriptions               | 70         |
| 2.8      | Research Areas                         | 74         |
| 2.9      | References                             | 78         |
| <b>3</b> | <b>Miscoding .....</b>                 | <b>79</b>  |
| 3.1      | Miscoding                              | 80         |
| 3.2      | Joint Miscoding                        | 83         |
| 3.3      | Decreasing Miscoding                   | 85         |
| 3.4      | Targetless Representations             | 87         |
| 3.5      | Miscoding of Areas                     | 89         |
| <b>4</b> | <b>Inaccuracy .....</b>                | <b>93</b>  |
| 4.1      | Inaccuracy                             | 94         |
| 4.2      | Conditional Inaccuracy                 | 97         |
| 4.3      | Decreasing Inaccuracy                  | 99         |
| 4.4      | Inaccuracy-Miscoding Rate of Change    | 101        |
| 4.5      | Inaccuracy of Areas                    | 104        |
| <b>5</b> | <b>Surfeit .....</b>                   | <b>107</b> |
| 5.1      | Surfeit                                | 108        |
| 5.2      | Redundancy                             | 111        |
| 5.3      | Conditional Surfeit                    | 112        |
| 5.4      | Decreasing Surfeit                     | 114        |
| 5.5      | Surfeit-inaccuracy rate of Change      | 115        |
| 5.6      | Surfeit of Areas                       | 118        |

|          |   |            |
|----------|---|------------|
| <b>6</b> | <b>Nescience</b>                        | <b>119</b> |
| 6.1      | <b>Nescience</b>                        | 120        |
| 6.2      | <b>Minimizing Nescience</b>             | 126        |
| 6.2.1    | Global Criterion .....                  | 127        |
| 6.2.2    | Weighting Method .....                  | 128        |
| 6.2.3    | The other method .....                  | 128        |
| 6.2.4    | Evolutionary Methods .....              | 128        |
| 6.3      | <b>Joint Nescience</b>                  | 128        |
| 6.4      | <b>Conditional Nescience</b>            | 129        |
| 6.5      | <b>Nescience of Areas</b>               | 129        |
| 6.6      | <b>Perfect Knowledge</b>                | 130        |
| 6.7      | <b>Current Best Description</b>         | 132        |
| 6.8      | <b>Nescience based on Datasets</b>      | 133        |
| 6.9      | <b>Unknown Unknown</b>                  | 134        |
| 6.10     | <b>Science vs. Pseudoscience</b>        | 134        |
| <b>7</b> | <b>Interesting Questions</b>            | <b>137</b> |
| 7.1      | <b>Other Metrics</b>                    | 139        |
| 7.2      | <b>Interesting Research Questions</b>   | 142        |
| 7.3      | <b>Relevance</b>                        | 144        |
| 7.4      | <b>Applicability</b>                    | 146        |
| 7.5      | <b>Interesting Questions</b>            | 150        |
| 7.6      | <b>New Research Topics</b>              | 152        |
| 7.7      | <b>Classification of Research Areas</b> | 153        |

## II

## Part 3: Applications

|          |                                 |            |
|----------|---------------------------------|------------|
| <b>8</b> | <b>Machine Learning</b>         | <b>159</b> |
| 8.1      | <b>Nescience Python Library</b> | 160        |
| 8.2      | <b>A Note About Compression</b> | 160        |
| 8.3      | <b>Miscoding</b>                | 163        |
| 8.4      | <b>Inaccuracy</b>               | 172        |
| 8.5      | <b>Surfeit</b>                  | 176        |
| 8.6      | <b>Nescience</b>                | 180        |

|             |  |            |
|-------------|--|------------|
| <b>8.7</b>  | <b>Auto Machine Classification</b>                             | <b>185</b> |
| 8.7.1       | Surfeit of Algorithms . . . . .                                | 186        |
| <b>8.8</b>  | <b>Auto Machine Regression</b>                                 | <b>186</b> |
| <b>8.9</b>  | <b>Time Series</b>   | <b>186</b> |
| 8.9.1       | Automiscoding, Crossmiscoding and Partial Automiscoding<br>187 |            |
| 8.9.2       | Auto Time Series . . . . .                                     | 189        |
| <b>8.10</b> | <b>Anomaly Detection</b>                                       | <b>192</b> |
| <b>8.11</b> | <b>Decision Trees</b>  | <b>195</b> |
| 8.11.1      | Algorithm Description . . . . .                                | 196        |
| 8.11.2      | Algorithm Evaluation . . . . .                                 | 199        |
| <b>8.12</b> | <b>Algebraic Model Selection</b>                               | <b>205</b> |
| <b>8.13</b> | <b>The Analysis of the Incompressible</b>                      | <b>208</b> |
| <b>9</b>    | <b>Analysis of Science</b> . . . . .                           | <b>211</b> |
| <b>9.1</b>  | <b>Describing Current Knowledge</b>                            | <b>211</b> |
| <b>9.2</b>  | <b>Measuring Knowledge</b>                                     | <b>214</b> |
| 9.2.1       | Surfeit . . . . .  | 216        |
| 9.2.2       | Inaccuracy . . . . .   | 219        |
| 9.2.3       | Nescience . . . . .  | 223        |
| 9.2.4       | Conclusion . . . . .   | 227        |
| <b>9.3</b>  | <b>Measuring Research Areas</b>                                | <b>227</b> |
| <b>9.4</b>  | <b>The Evolution of Knowledge</b>                              | <b>232</b> |
| <b>9.5</b>  | <b>The Demarcation Problem</b>                                 | <b>235</b> |
| <b>9.6</b>  | <b>Perfect knowledge</b>                                       | <b>241</b> |
| <b>9.7</b>  | <b>Unknown-unknown</b>   | <b>242</b> |
| <b>10</b>   | <b>Computational Creativity</b> . . . . .                      | <b>243</b> |
| <b>10.1</b> | <b>Maturity</b>  | <b>244</b> |
| 10.1.1      | Applicability . . . . .  | 245        |
| 10.1.2      | Relevance . . . . .  | 247        |
| <b>10.2</b> | <b>Interesting Research Questions</b>                          | <b>248</b> |
| 10.2.1      | Intradisciplinary Questions . . . . .                          | 251        |
| 10.2.2      | Interdisciplinary Questions . . . . .                          | 256        |
| <b>10.3</b> | <b>New Research Topics</b>                                     | <b>256</b> |
| <b>10.4</b> | <b>References</b>  | <b>257</b> |
| <b>10.5</b> | <b>Future Work</b>   | <b>257</b> |

|             |                                       |            |
|-------------|---------------------------------------|------------|
| <b>11</b>   | <b>Discrete Mathematics</b>           | <b>263</b> |
| 11.1        | Sets, Relations and Functions         | 264        |
| 11.2        | Strings and Languages                 | 268        |
| 11.3        | Counting Methods                      | 272        |
| 11.4        | Matrices                              | 273        |
| 11.5        | Graphs                                | 276        |
| <b>12</b>   | <b>Discrete Probability</b>           | <b>281</b> |
| 12.1        | Foundations of Probability Theory     | 282        |
| 12.2        | Conditional Probability               | 288        |
| 12.3        | Random Variables                      | 292        |
| 12.3.1      | Multivariate Distributions            | 297        |
| 12.3.2      | Marginal Probability Mass Function    | 299        |
| 12.3.3      | Conditional Probability Mass Function | 302        |
| <b>12.4</b> | <b>Characterizing Distributions</b>   | <b>305</b> |
| 12.4.1      | Measures of Central Tendency          | 306        |
| 12.4.2      | Measures of Dispersion                | 308        |
| 12.4.3      | Measures of Statistical Relationship  | 311        |
| <b>12.5</b> | <b>Common Distributions</b>           | <b>316</b> |
| 12.5.1      | Uniform Distribution                  | 316        |
| 12.5.2      | Bernoulli Distribution                | 316        |
| 12.5.3      | Binomial Distribution                 | 317        |
| 12.5.4      | Discrete Normal Distribution          | 318        |
| <b>12.6</b> | <b>Large Random Samples</b>           | <b>321</b> |
| 12.6.1      | Random Samples                        | 322        |
| 12.6.2      | Law of Large Numbers                  | 326        |
| 12.6.3      | Central Limit Theorem                 | 330        |
| <b>13</b>   | <b>Computability</b>                  | <b>335</b> |
| 13.1        | Turing Machines                       | 337        |
| 13.2        | Universal Turing Machines             | 342        |
| 13.3        | Non-Computable Problems               | 343        |
| 13.4        | Computable Functions and Sets         | 345        |
| 13.5        | Oracle Turing Machine                 | 347        |
| 13.6        | Computational Complexity              | 349        |

|           |  |            |
|-----------|--|------------|
| <b>14</b> | <b>Coding</b>                                    | <b>355</b> |
| 14.1      | Coding   | 356        |
| 14.2      | Kraft Inequality                                 | 360        |
| 14.3      | Optimal Codes                                    | 365        |
| 14.4      | Entropy  | 369        |
| 14.5      | Huffman Algorithm                                | 373        |
| <b>15</b> | <b>Complexity</b>                                | <b>379</b> |
| 15.1      | Strings Complexity                               | 380        |
| 15.2      | Properties of Complexity                         | 384        |
| 15.3      | Joint Kolmogorov Complexity                      | 385        |
| 15.4      | Conditional Kolmogorov complexity                | 387        |
| 15.5      | Information Distance                             | 389        |
| 15.6      | Incompressibility and Randomness                 | 392        |
| <b>16</b> | <b>Learning</b>                                  | <b>397</b> |
| 16.1      | Statistical Inference                            | 398        |
| 16.1.1    | Maximum Likelihood Estimator . . . . .           | 401        |
| 16.1.2    | Bayesian Inference . . . . .                     | 404        |
| 16.2      | Machine Learning                                 | 407        |
| 16.2.1    | Parametric vs. Non-parametric Models . . . . .   | 410        |
| 16.2.2    | Model Accuracy . . . . .                         | 411        |
| 16.2.3    | No free lunch theorem . . . . .                  | 412        |
| 16.2.4    | The bias-variance trade-off . . . . .            | 412        |
| 16.2.5    | Generative vs. discriminative models . . . . .   | 413        |
| 16.2.6    | Discretization of Continuous Variables . . . . . | 413        |
| 16.2.7    | k-means Clustering . . . . .                     | 416        |
| 16.2.8    | Decision Trees . . . . .                         | 416        |
| 16.2.9    | Time Series Analysis . . . . .                   | 418        |
| 16.3      | Minimum Message Length                           | 423        |
| 16.4      | Minimum Description Length                       | 425        |
| 16.4.1    | Refined MDL . . . . .                            | 427        |
| 16.5      | Multiobjective Optimization                      | 427        |
| 16.5.1    | Range of the Solutions . . . . .                 | 430        |
| 16.5.2    | Trade-offs . . . . .                             | 432        |
| 16.5.3    | Optimization Methods . . . . .                   | 432        |

|           |                              |            |
|-----------|------------------------------|------------|
| <b>17</b> | <b>Philosophy of Science</b> | <b>437</b> |
| 17.1      | What is Science              | 438        |
| 17.2      | What is an Entity            | 441        |
| 17.3      | Observation                  | 444        |
| 17.4      | Scientific Representation    | 446        |
| 17.5      | Scientific Discovery         | 450        |
| 17.6      | Scientific Explaination      | 453        |
| 17.7      | Scientific Justification     | 457        |
| 17.8      | The Limits of Science        | 463        |

## IV

## Appendix

|          |                                      |            |
|----------|--------------------------------------|------------|
| <b>A</b> | <b>Advanced Properties</b>           | <b>469</b> |
| A.1      | The Axioms of Science                | 470        |
| A.1.1    | Model Theory . . . . .               | 470        |
| A.1.2    | Type Theory . . . . .                | 475        |
| A.1.3    | Category Theory . . . . .            | 484        |
| A.2      | Scientific Method                    | 484        |
| A.2.1    | Science as a Language . . . . .      | 485        |
| A.3      | The Inaccuracy - Surfeit Trade-off   | 486        |
| A.4      | Graspness                            | 486        |
| A.5      | Effort                               | 487        |
| A.6      | Human Understanding                  | 487        |
| A.7      | Areas in Decay                       | 488        |
| <b>B</b> | <b>Advanced Mathematics</b>          | <b>491</b> |
| B.1      | Distinctiveness of Metrics           | 492        |
| B.2      | Distinctiveness of Metrics           | 492        |
| B.2.1    | Accuracy . . . . .                   | 492        |
| B.2.2    | Precision . . . . .                  | 493        |
| B.2.3    | Recall . . . . .                     | 494        |
| B.2.4    | F1 . . . . .                         | 495        |
| B.2.5    | Area under the ROC curve . . . . .   | 495        |
| B.3      | Statistical Significance of Analysis | 498        |

|          |                                      |            |
|----------|--------------------------------------|------------|
| <b>C</b> | <b>About Quotes and Photos .....</b> | <b>499</b> |
| C.0.1    | Quotes .....                         | 499        |
| C.0.2    | Photos .....                         | 502        |
| <b>D</b> | <b>Notation .....</b>                | <b>507</b> |
|          | <b>Bibliography .....</b>            | <b>511</b> |
|          | Books                                | 511        |
|          | Articles                             | 513        |
|          | <b>Index .....</b>                   | <b>515</b> |



## Preface

*Perfection is achieved not when there is nothing more to add,  
but when there is nothing left to take away.*  
Antoine de Saint-Exupéry

We live in an era that places a high value on knowledge and celebrates scientific advancement. The development of vaccines in record time, the detection of gravitational waves, and the use of artificial intelligence to model protein structures are just a few examples of the extraordinary progress science has made in recent years. However, this celebration of progress should not obscure the structural limitations within the scientific enterprise itself. Despite its accomplishments, science is constrained by the methodological and conceptual frameworks it employs. For example, the "publish or perish" culture in academia often results in a flood of low-quality publications, which can dilute the impact of meaningful scientific work. Another major constraint lies in the allocation of research funding, which often fails to prioritize investigations with the highest potential societal return.

This book is driven by a central aspiration: to overcome the methodological constraints of science in order to challenge established knowledge, explore the unknown, and progress towards the limits of scientific understanding. Our aim is to investigate not only well-known unsolved problems but

also those that remain outside our current awareness, the known unknowns and the unknown unknowns. But this pursuit is not merely an academic curiosity; it is a practical necessity. Rather than developing a purely theoretical framework, our goal is to create an approach that can be applied in real-world scientific contexts.

Our assumption is that to explore and ultimately reduce what we do not yet know, we need a structured framework for understanding ignorance itself. To this end, we propose a rigorous mathematical theory, named the Theory of Nescience, grounded in the principles of computability, randomness, and artificial intelligence. The theory builds upon a striking insight: perfect knowledge implies randomness. Although this may initially appear counterintuitive, given that science is traditionally associated with structure, order, and explanation, it becomes clearer when we consider the process of theory refinement. As scientific understanding improves, our descriptions of phenomena become more concise. Eventually, a theory becomes so concise that it cannot be simplified any further. Its representation becomes incompressible, which aligns with the mathematical definition of a random string.

Randomness sets the ultimate limit to our knowledge, since a random description cannot be refined any further and, assuming the theory is accurate, our understanding must be perfect. Far from being a handicap, the constraints that randomness imposes on knowledge pave the way for new possibilities in science and technology. By comprehending these limitations, we can address the most challenging open problems and discover completely new research topics.

From this theoretical foundation emerges a new mathematical framework for understanding nescience, or human ignorance, and the processes by which knowledge is acquired, organized, and refined. Some of the contributions of this book are:

- A mathematical theory to quantify our lack of scientific knowledge, based on computability and randomness, and on the assumption that measuring how much we do not know is easier than measuring how much we do know.
- A new collection of metrics to measure how much we do not know, whether about individual research topics or broader research areas, and how new research contributions either reduce or fail to reduce that ignorance.
- A practical framework to address some of the most relevant challenges in science, such as defining what constitutes perfect knowledge, identifying the limits of science, and discovering new, previously unknown, research topics.

- A software library that combines the Theory of Nescience with principles of artificial intelligence to automatically derive new knowledge and make predictions based on data.

The Theory of Nescience challenges us to rethink the aims of scientific inquiry. Instead of striving to maximize what we know, it encourages us to minimize what we do not know. By shifting our focus from accumulation to reduction, we open the door to more meaningful measures of scientific progress. The chapters that follow lay the theoretical groundwork for this new approach, one that begins with ignorance and ends in knowledge.

## **Research Agenda**

In this section, we present a focused list of research questions aimed at addressing important gaps in our current understanding of science and the scientific method. Our goal is to stimulate thoughtful investigation by drawing from a variety of academic perspectives. By pursuing these questions, we hope to push the boundaries of human knowledge and inspire innovative solutions that can benefit both scientific inquiry and society more broadly.

*Can we provide a quantitative characterization of our ignorance regarding a research topic?* Developing such a metric would give us a way to measure not just what we know, but how much we are still missing. This would be especially useful for evaluating scientific contributions, as we could measure how much each new idea or publication actually improves our understanding. By combining this with relevance metrics, we would gain a powerful tool for assessing the true value of scientific work.

*Can we compare the extent of our ignorance across disparate scientific fields?* If this is possible, we could evaluate and prioritize research based on how much there is left to understand, regardless of the field. This approach could help distribute scientific attention and funding more effectively, highlighting areas of science that are in greater need of exploration while still valuing fundamental research.

*Can we establish a systematic procedure to enhance our knowledge?* While the scientific method has long served as the cornerstone of research, it varies significantly across disciplines and often lacks clear, measurable criteria for success. Developing a unified and precise approach could improve the consistency and efficiency of scientific discovery. If successful, it would allow us to evaluate and refine how knowledge is generated, leading to more reliable and accelerated progress across all areas of inquiry.

*Can we devise a method for discovering new, previously unknown, and intriguing research entities and problems?* Many major discoveries arise from questions we had not thought to ask. If we could build a systematic way

to identify these unknown unknowns, we could unlock entirely new areas of research and significantly accelerate scientific progress.

*What constitutes perfect knowledge?* Understanding what it means to know something completely is essential to defining the limits of science. If we could precisely define and recognize perfect knowledge, we would know when a field is complete and when it is time to shift our focus elsewhere. Clarifying whether such completeness is always possible would also help set realistic goals for scientific inquiry.

*What effort is required to fully comprehend an unfamiliar subject?* Measuring the cost of understanding could fundamentally change how we plan and fund research. For example, knowing the minimal effort needed to improve our grasp of cancer treatment by a small percentage would help us allocate resources more wisely. Even an estimate would be useful for setting priorities and managing expectations.

*Do some research topics inherently possess a higher degree of complexity than others?* This question addresses a long-standing debate about intellectual differences across disciplines. If we could objectively measure topic complexity, we might dispel myths of superiority and develop more targeted educational strategies. It would also help explain why some areas advance faster than others.

*Are there research topics beyond the scope of human comprehension?* Human cognition has limits, and it is possible that some problems are fundamentally beyond our capacity to solve. Acknowledging these limits could help us better delegate scientific exploration to machines where appropriate, especially as artificial intelligence becomes more capable of original thought.

*How can we differentiate between science and pseudo-science?* Many fields present themselves as scientific but lack rigorous foundations. A mathematically grounded method for identifying what qualifies as science would have major implications for public policy, education, and research funding. It would also help protect scientific integrity by distinguishing valid inquiry from unsupported speculation.

Each chapter in this book engages with these questions from both theoretical and practical standpoints. Some offer detailed responses, while others lay the groundwork for future research. This openness is by design. The Theory of Nescience is not a closed doctrine but a platform for further exploration—an invitation to investigate the limits of science and to imagine new pathways of understanding. In this light, the book is not merely a treatise, but a research agenda, a provocation to conventional thinking, and a call to those who are driven to explore the boundaries of knowledge itself.

## Origins of the Theory of Nescience

It was in 1991, when I was eighteen years old, that I first encountered the statement, "*Computers are useless, they can only give you answers.*" This quote, attributed to Pablo Picasso, struck me as profoundly true. Of course, Picasso was referring to the early calculators of his time, not modern computers. But the underlying idea remains relevant: machines are designed to process predefined tasks, not to generate new and meaningful questions on their own. That realization stayed with me for years. However, it wasn't until 2014—more than two decades later—that I began to explore the implications of Picasso's observation from a practical and computational perspective.

The core ideas behind the Theory of Nescience came together during one particularly restless night. Concepts such as nescience, relevance, and the unknown unknown suddenly aligned. In hindsight, my long-standing interests in information theory and Kolmogorov complexity had likely prepared me for this moment. These disciplines proved essential in articulating the mathematical foundation for a theory that had first emerged as a series of intuitive insights.

Over the following weeks, I conducted a series of computer experiments to test these concepts. The results were promising, but the theory needed time, several years, to develop into a rigorous mathematical framework. Initially, my goal was quite focused: to devise a method for identifying interesting and underexplored scientific questions. But as I examined how ignorance, or nescience, changes over time within scientific disciplines, my scope expanded. I began to wonder whether it was possible to define perfect knowledge and to determine whether such a state could be formally described.

This line of inquiry led to an unexpected and striking insight: perfect knowledge could be expressed in terms of randomness. Specifically, when a theory's description cannot be compressed further, when it becomes algorithmically random, it may indicate that the theory is as complete as possible. This realization broadened my original focus into a general framework for studying the structure of ignorance, the development of understanding, and the mechanisms that drive scientific progress.

Encouraging early feedback from colleagues and researchers helped me refine the theory and explore new applications, particularly in data science and machine learning. Its potential to shed light on long-standing questions in the philosophy of science was especially motivating. For instance, it offered new ways to compare our understanding across fields like mathematics and sociology, and to examine why some research areas appear intrinsically more difficult than others.

Still, something was missing. Although the theory provided compelling

explanations, it initially lacked the ability to make testable predictions. To remedy this, I developed a predictive model: a function describing how nescience diminishes as research effort increases. This allowed me to estimate the expected gain in knowledge based on the resources devoted to studying a topic.

Yet even this was not enough. I sought a surprising prediction, one that could reshape how we think about inquiry. Continued mathematical development eventually led to such a discovery: in certain types of topics, additional research can actually increase our ignorance. In these cases, there may exist a critical threshold beyond which further investigation does not improve understanding, but instead introduces greater uncertainty.

This deepened my belief that grappling with ignorance is as fundamental as pursuing knowledge. The Theory of Nescience represents my response to Picasso's challenge: not merely to construct systems that answer questions, but to build frameworks capable of identifying which questions are most worth asking.

These developments and insights have shaped what has become a comprehensive and mature theory. One that I hope will not only reframe how we understand knowledge, but also inspire readers to question more deeply, think more broadly, and search more boldly. The chapters that follow present this journey in both its theoretical depth and practical implications.

## About the Book

The Theory of Nescience draws on concepts from multiple academic disciplines, including computability, complexity theory, artificial intelligence, and the philosophy of science. Despite the breadth of its foundations, this book is designed to be self-contained. Readers are expected to have only a basic understanding of first-year calculus and some experience with programming. The content is crafted to serve a wide technical audience, including mathematicians, computer scientists, engineers, and other scientifically inclined readers. The mathematical level is suitable for graduate students and advanced undergraduates.

The book is organized into three main parts: Foundations, Applications, and Mathematical Prerequisites. Readers who already have a background in the mathematics covered in the Mathematical Prerequisites may wish to begin directly with the Foundations. However, we recommend at least a brief review of the notation and key concepts introduced in those chapters. Once readers are familiar with the core ideas presented in the Foundations, they can proceed to the Applications. A detailed understanding of the underlying formalism is not essential; a general grasp of the main concepts and results is sufficient to engage with the practical examples and insights explored in

that part of the book.

- *Chapter 1 Introduction* provides a gentle entry point to the theory of nescience, presenting a brief overview of its main concepts and results. While it avoids the use of advanced mathematics, the ideas are introduced in a semi-formal manner. Although it is not recommended as a substitute for the full theoretical exposition, readers who find the mathematics challenging may choose to read only this chapter before proceeding directly to the Applications.

*PART I Foundations* presents a comprehensive account of the theory of nescience, including formal definitions of its core concepts and proofs of key theoretical results. This section forms the core of the book. Readers with prior knowledge in computability, complexity, information theory, probability, and artificial intelligence may choose to begin here directly.

- *Chapter 2 Fundamental Elements* includes the initial step toward quantifying our lack of knowledge, which involves the precise identification of the research entities under examination, determining how to represent them as strings of symbols, and identifying suitable models to explain them. The chapter introduces these fundamental components of the theory of nescience: entities, representations, and descriptions. It examines their properties and the relationships among them. The chapter also discusses how various representations and descriptions can be combined, how background knowledge influences research, and explores the link between perfect knowledge and randomness. It concludes by proposing a novel concept of a research area.
- *Chapter 3 Miscoding* explores the challenge of representing both abstract and concrete research entities as strings of symbols for research purposes. It formally introduces the concept of miscoding and examines its theoretical properties. Miscoding serves as a measure of the error introduced by inaccurate or inappropriate encodings of the entities being studied. The chapter also discusses strategies for minimizing the errors introduced by poor representations, thereby improving the accuracy and utility of symbolic encodings in research. Furthermore, it investigates issues associated with targetless representations, cases where a symbolic encoding exists without a clear or well-defined entity it aims to represent.
- *Chapter 4 Inaccuracy* presents a new interpretation of the classical concept of error, specifically how accurately a description reflects an entity. The concept is generalized to apply to a wide range of topics, including abstract ones. The chapter also introduces joint and conditional variants of inaccuracy and examines their theoretical

properties. It investigates techniques to reduce inaccuracy and explores how miscoding (errors resulting from poor representations) influences the degree of inaccuracy observed.

- *Chapter 5 Surfeit* investigates the redundancy present in a description, specifically how many unnecessary elements it contains. Surfeit serves as an indicator of how well we currently understand research topics and areas, since our lack of knowledge about the entity is typically reflected in the length of our prevailing description. The chapter also introduces joint and conditional variants of surfeit and derives a practical approximation that can be applied in real-world scenarios. Finally, it examines the relationship between miscoding, inaccuracy, and surfeit, and how these three components can be minimized simultaneously.
- *Chapter 6 Nescience* serves as the core of the book, presenting the mathematical foundations of the theory. It defines science as a non-linear multiobjective optimization problem in which the conflicting metrics of miscoding, inaccuracy, and surfeit are minimized simultaneously. A new metric, nescience, is introduced as a function of these three components. The chapter also explores key properties of this metric, including its evolution over time, the concept of perfect knowledge (zero nescience), and methods for identifying our current best model. In addition, it introduces a definition for the frontier of human knowledge and offers a characterization of what lies beyond that boundary.
- *Chapter 7 Interesting Questions* presents a methodology for identifying new research ideas, based on combinatorics and computational creativity, and focusing on how to address challenging open problems. It introduces two new metrics to measure a topic's relevance and its applicability to existing problems, and examines the properties of these metrics. The chapter also outlines a systematic approach to uncovering what lies hidden in the unknown unknown, illustrating how previously unrecognized research directions may be revealed.

*PART II Applications* presents a collection of practical uses of the concept of nescience in areas such as machine learning, the philosophy of science, and the discovery of new research topics. The included examples have been selected to illustrate the broad applicability of the theory, spanning from abstract research questions to more tangible problems grounded in datasets.

- *Chapter 8 Machine Learning* explores how the concept of nescience can be applied to entities represented by collections of measured samples. We introduce mnplib, a software library that can be used to analyze datasets, select relevant features, identify optimal model hy-

perparameters, and compute the errors of trained models. In addition, the chapter presents novel machine learning algorithms, including an innovative method for deriving optimal decision trees and for the automated construction of machine learning models.

- *Chapter 9 Analysis of Science* investigates how well we understand current research topics by applying the proposed metrics. Our aim is to assess the degree of understanding across different areas of science. To this end, we compare research topics within the same academic discipline, as well as across multiple disciplines. The chapter also addresses major open questions in the philosophy of science, including the demarcation problem (how to distinguish science from pseudoscience) and the nature of scientific progress.
- *Chapter 10 The Discovery of the Unknown* demonstrates the practical application of the theory of nescience for identifying promising research questions. Specifically, we show how nescience can be leveraged to generate new research ideas aimed at solving the most difficult open problems. We also propose a methodology for identifying new research topics—that is, for uncovering what lies hidden in the unknown unknown. Multiple examples of such research questions and novel research areas are provided.

*PART III Mathematical Prerequisites* introduces the mathematical foundations necessary to quantify our lack of knowledge about a research topic and to assess the randomness of a string. Its primary aim is to establish consistent notation, formally define key concepts, and present important theoretical results. While no prior expertise is required to follow the material, readers are encouraged to consult the standard references provided at the end of each chapter for deeper understanding. Additional mathematical elements are discussed in the appendices.

- *Chapter 11 Discrete Mathematics* offers a summary of the fundamentals of discrete mathematics needed to understand the more advanced topics discussed in the book. This chapter serves as a quick review of these concepts without providing formal definitions or proofs. Topics covered include sets, relations, strings, graphs, and counting methods. A section on linear algebra (matrices and vectors) is also included.
- *Chapter 12 Discrete Probability* introduces the foundational concepts of probability related to discrete events. Topics covered include conditional probability, random variables, distribution characterization, common distributions, and large random samples. This chapter aims to equip readers with the necessary background in probability to understand the more advanced statistical learning discussed later in the

book.

- *Chapter 13 Computability* presents a formal definition of the concept of algorithm. It introduces the idea of a universal Turing machine and shows that certain well-defined mathematical problems cannot be solved by computers. The chapter also examines the essential tools of oracle Turing machines and Turing reducibility. Key results in computational complexity, relevant to later chapters, are briefly reviewed.
- *Chapter 14 Coding* explores how codes function and how they enable us to compress text by eliminating redundant patterns without losing essential information. It shows that there is a limit to how much text can be compressed using this technique, and that this limit is determined by the entropy of the source. The relationship between optimal codes and discrete probabilities is also discussed.
- *Chapter 15 Complexity* introduces an absolute metric known as Kolmogorov complexity, which measures the amount of information contained in a string by calculating the length of the shortest computer program that can produce it. The chapter studies the properties of this metric in detail and explores the relationship between string complexity and randomness.
- *Chapter 16 Learning* offers a concise overview of the field of statistical learning, presenting key results from statistical inference and machine learning. It also examines the relationship between codes and probabilities, focusing on practical approaches that apply the concept of minimum string length. Additionally, the chapter introduces the concepts and notation associated with nonlinear multiobjective optimization problems.
- *Chapter 17 Philosophy of Science* provides a brief introduction to the field from a philosophical perspective. It reviews key concepts such as scientific representations, models, and theories, and identifies the essential components that any formal theory of science should include. The chapter also offers an overview of the current state of the scientific method.

Finally, the Appendices provide additional information that complements the content of the book.

- *Appendix A Research Agenda* contains a collection of all those elements of the theory of nescience that are still under heavy development. Some of these elements are more advanced than others, but all of them are insufficiently mature to be included in the main corpus of this book. They are also presented here as an invitation to those interested readers

---

to contribute to their development.

- *Appendix B Advanced Mathematics* offers a set of in-depth analyses that support the results presented in the preceding chapters. While these analyses are necessary to validate our findings, their inclusion in the main text would have disrupted the flow of the narrative. For this reason, they have been grouped together in this appendix.  
Finally, *Appendix C About the Photos* explains the origin and intended meaning of the carefully selected photographs featured at the beginning of each chapter.

## Acknowledgements

I would like to express my gratitude to everyone who has contributed their comments and ideas to the development of the theory of nescience. In particular, I am grateful to Antonio Fernández, Vincenzo Mancuso, and Paolo Casari, who believed in and supported this project from its very beginning when it was merely a far-fetched idea (although it may still be). Others who have provided contributions and valuable feedback include Héctor Cordobés, Luis F. Chiroque, Agustín Santos, Marco Ajmone, Pablo Rojo, Manuel Cebrián, Andrés Ortega, Emilio Amaya, Mattis Choummavong, Alexander Lynch, Andrés Carrillo, and Simon Bihoreau. The `mplib` library described in Chapter 8 has been partially funded by the IMDEA Networks Institute, the European Union’s Horizon 2020 research and innovation programme under grant agreement No 732667 RECAP, and Nokia Spain through the project NetPredict.

I would like to express my sincere gratitude to the open source community. This book would not have been possible without the extensive use of open source tools, which provided both the technical foundation and the flexibility required throughout its development. From the writing process, carried out using `TeX` and `LATeX`, to the data analysis and machine learning experiments—powered by libraries such as `scikit-learn`, `statsmodels`, `pandas`, `NumPy`, and `Matplotlib`, the contributions of countless developers and researchers who share their work openly have been indispensable. Their dedication to building and maintaining high-quality, freely available software is a testament to the collaborative spirit of scientific progress.

I would also like to acknowledge the role of ChatGPT in the preparation of this book. As an AI assistant, ChatGPT provided valuable support throughout the writing process, helping to refine ideas, clarify language, structure arguments, and review technical content. While all decisions and final content were ultimately my own, the ability to engage in thoughtful dialogue, receive constructive feedback, and explore alternative phrasings and

formulations greatly enhanced the clarity and coherence of the work. I am grateful for this tool, which has proven to be both efficient and intellectually stimulating.

Finally, I am grateful to my parents, who gave me the opportunities they never had in life, and to my wife and three children, who give ultimate meaning to my existence.

## **Disclaimer**

I would like to clarify that the vast majority of the ideas presented in this book are not my own. The inspiration comes from the brilliant minds of history—thinkers such as Occam, Llull, Leibniz, and Newton, as well as philosophers like Plato, Popper, Feyerabend, and Wittgenstein. I have also built upon the mathematical foundations laid by Turing, Church, Post, Shannon, Solomonoff, Chaitin, Kolmogorov, and many others. My original contribution, if any, lies in connecting some of these ideas and offering what I hope is a compelling reinterpretation of established concepts. The References section at the end of each chapter contains brief descriptions of the sources that have influenced my thinking.

Throughout the text, I use the passive voice ("it is defined") when referring to concepts whose origins I recognize, and the active voice ("we define") when I am unaware of any prior formulation.



# 1. Introduction

*If presented with a choice between indifferent alternatives,  
then one ought to select the simplest one.*  
Occam's razor principle

We find ourselves in an age where knowledge has become one of our most valued resources, driving both economic progress and societal well-being. Science continually provides us with deeper insights and practical solutions, shaping our daily lives through innovations ranging from advanced medical therapies to transformative digital technologies. However, the path of scientific exploration is neither straightforward nor free from significant barriers. Despite remarkable progress, our current scientific methodologies remain bounded by inherent limitations that restrict our ability to fully understand and address complex issues.

These limitations manifest in various ways, including fragmented research efforts, rigid disciplinary boundaries, and an often narrow approach to funding allocation, which tends to overlook bold, high-risk projects with transformative potential. Moreover, the conventional metrics of scientific success frequently incentivize incremental progress rather than genuinely groundbreaking discoveries, limiting innovation's pace and scope.

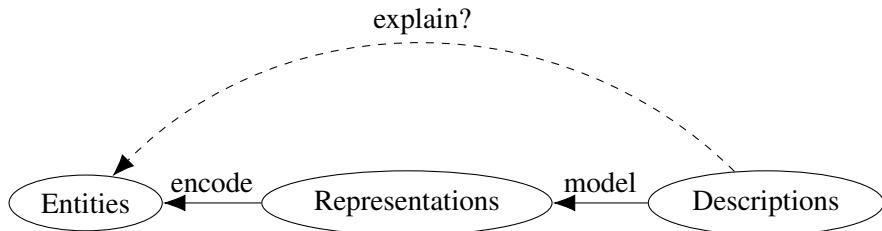


Figure 1.1: The Problem of Understanding

The theory of nescience emerges from the critical recognition of these challenges and the profound need to overcome them. It advocates a renewed approach to exploring the unknown, driven by curiosity and the willingness to question accepted knowledge rigorously. By systematically addressing the limitations inherent in contemporary scientific practice, our objective is to provide robust frameworks that significantly enhance our capability to solve real-world problems and catalyze meaningful advancements across diverse fields.

## 1.1 Scientific Knowledge

The pursuit of knowledge begins with identifying the *entities* we seek to understand, entities that are often extraordinarily diverse. Mathematicians focus on abstract concepts, biologists on living organisms, and engineers on machines. Our quest for understanding is fundamentally driven by our need to predict outcomes, solve problems, and navigate the complexities of the world around us. For instance, we know that applying sufficient heat to wood will ignite a fire, or that a fever may indicate a viral infection. Practical problem-solving, therefore, hinges crucially on our ability to recognize patterns and create simplified models of these entities. These models empower us to interact beneficially with the world.

Yet, while ideally, we strive to build precise *descriptions* capable of fully reconstructing the entities we study, in reality, such perfection often eludes us, particularly with abstract entities. Consequently, we rely on *representations*, texts or datasets capturing essential details of these entities. Physicists might represent an entity through experimental results; computer scientists through measured data; sociologists through observed facts. Figure 1.1 illustrates this critical interplay: although we aim to directly describe entities, practically, our descriptions are based on constructed representations.

Understanding the effectiveness of our descriptions involves examining potential errors deeply. Firstly, encoding methods can introduce errors,

what we call *misCoding*. Secondly, our descriptions themselves may fail to precisely recreate the encoded representations, resulting in *inaccuracy*. Lastly, due to human cognitive limitations, descriptions should not become unnecessarily complex, introducing *surfeit*.

Collectively, misCoding, inaccuracy, and surfeit hinder our quest for clear understanding and reliable predictions. By amalgamating these three error types into a single measure, known as *nescience*, we quantify our lack of knowledge, providing an essential tool for systematically addressing and reducing our ignorance.

## 1.2 Entities

At the core of our theory of nescience is the recognition that science is, at its essence, a quest to understand the world around us. Throughout history, scientists have examined an extraordinary variety of things (planets, particles, languages, ecosystems, human societies, and much more) in an effort to uncover patterns, formulate explanations, and make predictions. These things we seek to understand, which we refer to as *entities*, form the basis of all scientific activity. An entity might be a tangible object, such as a chemical compound or a cell, or something more abstract, like a mathematical function or a cultural practice.

The scope of what science might investigate is vast and continually evolving. New technologies, shifting societal needs, and fresh philosophical insights regularly bring new entities into view. What unifies this effort is a foundational belief: that some of these entities can be understood through science. That is, we hold that at least part of the unknown is ultimately knowable (see Section 17.2). This belief fuels the scientific drive to bring clarity to what was once obscure, to shed light on complexity, and to convert speculation into knowledge. Understanding how we fall short of this ideal, how ignorance persists and why, is the starting point for the theory of nescience.

One important conceptual difficulty we face is that the set of all entities under consideration, what we refer to symbolically as  $\mathcal{E}$ , cannot be rigorously defined in mathematical terms, except to say that it must be non-empty. This may seem like a minor point, but it carries deep implications.

In mathematics, the idea of a "set of everything" is fraught with contradictions. For instance, if we tried to construct a set that included absolutely all things (physical objects, abstract ideas, ...) we would quickly encounter logical problems. This is why we deliberately avoid the notion of a universal set in the theory of nescience. A key reason for this caution lies in Cantor's theorem, which we discuss further in Section 2.1. Cantor's result shows that

for any set, the collection of all its subsets is strictly larger in size than the set itself. As a consequence, it becomes impossible to form a set that includes everything without running into contradictions. Similarly, we steer clear of problematic constructions that lead to paradoxes, such as Russell's paradox, another example covered in Section 2.1, which illustrates how self-referential sets can collapse logical consistency.

These mathematical constraints are not mere formalities; they serve a vital purpose. By enforcing clear boundaries around the sets of entities we analyze, we ensure that the theory we are building remains coherent, consistent, and applicable. It allows us to focus our attention on meaningful, well-defined domains where real progress in understanding can be made.

In practice, we will work with well-defined sets, each associated with a specific domain of inquiry and its unique goals. These sets provide a practical framework for applying our theory to real-world contexts. For example, in mathematics, such a set might include different classes of abstract structures such as groups, functions, or topological spaces. In biology, it could encompass the vast diversity of living organisms, from microscopic bacteria to complex multicellular animals. In the realm of social sciences, the entities might include human behaviors, social systems, or economic models. And in computer science, we may focus on algorithms, data structures, and executable programs.

By tailoring our analysis to these different sets, we are able to apply a unified theoretical framework to a wide variety of disciplines. This adaptability is one of the strengths of our approach: it allows us to measure and reduce human ignorance, or nescience, in fields with very different kinds of entities. Our goal is not only to improve our theoretical understanding of these domains but also to contribute practical tools that can support deeper insights, better decision-making, and more effective problem-solving across the sciences and beyond.

### 1.3 Representations

In many instances, entities cannot be directly scrutinized through scientific analysis, particularly if they are abstract. As a result, we are compelled to rely on representations, i.e., symbolic encodings that stand in for the entities we aim to understand. We designate the collection of strings that encode the entities of  $\mathcal{E}$  as  $\mathcal{R}_{\mathcal{E}}$ . These strings, referred to as *representations*, may differ depending on the application of the theory of nescience. In certain scenarios, entities may inherently be string-based (e.g., computer programs), while others might be abstract objects that require encoding into string format (e.g., human needs). It is not uncommon for a single entity

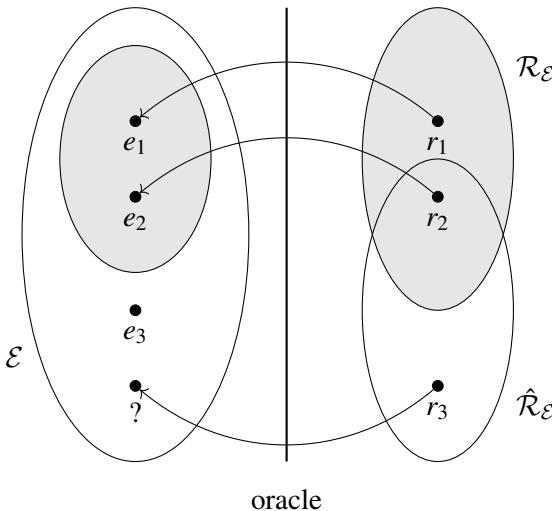


Figure 1.2: Entities and Representations.

$e \in \mathcal{E}$  to have multiple valid representations within  $\mathcal{R}_{\mathcal{E}}$ , for example, a text-based description, a diagram saved as a computer file, or a collection of empirical measurements. Each representation emphasizes different aspects of the entity and may be better suited to particular investigative goals or scientific approaches. Transforming abstract entities into symbol strings in a manner that faithfully captures their complexities and subtleties remains a formidable and unresolved challenge. Consequently, the exact composition of  $\mathcal{R}_{\mathcal{E}}$  often eludes us.

In a world where understanding depends on symbolic surrogates, the idea of constructing an encoding function  $f : \mathcal{E} \rightarrow \mathcal{R}_{\mathcal{E}}$  becomes not only attractive but essential. The function  $f$  represents an idealized encoding process: it assigns to each entity  $e \in \mathcal{E}$  one of its symbolic representations  $r \in \mathcal{R}_{\mathcal{E}}$ . In essence, this function models the act of transforming something we wish to understand (like a physical object, a biological system, or an abstract concept) into a format that can be studied, manipulated, or stored using symbols, typically as strings. In practice, such a function would allow us to systematically move from the domain of real-world or conceptual entities to their formal encodings, which are necessary for analysis in science, computation, and communication. However, defining such a function is no trivial task, precisely because  $\mathcal{E}$  itself is not well defined. The boundaries of what should or should not be included in  $\mathcal{E}$  are inherently blurred. For example, we still lack a precise and universally accepted definition of what constitutes a human need.

To grapple with this indeterminacy, one may resort to theoretical constructs such as the *oracle Turing machine* (see Chapter 13). While a standard *Turing machine* serves as a mathematical model of computation, the oracle variant introduces a conceptual leap: it simulates a computer with access to an external source of information. The oracle Turing machine can be imagined as a theoretical computer connected not to the actual internet of today, but to an idealized version, an omniscient information source containing perfect knowledge about everything that exists or could exist. This imaginary machine is allowed to submit string-based queries to this vast external database. For instance, it might ask whether a given string  $r$  encodes any entity in  $\mathcal{E}$ . Unfortunately, formulating the question “does  $r$  represent  $e$ ?” would require expressing  $e$  itself as a string of symbols. And since we do not know how to encode  $e$  in advance, we cannot construct such a query. This irony captures the very dilemma we aim to address, the tension between what can be queried, what can be known, and the inherent limitations of representation in scientific inquiry.

From a practical perspective, we typically approximate the set  $\mathcal{R}_{\mathcal{E}}$  with another set  $\hat{\mathcal{R}}_{\mathcal{E}}$  of strings, which we consider to be adequate representations of the entities of  $\mathcal{E}$ . In scientific practice, these representations have traditionally taken the form of illustrations or images (e.g., in biology), collections of factual data (e.g., in sociology), or experimental results (e.g., in physics). With recent significant advancements in the capability of computers to gather and store data, a novel and potent method for encoding entities has emerged: using vast data sets as representations. It’s essential to note that in the encoding process, our objective is not to find the shortest possible representation of the entities but to seek out high-quality representations.

It’s crucial to acknowledge that in numerous practical scenarios, the chosen representations of abstract entities may not fully encapsulate all nuances of the original objects. This means we are grappling with simplified abstractions of reality, which could potentially curtail our capacity to make sweeping assertions about nature (see Chapter 3).

■ **Example 1.1** If we’re studying animals (the set  $\mathcal{E}$ ), we could use a binary encoding of their DNA (the set  $\mathcal{R}_{\mathcal{E}}$ ) as representations. While our current technology doesn’t allow us to bring a creature to life solely from its DNA, theoretically, it could be feasible. However, DNA alone doesn’t fully replicate the original animal, as it doesn’t include life experiences. For instance, how would we represent a cat that only has three legs due to an accident? That detail is not recorded in its DNA. If our goal is to study the traits of certain species, working with the DNA of a representative sample of individuals within each species would be adequate. However, if we’re studying specific individuals within a species, we would also need a way to encode each

animal's history or the details not encapsulated by the DNA. ■

Working with strings as representations (the set  $\mathcal{R}_{\mathcal{E}}$ ) inevitably results in certain entities lacking any corresponding encoding (see the gray areas in Figure 1.2; specifically, entity  $e_3$  has no representation). This limitation becomes particularly evident when the set of entities is uncountable. For example, if  $\mathcal{E}$  is the set of real numbers, many elements cannot be represented, since we restrict representations to finite binary strings. Real numbers that require infinite precision (such as most irrational numbers) cannot be fully encoded. This mismatch reveals a deeper asymmetry: in many domains of knowledge, the space of conceivable problems or entities far exceeds the space of valid, encodable representations. Intuitively, this suggests that in such domains, the quantity of problems may exceed the number of solutions.

Using approximations of representations (the set  $\hat{\mathcal{R}}_{\mathcal{E}}$ ) can also result in some representations encoding the wrong entities, as illustrated by representation  $r_3$  in Figure 1.2. This occurs because our knowledge about the entities in  $\mathcal{E}$  is often incomplete or imprecise, which can lead us to construct representations that appear valid but fail to correspond to the intended entity.

Another issue with incomplete knowledge is the possibility of having unknown entities whose existence we are unaware of. For instance, representation  $r_1$  in Figure 1.2 is not part of the set  $\hat{\mathcal{R}}_{\mathcal{E}}$  and is therefore overlooked by researchers despite being the representation of a knowable entity  $e_1$ . One of the objectives of this book is to provide a procedure to uncover new, previously unknown, research entities from the set  $\mathcal{R}_{\mathcal{E}}$  (refer to Section ?? and Chapter 10).

## 1.4 Descriptions

Upon identifying the set  $\mathcal{R}$  of potential representations, we are faced with the deeper motivation that drives much of scientific inquiry: the desire to bring order to the complexity of the world. To do this, we must devise appropriate methods to describe these representations, an endeavor that underlies our attempt to form theories or models that articulate how the world operates. Our limited cognitive capacities as humans compel us to work with simplified, yet insightful, models of nature. These abstractions help us interpret phenomena and forecast the consequences of our actions. Descriptions also change over time, as our understanding of the entities studied improves.

■ **Example 1.2** To anchor these ideas in the concrete, consider the evolving effort to describe the macroscopic behavior of the physical universe (the entity  $e$ ). The sequence of proposed descriptions includes Aristotelian physics, Cartesian mechanics, Newtonian laws, Einstein's relativity, and, potentially,

superstring theory. Each successive model attempts to refine our grasp of reality. Among these, Einstein's theory currently stands as the most complete, given that alternatives like superstring theory still await experimental corroboration. ■

Defining a valid description for an entity is not merely a technical challenge; it embodies a fundamental limitation of knowledge. The Berry paradox serves as a compelling reminder of these philosophical intricacies. A phrase like "the smallest positive integer not definable in less than twelve words" becomes paradoxical by succeeding in doing just that within eleven words. To avoid such pitfalls, the theory of nescience imposes stricter demands: a valid description must be a finite symbol string that allows us to effectively and completely reconstruct a possible representation of the original entity. By "effectively," we mean that this reconstruction can be performed by a machine, or computer, without human intervention.

From Newton's formulation of classical mechanics to today's explorations, the scientific journey has been shaped by the pursuit of mathematical models. The theory of nescience follows this lineage but extends it by demanding that models be computable, i.e. executable by computers. This requirement of computability of descriptions allows us to remove many of the paradoxes traditionally associated with the concept of description. In this light, science becomes not only a quest for understanding but also a computable-driven endeavor to approximate reality.

Descriptions are typically divided into two components: a Turing machine  $TM$  (a computer program) that encapsulates all the regularities found in the entity's representation (the compressible part), and an input string  $a$  that contains a literal description of the remaining elements (the non-compressible part). This dualistic nature of descriptions parallels traditional distinctions in science, such as theories and assumptions, theories and initial conditions, problems and specific problem instances, species and individuals, and so on. For instance, a description might consist of a system-modeling set of differential equations (the compressible part), accompanied by a compilation of initial conditions (the non-compressible part). The precise interpretation of the pair  $TM, a$  relies on the specific characteristics of the entity set to which the theory is applied.

Figure 1.3 illustrates the relationship between entities, representations, and descriptions. The set of all potential descriptions is denoted by  $\mathcal{D}$ . However, not all strings qualify as valid descriptions: each must be grounded in a Turing machine, ensuring that it is computationally meaningful. Furthermore, not every valid description corresponds to a legitimate representation. Since representations  $r$  can be described in multiple ways, the overarching scientific goal emerges as the search for the shortest possible description  $d$

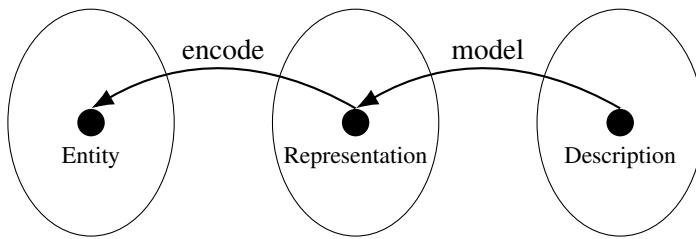


Figure 1.3: Entities, representations and descriptions.

that faithfully reconstructs the observed data.

However, a fundamental obstacle lies in the incomputability of this task. As discussed in Chapter 15, there exists no general procedure to determine the shortest program that outputs a given string. This impossibility extends to representations, rendering the pursuit of optimal scientific models as a challenge beyond computer capabilities. As a result, science must rely on heuristic strategies to approximate ideal solutions. Collectively, these heuristics define what we call the *scientific method*.

The theory of nescience is driven by the desire to understand, and ultimately quantify, the various errors that arise in the process of scientific discovery. Figure 1.3 provides the conceptual framework for this endeavor. In Sections 1.5, 1.6, and 1.7, we introduce metrics designed to capture distinct sources of error. These components are then synthesized in Section 1.8 into a single, unified measure: nescience. Although inherently uncomputable, this measure formally expresses the extent to which a given research entity remains poorly understood. It highlights not only the boundaries of current knowledge but also the areas most deserving of scientific attention.

## 1.5 MisCoding

As we've observed, in many scientific disciplines, the effort to understand the world often begins with an elusive challenge: the entities we wish to study, denoted as the set  $\mathcal{E}$ , do not always lend themselves to clear or complete representations. Some of these entities are too abstract, others too complex, and many remain partially known. This mismatch between the reality we wish to grasp and the means we have to represent it is not merely a technical limitation, it reflects the very heart of scientific inquiry.

The scientist's journey, then, often starts in uncertainty. We make do with approximations, crafting descriptions that we hope capture enough truth to be useful. Yet, we are aware that these representations carry errors. Our goal becomes not just to encode entities, but to quantify the error introduced by using these inaccurate representations.

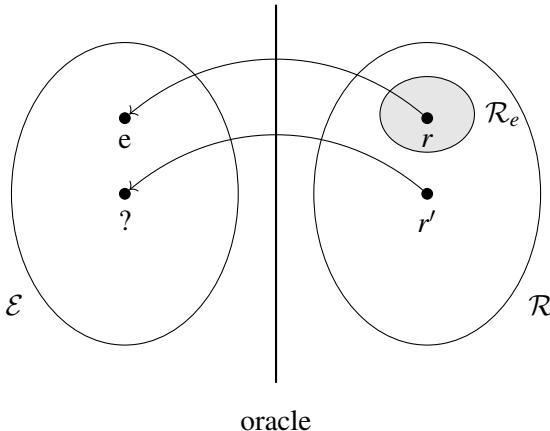


Figure 1.4: Miscoding of topics.

We propose to measure the *miscoding* of an inaccurate representation  $r'$  by assessing how difficult it is to transform this flawed representation into a correct one. In more technical terms, this difficulty is expressed as the length of the shortest computer program that, when given  $r'$  as input, is able to produce the accurate representation  $r$ . Figure 1.4 provides a visual depiction of this process. The set  $\mathcal{R}_e$  consists of all the strings that are considered accurate representations of the entity  $e$ . If falls outside this set, it means our understanding is flawed. Miscoding represents the cost of bridging that gap. Importantly, this cost is not about computational time or speed. It is about descriptive complexity, how much must be said, or programmed, to repair the mistakes. If  $r'$  contains inaccuracies, the required program must identify these deviations and correct them. If instead  $r'$  lacks key information altogether, the missing content must be embedded within the program itself. In this way, miscoding becomes a reflection of our ignorance: the larger the program needed, the more knowledge we have to include to arrive at the truth. Miscoding measures how much we still need to learn before our representations truly encode the entities we seek to understand.

However, the previous method of measuring miscoding does not fully capture our intuitive understanding of what it means for a representation to be flawed. There is a deeper issue that emerges when a representation includes more than what is necessary. In other words, it may not only be inaccurate by omission but also by addition. Consider a situation where the representation  $r'$  contains extra information that is irrelevant or unrelated to the entity  $e$  we are trying to understand. This surplus information is not just noise, it actively distorts the description by forcing any model based on  $e$  to account

for elements that have no bearing on the actual entity. The result is a bloated and misleading representation. The description becomes longer not because the entity itself is more complex, but because our flawed representation includes unnecessary baggage. This problem is not just theoretical. In real-world scientific practice, we frequently encounter such scenarios. Imagine an experiment where multiple variables are being recorded, but only a few of them actually influence the outcome. If we do not yet know which variables are relevant, our current models might treat all recorded features as potentially significant. This lack of understanding can lead us to build explanations and predictors around elements that are, in reality, unrelated to the entity or phenomenon of interest.

To account for this kind of misrepresentation, we must expand our definition of miscoding. We need to ask not only how much effort is required to fix an inaccurate representation, but also how much effort it would take to reconstruct that flawed version from the correct one  $r$ . The higher this effort, the worse the representation is, as it suggests that the inaccurate description deviates significantly from what is accurate. This leads us to introduce the a second measure, as the length of the shortest computer program that can output the incorrect description  $r'$  given the accurate one  $r$ . Only by considering both directions, how difficult it is to go from  $r'$  to  $r$ , and how difficult it is to go from  $r$  to  $r'$ , can we fully assess the degree of miscoding. We therefore define miscoding as the maximum of these two values. This revised definition acknowledges that misinformation can come in multiple forms. It captures both the missing and the misleading, recognizing that a poor representation might not only fail to say what is necessary, but might also say unnecessary things. In this way, miscoding becomes a more complete reflection of the divergence between what we currently believe and what truly is.

Nevertheless, this latest definition still poses practical challenges. In many cases, the same entity can be described in multiple, equally valid ways. This multiplicity creates a dilemma: what should we do when our inaccurate representation  $r'$  is far from one correct representation  $r_1$ , but quite close to another valid one  $r_2$ ? Judging solely by its distance to  $r_1$  could unfairly suggest a high level of miscoding, when in fact  $r_2$  may be a legitimate approximation of  $e$ .

The core issue here is that correctness is not always unique. Scientific and mathematical entities often admit many forms of expression—each highlighting different properties, or suited for different contexts. Penalizing a representation for not being similar to just one of these correct forms would ignore the richness and flexibility of representation.

■ **Example 1.3** Consider  $e$  as the abstract entity known as the "Pi constant",

the ratio of a circle's circumference to its diameter. Let  $r$  be the Wallis product, expressed as  $2\left(\frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdots\right)$ , a well-known infinite product that converges to  $\pi$ . Suppose  $r'$  is the infinite series  $4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots\right)$  which corresponds to the Leibniz series. Although  $r'$  is structurally very different from  $r$ , it also accurately represents the same entity  $e$ . To declare  $r'$  as highly miscoded with respect to  $r$  would be misleading, because  $r'$  is not an error, it's an alternative, equally correct expression. ■

As example 1.3 indicates, defining miscoding poses challenges because the set  $\mathcal{R}_e$  of valid representations for the entity  $e$  is generally unknown. Theoretically, we could rely again on the oracle Turing machine to solve this problem. However, as observed, we can't ask the oracle if the string  $r$  is a valid representation of our interested entity  $e$  (the set  $\mathcal{R}_e$ ), since that would require us to provide a valid encoding of  $e$  as a string of symbols, which typically can't be done. Perhaps, all we can do is ask this oracle how close a given string  $r$  is to being a valid description of some entity in the entire set of entities  $\mathcal{R}_{\mathcal{E}}$ .

With this constraint in mind, we have to define the miscoding of a representation as the smallest possible discrepancy, as judged by the oracle, between our given string and any valid representation of any entity. In this way, we allow the oracle to search through the universe of all correct representations, looking for the one that is most similar to our candidate. This definition enables us to talk about miscoding even in the absence of a known ground truth.

But there is one final complication we must confront. Because this definition of miscoding does not rely on knowing the actual entity being represented, it opens the door to a subtle yet critical problem: we might not be representing what we think we are. In other words, our descriptions might be well-formed and internally consistent, yet point to an entirely different entity than we had intended. This kind of mistaken identity is not just a philosophical curiosity, it has occurred repeatedly in the history of science. Researchers have often believed they were investigating one phenomenon, only to later discover that their results pertained to something entirely different.

■ **Example 1.4** In the late eighteenth century, chemist Joseph Priestley believed he was studying a substance called "phlogiston", which was thought to be a fire-like element released during combustion. All of his experiments and representations were constructed around this idea. However, in reality, Priestley was observing the properties of a completely different entity: oxygen. Though his descriptions were coherent and reproducible, they were ultimately anchored to the wrong conceptual foundation. ■

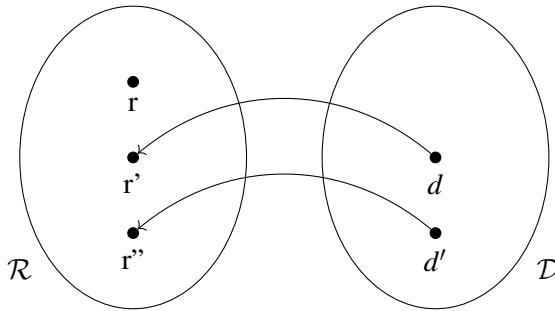


Figure 1.5: Inaccuracy of a description.

According to the theory of nescience, our role as researchers goes beyond the mere identification of correct representations for the entities we wish to understand. It also involves a deeper and more ambitious task: uncovering the principles by which an idealized oracle machine would reconstruct the original entities given their representations. In other words, it is not enough to arrive at accurate representations, we must also strive to understand why these representations are accurate, what makes them effective, and how they reflect the intrinsic structure of the underlying entities, by understanding how this hypothetical abstract oracle would work. This shift in perspective moves us from a practice of isolated trial and error to a more reflective inquiry into the nature of scientific representation itself. Our goal is not just to find a representation that works, but to understand the encoding process more deeply.

## 1.6 Inaccuracy

In the preceding section, we explored how ignorance may arise when the representation  $r'$  we use to encode an entity  $e$  does not match the actual entity's correct representation  $r$ . That was the problem addressed by miscoding. Now, we turn to another, equally important source of ignorance, what we call *inaccuracy*, which emerges not from selecting the wrong representation, but from failing to describe it adequately.

In an ideal scenario, we would have a description  $d$ , that is, a computer program, capable of fully reconstructing the representation  $r'$ , even if the true representation  $r$  remains unknown. However, this level of precision is rarely attainable in practice. More commonly, we rely on an approximate description  $d'$ , which produces a string  $r''$  that resembles  $r'$ , but is not exactly the same. In such cases, we refer to  $d'$  as an *inaccurate* description of the representation  $r'$  (see Figure 1.5).

If a description is inaccurate for a representation, it is useful to have a quantitative measure of how much we deviate from accurately modeling the representation. A viable method to define this measure could be calculating the effort needed to rectify the output of our inaccurate description. In this context, the inaccuracy could be determined by the length of the shortest computer program that can generate the correct representation when fed with the incorrect one produced by the description. However, similar to the case with miscoding, to have a holistic understanding of the error associated with the description  $d$ , we must also calculate the difficulty of generating the inaccurate representation given the correct one. It's possible that our description  $d$  models elements unrelated to the representation  $r'$ , and merely ignoring these elements won't solve the problem.

In other words, the inaccuracy measures how difficult it is to convert the output of the description into the intended representation, or the other way around, how difficult is to convert the representation in the output of our description. The larger the maximum of these two values, the more the description deviates from accurately capturing the intended representation.

We deliberately prefer the term *inaccuracy* over the term *error*. In the language of continuous systems, error includes both precision and accuracy. But in our discrete framework, where descriptions are finite symbol strings, precision loses its relevance. What matters here is how well the structure of the description aligns with the structure of the representation.

In practice, calculating the inaccuracy associated with the description of a representation is a challenging task. As previously mentioned, determining the length of the shortest computer program that can print a string is a non-computable problem. If the original entities are texts themselves, we could approximate the inaccuracy using compression algorithms. Here, the string complexity is approximated by the length of the compressed text using a compressor. If the topics are abstract entities, such as mathematical concepts, their descriptions could be derived from the result of an experiment. Hence, the inaccuracy could be based on the model's error (for instance, by calculating the length of additional information required to thoroughly describe the experiment's results given the model). In this regard, our definition of inaccuracy is a generalization of the concept of error. It can be applied to various types of entities, not only those that can be encoded as datasets.

■ **Example 1.5** Consider Newton's second law of motion,  $F = ma$ . Suppose we construct a dataset by applying known forces to objects of varying masses and measuring their resulting accelerations. If our goal is to study gravitational acceleration, the force and mass terms cancel, isolating acceleration as the variable of interest. Encoding this dataset in full would require a significant number of bits. Yet, recall that our objective with representations

is not to minimize string length but to ensure that the encoding captures the richness and structure of the underlying phenomenon.

In this example, we draw on a historical experiment conducted by the National Bureau of Standards in Washington D.C. between May 1934 and July 1935. The dataset includes 81 measurements of acceleration in centimeters per second squared, for instance, a value like 980,078. Using a uniform 20-bit encoding per measurement, the full dataset requires 1,620 bits. Suppose a model predicts a gravitational acceleration of  $980,000\text{cm/s}^2$  plus noise. If encoding the dataset using this model only requires 453 bits, the model's inaccuracy is estimated as:

$$\frac{453}{1,620} = 0.27$$

This tells us how much information must be added to the model to fully account for the empirical data. It quantifies the gap between representation and reality. ■

What this example illustrates is a profound ambiguity: when our models and our data disagree, we cannot determine, in general, whether the failure lies in the experiment (misCoding) or in the model (inaccuracy). This ambiguity is not a flaw of the framework, it is a reflection of the inherent uncertainty we face when attempting to describe the world. And it is precisely this uncertainty that the theory of nescience seeks to explore, quantify, and ultimately reduce.

## 1.7 Surfeit

In our pursuit of understanding, we often encounter a paradox: the more complex and verbose our explanations, the less confident we should be in the depth of our knowledge. Complexity, when unnecessary, signals confusion. When we struggle to explain a concept concisely, it is likely because our grasp of it remains partial. This observation reveals a deeper motivation behind scientific inquiry: the drive to eliminate what is superfluous, to strip our models down to their essential structure. By doing so, we gain clarity not only in explanation but also in prediction and control.

Science depends on descriptions, most often in the form of mathematical models, to interpret the past, predict the future, trace the connections between cause and effect, and engineer solutions to practical problems. But these models, to serve us effectively, must remain within the limits of our cognitive capacities. As scientists and engineers, we are compelled to seek out models that are not only correct but also minimal. This minimality is not an aesthetic preference; it is a cognitive and computational necessity. Even if, in the

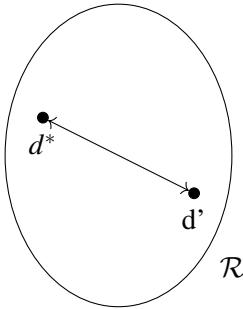


Figure 1.6: Surfeit of a model.

future, machines take over the work of scientific reasoning, and our concern for human comprehensibility fades, the idea of minimizing unnecessary complexity will remain a conceptual cornerstone, albeit perhaps relegated to a more theoretical realm.

The theoretical limit of our knowledge about a representation, denoted as its perfect description  $d^*$ , is given by the shortest possible computer program that can reconstruct that representation. In practice, we rarely achieve this ideal. The excess or surfeit of a description  $d'$  can be calculated by comparing the length of this specific description to the length of the shortest possible description  $d^*$  for that representation (refer to Figure 1.6). In other words, *surfeit* measures how much longer a description is compared to the most concise description we could ideally achieve. Regrettably, due to our incomplete knowledge, we generally do not know the shortest description of a representation, hence, surfeit is a value that must be estimated in practice.

This definition leads us to a profound insight: perfect knowledge implies randomness. If a description were perfect, it would be incompressible, any pattern or regularity would suggest redundancy that could be eliminated. According to the theory of computation, incompressible strings are indistinguishable from random sequences (see Section 15.6). Thus, within the framework of our theory, a perfect description of a phenomenon must be random.

This conclusion might seem counterintuitive. Traditionally, randomness has been associated with disorder and the absence of meaning. But in our context, a random description is not meaningless, it is simply maximally informative, packing as much content as possible into the smallest possible space. Still, not all random descriptions are perfect. A theory might become increasingly compressed and eventually appear random, only for a new representation to emerge, perhaps based on a different encoding, that offers a shorter and more insightful description.

Understanding randomness in this way reshapes the boundaries of what we consider knowable. Rather than seeing randomness as a wall, we see it as a marker of how far we've progressed. It signals the frontier of comprehension, a point where additional refinement is no longer possible unless we reconceptualize the problem. And crucially, this perspective doesn't just limit us to analyzing what we already know; it provides a strategy for identifying gaps in knowledge and pointing toward new avenues of discovery.

With our definition of surfeit, in which lengthier explanations are deemed inferior, we aren't implying that textbooks should always strive for utmost brevity. Contrarily, in certain situations, we anticipate textbooks to be highly redundant. A concise book contains an abundance of information in a very condensed space, making it challenging for humans to assimilate (understand) that information. However, a redundant textbook (such as this one) presents the same amount of information but in a larger space, hence, its content is easier to comprehend. Moreover, in fields outside of science, redundancy may be desirable. For instance, in law, redundancy aids lawyers in memorizing legal texts, and in music, repetition can contribute positively to harmony, as exemplified in a canon.

## 1.8 Nescience

*Nescience* is an old-fashioned English word meaning "lack of knowledge or awareness." At first glance, it appears to be synonymous with the word "ignorance." However, there is a subtle but important distinction: ignorance refers to the absence of knowledge when such knowledge exists and could be acquired (for instance, by reading a book), whereas nescience refers to the absence of knowledge when that knowledge does not yet exist, when it is unknown to everyone. The theory of nescience has been developed to quantitatively measure how much we do not know in situations where knowledge is not yet available, aiming to capture the extent of our collective ignorance as a species.

Intuitively, the extent of what we do not know about a research entity can be assessed through the metrics of miscoding, inaccuracy, and surfeit associated with a given representation and description. These three metrics capture the main types of errors we might make. Miscoding reflects how well the representation encodes the original entity under investigation; inaccuracy indicates how effectively the description models that representation; and surfeit measures the quality of the description itself in terms of unnecessary complexity or verbosity. The best combinations of representations and descriptions are those that exhibit low miscoding, low inaccuracy, and low surfeit.

However, these factors are often in tension with one another. Improving one dimension can inadvertently worsen another, making the process of refining knowledge inherently complex. For example, increasing the complexity of a description might improve its accuracy, bringing the output closer to the intended representation, but this gain may come at the cost of clarity and simplicity, thereby increasing surfeit. Similarly, modifying a representation to better match the original entity, thus reducing miscoding, can lead to an increase in the inaccuracy of the description, which may no longer align well with the new representation. In scientific research, such trade-offs are frequently encountered. More accurate models often require additional parameters, more sophisticated computations, or richer structures, which can obscure understanding, hinder reproducibility, or limit practical applicability. Recognizing these tensions helps researchers make more informed decisions about model selection, balancing the competing goals of simplicity, fidelity, and succinctness. The theory of nescience makes these trade-offs explicit, providing a structured way to analyze and optimize them.

A pair  $(d, r)$ , composed of a description and a representation, is said to be Pareto optimal if there is no other pair  $(d', r')$  that improves at least one of the three components of nescience—miscoding, inaccuracy, or surfeit—without worsening another. In this sense, a Pareto optimal pair represents a balance point where any improvement in one dimension would result in a degradation in another. This notion helps us identify a set of candidate  $(d, r)$  pairs that offer the best possible trade-offs and jointly provide strong explanatory value for an entity  $e$ .

However, in scientific practice, we typically aim to select a single description to serve as the model for a research entity. To make this selection, we must define a utility function that enables us to classify and rank the candidate descriptions. The exact form of this utility function depends on the field in which the theory of nescience is being applied. For instance, in machine learning, where entities are represented as datasets, a reasonable utility function might be the average of the three components of nescience: miscoding, inaccuracy, and surfeit. This provides a simple and effective way to evaluate and choose the most suitable description.

In traditional scientific approaches, it is common to fix a particular representation of the entity under investigation and then focus on selecting or developing a model that minimizes inaccuracy, and possibly surfeit. While this methodology is often effective within well-established domains, it risks overlooking better alternatives that arise from reconsidering the representation itself. The theory of nescience emphasizes that miscoding, inaccuracy, and surfeit must be minimized simultaneously. This holistic approach avoids the danger of settling into a local minimum—where a model seems optimal

given a fixed representation, but a better explanation might exist elsewhere in the space of possible (representation, description) pairs. By jointly considering and optimizing both the representation and description, the theory enables a more comprehensive and flexible exploration of scientific models.

## 1.9 Perfect Knowledge

In this book, we assume that the final objective of science is to achieve perfect knowledge, understood as the state in which our understanding of an entity is both accurate and efficient. This is equivalent to discovering pairs of descriptions and representations with the lowest possible nescience. Scientific progress, from this perspective, is inherently iterative: over time, new candidate pairs of descriptions and representations are proposed, each intended to bring us closer to this ideal state by reducing nescience.

There are two fundamental strategies for decreasing nescience. One is to develop new descriptions, these could stem from the formulation of novel theories, improvements upon existing explanations, or the elimination of unnecessary assumptions. The second is to discover better representations, ways of encoding or framing the entity under study that more faithfully reflect its essential properties. Both avenues are crucial and complementary, as a poor representation can obscure the merits of an otherwise sound description, and vice versa. Thus, progress in science involves the dynamic refinement of both how we represent and how we describe reality.

When the nescience of a pair composed of a representation and a description is equal to zero, we say that *perfect knowledge* about an entity has been achieved. This state corresponds to the simultaneous minimization of the three components of nescience: miscoding, inaccuracy, and surfeit. Miscoding being zero means the representation perfectly encodes the intended entity; inaccuracy being zero means the description fully and faithfully reconstructs the representation; and surfeit being zero means the description contains no redundancy, it is as concise as possible. In this ideal scenario, our understanding of the entity is complete, exact, and optimally efficient.

As previously described, representations aim for completeness: they must encapsulate all relevant aspects of an entity, even at the cost of introducing redundancy. This ensures that nothing essential is omitted and that the representation is faithful to the entity as it exists. In contrast, descriptions are governed by a strict economy of expression. Brevity is essential and is formally measured by surfeit. The ideal description is one that conveys the maximum possible information using the fewest possible symbols. In fact, perfect knowledge requires descriptions that are random in the algorithmic sense, meaning they are incompressible and contain no patterns that

would allow them to be expressed more succinctly. Such descriptions are optimally efficient: they cannot be improved upon. However, it is a mistake to assume that any random-looking description necessarily corresponds to perfect knowledge. Randomness alone does not guarantee that miscoding and inaccuracy are zero; a random string may still be describing the wrong representation or doing so poorly.

More critically, if a description is already random, if it is the shortest and most accurate possible for a given representation, then any attempt to replace it with an alternative will necessarily result in increased nescience. Continuing to search for new descriptions in such a case, without recognizing that the current one is already optimal, leads to a regression in understanding. We may introduce inaccuracy or redundancy, thereby increasing our ignorance rather than reducing it. This underlines the importance of knowing when to stop: in some cases, further research into new descriptions can obscure rather than clarify.

It is important to recognize that there may not be a single ultimate theory or uniquely optimal pair. Multiple combinations of representations and descriptions may yield a nescience of zero. The most suitable combination in any given context often depends on the intended practical application. Different applications may favor different aspects of a representation or description, such as interpretability, computational efficiency, or generalizability, leading to different but equally valid realizations of perfect knowledge.

If performed properly, the nescience of an entity should exhibit a strictly decreasing trend as new descriptions and representations are introduced. This principle relies on the notion that a newly proposed pair should only be accepted if it demonstrates a genuine improvement, namely, a reduction in overall nescience compared to the existing best-known alternative. It is possible that a new description may increase one component of nescience (miscoding, inaccuracy, or surfeit), but this increase must be compensated by a larger decrease in one of the others, so that total nescience does not rise. A simultaneous increase in all of them would clearly indicate regression.

Nevertheless, reality is more complicated. Our current measurements of miscoding, inaccuracy, and surfeit are merely estimations, approximations that are themselves subject to error and limited by our tools and understanding. Consequently, it is not always evident whether a particular refinement is truly superior. From a practical standpoint, we therefore relax the requirement of strict monotonic decrease and instead accept the weaker condition that nescience should decrease on average over time. Temporary setbacks or local increases are tolerable, as long as the general trajectory is toward a deeper and more refined understanding of the entity under study.

We can use this property of the reduction of nescience as a criterion to

distinguish between valid scientific disciplines and those that fall outside the scope of science, a challenge known as the *demarcation problem* in the philosophy of science. In scientific fields, successive refinements in descriptions and representations tend to yield a measurable decrease in nescience, signifying genuine progress in understanding. In contrast, non-scientific theories, including pseudosciences, typically do not exhibit this pattern. Despite the introduction of new descriptions or representations, there is no meaningful reduction in nescience over time. This implies that such disciplines fail to produce cumulative knowledge or deeper insights. In pseudoscientific domains, further research often leads to reinterpretations, embellishments, or rhetorical shifts, rather than the kind of substantive progress that characterizes scientific inquiry. As a result, these areas remain stagnant, unable to break new ground or approach perfect knowledge.

## 1.10 Unknown Unknown

We have previously discussed the existence of an *unknown unknown* area, comprising problems for which we not only lack solutions but whose very existence escapes our awareness. Within the framework of the theory of nescience, our goal is to develop a systematic procedure to identify and explore potential research entities hidden in this region. One approach could involve randomly generating binary strings and querying an oracle to assess whether any of them closely approximate the representation of a (hopefully unknown) entity. This conceptually embraces the idea of discovering new knowledge by pure chance. However, the sheer magnitude of the space of possible strings makes this brute-force strategy computationally impractical. Consequently, we must seek more efficient and guided methods for navigating this uncharted domain.

To discover what lies hidden in the unknown unknown, we must first delineate the area that encompasses everything already known. This known region comprises two types of topics. The first are the known knowns—topics that are well understood, where our descriptions are accurate, concise, and reliable. The second group includes the known unknowns, problems we are aware of but for which we still lack complete or satisfactory explanations. The boundary separating this region of known topics from what lies beyond is what we refer to as the *knowledge frontier*. It represents the outer limit of our current understanding, a conceptual demarcation where the known ends and the unexplored begins. Any entity that exists beyond this frontier—one that has not yet been identified or studied, constitutes a *new research entity*, residing in the domain of the unknown unknown.

Yet, identifying in practice the exact list of already known topics is

far from straightforward. The main challenge lies in determining which topics have already been studied and formally documented through published research. Scientific knowledge is dispersed across countless articles, journals, and disciplines, and there is no single, unified repository that captures the full extent of human understanding. Moreover, variations in terminology and differences in how topics are categorized further complicate the task. Thus, establishing the scope of the known demands careful analysis of bibliographic data, ontological classifications, and, often, expert consensus.

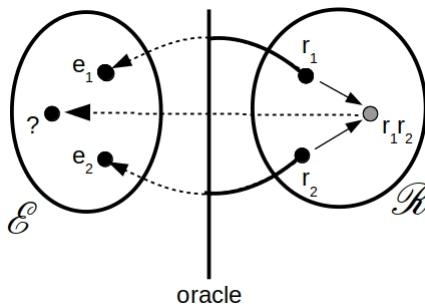


Figure 1.7: Discovering new research entities.

In this book, we explore an alternative strategy for identifying new research directions by combining concepts that are already known. The basic idea is straightforward: by taking two distinct representations, each corresponding to a different known entity, we identify a new entity by joining them into a single representation. To make this approach systematic, we assume that the space of valid representations is closed under such combinations—in other words, that combining any two representations will always yield another valid one. This assumption enables us to construct joint topics in a mechanical way and to explore their potential to uncover novel insights or previously unexamined questions. In practical terms, this involves computing all possible combinations of known entities and selecting those that appear to have the greatest potential to yield new and interesting research directions. However, the precise meaning and significance of any new entity formed in this way must still be determined through further investigation and reflection.

■ **Example 1.6** We could combine compelling topics from the field of theoretical computer science with those from phenomenology to identify promising new research directions. By merging the concepts of "minimum complexity computer programs" and "self-awareness," we arrive at a potential new research topic: "minimum complexity self-aware computers." This would involve investigating the minimum complexity required for a computer program to possess the capacity for self-awareness. ■

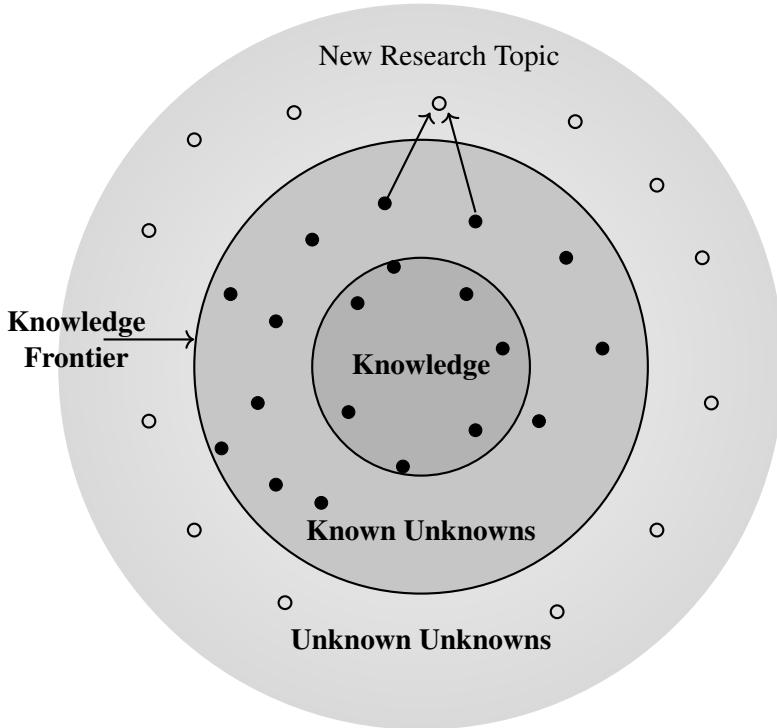


Figure 1.8: The structure of knowledge

As wisely stated by Saint John of the Cross, *to go where we do not know, we must go by a path we do not know*. This insight holds true in scientific discovery: the likelihood that a combination of two already known entities leads to a new entity located in the unknown unknown is greater when the entities being combined are themselves poorly understood. In contrast, combining well-understood entities is more likely to produce a result that remains within the bounds of current knowledge—that is, inside the knowledge frontier (see Figure 1.8). This is because the areas surrounding a well-known entity are often already thoroughly explored, leaving little room for the kind of novelty we are looking for, namely, ideas that lie beyond the current boundaries of knowledge and have the potential to open up entirely new lines of inquiry.

Another approach to increasing the chances of reaching the unknown unknown is by combining topics from two distinct fields of knowledge. The likelihood that such a combination has already been explored is relatively low, primarily because it would require someone with substantial expertise in both areas—a rarity in today’s academic landscape, where scientists are increasingly specialized in narrow domains. Interdisciplinary combinations,

therefore, offer a fertile ground for novel discoveries, as they may produce connections that have never been examined or even imagined within the confines of a single discipline.

## References

The theory of nescience builds upon several well-established foundations across information theory, computability, algorithmic complexity, and the philosophy of science. The following references provide the theoretical background and conceptual tools necessary to understand and contextualize the ideas introduced in this chapter.

Sipser's book [Sip12] is a widely respected introduction to formal languages, automata, and computability theory. It lays the groundwork for understanding which descriptions are computationally feasible, an essential aspect of the theory of nescience, which assumes that knowledge must be computable to be meaningful.

[LV13] is a comprehensive volume that presents the theory of Kolmogorov complexity, which formalizes the idea of description length using the shortest program capable of generating a given object. The concepts developed by Li and Vitányi are central to the theory of nescience, especially in defining metrics such as inaccuracy, miscoding, and surfeit based on algorithmic information.

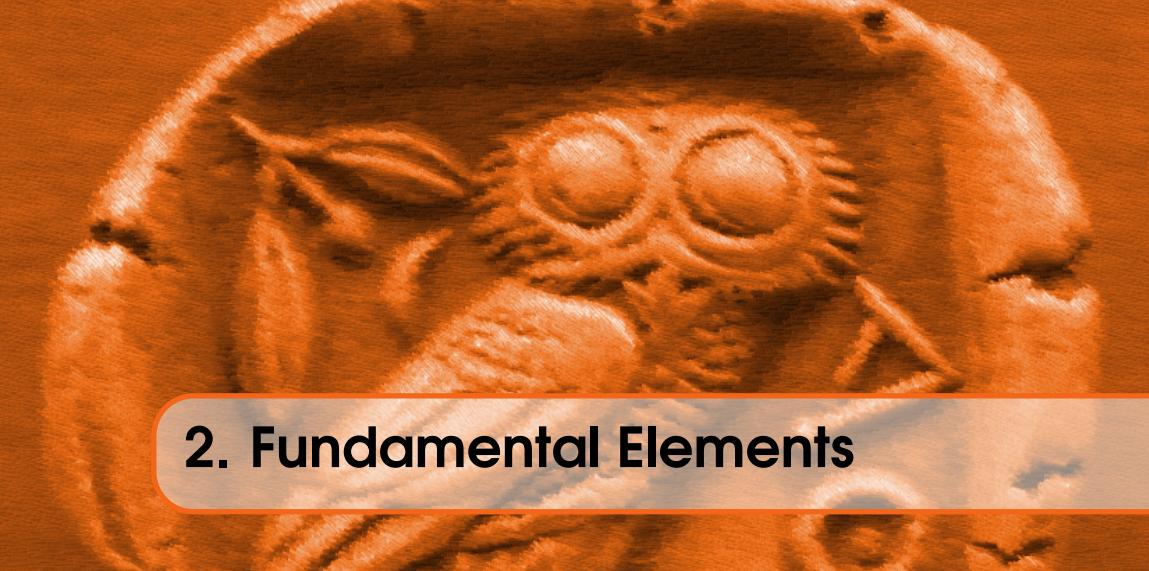
Chalmers' book [Cha13] offers a critical and historical introduction to the philosophy of science. It examines the assumptions, limits, and methodologies of scientific practice. This perspective is invaluable for situating the theory of nescience within the broader discourse on how scientific knowledge is constructed, evaluated, and refined over time.

[CT12] is a foundational text in information theory, providing the mathematical framework for understanding concepts such as entropy, mutual information, and data compression. It serves as a rigorous yet accessible introduction to how information can be quantified, transmitted, and encoded, ideas that relate the notions of description length in the theory of nescience.

|     |  |
|-----|--|
| 2.4 | Joint Representations                  |
| 2.5 | Descriptions                           |
| 2.6 | Descriptions for Joint Representations |
| 2.7 | Conditional Descriptions               |
| 2.8 | Research Areas                         |
| 2.9 | References                             |

|          |                                     |            |
|----------|-------------------------------------|------------|
| <b>3</b> | <b>Miscoding .....</b>              | <b>79</b>  |
| 3.1      | Miscoding                           |            |
| 3.2      | Joint Miscoding                     |            |
| 3.3      | Decreasing Miscoding                |            |
| 3.4      | Targetless Representations          |            |
| 3.5      | Miscoding of Areas                  |            |
| <b>4</b> | <b>Inaccuracy .....</b>             | <b>93</b>  |
| 4.1      | Inaccuracy                          |            |
| 4.2      | Conditional Inaccuracy              |            |
| 4.3      | Decreasing Inaccuracy               |            |
| 4.4      | Inaccuracy-Miscoding Rate of Change |            |
| 4.5      | Inaccuracy of Areas                 |            |
| <b>5</b> | <b>Surfeit .....</b>                | <b>107</b> |
| 5.1      | Surfeit                             |            |
| 5.2      | Redundancy                          |            |
| 5.3      | Conditional Surfeit                 |            |
| 5.4      | Decreasing Surfeit                  |            |
| 5.5      | Surfeit-inaccuracy rate of Change   |            |
| 5.6      | Surfeit of Areas                    |            |
| <b>6</b> | <b>Nescience .....</b>              | <b>119</b> |
| 6.1      | Nescience                           |            |
| 6.2      | Minimizing Nescience                |            |
| 6.3      | Joint Nescience                     |            |
| 6.4      | Conditional Nescience               |            |
| 6.5      | Nescience of Areas                  |            |
| 6.6      | Perfect Knowledge                   |            |
| 6.7      | Current Best Description            |            |
| 6.8      | Nescience based on Datasets         |            |
| 6.9      | Unknonwn Unknown                    |            |
| 6.10     | Science vs. Pseudoscience           |            |
| <b>7</b> | <b>Interesting Questions .....</b>  | <b>137</b> |
| 7.1      | Other Metrics                       |            |
| 7.2      | Interesting Research Questions      |            |
| 7.3      | Relevance                           |            |
| 7.4      | Applicability                       |            |
| 7.5      | Interesting Questions               |            |
| 7.6      | New Research Topics                 |            |
| 7.7      | Classification of Research Areas    |            |





## 2. Fundamental Elements

*We are all agreed that your theory is crazy.  
The question which divides us is whether it is crazy enough.*  
Niels Bohr

The first step in quantifying our lack of knowledge involves the precise identification of the research entities of interest. The elements of this collection depend largely on the specific application of the theory of nescience. Each application requires a distinct set of entities, whether mathematical objects, living organisms, human needs, or otherwise. Fortunately, the process of quantifying what we do not know remains consistent across all such domains.

The next step is to devise a procedure for representing the identified entities as strings of symbols. Accurately encoding a research entity is a complex and unresolved epistemological challenge. The solution proposed within the theory of nescience is based on the concept of oracle Turing machine. The practical feasibility of this solution depends largely on the abstractness of the entities being studied. For example, encoding abstract mathematical objects poses significant difficulties and often requires an approximation. In contrast, encoding computer programs is relatively straightforward, as they are already expressed as strings.

Once a suitable method for encoding the original entities into string-based representations has been established, the final step is to produce a description. This description should be both accurate and concise, reflecting our current understanding of the representation. In the theory of nescience, descriptions are required to be computable, meaning that a computer must be able to fully reconstruct the original representation from its description. However, descriptions refer to representations, that is, the way entities are encoded, not to the entities themselves. As a result, how well a description captures knowledge about an entity depends largely on the quality of the representation used.

In this chapter, we will formalize these core concepts: entities, representations, and descriptions, among others. We will also explore what constitutes a perfect description, how to compute the combined representation of multiple entities, and how to describe a representation assuming some prior background knowledge. Additionally, we will examine the concept of a research area and its associated properties.

## 2.1 Entities

Defining the nature of a research entity is a complex and unresolved philosophical problem. We approach this complexity from a fundamentally pragmatic perspective, rather than a philosophical-ontological one. Our theory starts from the premise that there exists a non-empty set of *entities* that we seek to understand.

**Notation 2.1.** *We represent the set of research entities of interest as  $\mathcal{E}$ .*

The contents of  $\mathcal{E}$  depend on the specific domain in which theory of nescience is employed, and are usually aligned with a specific area of knowledge. Examples of entity sets could include: research components in mathematics (abstract); the kingdom Animalia (living entities); known and unknown human needs (abstract); all potential computer programs (strings), etc. From a formal point of view,  $\mathcal{E}$  is not a well-defined set, since there is usually no rule or procedure for deciding which elements comprise this set.

The abstract nature of  $\mathcal{E}$  offers certain advantages, but also imposes significant restrictions. The main restriction is that our definition of nescience is, in most cases, a non-computable quantity, which requires the use of approximations in practice. The main advantage is that the new concepts and methods presented in this book can be applied to a wide range of problems, not limited only to the search for new scientific knowledge.

In the theory of nescience, the possibility of using universal sets is excluded; that is, the existence of a set  $\xi$  containing everything cannot be

assumed. The problem with universal sets is that they contravene Cantor's theorem (see the example 2.1). Cantor's theorem proves that the power set  $\mathcal{P}(\xi)$ , consisting of all possible subsets of  $\xi$ , has more elements than the original set  $\xi$ . This contradicts the assumption that  $\xi$  includes everything. In the theory of nescience, the set  $\mathcal{E}$  must be a specific set.

■ **Example 2.1 — Cantor's theorem.** Cantor's theorem proves that for any set  $A$ ,  $d(A) < d(\mathcal{P}(A))$ . Consider  $f : A \rightarrow \mathcal{P}(A)$ , a function that maps each element  $x \in A$  to the set  $\{x\} \in \mathcal{P}(A)$ ; evidently,  $f$  is injective, implying  $d(A) \leq d(\mathcal{P}(A))$ . To substantiate that the inequality is strict, let's assume there exists a surjective function  $g : A \rightarrow \mathcal{P}(A)$  and consider the set  $B = \{x \in A : x \notin g(x)\}$ . As  $g$  is surjective, there must exist a  $y \in A$  such that  $g(y) = B$ . This, however, raises a contradiction,  $y \in B \Leftrightarrow y \notin g(y) = B$ . Consequently, the function  $g$  cannot exist, therefore  $d(A) < d(\mathcal{P}(A))$ . ■

In the theory of nescience, not all conceivable sets are acceptable, as some may give rise to paradoxes. Take, for example, Russell's paradox, which proposes a set  $R$  consisting of all sets that are not members of themselves. The paradox arises when we try to discern whether  $R$  is a member of itself (see the example 2.2). To avoid such problems, the theory of nescience is based on the Zermelo-Fraenkel axiom set, along with the Axiom of Choice. The *axiom of separation* (if  $P$  is a property with parameter  $p$ , then for any set  $x$  and parameter  $p$  there exists a set  $y = \{u \in x : P(u)\}$  that includes all those sets  $u \in x$  that have property  $P$ ) allows the use of this notation only to construct sets that are subsets of already existing sets. A more extensive *axiom of comprehension* (if  $P$  is a property, then there exists a set  $y = \{u : P(u)\}$ ) would be required to allow sets like the one proposed by Russell's paradox. Russell's paradox arises from the use of an unrestricted comprehension principle. In the axioms of ZFC, and in the theory of nescience, the axiom of comprehension is considered false.

■ **Example 2.2 — Russell's Paradox.** Suppose  $R$  is the set of all sets not members of themselves, such that  $R = \{x : x \notin x\}$ . The contradiction arises when querying if  $R$  is a member of itself. If  $R$  is not a member of itself, by its own definition, it should be; conversely, if  $R$  is declared to be a member of itself, its definition dictates it should not be. Symbolically, this can be written as  $R \in R \Leftrightarrow R \notin R$ . ■

In the theory of nescience, we do not address the classic problems of ontology, that is, the classification of entities that exist in the world and can be known. Furthermore, we do not attempt to resolve epistemological questions, such as how scientific knowledge is validated by evidence, or what the nature of that evidence is.

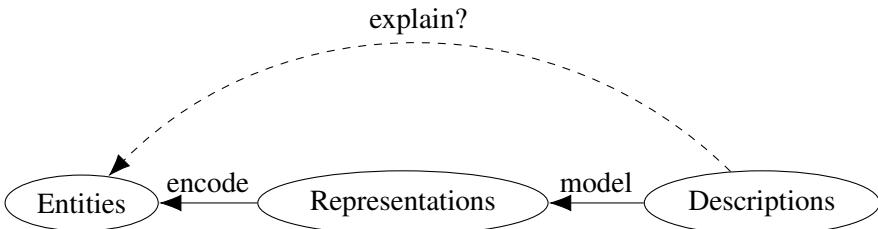


Figure 2.1: The Problem of Understanding

Once a set  $\mathcal{E}$  of entities has been selected, the next step is to uniquely encode them as strings of symbols, which will make them easier to describe. A method for doing this encoding efficiently is described in the following section.

## 2.2 Representations

The representation of the entities that compose the set  $\mathcal{E}$ , which can be abstract in nature, poses important epistemological challenges that have been the subject of research for more than two millennia (see Section 17.4 for a brief overview of the solutions proposed by the scientific community and their limitations). This book describes a possible solution to this complex problem, which has certain advantages, but also disadvantages. Our approach proposes dividing the problem of scientific representation into two interconnected subproblems: the encoding of entities by means of representations, and the modeling of these representations by means of descriptions. Thus, scientific models would explain the entities indirectly through their representations (see Figure 1.1 in the Introduction of the book, reproduced in this section for ease of reference). In this section we will focus on the aspects related to encoding, and in Section 2.5 on the descriptions.

Within the field of Kolmogorov complexity, the representation issue is tackled by presupposing that the set  $\mathcal{E}$  is well-defined, countable, and that a total encoding function  $f : \mathcal{E} \rightarrow \mathbb{N}$  exists, mapping the set of entities to the set of natural numbers (akin to Gödel numbering). The nescience theory similarly embraces this concept of encoding entities via numbers, or, in other words, strings of symbols. However, our approach to the problem of encoding an arbitrary set  $\mathcal{E}$  involves an inversion of the problem. This involves defining a partial function  $\mathcal{O}_{\mathcal{E}} : \mathcal{S}^* \rightarrow \mathcal{E}$  that maps the well-defined set  $\mathcal{S}^*$  (comprising all possible finite strings from an alphabet  $\mathcal{S}$ ) to the potentially non-countable and non-well defined set  $\mathcal{E}$  containing the entities of interest.

Our function  $\mathcal{O}_{\mathcal{E}}$ , which depends on the set  $\mathcal{E}$ , operates much like an

oracle (taking inspiration from the concept of an oracle Turing machine as per Definition 13.5.1). This function has the capacity to entirely reconstruct an entity  $\mathcal{O}_{\mathcal{E}}(x) \in \mathcal{E}$  using its encoding string  $x \in \mathcal{S}^*$ , without the need for additional information. Evidently, this oracle is an abstract concept, and its construction in the real world would be impossible for most sets  $\mathcal{E}$ . However, the theoretical existence of this oracle assists in the elucidation and verification of key attributes of the scientific discovery process. Due to its oracle nature,  $\mathcal{O}_{\mathcal{E}}$  has limitations with respect to the mathematical operations in which it can be used, as will be explained in the subsequent paragraphs.

Our premise is based on the assumption that there exists a single, objective physical world that is independent of observers. In addition, we assume that, for certain collections of entities  $\mathcal{E}$ , it is logically coherent to postulate the existence of an oracle  $\mathcal{O}_{\mathcal{E}}$  (see the discussion of the scientific representation problem in Section 17.4).

For the purposes of this book, without any loss of generality, we will exclusively regard binary strings as the means of encoding entities.

**Definition 2.2.1** Given  $\mathcal{E}$  as a set of entities, a *representation function* is defined as an oracle partial function  $\mathcal{O}_{\mathcal{E}} : \mathcal{B}^* \rightarrow \mathcal{E}$  that maps elements of the set  $\mathcal{B}^*$ , which comprises all conceivable finite binary strings, to the elements of the set  $\mathcal{E}$ , corresponding to the entities under examination.

Within this framework, only the elements of the set  $\mathcal{B}^*$ , that is, binary strings, are eligible to serve as representations of entities (see problem of ontology in Section 17.4). We do not permit other forms of physical models, including drawings, unless they can be transcribed into binary strings. We do not distinguish between scientific representations and other forms of representations (see representational demarcation in Section 17.4). The oracle assumes the responsibility of determining which entity is encoded by each representation. It is also pivotal to note that the oracle  $\mathcal{O}_{\mathcal{E}}$  is a partial function, meaning not all potential strings represent entities (targetless models are allowed in the theory of nescience, see Section 3.4 and Section 17.4).

■ **Example 2.3** For a given set of entities  $\mathcal{E}$ , the existence of a representation oracle  $\mathcal{O}_{\mathcal{E}}$  does not imply its uniqueness. For instance, a binary negation oracle (which transforms the zeros of a binary string into ones, and the ones into zeros), assigning to each string  $x \in \mathcal{B}^*$  the entity  $\mathcal{O}_{\mathcal{E}}(\neg x)$ , would also qualify as a representation function. ■

Rather than deploying individual oracles, we could have employed a universal oracle machine. This is a machine  $\mathcal{U}_{\mathcal{O}}$  which, given the encoding of an oracle  $\mathcal{O}_{\mathcal{E}}$  and a string  $s$  as input, computes  $\mathcal{U}_{\mathcal{O}}(\langle \mathcal{O}_{\mathcal{E}}, s \rangle) = \mathcal{O}_{\mathcal{E}}(s)$ .

Universal machines are particularly applicable to the universal set  $\xi$ , encompassing all entities. However, such an approach would complicate the process of scientific discovery in practical terms. Instead, we choose to work with entity sets  $\mathcal{E}$  corresponding to different areas of knowledge, choosing a single oracle  $\mathcal{O}_{\mathcal{E}}$  for each set  $\mathcal{E}$  (the most suitable one according to our current knowledge and our practical needs).

■ **Example 2.4** Consider the case when the subjects of study are animals. Initially, one might use detailed physical descriptions of the animals as encodings. In this scenario, the oracle would be a hypothetical machine capable of reconstructing the original animal from its description. As our understanding of biology advances, we might instead adopt an alternative encoding based on the animals' DNA. Both of these encodings serve as valid representations of the entities. ■

As illustrated in Example 2.3 and Example 2.4, the entities in  $\mathcal{E}$  can be encoded in multiple ways (see the problem of style in Section 17.4). Different oracles accommodate different encoding schemes. Some strings provide more accurate representations of an entity than others (see the standard of accuracy in Section 17.4). The optimal representation depends on the type of questions we aim to answer. Using a particular encoding style with an oracle designed for a different style may lead to unexpected outcomes. Our objective should be to employ representations that minimize the size of the oracle—that is, the amount of prior knowledge presumed to be embedded within it.

■ **Example 2.5** Consider  $\mathbf{X}_t$  as a time series consisting of  $m$  measurements  $x_1, \dots, x_m$  collected at fixed intervals from a physical phenomenon exhibiting sinusoidal behavior. Suppose we have trained a multi-layer perceptron neural network  $nn$  that perfectly fits the data, meaning it returns  $x_t$  when given a time  $t$  as input. By analyzing the cycles in the time series, one would naturally identify a sine function as the most appropriate model for the underlying physical process. However, no existing machine learning methodology would arrive at this conclusion if provided solely with the neural network's architecture (i.e., the number of layers, their sizes, and the trained weights). Clearly, the oracle is sufficiently powerful to recognize this underlying structure. ■

Remember that the oracle operates as a partial function, meaning that not every string must represent an entity. The less information a representation contains, the more difficult it becomes to understand how the oracle works. It is essential to be cautious with representations that fail to capture the full structure of the original entities, as this can introduce bias into our analyses.

Chapter 3 offers a detailed discussion of the types of errors arising from the miscoding of abstract entities, emphasizing that representations must not only correspond to entities, but also that entities must be relevant to their representations (see the requirement of directionality in Section 17.4).

We employ an oracle function rather than an oracle relation  $\mathcal{R}_{\mathcal{O}} \subset \mathcal{B}^* \times \mathcal{E}$ , not to merely associate strings with their corresponding entities, but to reconstruct an entity from its representation. The oracle's ability to recover original entities underpins our capacity to make hypotheses about entities based on their representations (see surrogate reasoning in Section 17.4). According to the theory of nescience, scientific inquiry involves not only learning how to encode entities properly but also understanding the mechanisms by which oracles decode them. For scientific inquiry to be practically effective, these oracles should ideally be minimal in size.

The purpose of encoding entities in the theory of nescience differs fundamentally from that in Shannon's information theory (see Chapter 14), as illustrated in Example 2.6.

■ **Example 2.6** Take, for instance, a set  $\mathcal{E}$  consisting of two books: "The Ingenious Nobleman Sir Quixote of La Mancha" and "The Tragedy of Romeo and Juliet". We might encode the first book with the string "0" and the second with the string "1". While these strings allow us to uniquely identify each book within the set, they do not qualify as valid encodings within the framework of the theory of nescience. In information theory, the goal is to uniquely identify an object based on a reference, assuming mutual agreement between the sender and the receiver about the mapping from references to objects. In contrast, the theory of nescience seeks representations that preserve the richness and detail of the original entities. For example, it would be impossible to hypothesize about Cervantes' influence on Shakespeare using only the strings "0" and "1". ■

One possible response to the limitation discussed in Example 2.6, where entities are encoded using overly simplistic strings that fail to capture their internal structure, would be to require the set  $\mathcal{E}$  to be infinite, as is done in Kolmogorov complexity. This requirement avoids pathological cases in which too many entities are assigned trivial or arbitrary encodings. However, merely increasing the size of  $\mathcal{E}$  does not solve the fundamental issue: even with an infinite set of entities, many encoding schemes still fall short of supporting surrogate reasoning. That is, they do not allow us to make meaningful inferences about the original entities based solely on their representations. To enable such reasoning, the encoding must preserve essential structural and semantic features of the entities, not just their identity.

We distinguish between *knowable* and *unknowable* entities. Knowable

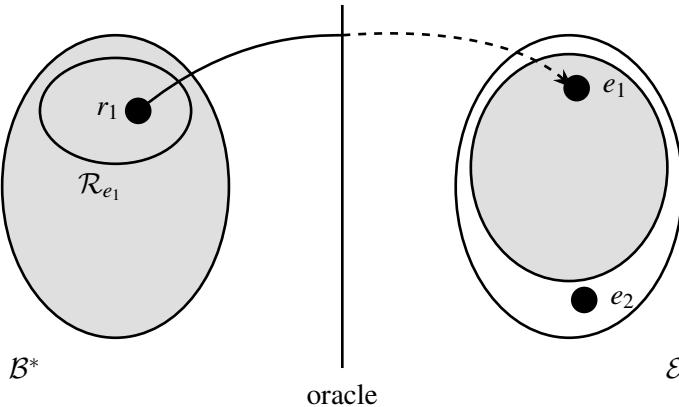


Figure 2.2: Encodings and Entities

entities are those that can, in principle, be understood through scientific inquiry (see Section 17.2). Unknowable entities, on the other hand, lie beyond the reach of human comprehension—either because they are inherently inaccessible to observation and reasoning, or because they exceed the cognitive or methodological limits of science.

**Definition 2.2.2** We say that an entity  $e \in \mathcal{E}$  is *knowable* if there exists at least one  $r \in \mathcal{B}^*$  such that  $\mathcal{O}_{\mathcal{E}}(r) = e$ . An entity  $e \in \mathcal{E}$  is *unknowable* if it is not knowable.

A priori, it is not possible to determine whether an entity  $e \in \mathcal{E}$  is knowable, unknowable, or only partially knowable. Identifying suitable knowable entities for study is a matter of trial and error. In this book, we say nothing further about unknowable entities, except to note that their unknowability can neither be proven nor discovered.

**Definition 2.2.3** Let  $e \in \mathcal{E}$  be a knowable entity. We define the *set of representations* for  $e$ , denoted by  $\mathcal{R}_e$ , as  $\{r \in \mathcal{B}^* : \mathcal{O}_{\mathcal{E}}(r) = e\}$ .

In Figure 2.2, we present a depiction of a hypothetical oracle mapping from the set of finite binary strings  $\mathcal{B}^*$  to the set of entities  $\mathcal{E}$ . The figure also highlights a particular entity  $e_1$ , the subset of strings  $\mathcal{R}_{e_1}$  that represent this entity, and a specific representation  $r_1$ . Since the set  $\mathcal{E}$  is, in general, not well defined, the inverse function  $\mathcal{O}_{\mathcal{E}}^{-1}(e_1)$  cannot be computed in practice.

A consequence of working with finite strings as representations is that some entities may not be encoded by any representation (see the gray areas in Figure 2.2; in particular, the entity  $e_2$  is not encoded by any string). Intuitively, this reflects the fact that, in certain domains of knowledge, the

number of possible problems far exceeds the number of available solutions.

■ **Example 2.7** If the collection of entities under study consists of real numbers, then there exist numbers that cannot be encoded using finite binary strings. This is because the set  $\mathbb{R}$  has the cardinality of the continuum, whereas the set  $\mathcal{B}^*$  is countable. ■

Since our knowledge of the entities and the internal workings of the oracle is incomplete, in practice we work with a different set of strings, denoted by  $\hat{\mathcal{R}}_e$ , which we believe are close approximations of the true representations  $\mathcal{R}_e$  that encode the entity  $e$ . The elements of this set typically change over time, as our understanding of the entities in  $\mathcal{E}$  and of how the oracle encodes them improves. The more abstract the set of entities is, the more difficult it becomes to approximate them as strings (see Chapter 3).

■ **Example 2.8** The entity "luminiferous ether" was a theoretical postulate describing a hypothetical medium through which light was believed to propagate. It was proposed to explain how wave-based light could travel through empty space. However, in 1887, the Michelson-Morley experiment provided strong evidence against the existence of ether. Later, Einstein's special theory of relativity offered a successful explanation for the propagation of light in a vacuum, leading to the complete abandonment of the ether concept. ■

If the chosen oracle is not minimal, determining whether a string is a valid representation of an entity requires understanding the internal workings of the oracle. A \*non-minimal oracle\* is one in which part of the information needed to reconstruct the original entity is embedded within the oracle itself, rather than being explicitly encoded in the representation string. Although using non-minimal oracles can simplify the research process by offloading complexity to the oracle, it obscures the boundary between what is encoded and what is assumed, making it harder to analyze or generalize the representation mechanism.



One of the fundamental challenges in science—and in human intellectual activity more broadly—is the tendency to confuse symbols with the things they represent. The theory of nescience has been carefully designed to avoid this issue by clearly distinguishing between research entities and their representations. However, making this distinction explicit at every step would render the book unnecessarily difficult to read. We have aimed to strike a balance between clarity of exposition and rigor of definition. Occasionally, especially when introducing new ideas, we use the term *topic* to refer broadly to an entity, a representation, or both. Nevertheless, in all mathematical definitions and propositions, this distinction is made unambiguous. In case of any uncertainty, the formal definitions should be taken as the definitive reference.

## 2.3 Invalid Representations

It may happen that the representation we are using for an entity  $e \in \mathcal{E}$  is incorrect. That is, instead of working with a string  $r \in \mathcal{B}^*$  that perfectly encodes  $e$ , we are studying another string  $r' \in \mathcal{B}^*$  which is, hopefully, close to  $r$  but not necessarily identical. We are interested in computing the distance between the string  $r'$  and the correct representation  $r$  as a quantitative measure of the error introduced by using an incorrect encoding (see Chapter 3). Unfortunately, we do not know  $r$ , since in most practical applications there does not exist a computable function from  $\mathcal{E}$  to  $\mathcal{B}^*$ . The only tool at our disposal is an abstract oracle machine  $\mathcal{O}_{\mathcal{E}}$  that knows which strings represent each entity. Recall from Section 2.2 that we propose to address the scientific representation problem by introducing an abstract oracle  $\mathcal{O}_{\mathcal{E}} : \mathcal{B}^* \rightarrow \mathcal{E}$ , which maps the set  $\mathcal{B}^*$  of all finite binary strings to the set  $\mathcal{E}$  of entities under study.

We begin by distinguishing between valid representations, strings that contain all the information required by the selected oracle to reconstruct an entity, and non-valid representations.

**Definition 2.3.1** Let  $\mathcal{E}$  be a collection of entities, and let  $\mathcal{O}_{\mathcal{E}}$  be a decoding function. We define the set of *valid* representations for  $\mathcal{E}$  with respect to the encoding function  $\mathcal{O}_{\mathcal{E}}$ , denoted by  $\mathcal{R}_{\mathcal{O}_{\mathcal{E}}}^*$ , as the subset of representations that perfectly encode an entity in  $\mathcal{E}$ , according to  $\mathcal{O}_{\mathcal{E}}$ .

The set  $\mathcal{R}_{\mathcal{O}_{\mathcal{E}}}^*$  is generally unknown, as it is a subset of  $\mathcal{B}^*$  whose definition depends on an abstract and not well-defined oracle. Intuitively, a representation is considered valid if it contains all the necessary information for the oracle to reconstruct the original entity without relying on any external sources. A valid representation must not include incorrect symbols, nor omit any relevant information, otherwise, the oracle would be unable to reconstruct the entity accurately.

Furthermore, we require that the oracle can not only reconstruct an entity from a representation, but also derive the set of valid representations for a given entity. From this perspective, a representation that includes non-relevant information cannot be considered valid, since that extraneous part cannot be generated by the oracle solely from the entity.

■ **Example 2.9** The astronomical data used during the time of Ptolemy to record the positions of celestial bodies throughout the year constituted a non-valid encoding of the entity "position of celestial bodies," due to its inaccuracy. A better encoding was later provided by the more precise observations of Tycho Brahe. Today, we possess even more accurate encodings. Adding the name of the person who collected the data to the dataset would be an example of a representation that includes non-relevant information. ■

Recall that, in our framework, we only allow representations of entities in the form of finite binary strings. Drawings, mathematical formulas, or datasets may serve as representations, provided they can be encoded as binary strings and are recognized as valid by the selected oracle. Each string represents one, and only one, entity. Naturally, different oracles may define different sets of valid representations. Throughout this work, we assume that a specific representation oracle  $\mathcal{O}_{\mathcal{E}}$  has been chosen to encode the set  $\mathcal{E}$  of entities under study.

**Notation 2.2.** We will denote the set of valid representations of  $\mathcal{E}$  by  $\mathcal{R}_{\mathcal{E}}^*$  when there is no ambiguity regarding the oracle  $\mathcal{O}_{\mathcal{E}}$  in use.

We can similarly define the set of valid representations for an individual entity  $e$ .

**Definition 2.3.2** We define the set of valid representations for an entity  $e \in \mathcal{E}$ , denoted by  $\mathcal{R}_e^*$ , as the subset of representations that perfectly encode the entity  $e$ ; that is,  $\mathcal{R}_e^* = \mathcal{R}_{\mathcal{E}}^* \cap \mathcal{R}_e$ .

As we have seen in Example 2.4 there could be more than one valid representation for some entities, even if the oracle is restricted to a particular style. It might also happen that the set of valid representations is empty, that is, there is no valid representation for that entity. In that case the entity will be unknowable.

**Notation 2.3.** We denote by  $r_e^* \in \mathcal{R}_{\mathcal{E}}^*$  the fact that  $r$  is a valid representation of the entity  $e$ .

A consequence of working with approximations instead of true representations is that some of the candidate strings currently in use may encode a different entity than the one we intended to study.

■ **Example 2.10** In 1961, Soviet physicist Nikolai Fedyakin conducted a series of experiments that appeared to reveal a new form of water. This substance, named polywater, exhibited unusual properties: a higher boiling point, a lower freezing point, and much greater viscosity than ordinary water. However, later experiments demonstrated that polywater was simply ordinary water contaminated with small amounts of impurities. ■

We can safely assume (without logical contradiction) that the oracle machine not only knows which representations correspond to which entities, but also how far any string  $r \in \mathcal{B}^*$  is from being a valid representation  $r_e^* \in \mathcal{R}_{\mathcal{E}}^*$ .

If a representation  $r$  is non-valid, the oracle will identify the closest valid representation and assume they both refer to the same entity.

The oracle defines an equivalence relation that partitions the set  $\mathcal{B}^*$  into equivalence classes. Each class corresponds to an entity in  $\mathcal{E}$ , although it may happen that some entities in  $\mathcal{E}$  do not have any associated class, these are what we call the unknowable unknowns. Every string is associated with some entity, but some representations are better than others (i.e., they yield lower nescience). No string can belong to more than one entity. Moreover, for any given collection  $\mathcal{E}$ , multiple valid oracles may exist.

**Definition 2.3.3** Let  $\mathcal{B}^*/\mathcal{O}$  be the quotient set defined by the oracle's equivalence relation over the set of binary strings. We call each class in this quotient set an *entity class*, denoted by  $[e]$ , that is,  $[e] \in \mathcal{B}^*/\mathcal{O}$ .

## 2.4 Joint Representations

We saw in the previous section that there is more than one way to encode an entity  $e \in \mathcal{E}$ , this collection of encodings is what we called the set  $\mathcal{R}_e$  of representations of the entity. Some of these representations are of high quality, in the sense that they contain all the information required by the oracle to reconstruct (by whatever means the oracle employs) the original entity. However, the set  $\mathcal{R}_e$  also includes low-quality representations, that is, representations that lack many of the details necessary to fully reconstruct the entity. Moreover,  $\mathcal{R}_e$  may contain representations that include incorrect information: symbols that the oracle will automatically disregard during reconstruction but that may mislead us when trying to understand the entity.

If we want to increase our knowledge about an entity, we must identify the best possible representation for that entity, that is, one that is both complete and correct. One way to achieve this is to try different strings until we discover a high-quality representation. However, this method can be extremely time-consuming and impractical. A more efficient approach is to enhance a poor representation by adding missing symbols or by combining known representations that each contain partial information. Both approaches require introducing the concept of a joint representation.

**Definition 2.4.1** Let  $s, t \in \mathcal{R}_{\mathcal{E}}$  be two different representations. The *joint representation* of  $s$  and  $t$  is defined as the concatenated string  $st$ .

**Example 2.11** Suppose the research entity  $e$  of interest is the set of causes of lung cancer. To study this entity, we have measured a collection of risk factors in a random sample of the population (smoking, exercise, diet, age, etc.). However, due to a flaw in the sampling procedure, all the samples correspond to a specific subgroup of the population, for instance, males. This dataset constitutes a representation  $s$  of our entity  $e$ , but a poor one, as it is strongly biased. If we have a second representation  $t$ , corresponding to

sample data from females, the joint representation  $st$  will be superior to either  $s$  or  $t$  considered in isolation. ■

It would be desirable for the set of representations  $\mathcal{R}_\mathcal{E}$  to be closed under the operation of concatenation, meaning that if  $s$  and  $t$  are representations of an entity  $e$ , then their concatenation  $st$  would also be a representation of some entity (not necessarily the same one). Unfortunately, this is not the case, since the oracle function is partial; it does not assign an entity to every possible binary string. Moreover, we do not even require the set of representations  $\mathcal{R}_e$  for a given entity  $e$  to be closed under concatenation. It is entirely possible that two strings  $s \in \mathcal{R}_e$  and  $t \in \mathcal{R}_e$  produce a concatenated string  $st$  that does not belong to  $\mathcal{R}_e$ .

The concept of joint representation is defined for any pair of strings  $s, t \in \mathcal{R}_\mathcal{E}$ , even if they do not belong to the same set of representations  $\mathcal{R}_e$ , that is, even when  $\mathcal{O}_\mathcal{E}(s) \neq \mathcal{O}_\mathcal{E}(t)$ . This ability to combine representations of different entities will prove particularly useful for the discovery of new research entities that are currently unknown (see Section 7.6).

The operation of string concatenation is associative; that is, for all  $r, s, t \in \mathcal{R}_\mathcal{E}$ , we have  $(rs)t = r(st)$ . This property underlies the algebraic structure of the set of representations.

**Proposition 2.4.1** The set  $\mathcal{R}_\mathcal{E}$  of representations, together with the operation of concatenation, has the structure of a free monoid.

*Proof.* As shown in Section 11.2, the operation of string concatenation on  $\mathcal{R}_\mathcal{E}$  is associative, and the empty string  $\lambda$  serves as the identity element. ■

We do not require the operation of joining representations to be commutative with respect to the oracle function. Given representations  $s, t \in \mathcal{R}_\mathcal{E}$ , it may happen that the concatenated strings  $st$  and  $ts$  represent different entities—that is,  $\mathcal{O}_\mathcal{E}(st) \neq \mathcal{O}_\mathcal{E}(ts)$ .

The concept of joint representation can be extended to any arbitrary, but finite, collection of representations. This allows us to incorporate multiple partial representations into our research or to use them in the process of discovering new entities.

**Definition 2.4.2** Let  $r_1, r_2, \dots, r_n \in \mathcal{R}_\mathcal{E}$  be a finite collection of representations. The *joint representation* of  $r_1, r_2, \dots, r_n$  is defined as the concatenated string  $r_1r_2\dots r_n$ .

Unfortunately, the set  $\mathcal{R}_\mathcal{E}$  is not closed under the operation of concatenation of multiple, finite, representations.

We have seen that the concatenation of two or more representations of an entity can result in a string that encodes a different entity from the

original. Remarkably, this phenomenon can occur even when the individual representations being concatenated all correspond to the same entity. This non-trivial behavior opens the door to a novel strategy for scientific discovery: the generation of new, previously unknown entities through the systematic combination of representations of known entities.

■ **Example 2.12** Consider a research scenario in which the entities under study are chemical compounds, and the representations are strings encoding molecular descriptors. Suppose  $s$  encodes a compound known for its anti-inflammatory properties, and  $t$  encodes a structurally similar compound with anti-viral properties. While both  $s$  and  $t$  individually map to well-known compounds via the oracle, the concatenation  $st$  may correspond to a novel hybrid molecule whose biological activity has not yet been characterized. In this case,  $st$  represents a new research entity, possibly a candidate for drug discovery—that was not explicitly part of the original knowledge base. ■

## 2.5 Descriptions

So far, our goal in working with strings from  $\mathcal{B}^*$  has been to construct encodings, or representations, that are as complete and detailed as possible for the entities in  $\mathcal{E}$ , regardless of their length. However, as stated in the preface of this book, human understanding requires the formulation of concise models of these entities, since human reasoning cannot operate effectively on lengthy representations.

■ **Example 2.13** In Example 2.11, we showed that a good representation of the entity "lung cancer" could be a dataset in which various risk factors are measured. However, smokers do not decide to quit smoking because they have studied and understood this extensive dataset. Rather, they do so because they understand the much simpler derived model: "smoking increases the risk of lung cancer." ■

A description or model<sup>1</sup> is a finite binary string that is mapped to a representation of an entity (see Figure 1.3 in Chapter 1). Importantly, descriptions do not directly model the entities themselves (i.e., the target systems); instead, they operate on representations of those entities (encodings in the form of strings) serving as approximations of the original entities through those representations.

In the theory of nescience, we require that descriptions be computable, so that the original representations can be fully and effectively reconstructed from them. This requirement of computability allows us to clearly define the limits of the concept of a "description." For example, paradoxes involving

---

<sup>1</sup>In the theory of nescience, the terms "description" and "model" are used interchangeably.

self-reference, such as the Berry paradox (i.e., "the smallest positive integer not definable in less than twelve words," see Chapter 1), can be addressed within the framework of computability.

**Definition 2.5.1 — Model.** Let  $d \in \mathcal{B}^*$  be a binary string of the form  $d = \langle TM, a \rangle$ , where  $TM$  is the encoding of a prefix-free Turing machine and  $a$  is the input string to that machine. If  $TM(a)$  is defined, then  $d$  is called a *description*.

Intuitively, a description consists of two parts: a Turing machine that captures and compresses the regularities present in the representation, and a string that contains what remains, that is, the incompressible or random part.

**Definition 2.5.2** We define the *set of descriptions*, denoted by  $\mathcal{D}$ , as:

$$\mathcal{D} = \{d \in \mathcal{B}^* : d = \langle TM, a \rangle \wedge TM(a) \downarrow\}.$$

Let  $r \in \mathcal{B}^*$  be a representation. We define the set of *descriptions for r*, denoted by  $\mathcal{D}_r$ , as:

$$\mathcal{D}_r = \{d \in \mathcal{D} : TM(a) = r\}.$$

Finally, given an entity  $e \in \mathcal{E}$ , we define the set of *descriptions for e*, denoted by  $\mathcal{D}_e$ , as:

$$\mathcal{D}_e = \{d \in \mathcal{D} : \exists r \in \mathcal{R}_e, TM(a) = r\}.$$

From an ontological point of view, descriptions are string-based representations that satisfy the additional requirement of being computable. In this sense, descriptions are a subset of representations, and thus, there can exist descriptions that describe other descriptions. However, in practice, it is not advisable to use descriptions as representations of entities, since what we seek in a good representation is the inclusion of as many details as possible about the original entities, not a concise encoding. Using descriptions in place of representations would make the task of scientific discovery considerably more difficult for humans.

Since each description corresponds to one, and only one, representation, we can define a function that maps descriptions to representations. Given that descriptions are encoded Turing machines, it is natural to define this mapping using a universal Turing machine. As a result, not only are individual descriptions of representations computable, but the function that maps descriptions to representations is also computable.

**Definition 2.5.3** We call *description function*, denoted by  $\delta$ , any universal Turing machine  $\delta : \mathcal{D} \rightarrow \mathcal{B}^*$  that maps descriptions to their corresponding representations.

If  $d = \langle TM, a \rangle$  is a description of the representation  $r$ , then we have that  $\delta(d) = \delta(\langle TM, a \rangle) = TM(a) = r$ .

Inspired by the Occam's razor principle<sup>2</sup>, if two explanations are equivalent, we should prefer the shorter one. Accordingly, the limit of what can be known, or understood, about a representation, that is, its perfect model, is given by the shortest description that allows us to reconstruct that representation.

**Definition 2.5.4** Let  $\mathcal{D}_r$  be the set of descriptions of a representation  $r \in \mathcal{R}_{\mathcal{E}}$ , and let  $d \in \mathcal{D}_r$  be a description of  $r$ . We say that  $d$  is a *perfect description* of the representation  $r$  if there is no other description  $d' \in \mathcal{D}_r$  such that  $l(d') < l(d)$ .

Recall that what we know about an entity  $e$  depends on the quality of the representation  $r$  used. If the representation  $r$  is incorrect, we cannot achieve perfect knowledge of  $e$ , even if we have found the perfect description  $d$  for  $r$ .

**Notation 2.4.** We denote by  $d_r^*$  that the description  $d$  is a perfect description of the representation  $r$ .

The perfect description of a representation may not be unique; that is, there could be multiple optimal ways to compute  $r$ .

**Definition 2.5.5** Let  $\mathcal{D}_r$  be the set of descriptions of a representation  $r \in \mathcal{R}_{\mathcal{E}}$ . We define the *set of perfect descriptions* for  $r$ , denoted by  $\mathcal{D}_r^*$ , as the subset of  $\mathcal{D}_r$  consisting of all perfect descriptions of  $r$ .

Unfortunately, the set of perfect descriptions of a representation is generally unknown, and as Proposition 2.5.1 shows, there exists no algorithm to compute it. In practice, we must rely on approximations to estimate how far our current best description is from a perfect one, that is, to quantify how much we do not know about a particular representation of an entity (see Chapter 5).

**Proposition 2.5.1** Let  $r \in \mathcal{B}^*$  be a representation and let  $d_r^*$  be a perfect description of  $r$ . Then we have  $l(d_r^*) = K(r)$ .

*Proof.* Apply Definition 15.1.2 and note that the Turing machines  $TM$  used in descriptions of the form  $\langle TM, a \rangle$  are required to be prefix-free. ■

---

<sup>2</sup>The Occam's razor principle refers to the number of assumptions in an explanation, not to the length of the explanation itself.

The actual length of a description  $l(d)$  for a representation  $r$  depends on the specific encoding of Turing machines used. This encoding method is determined by the chosen description function  $\delta$ . Fortunately, if we replace our description function with a different one, the length of perfect descriptions remains essentially unchanged, up to an additive constant that does not depend on the representation itself.

**Corollary 2.5.2** Let  $r \in \mathcal{R}_{\mathcal{E}}$  be a representation, and let  $\delta$  and  $\dot{\delta}$  be two different description functions. Let  $d_r^*$  be a perfect description of  $r$  under  $\delta$ , and  $\dot{d}_r^*$  a perfect description under  $\dot{\delta}$ . Then  $l(d_r^*) \leq l(\dot{d}_r^*) + c$ , where  $c$  is a constant that does not depend on  $r$ .

*Proof.* Apply Proposition 2.5.1 and Theorem 15.1.1. ■

In general, within the theory of nescience, we are not concerned with computing the exact value of nescience for an entity given a specific description and representation. Instead, our interest lies in the relative ordering of different possible pairs of descriptions and representations according to their nescience. In this sense, the specific details of the universal Turing machine used in practice are not relevant<sup>3</sup>. For the remainder of this book, we will assume that  $\delta$  is fixed to a reference universal Turing machine. Alternatively, the reader may consider that all theorems in this book involving the length of the shortest models are valid up to an additive constant that does not depend on the topics themselves.

A remarkable consequence of Proposition 2.5.1 is that perfect descriptions must be incompressible; that is, *perfect knowledge implies randomness* (see Section 15.6).

**Corollary 2.5.3** Let  $d_r^*$  be a perfect description of a representation  $r$ , then  $K(r) = l(d_r^*)$ .

*Proof.* If  $K(r) < l(d_r^*)$ , then  $d_r^*$  could be replaced by a shorter description, contradicting its minimality as a perfect description. ■

The converse does not generally hold: a description can be random without being the shortest possible one. That is, we may have a description  $d$  of a representation  $r$  such that  $l(d) = K(d)$ , yet  $l(d_r^*) < l(d)$ .

---

<sup>3</sup>Do not confuse the internal workings of the universal Turing machine that maps descriptions to representations, which are not of interest, with the internal workings of the universal oracle Turing machine that maps representations to entities, which are of interest, as understanding this mechanism is crucial to understanding how things work.

■ **Example 2.14** Consider a deep neural network with an input layer of one thousand nodes, ten hidden layers of fifty thousand nodes each, and an output layer of one thousand nodes. Suppose the network is trained to output a fixed string of one thousand 1's for any given input. The Kolmogorov complexity of this neural network is much greater than that of the output string itself, which consists of one thousand identical bits. ■

There is little value on descriptions that are longer than the representations they describe, that is, descriptions that do not compress the representations.

**Definition 2.5.6** Let  $r \in \mathcal{B}^*$  be a representation, and  $d \in \mathcal{D}_r$  one of its descriptions. If  $l(d) \geq l(r)$ , we say that  $d$  is a *pleonastic description* of the representation  $r$ .

■ **Example 2.15** Consider the set of all possible finite graphs. Since graphs are abstract mathematical objects, we must represent them as strings, for instance, using a binary encoding of their adjacency matrices (see Section 11.5 for an introduction to graphs). The description  $d = \langle TM, r \rangle$ , where  $r$  is the representation of a graph and  $TM$  is a Turing machine that simply halts, belongs to  $\mathcal{D}_r$  because  $TM(r) = r$ . However, this description is of limited interest, as it is likely not the shortest possible description of  $r$ . ■

It may happen that there is no shorter possible description of a representation than the representation itself. This occurs when the representation is a random, or incompressible, string. As discussed in Section 15.6, the overwhelming majority of strings are incompressible. Conducting research on random representations is unproductive, as it is not possible to find shorter models for such representations.

The concept of a perfect description can be generalized from individual representations to entire entities. This generalization allows us to study the nature and properties of the entities themselves.

**Definition 2.5.7** Let  $\mathcal{D}_e$  be the set of descriptions of an entity  $e \in \mathcal{E}$ . We define the *set of perfect descriptions* of the entity  $e$ , denoted by  $\mathcal{D}_e^*$  as the subset of  $\mathcal{D}_e$  consisting of perfect descriptions. The elements of  $\mathcal{D}_e^*$  are denoted by  $d_e^*$ .

If  $d_e^* \in \mathcal{D}_e^*$  there must exist a representation  $r \in \mathcal{R}_e^*$  such that  $d_e^* \in \mathcal{D}_r^*$ .

An interesting case arises when all the descriptions in  $\mathcal{D}_e$  are pleonastic, that is, there exists no model shorter than the representation for any of the possible representations of the entity. This situation would occur if all representations of the entity  $e$  are random strings. In such a case, scientific research would be fundamentally limited, as it would be impossible to find a

suitable model for  $e$ . Our ability to understand and make predictions about  $e$  would then be constrained by the length of its incompressible representations.

## 2.6 Descriptions for Joint Representations

In Section 2.4, we introduced the concept of a joint representation  $ts$ , formed by combining two individual representations  $t$  and  $s$ . In this section, we aim to study how the length of the perfect description of a joint representation relates to the lengths of the perfect descriptions of the individual representations.

The length of the perfect description of a joint representation is greater than or equal to the length of the perfect description of either individual representation. In other words, the more information a representation contains, the longer it takes to describe.

**Proposition 2.6.1** Let  $t, s \in \mathcal{R}_{\mathcal{E}}$  be two representations, and let  $m_t^*$ ,  $m_s^*$ , and  $m_{ts}^*$  denote the perfect descriptions of the representations  $t$ ,  $s$ , and the joint representation  $ts$ , respectively. Then:  $l(m_{ts}^*) \geq l(m_t^*)$  and  $l(m_{ts}^*) \geq l(m_s^*)$ .

*Proof.* The inequality  $l(m_{ts}^*) \geq l(m_t^*)$  is equivalent to  $K(ts) \geq K(t)$ . The result then follows from Proposition 15.3.3. ■

Intuitively, adding more information to a representation is beneficial if the additional information is relevant to describing the entity of interest. However, including irrelevant information leads to unnecessarily long models. Recall that joining representations can serve either to concatenate two partial representations of the same entity or to enrich a representation by adding missing symbols.

If the selected representations partially overlap, we can exploit this redundancy to produce a joint description that is shorter than the mere concatenation of the individual descriptions. In the worst-case scenario, the perfect description of a joint representation would be equal in length to the sum of the perfect descriptions of the individual representations.

**Proposition 2.6.2** Let  $t, s \in \mathcal{R}_{\mathcal{E}}$  be two representations, and let  $m_t^*$ ,  $m_s^*$ , and  $m_{ts}^*$  denote the perfect descriptions of the representations  $t$ ,  $s$ , and the joint representation  $ts$ , respectively. Then:  $l(m_{ts}^*) \leq l(m_t^*) + l(m_s^*)$ .

*Proof.* The inequality  $l(m_{ts}^*) \leq l(m_t^*) + l(m_s^*)$  is equivalent to  $K(ts) \leq K(t) + K(s)$ . The result follows from Proposition 15.3.2. ■

One interpretation of Proposition 2.6.2 is that including redundant information in the representation of an entity does not hinder our ability to find its shortest possible description. From the perspective of compression,

redundancy can be eliminated during the modeling process. Therefore, in practice, we may prefer to work with representations that are longer but make the process of scientific discovery, i.e., finding the best model, easier, even if they contain superfluous information. In contrast, Proposition 2.6.1 highlights a different concern: adding irrelevant or non-informative symbols to a representation should be avoided, as they increase the complexity of the description without contributing useful information about the entity.

Finally, the following proposition shows that the order of the representations in the perfect description of a joint representation does not affect its length.

**Proposition 2.6.3** Let  $t, s \in \mathcal{R}_{\mathcal{E}}$  be two representations, and let  $m_{ts}^*$  and  $m_{st}^*$  be the perfect descriptions of the joint representations  $ts$  and  $st$ , respectively. Then:  $l(m_{ts}^*) = l(m_{st}^*)$ .

*Proof.* The equality  $l(m_{ts}^*) = l(m_{st}^*)$  is equivalent to  $K(ts) = K(st)$ . The result follows from Proposition 15.3.1. ■

It is important to note, however, that joining representations is not a commutative operation, there is no guarantee that the strings  $ts$  and  $st$  encode the same entity. Moreover, given only the concatenated string  $ts$ , it is generally not possible to recover the original representations  $t$  and  $s$ , since they are not self-delimiting.

Propositions 2.6.1, 2.6.2 and 2.6.3 can be generalized to any arbitrary, but finite, collection of representations  $t_1, t_2, \dots, t_n$ .

**Proposition 2.6.4** Let  $t_1, t_2, \dots, t_n \in \mathcal{R}_{\mathcal{E}}$  be a finite collection of representations. Then, we have that:

- i  $l(m_{t_1 t_2 \dots t_n}^*) \geq l(m_{t_i}^*) \forall 1 \leq i \leq n$ ,
- ii  $l(m_{t_1 t_2 \dots t_n}^*) \leq l(m_{t_1}^*) + l(m_{t_2}^*) + \dots + l(m_{t_n}^*)$ ,
- iii  $l(m_{t_1 \dots t_i \dots t_j \dots t_n}^*) = l(m_{t_1 \dots t_j \dots t_i \dots t_n}^*) + c \forall 1 \leq i \leq j \leq n$ ,
- iv  $l(m_{t_1 \dots t_{n-1}}^*) \leq l(m_{t_1 \dots t_{n-1} t_n}^*)$ .

*Proof.* Apply Propositions 2.6.1, 2.6.2 and 2.6.3 to individual pairs of representations  $i$  and  $j$ . ■

## 2.7 Conditional Descriptions

It is often cumbersome to include all the information required to reconstruct an entity within a single description, as this would typically result in very long strings for most entities. A more practical approach is to assume the existence of some background knowledge and to quantify our lack of knowledge about an entity relative to that background. In this section, we study the concept of

*conditional descriptions*, that is, constructing a description given some prior description. Conditional descriptions also play a crucial role in the discovery of new knowledge: if conditioning a description on some prior knowledge significantly reduces the inaccuracy of a model, it indicates that this prior knowledge is relevant to understanding the entity.

**Definition 2.7.1** Let  $r, d, s \in \mathcal{B}^*$  be strings. We say that the string  $\langle d, s \rangle$  is a *valid conditional description* of the representation  $r$  given the string  $s$ , denoted by  $d_{r|s}$ , if  $d = \langle TM, a \rangle$  is a description, and  $TM(\langle a, s \rangle) = r$ .

The conditional description  $d_{r|s}$  relies on two distinct strings:  $a$  and  $s$ , each fulfilling a different role. The string  $a$  is provided as input to the Turing machine  $TM$  and is intended to contain the portion of the representation  $r$  that cannot be derived from prior knowledge, that is, the incompressible or novel part. In contrast, the string  $s$  represents background knowledge: it is a description or representation of another entity that is assumed to be already known and that can facilitate the reconstruction or understanding of  $r$ . For example, as we will explain in Chapter 5, when evaluating the redundancy of a conditional description, the contribution of the string  $s$  is disregarded—only the length and content of  $a$  are taken into account.

Note that the conditional description  $d_{r|s}$  does not belong to the set of valid descriptions  $\mathcal{D}$  for the representation  $r$ , since computing  $r$  requires the additional string  $s$ , which is not part of the description itself. Therefore, a new definition is needed to formally capture this concept.

**Definition 2.7.2** Let  $r \in \mathcal{B}^*$  be a representation and  $s \in \mathcal{B}^*$  an arbitrary string. We define the *set of conditional descriptions* of  $r$  given  $s$ , denoted by  $\mathcal{D}_{r|s}$ , as:

$$\mathcal{D}_{r|s} = \{d \in \mathcal{B}^*, d = \langle TM, a \rangle : TM(\langle a, s \rangle) = r\}.$$

For each representation  $r \in \mathcal{B}^*$ , there always exists a conditional description  $d_{r|s}$  that describes  $r$ , as the following proposition shows.

**Proposition 2.7.1** Let  $r \in \mathcal{B}^*$  be a representation and  $s \in \mathcal{B}^*$  an arbitrary string. If  $d \in \mathcal{D}_r$  then  $d \in \mathcal{D}_{r|s}$ .

*Proof.* We can construct a conditional description  $\langle \langle TM, a \rangle, s \rangle$  based on a Turing machine  $TM$  such that, when given the input  $\langle a, s \rangle$ , the machine safely ignores the string  $s$ . ■

The converse of Proposition 2.7.1 is not true. The fact that  $d$  is a conditional description (i.e.,  $d \in \mathcal{D}_{r|s}$ ) does not imply that  $d$  is also a valid

description (i.e.,  $d \in \mathcal{D}_r$ ). Indeed, while we require that  $TM(\langle a, s \rangle) = r$ , we do not require that  $TM(a) = r$ , and in general, this may not hold.

We are interested in the concept of a perfect conditional description. The perfect conditional description of a representation, given some prior knowledge, is the shortest possible string that allows us to fully reconstruct the representation, assuming that the prior knowledge is already known.

**Definition 2.7.3** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d_{r|s}^*$  be the shortest possible description of  $r$  given the string  $s$ . We call  $d_{r|s}^*$  the *perfect conditional description* of the representation  $r$  given the string  $s$ , or simply perfect conditional description of  $r$  given  $s$  for short.

Note that  $d_{r|s}^*$  is a perfect description of the representation  $r$  \*conditional on\* the string  $s$ . This does not imply that  $s$  is a perfect description itself; it may be an incomplete or partially irrelevant representation. In such a case, we would have achieved perfect knowledge with respect to the  $d$  component, but not with respect to the  $s$  component of the combined string  $\langle d, s \rangle$ .

The length of a perfect conditional description is always less than or equal to that of its unconditional counterpart. In other words, assuming the existence of some background knowledge can reduce the effort required to describe a representation.

**Proposition 2.7.2** Let  $r \in \mathcal{B}^*$  be a representation and  $s \in \mathcal{B}^*$  an arbitrary string. Then  $l(d_{r|s}^*) \leq l(d_r^*)$ .

*Proof.* The inequality  $l(d_{r|s}^*) \leq l(d_r^*)$  is equivalent to the well-known result  $K(r | s) \leq K(r)$ . The proposition follows directly by applying Proposition 15.4.3. ■

The notions of unconditional, conditional, and joint descriptions are closely related. In particular, the availability of prior knowledge (as captured by a conditional description) can reduce the length of a description, while describing multiple entities jointly (via a joint description) typically requires more information than describing a single entity. The following proposition formalizes these relationships by comparing the lengths of the perfect conditional description, the perfect (unconditional) description, and the perfect joint description.

**Proposition 2.7.3** Let  $r, s \in \mathcal{B}^*$  two different representations. Then:

$$l(d_{r|s}^*) \leq l(d_r^*) \leq l(d_{rs}^*)$$

*Proof.* The inequality  $l(d_{r|s}^*) \leq l(d_r^*) \leq l(d_{rs}^*)$  is equivalent to the Kolmogorov complexity relations  $K(r | s) \leq K(r)$  and  $K(r) \leq K(rs)$ . The result follows directly by applying Proposition 15.4.4. ■

As was the case with joint descriptions, the concept of conditional description can be naturally extended to finite collections of representations.

**Definition 2.7.4** Let  $r, d, s_1, s_2, \dots, s_n \in \mathcal{B}^*$  be strings. We say that the string  $\langle d, s_1, s_2, \dots, s_n \rangle$  is a *valid conditional description* of the representation  $r$  given the strings  $s_1, s_2, \dots, s_n$ , denoted by  $d_{r|s_1, s_2, \dots, s_n}$ , if  $d = \langle TM, a \rangle$  is a description, and  $TM(\langle a, s_1, s_2, \dots, s_n \rangle) = r$ .

The following definition generalizes the notion of a perfect conditional description to the case of multiple conditioning strings.

**Definition 2.7.5** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d_{r|s_1, s_2, \dots, s_n}^*$  be the shortest possible description of  $r$  given the strings  $s_1, s_2, \dots, s_n$ . We call  $d_{r|s_1, s_2, \dots, s_n}^*$  the *perfect conditional description* of the representation  $r$  given the string  $s_1, s_2, \dots, s_n$ , or perfect conditional description of  $r$  given  $s_1, s_2, \dots, s_n$  for short.

The next proposition generalizes Propositions 2.7.2 and 2.7.3 to any arbitrary (but finite) collection of strings  $s_1, s_2, \dots, s_n$ . In particular, it shows that the more background knowledge we assume for a given representation, the shorter its perfect description becomes.

**Proposition 2.7.4** Let  $r, s_1, s_2, \dots, s_n \in \mathcal{B}^*$  be a finite collection of strings. Then:

$$l(d_{r|s_1, s_2, \dots, s_n}^*) \leq l(d_r^*) \leq l(d_{r, s_1, s_2, \dots, s_n}^*)$$

*Proof.* This follows from the Kolmogorov complexity inequalities  $K(r | s_1, s_2, \dots, s_n) \leq K(r) \leq K(r, s_1, s_2, \dots, s_n)$ , which generalize the results stated in Propositions 2.7.2 and 2.7.3. ■

The following proposition further generalizes the idea that assuming additional background knowledge cannot increase the length of a perfect conditional description.

**Proposition 2.7.5** Let  $r, s_1, s_2, \dots, s_n, s_{n+1} \in \mathcal{B}^*$  be a finite collection of strings. Then:

$$l(d_{r|s_1, s_2, \dots, s_n, s_{n+1}}^*) \leq l(d_{r|s_1, s_2, \dots, s_n}^*)$$

*Proof.* This follows directly from the monotonicity property of conditional Kolmogorov complexity: adding more conditioning information cannot increase the complexity. Formally,  $K(r \mid s_1, s_2, \dots, s_n, s_{n+1}) \leq K(r \mid s_1, s_2, \dots, s_n)$ . ■

## 2.8 Research Areas

Entities can be grouped into research areas. The concept of an area is useful insofar as all the entities included in the area are related to a common subdomain of knowledge or share a common property. The specific criteria used for grouping depend on the practical application of the theory of nescience.

**Definition 2.8.1** Given a set of entities  $\mathcal{E}$ , we define a *research area*  $\mathcal{A}$  as a subset of entities,  $\mathcal{A} \subset \mathcal{E}$ .

If we want to quantify how much we do not know about a research area, we must first provide a representation for that area. In general, research areas are infinite, but the number of known representations is finite. Therefore, we can only describe an area with respect to our current state of knowledge.

**Definition 2.8.2** Let  $\mathcal{A} \subset \mathcal{E}$  be a research area. We define the *known subset of the area*  $\mathcal{A}$ , denoted by  $\hat{\mathcal{A}}$ , as the set of entities  $e_1, e_2, \dots, e_n \in \mathcal{A}$  for which at least one non-pleonastic description is known.

We must distinguish between the knowable subset of  $\mathcal{A}$ , composed of those entities for which a representation exists, and the known subset of  $\mathcal{A}$ , composed of those entities for which at least one non-pleonastic description is known, that is, entities about which some research has already been conducted. Clearly, the set of known entities is a subset of the set of knowable entities.

As our understanding of a research area evolves, the number of entities included in its known subset also changes. Throughout this book, the properties of research areas will always be considered relative to our current state of knowledge.

**Definition 2.8.3** Let  $\mathcal{A} \subset \mathcal{E}$  be a research area with known subset  $\hat{\mathcal{A}} = \{e_1, e_2, \dots, e_n\}$ , and let  $R_{\hat{\mathcal{A}}} = \{r_1, r_2, \dots, r_n\}$  be a set of representations such that  $r_i \in \mathcal{R}_{e_i}$ . We call  $R_{\hat{\mathcal{A}}}$  a *representation of the area*  $\mathcal{A}$  given the known subset  $\hat{\mathcal{A}}$ , abbreviated as *representation of*  $\mathcal{A}$ .

In a similar manner to how we describe individual entities, we can also introduce the concept of a description for an entire research area. Since a research area is represented by a collection of known representations corresponding to its known subset, a description of the area must account

for the generation of this entire set. Thus, we define a description of a research area as a program that, when executed, produces the sequence of representations associated with the known entities in that area.

**Definition 2.8.4** Let  $R_{\hat{A}} = \{r_1, r_2, \dots, r_n\}$  be the representation of an area  $A$ . We call a *description of the area A* given the known subset  $\hat{A}$ , abbreviated as *description of A*, and denoted by  $d_{\hat{A}}$ , to any string in the form  $\langle TM, a \rangle$  such that the Turing machine  $TM$ , when given input  $a$ , outputs the sequence  $\langle r_1, r_2, \dots, r_n \rangle$ .

We can also consider all possible descriptions that generate the full set of known representations for a given research area. These descriptions differ in structure, length, or computational efficiency, but they all produce the same output: the sequence of representations associated with the known entities in the area.

**Definition 2.8.5** Let  $R_{\hat{A}} = \{r_1, r_2, \dots, r_n\}$  be the representation of an area  $A$ . We define the set of *descriptions for  $R_{\hat{A}}$* , denoted by  $\mathcal{D}_{R_{\hat{A}}}$ , as:

$$\mathcal{D}_{R_{\hat{A}}} = \{d \in \mathcal{D} : TM(a) = \langle r_1, r_2, \dots, r_n \rangle\}.$$

Finally, we are interested in identifying the perfect model for a research area, that is, the shortest possible string that fully describes its known subset. According to Definition 2.5.6, if we are aware of the existence of an entity  $e \in A$ , then  $e$  should be included in the known subset  $\hat{A}$ , even if no research has yet been conducted on that specific topic.

**Definition 2.8.6** Let  $A \subset \mathcal{E}$  be an area with known subset  $\hat{A}$ , and let  $d_{\hat{A}}^* \in \mathcal{D}_{R_{\hat{A}}}$  be the shortest possible description of  $A$ . We call  $d_{\hat{A}}^*$  the *perfect description of the area A* given the known subset  $\hat{A}$ , abbreviated as *perfect description of A*.

The following proposition shows the relationship between the description of a research area and the descriptions of the individual entities that compose its known subset. In general, the models for an area are not simply the collection of the models for each individual topic; instead, a joint model may offer a more concise description.

**Proposition 2.8.1** Let  $A \subset \mathcal{E}$  be an area with known subset  $\hat{A} = \{e_1, e_2, \dots, e_n\}$ , then we have that  $l(d_{\hat{A}}^*) \leq l(d_{e_1}^*) + l(d_{e_2}^*) + \dots + l(d_{e_n}^*)$ .

*Proof.* Apply Proposition 2.6.4-ii. ■

Moreover, as shown in Proposition 2.6.4, the order in which the representations are listed in the description of an area does not affect the length of

its perfect model.

Research areas can overlap; that is, given two areas  $A$  and  $B$ , it may be the case that  $A \cap B \neq \emptyset$ . Furthermore, one area can be a subset of another, forming a hierarchy of areas. In this context, we are particularly interested in how the length of perfect models for some areas compares to the length of perfect models for related areas.

**Proposition 2.8.2** Let  $A, B \subset \mathcal{E}$  be two areas such that  $A \subset B$ , and let  $\hat{A}$  and  $\hat{B}$  be their known subsets respectively, then we have that  $l(d_{\hat{A}}^*) \leq l(d_{\hat{B}}^*)$ .

*Proof.* Since  $A \subset B$ , it follows that  $\hat{A} \subset \hat{B}$ . Let

$$R_{\hat{A}} = \{r_1, r_2, \dots, r_m\} \quad \text{and} \quad R_{\hat{B}} = \{r_1, r_2, \dots, r_m, r_{m+1}, \dots, r_n\}$$

be the sets of representations corresponding to  $\hat{A}$  and  $\hat{B}$  respectively, and let  $d_{\hat{B}}^*$  be the perfect description of  $\hat{B}$ . Then, we can construct a description  $d'$  of  $\hat{A}$  by modifying  $d_{\hat{B}}^*$  to output only the subset  $R_{\hat{A}}$ . This can be achieved by appending a simple postprocessing step that discards the extra representations. The additional cost of this truncation is at most a constant number of bits, independent of the specific contents of  $R_{\hat{B}}$ .

Formally, we have:

$$l(d_{\hat{A}}^*) \leq l(d') \leq l(d_{\hat{B}}^*) + c$$

for some constant  $c$ . But since  $d_{\hat{A}}^*$  is the shortest possible description of  $R_{\hat{A}}$ , we conclude:

$$l(d_{\hat{A}}^*) \leq l(d_{\hat{B}}^*).$$

■

The following proposition shows how the length of the shortest possible description of two areas relates to the length of the description of their union and intersection.

**Proposition 2.8.3** Let  $A, B \subset \mathcal{E}$  be two areas with known subsets  $\hat{A}$  and  $\hat{B}$  respectively, then we have that  $l(d_{\hat{A} \cup \hat{B}}^*) = l(d_{\hat{A}}^*) + l(d_{\hat{B}}^*) - l(d_{\hat{A} \cap \hat{B}}^*)$ .

*Proof.* Let  $R_{\hat{A}}$ ,  $R_{\hat{B}}$ , and  $R_{\hat{A} \cap \hat{B}}$  be the sets of representations corresponding to the known subsets  $\hat{A}$ ,  $\hat{B}$ , and  $\hat{A} \cap \hat{B}$ , respectively.

From the theory of Kolmogorov complexity, the minimal description length of the union of two finite sets of strings satisfies the following identity:

$$K(R_{\hat{A} \cup \hat{B}}) = K(R_{\hat{A}}) + K(R_{\hat{B}}) - K(R_{\hat{A} \cap \hat{B}}),$$

Since, by definition, the perfect description  $d_{\hat{A}}^*$  satisfies:

$$l(d_{\hat{A}}^*) = K(R_{\hat{A}}), \quad l(d_{\hat{B}}^*) = K(R_{\hat{B}}), \quad l(d_{\hat{A} \cap \hat{B}}^*) = K(R_{\hat{A} \cap \hat{B}}),$$

it follows that:

$$l(d_{\hat{A} \cup \hat{B}}^*) = K(R_{\hat{A} \cup \hat{B}}) = l(d_{\hat{A}}^*) + l(d_{\hat{B}}^*) - l(d_{\hat{A} \cap \hat{B}}^*).$$

■

A consequence of Proposition 2.8.3 is that  $l(d_{\hat{A} \cup \hat{B}}^*) \leq l(d_{\hat{A}}^*) + l(d_{\hat{B}}^*)$ , that is, when we combine two different research areas, how much we do not know about these areas decreases.

Just as we introduced a chain rule for entropy in Proposition 14.4.5, we can also establish a chain rule for the shortest length of a description of a research area.

**Proposition 2.8.4** Let  $A, B \subset \mathcal{E}$  be two areas with known subsets  $\hat{A}$  and  $\hat{B}$ , then we have that  $l(d_{\hat{A} \cup \hat{B}}^*) = l(d_{\hat{A}}^*) + l(d_{\hat{B} \setminus \hat{A}}^*)$ .

*Proof.* Let  $R_{\hat{A}}$  be the set of representations associated with  $\hat{A}$ , and let  $R_{\hat{B} \setminus \hat{A}}$  be the set of representations corresponding to entities in  $\hat{B}$  that are not in  $\hat{A}$ .

By definition, the known subset of the union  $\hat{A} \cup \hat{B}$  corresponds to the set of representations:

$$R_{\hat{A} \cup \hat{B}} = R_{\hat{A}} \cup R_{\hat{B} \setminus \hat{A}}.$$

Let  $d_{\hat{A}}^*$  be the shortest (perfect) description that generates  $R_{\hat{A}}$ , and let  $d_{\hat{B} \setminus \hat{A}}^*$  be the shortest description that generates  $R_{\hat{B} \setminus \hat{A}}$ . Because the two subsets are disjoint, we can concatenate these two descriptions to produce a description of  $R_{\hat{A} \cup \hat{B}}$ .

Hence, the length of the shortest description of the union satisfies:

$$l(d_{\hat{A} \cup \hat{B}}^*) \leq l(d_{\hat{A}}^*) + l(d_{\hat{B} \setminus \hat{A}}^*).$$

To prove equality, assume there exists a shorter description  $d'$  for  $\hat{A} \cup \hat{B}$  such that:

$$l(d') < l(d_{\hat{A}}^*) + l(d_{\hat{B} \setminus \hat{A}}^*).$$

Then, one could extract from  $d'$  both  $R_{\hat{A}}$  and  $R_{\hat{B} \setminus \hat{A}}$ , which would imply that at least one of  $d_{\hat{A}}^*$  or  $d_{\hat{B} \setminus \hat{A}}^*$  is not minimal—contradicting the assumption that they are perfect descriptions.

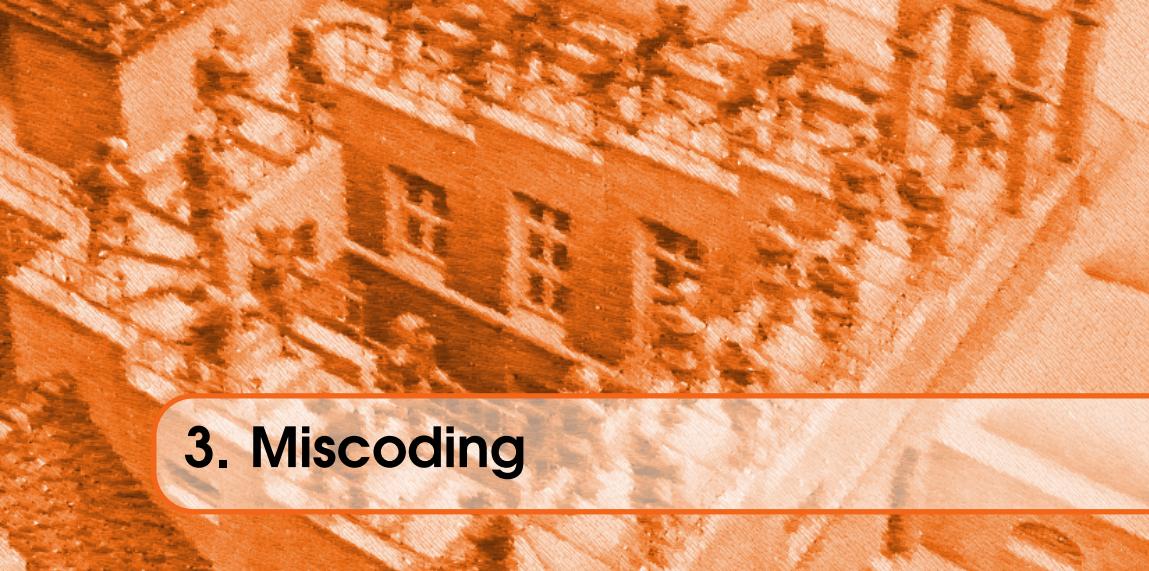
Therefore:

$$l(d_{\hat{A} \cup \hat{B}}^*) = l(d_{\hat{A}}^*) + l(d_{\hat{B} \setminus \hat{A}}^*).$$

■

## 2.9 References

For more information about Russell’s paradox, Cantor’s theorem, and universal sets, refer, for example, to [Jec13]. This book also covers the Zermelo–Fraenkel system of axioms, including the Axiom of Choice. The idea of using a function to assign to each symbol and well-formed formula of a formal language a unique natural number (called a Gödel number) was introduced by Kurt Gödel in his proof of the incompleteness theorems [Göd31]. A detailed discussion of the Berry paradox from the perspective of computability can be found in [Cha95]. For a review of the problem of representation in Kolmogorov complexity, as well as a detailed account of the implications of Kolmogorov complexity being defined only up to an additive constant, see [LV13]. That oracle machines are not mechanical was originally stated by Turing when he introduced the concept of the oracle machine in [Tur39]. For a comprehensive review of oracle Turing machines, refer to [Rob15].



## 3. Miscoding

*All great work is the fruit of patience and perseverance,  
combined with tenacious concentration on a subject  
over a period of months or years.*

Santiago Ramón y Cajal

In many areas of science, the entities we aim to study (the set  $\mathcal{E}$ ) often include abstract concepts or complex phenomena that are difficult to investigate directly. These may be ideas or processes that resist straightforward description or modeling. To gain insight into such entities, we must often rely on indirect methods. As discussed in the previous chapter, the theory of nescience suggests using representations, that is, sequences of symbols, rather than attempting to interact directly with the entities themselves.

However, this approach introduces its own challenges. Because our understanding of the elements in  $\mathcal{E}$  is incomplete (otherwise, further research would be unnecessary), the representations we construct are typically imperfect, they lack the completeness and precision we would ideally desire. These imperfections can lead to significant problems: inaccuracies in representation can propagate into the models derived from them, potentially distorting our interpretation and understanding. It is therefore essential to recognize this source of error and to carefully consider its implications.

To address this issue, we introduce the concept of *miscoding*, a measure used to quantify the error that arises from misrepresenting or inaccurately encoding entities. Miscoding is defined in terms of the length of the shortest computer program capable of transforming an incorrect representation into a correct one. In essence, it measures the amount of effort, as reflected by the program's length, required to correct a faulty representation.

However, there is a significant complication: applying this idea in practice is not straightforward, since the perfect representations of entities are unknown. As a result, we cannot directly measure the gap between a given representation and an ideal one. Nevertheless, we propose a theoretical framework based on the oracle machine (an abstract construct) used to define the set  $\mathcal{E}$ . This oracle machine is assumed to be capable of recognizing valid representations for all entities in the set. It is important to note, however, the limitations of this approach; for instance, we cannot query the oracle about a specific entity unless we already possess a valid representation of it.

Getting a better grasp on scientific representation and the challenges in reducing miscoding could really push forward scientific research, helping us create better, more thorough models of the natural world. In this chapter, we are going to properly introduce the idea of miscoding and look into its various characteristics. We will examine how miscoding behaves when it comes to combined representations and discuss methods to reduce miscoding. We will delve into the idea that there are strings that do not represent any entity. And we will dig into why miscoding is important in different research areas and talk about its potential to open up new lines of inquiry.

Gaining a deeper understanding of scientific representation and the difficulties involved in reducing miscoding can significantly advance scientific research, enabling the development of more accurate and comprehensive models of the natural world. In this chapter, we formally introduce the concept of *miscoding* and explore its key properties. We will analyze how miscoding behaves in the context of combined representations and consider strategies for reducing it. We will also examine the notion that some strings fail to represent any entity and investigate the implications of this idea. Finally, we will highlight the relevance of miscoding across various research domains and discuss its potential to inspire new lines of inquiry.

### 3.1 Miscoding

In an ideal scenario, we would ask the oracle to determine how far a particular string  $r$  is from perfectly encoding an entity  $e$ . This, however, presents a complication: we can only refer to entity  $e$  through a valid representation in the set  $\mathcal{R}_e^*$ . Unfortunately, we typically do not have access to these perfect

representations and, therefore, cannot use them in our query.

To overcome this limitation, we propose an alternative approach. Rather than asking the oracle to assess the discrepancy between our current representation  $r$  and a valid representation of the specific entity  $e$ , we instead ask the oracle to identify the smallest distance between  $r$  and all valid representations of all entities in the set  $\mathcal{E}$ , that is, the elements of  $\mathcal{R}_\mathcal{E}^*$ . As discussed in detail in Section 2.3, this type of query can, in principle, be handled by the oracle.

**Definition 3.1.1 — Miscoding.** Let  $r \in \mathcal{B}^*$  be a representation. We define the *miscoding* of  $r$ , denoted by  $\mu(r)$ , as:

$$\mu(r) = \min_{r_e^* \in \mathcal{R}_\mathcal{E}^*} \frac{\max\{K(r | r_e^*), K(r_e^* | r)\}}{\max\{K(r), K(r_e^*)\}}$$

where  $\min^o$  indicates that the minimum is to be computed by an oracle.

Intuitively, our lack of knowledge about an entity is reflected in a higher miscoding value of its current representation. Conversely, a deeper understanding of the entity should lead to an encoding that is closer to a valid representation, and thus exhibits lower miscoding.

Miscoding is computed using a bidirectional approach: we require the oracle to determine both the length of the shortest computer program that can generate the valid representation  $r_e^*$  from our current representation  $r$  (i.e.,  $K(r_e^* | r)$ ), and the length of the shortest program that can generate  $r$  from  $r_e^*$  (i.e.,  $K(r | r_e^*)$ ). This bidirectionality captures the maximum difficulty of transforming one representation into the other. A high-quality representation should allow for an easy reconstruction of the correct representation and vice versa, implying that both of these conditional complexities are low. In other words, an ideal representation contains all the information needed to recover the correct encoding of the entity, without introducing any erroneous or irrelevant symbols.

While miscoding concerns whether the symbols in a representation are relevant, that is, whether they correctly refer to the intended entity, it is not concerned with how many symbols are used. In contrast, surfeit (as discussed in Chapter 5) focuses on whether the symbols are essential, aiming to minimize the number of unnecessary or redundant symbols. In this sense, miscoding relates to the correctness of the content, while surfeit relates to the efficiency of its expression. Although both concepts address flaws in representation, they capture fundamentally different types of representational inefficiency.

Our definition of miscoding is formulated as a relative measure rather than an absolute one. Instead of providing an isolated score for a single

representation, it quantifies how far a representation is from the nearest valid representation, normalized by their respective lengths. This relative formulation allows us to meaningfully compare the miscoding of different representations of the same entity, and also to compare the miscoding values across representations of different entities, regardless of their size or complexity.

The miscoding value of a representation  $r$  always lies within the interval  $[0, 1]$ .

**Proposition 3.1.1** For all  $r \in \mathcal{B}^*$ , it holds that  $0 \leq \mu(r) \leq 1$ .

*Proof.* This follows directly from the inequality  $0 \leq \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}} \leq 1$  which holds for all  $x, y \in \mathcal{B}^*$ , as stated in Proposition 15.5.4. ■

Miscoding is zero if, and only if, the representation  $r$  is a valid representation of some entity  $e$ .

**Proposition 3.1.2** Given a representation  $r \in \mathcal{B}^*$ , we have  $\mu(r) = 0$  if, and only if,  $r \in \mathcal{R}_e^*$ .

*Proof.* If  $r \in \mathcal{R}_e^*$ , then there exists an entity  $e \in \mathcal{E}$  such that  $r = r_e^*$ . In this case,  $K(r | r_e^*) = 0$ , and thus  $\mu(r) = 0$ .

Conversely, if  $\mu(r) = 0$ , then there must exist some  $r_e^* \in \mathcal{R}_e^*$  such that  $K(r | r_e^*) = 0$  and  $K(r_e^* | r) = 0$ , which implies that  $r = r_e^*$ , and hence  $r \in \mathcal{R}_e^*$ . ■

According to Proposition 3.1.2, a miscoding value of zero implies that  $r$  perfectly encodes some entity  $e$ . However, identifying the exact entity that  $r$  represents remains a challenge. While scientific intuition may offer plausible hypotheses about the encoded entity, such guesses cannot be confirmed mathematically. Moreover, as scientific understanding evolves, our interpretation of what  $r$  encodes may also change over time (see Example 2.10).

It has been observed that an entity  $e \in \mathcal{E}$  may possess multiple valid representations, as captured by the set  $\mathcal{R}_e^*$ . Fortunately, the value of miscoding is invariant under the choice of valid representation (see the discussion on style in Section 17.4).

**Proposition 3.1.3** For any valid representation  $r_e^* \in \mathcal{R}_e^*$ , it holds that  $\mu(r_e^*) \leq \mu(r)$  for all  $r \in \mathcal{B}^*$ .

*Proof.* This follows from the fact that  $\mu(r_e^*) = 0$  and  $\mu(r) \geq 0$  for all  $r \in \mathcal{B}^*$ . ■

For a given entity  $e$ , all valid representations in the set  $\mathcal{R}_e^*$  are equally adequate with respect to miscoding, as each yields a miscoding value of 0. From a practical standpoint, however, the most suitable representation is the one that best facilitates the discovery of new knowledge about the entity—specifically, the representation that most effectively supports the construction of explanatory models.

## 3.2 Joint Miscoding

Section 2.4 introduced the notion of a *joint representation*, which arises when two representations  $s, t \in \mathcal{B}^*$  are concatenated to form a new string  $st$ . This section explores the properties of miscoding as they apply to such joint representations.

Since the concatenated string  $st$  is itself a valid representation, it is not necessary to introduce a separate definition of miscoding for joint representations. The miscoding of the joint representation is given by:

$$\mu(st) = \min_{r_e^* \in \mathcal{R}_e^*} \frac{\max\{K(st | r_e^*), K(r_e^* | st)\}}{\max\{K(st), K(r_e^*)\}}$$

Consistent with Proposition 3.1.1, the miscoding of a joint representation is a value between 0 and 1, i.e.,  $0 \leq \mu(st) \leq 1$ .

It is important to note that supplementing an incomplete representation, that is, one with a positive miscoding value, with additional symbols does not necessarily lead to a reduction in miscoding. This is because the added symbols may introduce irrelevant or incorrect information. Conversely, miscoding does not necessarily increase either, since incorporating relevant and accurate symbols can reduce the miscoding of an incomplete representation.

From a formal perspective, given two arbitrary representations  $s, t \in \mathcal{B}^*$ , there is no guarantee that any of the following inequalities will always hold:  $\mu(ts) \geq \mu(t)$ ,  $\mu(ts) \leq \mu(t)$ ,  $\mu(ts) \geq \mu(s)$ , or  $\mu(ts) \leq \mu(s)$ .

**Example 3.1** Consider the text  $r$  of a biology research article that succinctly and accurately describes a newly discovered specimen, resulting in a low miscoding value  $\mu(r)$ . However, when additional text  $s$ , taken from a completely unrelated article about a second specimen from a different species, is appended, the resulting concatenated representation  $rs$  muddles the original focus. This inclusion of unrelated content increases the miscoding value to  $\mu(rs)$ , illustrating a case in which  $\mu(rs) \geq \mu(r)$ . Although the added text is scientifically valid on its own, it dilutes the clarity and precision of the original article, thereby increasing its miscoding.

Conversely, in a second scenario, consider another research article  $r'$  that exhibits a relatively high miscoding value  $\mu(r')$ , due to an incomplete

description of the specimen. If this article is supplemented with additional text  $s'$ , containing essential information previously omitted, the resulting concatenated representation  $r's'$  becomes significantly more informative and coherent. As a result, the miscoding value decreases, demonstrating a case where  $\mu(r's') \leq \mu(r')$ . This highlights that enriching an incomplete representation with relevant and focused content can yield a more accurate and lower-miscoding representation. ■

Furthermore, it should not be assumed that the miscoding of a joint representation is less than or equal to the sum of the miscodings of the individual representations, i.e.,  $\mu(st) \leq \mu(s) + \mu(t)$ . This inequality does not necessarily hold, even when both  $s$  and  $t$  encode the same entity. The reason is that the joint representation  $\$st\$$  may end up encoding an entirely different entity from those represented by  $s$  and  $t$  individually.

■ **Example 3.2** In a medical research context, representation  $r$  describes a new drug compound  $A$  for treating a specific type of cancer, while representation  $s$  outlines a genetic mutation  $B$  associated with that cancer type. Both  $r$  and  $s$  have low miscoding values, denoting accurate, focused content. However, when concatenated, the combined representation  $rs$  unintentionally suggests a third entity  $C$ , implying a causal relationship between  $A$  and  $B$ . This unintended implication stems from the mixture of distinct information, leading to a higher miscoding value. In this instance, each of  $r$ ,  $s$ , and  $rs$  essentially represents different research entities.  $r$  and  $s$  are clear in their individual contexts, but  $rs$  is ambiguous, exemplifying a case where concatenated information can inadvertently create a representation of an entirely different, unintended entity, thus increasing miscoding. ■

Given the non-commutative nature of joining two representations, it is not guaranteed that  $\mu(ts) = \mu(st)$ . This is because the composite strings  $ts$  and  $st$  may encode entirely different entities. This property holds true even when we restrict our attention to valid representations of a specific entity  $e$ . For example, if  $r_e^*, s_e^* \in \mathcal{R}_e^*$  are two valid representations of the entity  $e$ , there is no assurance that the concatenated string  $r_e^*s_e^*$  will itself be a valid representation of  $e$ .

■ **Example 3.3** In environmental science, representation  $r$  thoroughly examines microplastics, their nature and harmful impacts on marine ecosystems, while representation  $s$  focuses on specific technologies and methodologies for detecting these pollutants in water. The concatenation  $rs$  presents a logical progression: first describing the pollutant and its effects, then outlining detection methods. This creates a coherent representation of a broader entity, one concerned with the pollutant, its impact, and how to detect it, an entity

which may not yet be fully understood, hence yielding a higher miscoding value.

Conversely, the concatenation  $sr$  begins with technical methodologies for detection and then provides the contextual background on microplastics and their environmental effects. This ordering may be more familiar within a methodological framework and results in a more coherent and well-understood entity, yielding a lower miscoding value.

In this particular case, we observe that  $\mu(rs) > \mu(sr)$ . ■

The concept of miscoding can be naturally extended to joint representations formed from any finite collection of representations. Let  $r_1, r_2, \dots, r_n \in \mathcal{B}^*$  be a finite collection of representations. The miscoding of their concatenation, denoted by  $r_1r_2\dots r_n$ , is defined as:

$$\mu(r_1r_2\dots r_n) = \min_{r_e^* \in \mathcal{R}_e^*} \frac{\max\{K(r_1r_2\dots r_n | r_e^*), K(r_e^* | r_1r_2\dots r_n)\}}{\max\{K(r_1r_2\dots r_n), K(r_e^*)\}}$$

As in the case of concatenating two representations, the miscoding of a joint representation composed of  $\$$  elements remains bounded within the unit interval  $0 \leq \mu(r_1r_2\dots r_n) \leq 1$ . However, several caveats must be noted. There is no guarantee that  $\mu(r_1r_2\dots r_n) \leq \mu(r_i)$  or  $\mu(r_1r_2\dots r_n) \geq \mu(r_i)$  for any  $i = 1, \dots, n$ . Moreover, the miscoding of the joint representation is not necessarily less than or equal to the sum of the individual miscodings  $\mu(r_1r_2\dots r_n) \not\leq \mu(r_1) + \mu(r_2) + \dots + \mu(r_n)$ . Finally, no general conclusions can be drawn regarding how the miscoding of a joint representation is affected by permutations of its components. That is, reordering the representations may increase, decrease, or leave unchanged the overall miscoding.

### 3.3 Decreasing Miscoding

A valid representation of an entity is a string that contains all the essential information required by the oracle to reproduce the entity, no more, no less. In contrast, an invalid representation, indicated by a miscoding value greater than zero, may result from missing critical information, the inclusion of incorrect symbols, or the presence of irrelevant symbols. To reduce the miscoding of a representation, one may attempt to supply the missing information or remove erroneous or non-pertinent symbols. However, it is generally not possible to determine in advance which of these actions will be effective. Nevertheless, as the following theorem shows, at least one of these approaches must be applicable.

**Theorem 3.3.1** Let  $r \in \mathcal{B}^*$  be a representation such that  $\mu(r) > 0$ , then one or both of the following conditions must hold:

- (i) There exists a  $s \in \mathcal{B}^*$  such that  $\mu(rs) < \mu(r)$  or  $\mu(sr) < \mu(r)$ ,
- (ii) There exists an  $s \in \mathcal{B}^*$  in the form  $r = \alpha s \beta$  with  $\alpha, \beta \in \mathcal{B}^*$  such that  $\mu(s) < \mu(r)$ .

*Proof.* Let  $r \in \mathcal{B}^*$  be a representation with

$$\mu(r) = \min_{r_e^* \in \mathcal{R}_e^*} \frac{\max\{K(r | r_e^*), K(r_e^* | r)\}}{\max\{K(r), K(r_e^*)\}} > 0.$$

Fix one valid representation  $r^* \in \mathcal{R}_e^*$  at which the minimum is attained, and write

$$A := K(r | r^*), \quad B := K(r^* | r), \quad L := \max\{K(r), K(r^*)\},$$

so that  $\mu(r) = \frac{\max\{A, B\}}{L}$ .

We distinguish two (exhaustive) cases. Case i).  $B > 0$ , that is, some information is missing in  $r$ . Because  $B > 0$ , a shortest self-delimiting program  $p$  of length  $B$  exists that, given  $r$ , outputs  $r^*$ . Set  $s := p$  and consider the concatenation  $r' := rs$ . Because  $s$  already is the program that converts  $r$  into  $r^*$ ,

$$K(r^* | r') = O(1), \quad K(r' | r^*) \leq A + B + O(1).$$

Hence

$$\max\{K(r^* | r'), K(r' | r^*)\} \leq A + B + O(1) < A + L.$$

The description length of  $r'$  satisfies  $K(r') \leq K(r) + B + O(\log B)$ , so

$$\max\{K(r'), K(r^*)\} \geq \max\{K(r), K(r^*)\} = L.$$

Thus

$$\mu(r') = \frac{\max\{K(r^* | r'), K(r' | r^*)\}}{\max\{K(r'), K(r^*)\}} < \frac{A + B + O(1)}{L} \leq \frac{\max\{A, B\}}{L} = \mu(r),$$

so either  $\mu(rs) < \mu(r)$  (if we append) or, by symmetric reasoning,  $\mu(sr) < \mu(r)$  (if we prepend). Hence condition (i) is satisfied.

Case ii).  $B = 0$ , that is, all information needed to produce  $r^*$  is contained in  $r$ ; extra symbols remain. Now  $A > 0$  because  $\mu(r) > 0$ . Since  $K(r^* | r) = 0$ , a deterministic algorithm can scan  $r$  from left to right and halt as soon as it has seen a \*\*shortest prefix\*\* that already allows it to output  $r^*$ . Let  $s$

be that prefix and write  $r = \alpha s \beta$  with  $\beta \neq \lambda$  (otherwise  $r = s$  and  $A = 0$ , contradicting  $\mu(r) > 0$ ).

$$K(r^* | s) = 0, \quad K(s | r^*) \leq K(r | r^*) = A.$$

Because  $s$  is a proper prefix of  $r$ ,  $K(s) \leq K(r)$  (up to an  $O(\log K(r))$  term). Therefore

$$\mu(s) = \frac{\max\{K(s | r^*), K(r^* | s)\}}{\max\{K(s), K(r^*)\}} \leq \frac{A}{\max\{K(s), K(r^*)\}} < \frac{\max\{A, B\}}{L} = \mu(r),$$

so condition (ii) holds with that substring  $s$ . ■

Refer to Example 2.11 for a case in which appending additional information to a representation improves its quality, that is, reduces its miscoding. Also, the following example illustrates the opposite situation: a case in which removing certain symbols from a representation can lead to a decrease in miscoding.

**■ Example 3.4** A historian compiles a representation  $r$  intended to narrate the events surrounding the signing of a pivotal treaty. However, the inclusion of speculative statements and personal interpretations about the intentions of the involved parties, statements not supported by primary sources, results in a high miscoding value. Although the document is rich in content, it is contaminated by erroneous or unverifiable information, rendering it an invalid representation of the historical event.

By removing these speculative and unsubstantiated segments, a revised version  $r'$  is produced. This refined representation presents a concise and factual account based solely on verifiable data and primary sources. The removal of misleading content reduces the miscoding value, making  $r'$  a more valid representation that more accurately reflects the historical event without the noise of uncorroborated interpretations. ■

## 3.4 Targetless Representations

In the most extreme scenario, the miscoding value can reach its upper bound of 1. This occurs when the current representation  $r$  does not contain a single symbol that contributes to the encoding of any entity in the set  $\mathcal{E}$ . In such a case,  $r$  is said to be a targetless representation, an abstract string with no concrete or meaningful interpretation.

**Definition 3.4.1** Let  $r \in \mathcal{B}^*$  be a representation. If  $\mu(r) = 1$  we say that  $r$  is a *targetless* representation.

Although one could, in principle, assign a targetless representation to an arbitrary entity, doing so would violate the essential requirement of surrogate reasoning (see Section 17.4). The following proposition formalizes this idea.

**Proposition 3.4.1** Given a representation  $r \in \mathcal{B}^*$ , the miscoding  $\mu(r)$  reaches its maximum value of 1 if and only if no symbol in  $r$  contributes to the encoding of any entity in  $\mathcal{E}$ .

*Proof.* Assume that  $r$  contains no symbol that contributes to the encoding of any entity  $e \in \mathcal{E}$ . Let  $r_e^*$  be a valid representation of some  $e \in \mathcal{E}$ . Then, the shortest program capable of generating  $r_e^*$  from  $r$  must encode the entirety of  $r_e^*$  without reuse of any symbols from  $r$ . Thus,  $K(r_e^* | r) = K(r_e^*)$ . Similarly, since no symbol in  $r_e^*$  contributes to the generation of  $r$ , we have  $K(r | r_e^*) = K(r)$ . Substituting into the definition of miscoding:

$$\mu(r) = \frac{\max\{K(r | r_e^*), K(r_e^* | r)\}}{\max\{K(r), K(r_e^*)\}} = \frac{\max\{K(r), K(r_e^*)\}}{\max\{K(r), K(r_e^*)\}} = 1$$

assuming  $K(r) \neq 0$  and  $K(r_e^*) \neq 0$ . Conversely, suppose that  $\mu(r) = 1$ . Then:

$$\max\{K(r | r_e^*), K(r_e^* | r)\} = \max\{K(r), K(r_e^*)\}$$

which implies that neither  $r$  can help generate  $r_e^*$  nor  $r_e^*$  can help generate  $r$ . Therefore, there is no overlap in information, and  $r$  contains no symbol that contributes to the encoding of any entity in  $\mathcal{E}$ . ■

For every finite or countably infinite set of entities  $\mathcal{E}$ , there exists an infinite number of targetless representations. We will prove this result constructively by showing that at least one targetless representation exists and then defining a method to generate infinitely many more from it.

**Proposition 3.4.2** For every finite or countably infinite set of entities  $\mathcal{E}$ , there exists an uncountably infinite set of targetless representations.

*Proof.* Let  $\mathcal{R}$  denote the set of all valid representations of entities in  $\mathcal{E}$ , such that for each entity  $e \in \mathcal{E}$ , there exists a representation  $r \in \mathcal{R}$  and vice versa. Since  $\mathcal{E}$  is finite or countably infinite,  $\mathcal{R}$  is also countable.

Now consider the set  $\mathcal{B}^*$  of all finite binary strings. This set is countably infinite, as it includes all possible finite sequences over the alphabet 0, 1. Because  $\mathcal{R}$  is a subset of  $\mathcal{B}^*$  and is countable, and  $\mathcal{B}^* \setminus \mathcal{R}$  is non-empty and infinite, there must exist binary strings that are not valid representations of any entity in  $\mathcal{E}$ .

Let  $r_0 \in \mathcal{B}^* \setminus \mathcal{R}$  be any such string. By definition,  $r_0$  is a targetless representation.

Given any targetless representation  $r_i$ , we can construct a new targetless representation  $r_{i+1}$  by appending a symbol (either '0' or '1') to  $r_i$ . Since  $r_i$  does not encode any entity in  $\mathcal{E}$ , and  $\mathcal{R}$  is prefix-free with respect to targetless extensions (i.e., extending a non-representation cannot turn it into a valid one), the new string  $r_{i+1}$  also does not correspond to any entity. Therefore,  $r_{i+1}$  is also targetless.

By iterating this process, we generate an infinite sequence of distinct targetless representations  $r_0, r_1, r_2, \dots \subset \mathcal{B}^* \setminus \mathcal{R}$ . This proves that there exists a countably infinite set of targetless representations corresponding to any finite or countably infinite set of entities  $\mathcal{E}$ . ■

What constitutes a targetless representation for one oracle may not necessarily be targetless for another, as the following example illustrates.

■ **Example 3.5** Imagine two machines, A and B, each controlling a robotic arm capable of manufacturing nuts and bolts. Machine A operates using a low-level assembly language, whereas Machine B uses a more sophisticated high-level programming language. As a result, a particular set of instructions that fails to produce a valid output on Machine A—thus being considered a targetless representation—could be perfectly interpretable by Machine B, successfully yielding a finished bolt. ■

The normalized compression distance between a targetless representation and the closest valid (non-targetless) representation may be less than one. This indicates that the targetless representation shares some information with a valid one. However, this is not sufficient to ensure scientific progress, as the computational effort, reflected in the size or complexity of the Turing machine required to extract useful knowledge, may be larger.

## 3.5 Miscoding of Areas

The concept of miscoding can be extended to research areas to quantitatively measure the amount of effort required to correct an inaccurate representation of an area. Unfortunately, as discussed in Section 2.8, there is no reliable way to verify whether the strings included in an  $n$ -fold representation actually correspond to entities within that area. Moreover, it is not possible to prevent cases in which some of these strings represent the same entity.

Consider the strings  $r_1, r_2, \dots, r_n$ , where each  $r_i \in \mathcal{B}^*$  for  $i = 1, 2, \dots, n$ . Recall that the expression  $r_1r_2\dots r_n$  refers to the concatenation of these strings into a single binary string. This operation may merge the individual components in a way that makes them inseparable. In contrast, the notation  $\langle r_1, r_2, \dots, r_n \rangle$  denotes a re-encoding of the individual strings into a single string that preserves the ability to recover each original component.

Furthermore, the joint representation  $r_1 r_2 \dots r_n$  is assumed to encode a single entity, while the  $n$ -fold representation  $\langle r_1, r_2, \dots, r_n \rangle$  may encode up to  $n$  distinct entities.

The following definition extends the concept of miscoding to  $n$ -fold representations.

**Definition 3.5.1** Let  $R = (r_1, r_2, \dots, r_n) \in \mathcal{B}^* \times \mathcal{B}^* \times \dots \times \mathcal{B}^*$  be an  $n$ -fold representation. We define the *miscoding* of  $R$ , denoted by  $\mu(R)$ , as:

$$\mu(R) = \min_{(r_{e_1}^*, \dots, r_{e_n}^*) \in \mathcal{R}_{\mathcal{E}}^* \times \dots \times \mathcal{R}_{\mathcal{E}}^*} \frac{\max \{ K(\langle r_1, \dots, r_n \rangle | \langle r_{e_1}^*, \dots, r_{e_n}^* \rangle), K(\langle r_{e_1}^*, \dots, r_{e_n}^* \rangle | \langle r_1, \dots, r_n \rangle) \}}{\max \{ K(\langle r_1, \dots, r_n \rangle), K(\langle r_{e_1}^*, \dots, r_{e_n}^* \rangle) \}}$$

where  $\min^o$  indicates that the minimum is to be computed by an oracle.

The miscoding of the representation of an area falls within the range  $[0, 1]$ , as demonstrated by the following proposition.

**Proposition 3.5.1** For all known subsets  $R = (r_1, r_2, \dots, r_n) \in \mathcal{B}^* \times \mathcal{B}^* \times \dots \times \mathcal{B}^*$ , it holds that  $0 \leq \mu(R) \leq 1$ .

*Proof.* Since  $\langle r_1, r_2, \dots, r_n \rangle$  is a string in  $\mathcal{B}^*$ , we can apply Proposition 15.5.4, which guarantees that the normalized compression-based distance lies between 0 and 1. ■

By extending the concept of miscoding to cover research areas, we gain a quantitative means of evaluating the quality of representations for specific subsets of entities. This mathematical framework provides a rigorous tool to assess and correct inaccuracies, both at the level of individual entities and across broader scientific domains.

## References

Misrepresentation or inaccuracies in scientific representation have significant implications for scientific discovery, technological progress, policymaking, and other domains. However, no book or paper explicitly addresses the topic of "incorrect representations in science" from the perspective adopted in this work. Here, we decompose the problem of scientific representation into two complementary subproblems: the representation of entities and the description of those representations.

[Sup02] provides a formal foundation for scientific representation and emphasizes the role of structure-preserving mappings; his analysis is crucial

for understanding when and how representations fail to capture relevant features of phenomena. [Van80] introduces the concept of "constructive empiricism" and highlights the model-dependent nature of scientific representation. [LSW13] offers an ethnographic study of how scientific representations are socially constructed—and, at times, distorted—in scientific practice.

Although the specific topic of incorrect representations has not been directly examined in this way, various researchers have addressed related concerns indirectly. Their discussions often focus on issues such as scientific fraud, the replication crisis, and the use of incorrect or misleading models. For example, [Ioa05] presents a widely cited empirical and philosophical analysis of how methodological biases, poor model design, and selective reporting contribute to unreliable or misleading scientific results.





## 4. Inaccuracy

*A little inaccuracy sometimes saves tons of explanations.*

Saki

In Section 2.5, we introduced the notion of a description, or model, of an entity as a computer program. When executed, this program reproduces one of the representations encoding the entity in question. More precisely, a description  $d$  for a representation  $r$  of an entity  $e$  is a Turing machine that produces the string  $r$  when interpreted by a universal Turing machine  $\delta$ . However, due to our typically incomplete understanding of the entity  $e$ , the actual output of the description, denoted as  $r' = \delta(d)$ , will generally resemble but not exactly match  $r$ .

In this chapter, we investigate the error introduced by flawed models, specifically, how closely the output  $r'$  approximates the intended representation  $r$ . We refer to this type of error as the *inaccuracy* of the description  $d$ .

Inaccuracy constitutes the second metric for assessing our understanding of a research entity. The underlying idea is that the more accurate our model, the closer  $r'$  is to  $r$ , and thus the better our understanding of the entity. Formally, the inaccuracy of a description  $d$  is defined as the normalized

information distance between the original representation  $r$  and the output representation  $r'$  produced by the description. That is, inaccuracy measures the length of the shortest computer program needed to transform the erroneous output  $r'$  into the correct representation  $r$ .

Inaccuracy evaluates how well the output of a description aligns with the selected representation encoding the entity. However, as discussed in the preceding chapter, the representation itself may be flawed. Inaccuracy focuses exclusively on the description  $d$ , without accounting for potential miscoding in the representation  $r$ . Moreover, although its computation does not require an oracle, inaccuracy cannot always be calculated exactly in practice; instead, it must often be estimated—a topic we will address in Part II of this book.

In this chapter, we formally define inaccuracy and examine its key properties. We also analyze how inaccuracy behaves when conditional descriptions are used in place of unconditional ones. Finally, we extend the notion of inaccuracy from individual entities to entire research areas.

This investigation is not purely theoretical, it has significant practical relevance. Accurate models are essential across a wide range of domains, from climate prediction to the development of artificial intelligence systems. Understanding and quantifying inaccuracy can thus lead to better models, ultimately improving our ability to make reliable predictions and informed decisions.

## 4.1 Inaccuracy

In the process of studying an entity  $e \in \mathcal{E}$  through a representation  $r \in \mathcal{R}_e$ , we may encounter situations in which our proposed description  $d$  fails to accurately produce  $r$ . That is,  $d \notin \mathcal{D}_r$  (see Definition 2.5.2). In such cases, when the universal Turing machine  $\delta$  receives  $d$  as input, it produces a string  $r'$  that differs from the original representation  $r$ .

Intuitively, one might say that  $d$  is an inaccurate description of the entity  $e$ . However, because descriptions refer to entities only indirectly, via representations, our formal notion of inaccuracy must be defined in terms of the representation, not the entity itself. Furthermore, we must account for the possibility that the representation  $r$  is itself flawed, as previously discussed through the concept of miscoding.

With these considerations in mind, we introduce the following definition of an inaccurate description.

**Definition 4.1.1** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d \in \mathcal{D}$  be a description, where  $d = \langle TM, a \rangle$ . If the output of  $TM(a)$  is a string  $r'$  such that  $r \neq r'$ , we say that  $d$  is an *inaccurate* description for  $r$ .

Our proposed description  $d$  may fall outside the set of valid descriptions  $\mathcal{D}_r$  for  $r$  (indicating positive inaccuracy), and the representation  $r$  may not belong to the set of valid representations  $\mathcal{R}_e^*$  for the entity  $e$  (indicating positive miscoding).

When a description is inaccurate, we aim to quantify the degree of inaccuracy. Within the computational framework, a natural approach is to measure how difficult it is to transform the incorrect representation  $r'$ , obtained by executing  $d$  on the universal Turing machine, into the original representation  $r$ . This difficulty is captured by the normalized information distance between  $r'$  and  $r$ .

**Definition 4.1.2 — Inaccuracy.** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d \in \mathcal{D}$  be a description, where  $d = \langle TM, a \rangle$ . We define the *inaccuracy* of the description  $d$  with respect to the representation  $r$ , denoted by  $\iota(d, r)$ , as:

$$\iota(d, r) = \frac{\max \{K(r \mid \delta(d)), K(\delta(d) \mid r)\}}{\max \{K(r), K(\delta(d))\}}$$

The use of a relative measure of inaccuracy, rather than an absolute one, enables meaningful comparisons between inaccuracies across different descriptions of the same representation, as well as between descriptions of different representations.

Similar to miscoding (see Definition 3.1.1), inaccuracy is computed using a bidirectional approach: we calculate the length of the shortest computer program that can generate the correct representation  $r$  from the erroneous one  $r'$ , and vice versa, that is, the shortest program that can generate  $r'$  from  $r$ . In essence, a valid description should produce a representation that contains all the necessary information to reconstruct the intended entity, while excluding any erroneous or irrelevant content.

■ **Example 4.1** Inaccuracy primarily concerns the difficulty of correcting the *output* of a description, that is, the result produced by a computable model, rather than the difficulty of modifying the description itself.

For example, suppose we have a dataset generated by a system that is perfectly described by a quadratic function, but we choose a linear function as our model. In this case, inaccuracy evaluates how different the predicted dataset (produced by the linear model) is from the original quadratic dataset. It does not measure how difficult it is to transform the incorrect linear model into the correct quadratic one.

In this sense, if the original dataset consists of 10 points, a polynomial of degree ten that perfectly fits the data would also yield an inaccuracy of zero. Determining which model is preferable, the quadratic model with

zero inaccuracy or the more complex degree-ten polynomial, also with zero inaccuracy, is a matter addressed by the surfeit metric (see Chapter 5). ■

Given its basis in Kolmogorov complexity, inaccuracy is a quantity that, in general, cannot be computed exactly and must instead be approximated. The method used to approximate inaccuracy depends on the specific characteristics of the entities under study and the nature of their representations.

Conveniently, the inaccuracy of a description always falls within the range  $[0, 1]$ , as established by the following proposition.

**Proposition 4.1.1** For all representations  $r \in \mathcal{B}^*$  and all descriptions  $d \in \mathcal{D}$ , it holds that  $0 \leq i(d, r) \leq 1$ .

*Proof.* This follows from the general inequality

$$0 \leq \frac{\max\{K(x | y), K(y | x)\}}{\max\{K(x), K(y)\}} \leq 1$$

for all  $x, y \in \mathcal{B}^*$ , as stated in Proposition 15.5.4. ■

The proposition above applies to all pairs of descriptions  $d$  and representations  $r$ , even in cases where  $d$  is not intended to model  $r$ . In such instances, the inaccuracy  $i(d, r)$  will typically be close to one.

Inaccuracy is exactly zero if and only if the description  $d$  is one of the valid descriptions of the representation  $r$ .

**Proposition 4.1.2** Given a description  $d \in \mathcal{D}$  and a representation  $r \in \mathcal{B}^*$ , we have that  $i(d, r) = 0$  if and only if  $d \in \mathcal{D}_r$ , i.e.,  $d$  is a valid description of  $r$ .

*Proof.* If  $d \in \mathcal{D}_r$ , then by definition,  $\delta(d) = r$ . Consequently, we have  $K(r | \delta(d)) = K(\delta(d) | r) = 0$ , and thus  $i(d, r) = 0$ .

Conversely, suppose  $i(d, r) = 0$ . Then,

$$\max\{K(r | \delta(d)), K(\delta(d) | r)\} = 0,$$

which implies that both conditional complexities are zero. Therefore,  $r = \delta(d)$ , and it follows that  $d \in \mathcal{D}_r$ . ■

Finally, given two representations  $r$  and  $s$ , we may also be interested in evaluating the inaccuracy of a description  $d$  when it is used to describe their joint representation  $rs$ . Since we require that  $rs$  is itself a valid representation, the extension of the inaccuracy concept to joint representations is straightforward and does not require a new definition:

$$i(d, rs) = \frac{\max\{K(rs | \delta(d)), K(\delta(d) | rs)\}}{\max\{K(rs), K(\delta(d))\}}$$

As a direct consequence of Proposition 4.5.1, for any representations  $r, s \in \mathcal{B}^*$  and any description  $d \in \mathcal{D}$ , we have:

$$0 \leq \iota(d, rs) \leq 1.$$

## 4.2 Conditional Inaccuracy

In this section, we delve deeper into the concept of inaccuracy by considering its application to conditional descriptions. Specifically, we explore the inaccuracy of a description when evaluated in conjunction with pre-existing background knowledge, a notion we term *conditional inaccuracy*. As we will see, the inaccuracy of a conditional description can never exceed that of its unconditional counterpart; at worst, it remains unchanged. This property makes conditional inaccuracy a useful tool for evaluating new concepts or models and for assessing their explanatory power with respect to the entity of interest.

In Definition 2.7.1, we introduced the concept of a conditional description  $d$  for a representation  $r$ , given an arbitrary background string  $s$ . This is denoted by  $d | s$  and defined as the self-delimiting concatenated string  $\langle d, s \rangle$ , where  $d = \langle TM, a \rangle$  and  $TM(\langle a, s \rangle) = r$ . If  $TM(\langle a, s \rangle) = r'$  with  $r \neq r'$ , then  $d | s$  is referred to as an *inaccurate* conditional description of  $r$ .

It is important to note that  $d | s$  must be defined for all possible background strings  $s$ . Additionally, we refer to the case in which  $s$  is the empty string, denoted  $d | \lambda$ , as the *unconditional* version of the description.

Building on the concept of inaccuracy introduced in Definition 4.1.2, we now define the notion of *conditional inaccuracy* to capture the error introduced when using an inaccurate conditional description.

**Definition 4.2.1** Let  $r \in \mathcal{B}^*$  be a representation,  $s \in \mathcal{B}^*$  a background string, and  $d | s$  an inaccurate conditional description. We define the *conditional inaccuracy* of the description  $d$  for the representation  $r$  given the string  $s$ , denoted by  $\iota(d | s, r)$ , as:

$$\iota(d | s, r) = \frac{\max \{K(r | \delta(d | s)), K(\delta(d | s) | r)\}}{\max \{K(r), K(\delta(d | s))\}}$$

Conditional inaccuracy is thus defined as the normalized information distance between the original representation  $r$  and the string produced by the conditional description  $d | s$ .

As a normalized measure, the conditional inaccuracy of a description lies within the interval  $[0, 1]$ .

**Proposition 4.2.1** Let  $r \in \mathcal{B}^*$  be a representation,  $s \in \mathcal{B}^*$  a string, and  $d | s$  a conditional description of  $r$  given  $s$ . Then  $0 \leq \iota(d | s, r) \leq 1$ .

*Proof.* This follows directly from the general inequality:

$$0 \leq \frac{\max\{K(x | y), K(y | x)\}}{\max\{K(x), K(y)\}} \leq 1$$

for all  $x, y \in \mathcal{B}^*$ , as established in Proposition 15.5.4. ■

The conditional inaccuracy assumes a value of zero if and only if the conditional description  $d | s$  is a valid model that correctly produces the representation  $r$ .

**Proposition 4.2.2** Let  $r \in \mathcal{B}^*$  be a representation,  $s \in \mathcal{B}^*$  a string, and  $d | s$  a conditional description of  $r$  given  $s$ , where  $d = \langle TM, a \rangle$ . Then  $\iota(d | s, r) = 0$  if and only if  $TM(\langle a, s \rangle) = r$ .

*Proof.* If  $TM(\langle a, s \rangle) = r$ , then  $\delta(d | s) = r$ , which implies that

$$K(r | \delta(d | s)) = K(\delta(d | s) | r) = 0.$$

Hence,  $\iota(d | s, r) = 0$ . Conversely, if  $\iota(d | s, r) = 0$ , then

$$\max\{K(r | \delta(d | s)), K(\delta(d | s) | r)\} = 0,$$

which implies both conditional complexities are zero. Therefore,  $\delta(d | s) = r$ , which means  $TM(\langle a, s \rangle) = r$ . ■

Incorporating established prior knowledge into research does not increase the inaccuracy of a description. If this background knowledge is relevant to the representation being described, the oracle will use it accordingly. Conversely, if the prior knowledge is irrelevant, the oracle will simply disregard it. The following theorem formalizes this idea.

**Theorem 4.2.3** Let  $r \in \mathcal{B}^*$  be a representation, and  $d \in \mathcal{D}$  a conditional description of  $r$ . Then

$$\iota(d | s, r) \leq \iota(d, r)$$

for all strings  $s \in \mathcal{B}^*$ .

*Proof.* Since  $\iota(d, r)$  is equivalent to  $\iota(d | \lambda, r)$ , we need to prove that

$$\frac{\max\{K(r | \delta(d | s)), K(\delta(d | s) | r)\}}{\max\{K(r), K(\delta(d | s))\}} \leq \frac{\max\{K(r | \delta(d | \lambda)), K(\delta(d | \lambda) | r)\}}{\max\{K(r), K(\delta(d | \lambda))\}}.$$

This inequality follows from the fact that

$$K(r | \langle \delta(d), s \rangle) \leq K(r | \delta(d)),$$

as shown in Proposition 15.4.3. ■

Theorem 4.2.3 represents a foundational result in the theory of nescience. It establishes the basis for developing a robust methodology aimed at deepening our understanding (i.e., reducing inaccuracy) of a research entity. In practical applications, our primary focus will typically be on prior knowledge directly related to the subject of study. However, the core insight of Theorem 4.2.3 is that incorporating concepts from seemingly unrelated domains will not compromise the accuracy of our investigation. This theorem becomes particularly powerful when such exploratory processes are automated (see Chapter 10).

■ **Example 4.2** The P vs. NP problem stands as one of the most significant unresolved questions in computer science. It asks whether every problem whose solution can be verified in polynomial time (class NP) can also be solved in polynomial time (class P). The relationship between these two complexity classes remains unsolved. Constructing a comprehensive, self-contained solution to this problem in a formal language is an immense challenge. However, leveraging relevant prior knowledge can significantly reduce the complexity of the required description. For instance, insights from Algorithm Theory, which deals with the classification and efficiency of algorithms, and from Formal Language Theory, which addresses the structure of computational problems (e.g., regular and context-free languages) and highlights the role of Turing machines, can be instrumental. Drawing upon such established knowledge may not only simplify our descriptions but also facilitate a deeper understanding and potentially contribute to resolving the P vs. NP problem. ■

Finally, given two representations  $r$  and  $t$ , the formalization of the concept of conditional inaccuracy, when applied to the joint representation  $rt$ , is straightforward and does not require a new definition:

$$\iota(d | s) = \frac{\max\{K(rt | \delta(d | s)), K(\delta(d | s) | rt)\}}{\max\{K(rt), K(\delta(d | s))\}}$$

As a normalized measure,  $\iota(d | s)$  always takes a value in the interval  $[0, 1]$ .

## 4.3 Decreasing Inaccuracy

Our objective is to reduce the inaccuracy of the current description  $d_1$ , thereby improving our understanding of the original entity. This improvement may involve either modifying  $d_1$  to correct or eliminate its inaccuracies, or developing a completely new description based on a different approach to modeling the entity. In either case, the result is a new description  $d_2$ . In this section, we aim to analyze how the introduction of a new description  $d_2$  affects the inaccuracy compared to the original description  $d_1$ .

**Definition 4.3.1** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d_1, d_2 \in \mathcal{D}$  be two descriptions. We define the *variation of inaccuracy* between the descriptions  $d_1$  and  $d_2$ , with respect to  $r$ , denoted by  $\Delta_t^a(d_1, d_2, r)$ , as:

$$\Delta_t^a(d_1, d_2, r) = \iota(d_1, r) - \iota(d_2, r)$$

Since inaccuracy is bounded between 0 and 1, the maximum possible variation in inaccuracy is  $\pm 1$ . A positive value of  $\Delta_t$  indicates that description  $d_2$  is preferable to  $d_1$  in terms of accuracy. Conversely, a negative value suggests that  $d_1$  is more accurate than  $d_2$ . It is important to note that the new description may also introduce a substantial increase in surfeit, potentially outweighing the improvement in inaccuracy. For a detailed discussion of surfeit, refer to Chapter 5, and for an explanation of how inaccuracy and surfeit combine into the unified metric of nescience, see Chapter 6.

We can also introduce a relative measure of the variation in inaccuracy, when moving from description  $d_1$  to  $d_2$

**Definition 4.3.2** Let  $r \in \mathcal{B}^*$  be a representation, and  $d_1, d_2 \in \mathcal{D}$  be two descriptions. We define the *relative variation of inaccuracy* between descriptions  $d_1$  and  $d_2$ , denoted by  $\Delta_t^r(d_1, d_2, r)$ , as:

$$\Delta_t^r(d_1, d_2, r) = \frac{\iota(d_1, r) - \iota(d_2, r)}{\iota(d_1, r)}$$

provided that  $\iota(d_1, r) \neq 0$ .

A value of 0 indicates no change, while a value of 1 corresponds to a complete elimination of inaccuracy. Negative values, on the other hand, signify an increase in inaccuracy, indicating that the new description  $d_2$  is less accurate than the original  $d_1$ . Note that the relative variation can be arbitrarily negative, diverging to  $-\infty$  as  $\iota(d_2, r)$  increases and  $\iota(d_1, r)$  approaches zero.

As inaccuracy approaches zero, relative variations become increasingly unstable. A small absolute change can produce a large relative variation if the initial inaccuracy is very low. For instance, if  $\iota(d_1, r) = 0.1$  and the inaccuracy decreases by 0.05, this corresponds to a relative improvement of 50%. In contrast, if  $\iota(d_1, r) = 0.9$  and the same absolute improvement occurs, the relative reduction is only about 5.6%. Both absolute and relative variations are essential for evaluating the significance and magnitude of improvements.

An alternative strategy for reducing uncertainty about an entity involves modifying its representation rather than altering its description. While such a change might increase the miscoding of the entity, the potential reduction in inaccuracy could outweigh this drawback.

**Definition 4.3.3** Let  $r_1, r_2 \in \mathcal{B}^*$  be two representations, and let  $d \in \mathcal{D}$  be a description. We define the *variation of inaccuracy* between representations  $r_1$  and  $r_2$ , denoted by  $\Delta_t^a(d, r_1, r_2)$ , as:

$$\Delta_t^a(d, r_1, r_2) = \iota(d, r_1) - \iota(d, r_2)$$

As before, since inaccuracy is bounded between 0 and 1, the maximum possible variation is 1. A positive value of  $\Delta_t^a$  indicates a preference for representation  $r_2$  over  $r_1$  with respect to the fixed description  $d$ . Conversely, a negative value suggests that  $r_1$  is more accurate than  $r_2$ . This comparison is based solely on inaccuracy, as miscoding is not considered here. Additionally, since the description remains unchanged, there is no risk of variation in surfeit.

Finally, we can also introduce a relative variation of inaccuracy with respect to a change in representation

**Definition 4.3.4** Let  $r_1, r_2 \in \mathcal{B}^*$  be two representations, and  $d \in \mathcal{D}$  be a description. We define the *relative variation of inaccuracy* of the representations  $r_1, r_2$ , denoted by  $\Delta_t^r(d, r_1, r_2)$ , as:

$$\Delta_t^r(d, r_1, r_2) = \frac{\iota(d, r_1) - \iota(d, r_2)}{\iota(d, r_1)}$$

provided that  $\iota(d, r_1) \neq 0$ .

This quantity measures the proportional reduction in inaccuracy resulting from replacing representation  $r_1$  with  $r_2$ , while keeping the description fixed. A value of 0 indicates no change in inaccuracy, and a value of 1 corresponds to a complete elimination of inaccuracy. However,  $\Delta_t^r$  can also take negative values, potentially diverging to  $-\infty$ , when the new representation  $r_2$  performs worse than  $r_1$ . As the inaccuracy of the initial representation  $r_1$  approaches zero, even minor absolute changes in inaccuracy can lead to large swings in the relative variation, making it increasingly volatile.

## 4.4 Inaccuracy-Miscoding Rate of Change

In the preceding section, we examined how the inaccuracy of a representation can be reduced by selecting a different description. We also explored an alternative strategy in which inaccuracy is minimized not by modifying the description, but by changing the representation itself. In this section, we turn our attention to a more general approach for reducing the inaccuracy associated with an entity. Rather than altering the description or the representation in isolation, it may be more effective to modify both simultaneously.

The balance between the amount of miscoding we are willing to accept in order to achieve a reduction in inaccuracy is referred to as the *miscoding-inaccuracy trade-off*. For a broader discussion of trade-offs in multi-objective optimization, refer to Section 16.5.2.

**Definition 4.4.1** Let  $e \in \mathcal{E}$  be an entity, and  $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{R}_e \times \mathcal{D}$  be two hypotheses, with  $\mathbf{x}_1 = (r_1, d_1)$  and  $\mathbf{x}_2 = (r_2, d_2)$ . We define the *rate of change between the inaccuracy and the miscoding* of the hypothesis  $\mathbf{x}_1, \mathbf{x}_2$ , denoted by  $\Delta_{l\mu}(\mathbf{x}_1, \mathbf{x}_2)$  as:

$$\Delta_{l\mu}(\mathbf{x}_1, \mathbf{x}_2) = \frac{\iota(d_2, r_2) - \iota(d_1, r_1)}{\mu(r_2) - \mu(r_1)}$$

provided that  $\mu(r_2) - \mu(r_1) \neq 0$ .

The ratio  $\Delta_{l\mu}$  represents the rate of change between inaccuracy and miscoding when transitioning from the first hypothesis to the second. A positive value of  $\Delta_{l\mu}$  implies that both quantities, miscoding and inaccuracy, either decrease (which is desirable) or increase (which is undesirable). The interpretation becomes more nuanced when  $\Delta_{l\mu}$  is negative, indicating that one of the quantities decreases while the other increases. In such cases, two scenarios must be considered:

- (i) Inaccuracy decreases and miscoding increases, we aim for  $\Delta_{l\mu}(\mathbf{x}_1, \mathbf{x}_2) < M$ , where  $M < -1$ , thereby ensuring that the reduction in inaccuracy compensates for the increase in miscoding.
- (ii) Inaccuracy increases and miscoding decreases, we aim for  $\Delta_{l\mu}(\mathbf{x}_1, \mathbf{x}_2) > M$ , where  $-1 < M < 0$ , thereby ensuring that the reduction in miscoding justifies the increase in inaccuracy.

In both cases, caution is warranted when the change in miscoding is small, as it can disproportionately affect the ratio and potentially lead to misleading conclusions.

■ **Example 4.3** Let  $\mathbf{x}_1$  and  $\mathbf{x}_2$  be two hypotheses. For  $\mathbf{x}_1$ , the inaccuracy  $\iota(d_1, r_1)$  is 0.40, and the miscoding  $\mu(r_1)$  is 0.15. For  $\mathbf{x}_2$ , the inaccuracy  $\iota(d_2, r_2)$  is 0.20 (a decrease from 0.40), and the miscoding  $\mu(r_2)$  is 0.25 (an increase from 0.15). Using the definition of the rate of change, we compute:

$$\Delta_{l\mu}(\mathbf{x}_1, \mathbf{x}_2) = \frac{0.40 - 0.20}{0.15 - 0.25} = \frac{0.20}{-0.10} = -2$$

In this case, transitioning from hypothesis  $\mathbf{x}_1$  to  $\mathbf{x}_2$  results in a decrease of inaccuracy by 0.20 units and an increase in miscoding by 0.10 units. The rate of change is  $-2$ . ■

Having a very small change in miscoding can significantly amplify the

value of the rate of change, potentially giving the misleading impression that inaccuracy and miscoding are varying at an extreme rate, even when the actual changes are minor. This phenomenon is illustrated in the following example.

■ **Example 4.4** Let  $\mathbf{x}_1$  be a hypothesis with an inaccuracy  $\iota(d_1, r_1)$  of 0.35 and a miscoding  $\mu(r_1)$  of 0.20. Let  $\mathbf{x}_2$  be a second hypothesis with an inaccuracy  $\iota(d_2, r_2)$  of 0.30 (a slight decrease from 0.35) and a miscoding  $\mu(r_2)$  of 0.2001 (a very small increase from 0.20). Applying the definition of the rate of change:

$$\Delta_{\iota\mu}(\mathbf{x}_1, \mathbf{x}_2) = \frac{0.35 - 0.30}{0.20 - 0.2001} = \frac{0.05}{-0.0001} = -500$$

The rate of change is  $-500$ , which may misleadingly suggest a dramatic shift. In reality, the inaccuracy only decreased by 0.05 units, and the miscoding increased by a negligible 0.0001 units. The extremely small denominator inflates the result, making the change appear far more significant than it actually is. ■

As we attempt to optimize both inaccuracy and miscoding, we inevitably encounter configurations (hypotheses) where improving one objective necessitates compromising the other. The set of such "best trade-off" configurations constitutes the Pareto frontier (see Section 16.5). Points on the Pareto frontier are said to be Pareto optimal because any attempt to improve one objective leads to a deterioration in the other.

**Definition 4.4.2** Let  $e \in \mathcal{E}$  be an entity, and let  $\mathbf{x} = (r, d) \in \mathcal{R}_e \times \mathcal{D}$  be a hypothesis. The hypothesis  $\mathbf{x}$  is said to be a Pareto point with respect to inaccuracy  $\iota$  and miscoding  $\mu$  if there does not exist another hypothesis  $\mathbf{x}' = (r', d')$  such that:

- 1  $\iota(d', r') \leq \iota(d, r)$  and  $\mu(r') \leq \mu(r)$ , and
- 2  $\iota(d', r') < \iota(d, r)$  or  $\mu(r') < \mu(r)$ .

In simpler terms, a hypothesis  $\mathbf{x}$  is Pareto optimal if: (i) no other hypothesis is better in both inaccuracy and miscoding, and (ii) any improvement in one metric necessarily results in a worsening of the other.

The rate of change  $\Delta_{\iota\mu}$  between any two Pareto optimal points provides insight into how the trade-off between inaccuracy and miscoding evolves along the Pareto frontier. If decision-makers are more sensitive to changes in inaccuracy than to miscoding, they may prefer configurations with a less negative  $\Delta_{\iota\mu}$ . Conversely, if miscoding is of greater concern, they may accept solutions where  $\Delta_{\iota\mu}$  indicates a larger increase in inaccuracy in exchange for a smaller gain in miscoding.

## 4.5 Inaccuracy of Areas

An area  $\mathcal{A}$  (see Section 2.8) is a subset  $\mathcal{A} \subset \mathcal{E}$  of entities that are related or share a common property. The concept of inaccuracy can be extended to research areas in order to quantitatively measure the effort required to correct an inaccurate description of an area.

Given the strings  $r_1, r_2, \dots, r_n$ , where each  $r_i \in \mathcal{B}^*$  for  $i = 1, 2, \dots, n$ , recall that we use  $\langle r_1, r_2, \dots, r_n \rangle$  to denote a self-delimited encoding of the individual strings  $r_i$  into a unified string, such that the original components can be fully recovered.

The following definition extends the concept of inaccuracy to research areas.

**Definition 4.5.1** Let  $\mathcal{A} \subset \mathcal{E}$  be an area with a known subset  $\hat{\mathcal{A}} = r_1, r_2, \dots, r_n$ , and let  $d \in \mathcal{D}$  be a description. We define the *inaccuracy of the area* given the description  $d$ , denoted by  $\iota(d, \hat{\mathcal{A}})$ , as:

$$\iota(d, \hat{\mathcal{A}}) = \frac{\max\{K(\langle r_1, r_2, \dots, r_n \rangle \mid \delta(d)), K(\delta(d) \mid \langle r_1, r_2, \dots, r_n \rangle)\}}{\max\{K(\langle r_1, r_2, \dots, r_n \rangle), K(\delta(d))\}}$$

The inaccuracy of a description for an area falls within the interval from 0 to 1, as established by the following proposition.

**Proposition 4.5.1** For all known subsets  $\hat{\mathcal{A}} = r_1, r_2, \dots, r_n$  and all descriptions  $d \in \mathcal{D}$ , we have that  $0 \leq \iota(d, \hat{\mathcal{A}}) \leq 1$ .

*Proof.* The result follows directly from the fact that  $\langle r_1, r_2, \dots, r_n \rangle$  is a string in  $\mathcal{B}^*$ , and from Proposition 15.5.4. ■

By extending the concept of inaccuracy to cover areas, we can quantitatively evaluate the quality of a description for a specific subset of entities. This mathematical framework provides a rigorous tool for assessing and correcting inaccuracies not only at the level of individual entities but also across broader research domains.

## References

A good introduction to the study of uncertainties (i.e., error analysis in models) in science (particularly in physics, chemistry, and engineering) is the best-selling textbook by Taylor [TAY22], which also features the same image of a crashed train used in the introduction to this chapter. Another excellent entry-level reference on error analysis, aimed at undergraduate students in science and technology, is the book by Hughes and Hase [HH10].

From a more philosophical perspective, Popper's work [Pop14] is highly influential. In it, he introduces the concept of falsifiability, asserting that for a theory to be regarded as scientific, it must be testable and subject to refutation.





## 5. Surfeit

*Everything should be made as simple as possible,  
but not simpler.*  
Albert Einstein

Surfeit is the final metric we introduce to quantitatively assess our understanding of a research entity. It measures the presence of superfluous symbols in the description used to model that entity. Intuitively, our lack of knowledge is often reflected in the length (i.e., number of symbols) of our current description. Lengthy descriptions tend to include erroneous or redundant elements. As our understanding of the subject improves, we should be able to identify and remove these unnecessary symbols, resulting in a more concise and accurate description.

We define the surfeit of a description for an entity as the difference in length between the given description and the optimal (i.e., shortest) one for that entity. Within the framework of the theory of nescience, we assume that the pinnacle of knowledge about an entity, its perfect description, is represented by the shortest description capable of fully reconstructing the entity's representation. This notion of perfection relies on both the validity of the representation and the accuracy of the description.

The length of the most concise description of an entity is determined by the Kolmogorov complexity of a representation of that entity. In practical scenarios, given that our knowledge of entities is typically incomplete, the most concise possible description remains unknown. Moreover, as previously discussed, Kolmogorov complexity is not computable in general. Consequently, surfeit is a metric that must be approximated in practice.

If we could construct a perfect description of an entity, it would necessarily be a random string; otherwise, it would contain redundant elements that could be eliminated. Within the framework of the theory of nescience, attaining perfect knowledge corresponds to reaching a state of randomness. This inherent randomness defines a boundary on the depth of understanding achievable for a given research topic. However, rather than constituting a limitation, recognizing and understanding this boundary opens new opportunities in both science and technology. For example, by assessing how far our current description deviates from a random string, we can estimate our proximity to realizing a perfect description.

In this chapter, we formally introduce the concept of surfeit and examine its properties, including conditional surfeit. We will also present the notion of redundancy as a practical approximation of surfeit. Strategies for reducing both surfeit and redundancy will be discussed, as well as the relationship between reductions in surfeit and changes in inaccuracy or miscoding. Finally, we extend the concept of surfeit to support the analysis of entire research areas.

## 5.1 Surfeit

Given the length of a description of a representation for an entity and the length of its shortest possible description, we can introduce a relative measure to quantify the unnecessary effort involved in explaining the entity using that particular description. We call this quantity *surfeit*. Surfeit is a key component of our definition of nescience, as it reflects the degree to which our current understanding of the research entity includes superfluous or redundant information.

**Definition 5.1.1 — Surfeit.** Given a representation  $r \in \mathcal{B}^*$ , and a description  $d \in \mathcal{D}$  for  $r$ , we define the *surfeit of the description  $d$  for the representation  $r$* , denoted by  $\sigma(d, r)$ , as

$$\sigma(d, r) = \frac{|l(d) - K(r)|}{l(d)}$$

For most descriptions, the length of the description  $l(d)$  for  $r$  will exceed the length of its shortest possible description  $K(r)$ . Intuitively, the less we

know about an entity, the longer our description tends to be. As our understanding of the entity improves, we should be able to remove all redundant elements from the description. There may also be cases where the description is shorter than the optimal one. In such instances, the description oversimplifies the problem, which can be equally problematic. This justifies the use of the absolute value  $|l(d) - K(r)|$  rather than simply  $l(d) - K(r)$ . Naturally, the current description might also be inaccurate, or the representation may be invalid. These concerns are addressed separately by the metrics of inaccuracy and miscoding.

In our definition of surfeit, we chose a relative measure rather than an absolute one (i.e.,  $|l(d) - K(r)|$ ) because we aim to compare the surfeit not only among different models of the same entity but also across models of different entities. We prefer to use  $K(r)$  instead of the equivalent  $l(r^*)$  to maintain consistency with the definition of inaccuracy provided in Section 4.1.

■ **Example 5.1** In a practical machine learning scenario, consider the task of classifying images of cats and dogs, where the representation consists of a set of training images. An initial complex model, burdened with excessive parameters and irrelevant features, can be seen as a lengthy "description" of the problem. The optimal model, by contrast, achieves a balance between accuracy and simplicity. Surfeit quantifies the excess complexity of the initial model relative to this optimal one. It measures the "extra" components that are not essential for accurate classification. A high surfeit indicates that the model is overly complicated, highlighting the need for simplification to improve both efficiency and generalization, a fundamental principle in machine learning for building models that perform well on unseen data. ■

The surfeit of a description is a number between 0 and 1.

**Proposition 5.1.1** Let  $r \in \mathcal{B}^*$  be a representation, and  $d \in \mathcal{D}_r^*$  one of its valid descriptions, then we have that  $0 \leq \sigma(d, r) \leq 1$ .

*Proof.* The numerator  $|l(d) - K(r)|$  is non-negative for all  $d$  and  $r$ , and the denominator  $l(d) > 0$  (as descriptions must be non-empty). Hence, the entire expression is non-negative  $\sigma(d, r) \geq 0$ .

To show that  $\sigma(d, r) \leq 1$ , observe that:

$$|l(d) - K(r)| = l(d) - K(r) \leq l(d).$$

since  $d$  is a valid description of  $r$ , and so  $l(d) \geq K(r)$ . Thus:

$$\sigma(d, r) = \frac{|l(d) - K(r)|}{l(d)} \leq \frac{l(d)}{l(d)} = 1.$$



The surfeit is zero when the length of the description  $l(d)$  equals the Kolmogorov complexity  $K(r)$  of the representation  $r$  of the entity, indicating that the description has achieved theoretical conciseness.

**Proposition 5.1.2** Let  $r \in \mathcal{B}^*$  be a representation, and  $d \in \mathcal{D}_r^*$  a valid description for  $r$ , then we have that  $\sigma(d, r) = 0$  if and only if  $l(d) = l(d^*)$ .

*Proof.* Assume  $l(d) = l(d^*) = K(r)$ . Then:

$$\sigma(d, r) = \frac{|l(d) - K(r)|}{l(d)} = \frac{|K(r) - K(r)|}{K(r)} = \frac{0}{K(r)} = 0.$$

Assume  $\sigma(d, r) = 0$ . Then:

$$\frac{|l(d) - K(r)|}{l(d)} = 0.$$

This implies that the numerator must be zero, i.e.:

$$|l(d) - K(r)| = 0 \Rightarrow l(d) = K(r) = l(d^*).$$

■

It is important to distinguish between conciseness and correctness. A surfeit of zero indicates that the description is as brief as theoretically possible, but not necessarily accurate. The accuracy of a description is evaluated using the inaccuracy metric. Thus, while a zero surfeit reflects the optimal compactness of a description, its correctness and reliability must be assessed separately through inaccuracy. Together, surfeit and inaccuracy provide a more complete assessment of the description's efficiency and validity.

The following example underscores the need to balance minimal surfeit with low inaccuracy in order to develop models that are both efficient and reliable.

■ **Example 5.2** In a machine learning scenario, consider a model designed to classify emails as spam or not. Suppose this model achieves a surfeit of zero, indicating optimal conciseness with no superfluous elements in its description. However, despite this streamlined complexity, the model exhibits high inaccuracy, frequently misclassifying emails. This illustrates the distinction between surfeit and inaccuracy: while the model is theoretically as concise as possible, reflected in its zero surfeit, its practical utility is compromised by incorrect classifications, as captured by the inaccuracy metric.

■

It is important to note that, in theory, more than one description may yield a surfeit of zero for the same representation. This occurs when two distinct, yet incompressible, descriptions have exactly the same length. In such cases, both descriptions are equally concise and minimal, meaning that neither contains redundant information that could be further compressed. Since surfeit measures the relative difference between the actual length of a description and the shortest possible one, any description whose length equals the Kolmogorov complexity  $K(r)$  of the representation will result in  $\sigma(d, r) = 0$ . Therefore, while the perfect description is often referred to as unique, from a theoretical standpoint, multiple descriptions of equal minimal length and incompressibility may coexist.

## 5.2 Redundancy

Our definition of surfeit compares the length of a description with the Kolmogorov complexity of the representation, not with the Kolmogorov complexity of the description itself (i.e.,  $K(d)$ ). In other words, surfeit is not a measure of the redundancy within a description. It is possible to construct an incompressible description (i.e., one without any internal redundancy) that is not the shortest possible description of the representation it refers to (see Example 2.14). Such a description would not be redundant in the traditional sense, yet it would still exhibit surfeit according to the theory of nescience.

Moreover, it might happen that the description  $d$  under consideration does not describe the representation  $r$ ; in other words,  $d \notin \mathcal{D}_r$ . For practical applications, it is useful to introduce an alternative, and arguably weaker, notion of redundancy that applies solely to the description itself, independently of any particular representation.

**Definition 5.2.1 — Redundancy.** Given a description  $d \in \mathcal{D}$ , we define the *redundancy* of the description  $d$ , denoted by  $\rho(d)$ , as

$$\rho(d) = 1 - \frac{K(d)}{l(d)}$$

The redundancy of a description  $d$  is a quantity related to the description itself, and it does not depend on the representation  $r$  being described. The following example shows that a description can have low redundancy (being incompressible) and still have high surfeit if it is longer than necessary to describe the underlying task.

■ **Example 5.3** Consider the task of computing the first  $n$  digits of  $\pi$ . Description  $d_1$  is a program that implements an algorithm to generate and print these  $n$  digits. Description  $d_2$  is a self-extracting compressed program that

simply stores the first  $n$  digits of  $\pi$  as a hard-coded string, followed by a basic routine to print them.

Description  $d_2$  does not exploit the mathematical structure of  $\pi$ , it merely reproduces the result. Therefore, relative to the representation of the task (computing  $\pi$ ), the surfeit  $\sigma(d_2, r)$  is high because  $l(d_2) \gg K(r)$ .

In contrast, description  $d_1$  is significantly shorter since it captures the generative process behind  $\pi$ . While  $d_1$  may still contain some redundancy  $\rho(d_1) > 0$ , for instance due to suboptimal coding practices, its surfeit remains low since it is near the minimal length required to describe the task effectively.

■

The redundancy of a description always falls within the range  $[0, 1]$ .

**Proposition 5.2.1** We have that  $0 \leq \rho(d) \leq 1$  for all  $d \in \mathcal{D}$ .

*Proof.* Since Kolmogorov complexity is always less than or equal to the length of the string, we have  $0 \leq K(d) \leq l(d)$ , which implies that  $0 \leq \frac{K(d)}{l(d)} \leq 1$ . Subtracting from 1 reverses the inequalities. ■

Finally, next proposition formalizes our intuition that the surfeit of a description is greater or equal than its redundancy.

**Proposition 5.2.2** Let  $r \in \mathcal{B}^*$  be a representation , and  $d \in \mathcal{D}_r^*$  one of its valid descriptions, then we have that  $\rho(d) \leq \sigma(d, r)$ .

*Proof.* Proving that  $\rho(d) \leq \sigma(d, r)$  is equivalent to prove that  $K(d) \geq K(r)$  for all  $d$ . Lets assume that there exist a  $d$  such that  $K(d) < K(r)$ , that would mean there exists a Turing machine  $\langle TM, a \rangle$  such that  $TM(a) = r$  but  $l(\langle TM, a \rangle) < K(r)$ . That is a contradiction with the fact that  $K(r)$  is the length of the shortest possible Turing machine that prints  $r$ . ■

It would be very nice if Proposition 5.2.2 applies to all possible description. Unfortunately, the proposition is true only when we deal with valid descriptions (from  $\mathcal{D}_r^*$ ).

### 5.3 Conditional Surfeit

We are interested in studying how the surfeit of a description for a representation is affected when some background knowledge is assumed. In particular, we examine the case where a description is constructed under the assumption that some prior information is already known. The surfeit of such a description is referred to as the *conditional surfeit*.

**Definition 5.3.1** Let  $r \in \mathcal{B}^*$  be a representation,  $s \in \mathcal{B}^*$  a string, and  $d \in \mathcal{D}$  a description of  $r$  given  $s$ . We define the *conditional surfeit* of the conditional description  $d_{r|s}$ , denoted by  $\sigma(d_{r|s})$ , as:

$$\sigma(d_{r|s}) = 1 - \frac{K(r|s)}{l(d_{r|s})}$$

This definition is primarily motivated by practical considerations. When we assume perfect knowledge of  $s$ , we can isolate and study the informational content of  $r$  that is not already covered by  $s$ , that is, the new knowledge introduced by  $r$  relative to  $s$ . Conditional surfeit thus allows us to quantify the conciseness of a description in light of what is already known.

Conditional surfeit, being a relative measure, is a number between 0 and 1.

**Proposition 5.3.1** Let  $r \in \mathcal{B}^*$  be a representation,  $s \in \mathcal{B}^*$  be a string, and  $d \in \mathcal{D}$  be a description of  $r$  given  $s$ . We have that  $0 \leq \sigma(d_{r|s}) \leq 1$ .

*Proof.* Given that  $l(d_{r|s}) > 0$  and that  $K(r|s) > 0$ , since they are the lengths of non-empty strings, and that  $l(d_{r|s}) \geq K(r|s)$ . ■

Intuition tell us that the surfeit of a description could only decrease if we assume the background knowledge given by the description of another topic. This is because we require that this background knowledge must be a perfect description (it presents no surfeit). However, as it was the case of joint surfeit, we have to wait until Chapter 6 to formalize this intuition.

In the same way we introduced the concept of redundancy of a description as a weaker version of the concept of surfeit, we can also introduce the concept of conditional redundancy as a weaker version of the concept of conditional surfeit.

**Definition 5.3.2** Let  $s \in \mathcal{B}^*$  be a string, and  $d \in \mathcal{D}$  be a conditional description given  $s$ . We define the *conditional surfeit* of the conditional description  $d_{t|s^*}$ , denoted by  $\rho(d_{t|s^*})$ , as:

$$\rho(d_{t|s}) = 1 - \frac{K(d_{t|s^*})}{l(d_{t|s^*})}$$

Conditional surfeit is a relative measure, and so, a number between 0 and 1.

**Proposition 5.3.2** We have that  $0 \leq \rho(d_{t|s^*}) \leq 1$  for all  $t, s$  and all  $d_{t,s}$ .

*Proof.* Given that  $K(d_{t|s^*}) \leq l(d_{t|s^*})$  we have that  $\frac{K(d_{t|s^*})}{l(d_{t|s^*})} \leq 1$  and so,  $1 -$

$\frac{K(d_{t|s^*})}{l(d_{t|s^*})} \geq 0$ . Also, since  $\frac{K(d_{t|s^*})}{l(d_{t|s^*})} > 0$  (both quantities are positive integers), we have that  $1 - \frac{K(d_{t|s^*})}{l(d_{t|s^*})} \leq 1$ . ■

Finally, we can extend our concepts of conditional surfeit and conditional redundancy to multiple, but fine, number of topics.

**Definition 5.3.3** Let  $t, s_1, s_2, \dots, s_n \in \mathcal{T}$  be a finite collection of topics, and let  $d_{t|s_1^*, s_2^*, \dots, s_n^*}$  any conditional description of  $t$  given  $s_1, s_2, \dots, s_n$ . We define the *conditional surfeit* of the description  $d_{t|s_1^*, s_2^*, \dots, s_n^*}$ , denoted by  $\sigma(d_{t|s_1^*, s_2^*, \dots, s_n^*})$ , as:

$$\sigma(d_{t|s_1^*, s_2^*, \dots, s_n^*}) = 1 - \frac{K(t | s_1^*, s_2^*, \dots, s_n^*)}{l(d_{t|s_1^*, s_2^*, \dots, s_n^*})}$$

And the *conditional redundancy* of the description  $d_{t_1, t_2, \dots, t_n}$ , denoted by  $\rho(d_{t_1, t_2, \dots, t_n})$ , as:

$$\rho(d_{t_1, t_2, \dots, t_n}) = 1 - \frac{K(d_{t_1, t_2, \dots, t_n})}{l(d_{t_1, t_2, \dots, t_n})}$$

It is easy to show that the properties of conditional surfeit and conditional redundancy apply to the case of multiple topics as well.

## 5.4 Decreasing Surfeit

Our objective is to reduce the surfeit of our current best description  $d_1$ , thereby enhancing our understanding of the original entity. This improvement may involve either refining  $d_1$  to eliminate redundant symbols or developing an entirely new description based on a different modeling approach. In either case, the result is a new description  $d_2$ . In this section, we analyze how the introduction of a new description  $d_2$  affects the surfeit relative to that of the original description  $d_1$ .

**Definition 5.4.1** Let  $r \in \mathcal{B}^*$  be a representation, and let  $d_1, d_2 \in \mathcal{D}$  be two descriptions. We define the *variation of surfeit* between the descriptions  $d_1$  and  $d_2$ , with respect to  $r$ , denoted by  $\Delta_\sigma^a(d_1, d_2, r)$ , as:

$$\Delta_\sigma^a(d_1, d_2, r) = \sigma(d_1, r) - \sigma(d_2, r)$$

Since surfeit is bounded between 0 and 1, the maximum possible variation is  $\pm 1$ . A positive value of  $\Delta_\sigma$  indicates that  $d_2$  is preferable to  $d_1$  in terms of surfeit. Conversely, a negative value suggests that  $d_1$  is more concise than  $d_2$ .

It is important to emphasize that a new description may also introduce a substantial increase in inaccuracy, potentially offsetting the improvement in surfeit. For a detailed discussion of inaccuracy, refer to Chapter 4, and for an explanation of how inaccuracy and surfeit combine into the unified metric of nescience, see Chapter 6.

We can also introduce a relative measure of the variation in surfeit when transitioning from description  $d_1$  to  $d_2$ .

**Definition 5.4.2** Let  $r \in \mathcal{B}^*$  be a representation, and  $d_1, d_2 \in \mathcal{D}$  be two descriptions. We define the *relative variation of surfeit* between descriptions  $d_1$  and  $d_2$ , denoted by  $\Delta_\sigma^r(d_1, d_2, r)$ , as:

$$\Delta_\sigma^r(d_1, d_2, r) = \frac{\sigma(d_1, r) - \sigma(d_2, r)}{\sigma(d_1, r)}$$

provided that  $\sigma(d_1, r) \neq 0$ .

A value of 0 indicates that there is no change in surfeit between the two descriptions. A value of 1 corresponds to a complete elimination of surfeit, meaning that the new description  $d_2$  has zero surfeit, while the original description  $d_1$  had a nonzero surfeit. Negative values indicate that surfeit has increased, suggesting that the new description  $d_2$  either includes more irrelevant or redundant symbols than  $d_1$ , or is too short to adequately represent the entity, thus omitting important information. It is important to note that the relative variation can become arbitrarily negative. This occurs when the surfeit of the original description  $\sigma(d_1, r)$  is close to zero, while the surfeit of the new description  $\sigma(d_2, r)$  is significantly larger. In such cases, the denominator becomes very small, causing the relative variation to diverge toward  $-\infty$ .

As surfeit approaches zero, relative variations become increasingly unstable. A small absolute change can result in a large relative variation when the initial surfeit is very low. For example, if  $\sigma(d_1, r) = 0.1$  and surfeit decreases by 0.05, the relative improvement is 50%. However, if  $\sigma(d_1, r) = 0.9$  and the same absolute reduction occurs, the relative improvement is only about 5.6%. Therefore, both absolute and relative variations are important for assessing the significance and scale of changes in surfeit.

## 5.5 Surfeit-inaccuracy rate of Change

In the preceding section, we examined how the surfeit of a description can be reduced by modifying that description, either by adding missing symbols, or by removing redundant, or irrelevant symbols. In this section, we turn our attention to a more general approach for reducing the surfeit associated with

an entity, by allowing a increase in the inaccuracy of the description. Rather than reducing the surfeit in isolation, it may be more effective to modify both, surfeit and inaccuracy, simultaneously.

The balance between the amount of inaccuracy we are willing to accept in order to achieve a reduction in surfeit is referred to as the *surfeit-inaccuracy trade-off*. For a broader discussion of trade-offs in multi-objective optimization, refer to Section 16.5.2.

**Definition 5.5.1** Let  $d_1, d_2 \in \mathcal{D}$  be two descriptions, and  $r \in \mathcal{B}^*$  a representation. We define the *rate of change between the surfeit and inaccuracy* of the descriptions  $d_1, d_2$  given the representation  $r$ , denoted by  $\Delta_{\sigma\iota}(d_1, d_2, r)$  as:

$$\Delta_{\sigma\iota}(x_1, x_2) = \frac{\sigma(d_2, r) - \sigma(d_1, r)}{\iota(d_2, r) - \iota(d_1, r)}$$

provided that  $\iota(d_2, r) - \iota(d_1, r) \neq 0$ .

The ratio  $\Delta_{\sigma\iota}$  represents the rate of change between surfeit and inaccuracy when transitioning from the first description to the second. A positive value of  $\Delta_{\sigma\iota}$  implies that both quantities, surfeit and inaccuracy, either decrease (which is desirable) or increase (which is undesirable). The interpretation becomes more nuanced when  $\Delta_{\sigma\iota}$  is negative, indicating that one of the quantities decreases while the other increases. In such cases, two scenarios must be considered:

- (i) Surfeit decreases and inaccuracy increases, we aim for  $\Delta_{\sigma\iota}(d_1, d_2, r) < M$ , where  $M < -1$ , thereby ensuring that the reduction in surfeit compensates for the increase in inaccuracy.
- (ii) Surfeit increases and inaccuracy decreases, we aim for  $\Delta_{\sigma\iota}(d_1, d_2, r) > M$ , where  $-1 < M < 0$ , thereby ensuring that the reduction in inaccuracy justifies the increase in surfeit.

In both cases, caution is warranted when the change in inaccuracy is small, as it can disproportionately affect the ratio and potentially lead to misleading conclusions.

**■ Example 5.4** Let  $d_1, d_2 \in \mathcal{D}$  be two descriptions, and  $r \in \mathcal{B}^*$  a representation. For  $d_1$ , the surfeit  $\sigma(d_1, r_1)$  is 0.40, and the inaccuracy  $\iota(d_1, r)$  is 0.15. For  $d_2$ , the surfeit  $\sigma(d_2, r)$  is 0.20 (a decrease from 0.40), and the inaccuracy  $\iota(d_2, r)$  is 0.25 (an increase from 0.15). Using the definition of the rate of change, we compute:

$$\Delta_{\sigma\iota}(d_1, d_2, r) = \frac{0.40 - 0.20}{0.15 - 0.25} = \frac{0.20}{-0.10} = -2$$

In this case, transitioning from description  $d_1$  to  $d_2$  results in a decrease of

surfeit by 0.20 units and an increase in inaccuracy by 0.10 units. The rate of change is  $-2$ . ■

Having a very small change in inaccuracy can significantly amplify the value of the rate of change, potentially giving the misleading impression that surfeit and inaccuracy are varying at an extreme rate, even when the actual changes are minor. This phenomenon is illustrated in the following example.

■ **Example 5.5** Let  $d_1 \in \mathcal{D}$  be a description and  $r \in \mathcal{B}^*$  a representation with an surfeit  $\sigma(d_1, r)$  of 0.35 and a inaccuracy  $\iota(d_1, r)$  of 0.20. Let  $d_2 \in \mathcal{D}$  be a second description with an surfeit  $\sigma(d_2, r)$  of 0.30 (a slight decrease from 0.35) and a inaccuracy  $\iota(d_2, r)$  of 0.2001 (a very small increase from 0.20). Applying the definition of the rate of change:

$$\Delta_{\sigma\iota}(d_1, d_2, r) = \frac{0.35 - 0.30}{0.20 - 0.2001} = \frac{0.05}{-0.0001} = -500$$

The rate of change is  $-500$ , which may misleadingly suggest a dramatic shift. In reality, the surfeit only decreased by 0.05 units, and the inaccuracy increased by a negligible 0.0001 units. The extremely small denominator inflates the result, making the change appear far more significant than it actually is. ■

As we attempt to optimize both surfeit and inaccuracy, we inevitably encounter descriptions where improving one objective necessitates compromising the other. The set of such "best trade-off" configurations constitutes the Pareto frontier (see Section 16.5). Points on the Pareto frontier are said to be Pareto optimal because any attempt to improve one objective leads to a deterioration in the other.

**Definition 5.5.2** Let  $d_1 \in \mathcal{D}$  be a description and  $r \in \mathcal{B}^*$  a representation. The description  $d_1$  is said to be a Pareto point with respect to surfeit  $\sigma$  and inaccuracy  $\iota$  if there does not exist another description  $d_2$  such that:

- 1  $\sigma(d_2, r) \leq \sigma(d_1, r)$  and  $\iota(d_2, r) \leq \iota(d_1, r)$ , and
- 2  $\sigma(d_2, r) < \sigma(d_1, r)$  or  $\iota(d_2, r) < \iota(d_1, r)$ .

In simpler terms, a description is Pareto optimal if: (i) no other description is better in both surfeit and inaccuracy, and (ii) any improvement in one metric necessarily results in a worsening of the other.

The rate of change  $\Delta_{\sigma\iota}$  between any two Pareto optimal points provides insight into how the trade-off between surfeit and inaccuracy evolves along the Pareto frontier. If decision-makers are more sensitive to changes in surfeit than to inaccuracy, they may prefer configurations with a less negative  $\Delta_{\sigma\iota}$ . Conversely, if inaccuracy is of greater concern, they may accept solutions where  $\Delta_{\sigma\iota}$  indicates a larger increase in surfeit in exchange for a smaller gain in inaccuracy.

## 5.6 Surfeit of Areas

The concept of surfeit can be extended to research areas, to quantitative measure the amount of extra effort we are using to describe the topics of the area.

**Definition 5.6.1** Let  $A \subset \mathcal{T}$  be an area with known subset  $\hat{A} = \{t_1, t_2, \dots, t_n\}$ , and let  $d_{\hat{A}}$  be a description. We define the *surfeit of the description*  $d_{\hat{A}}$  as:

$$\sigma(d_{\hat{A}}) = 1 - \frac{K(\langle t_1, t_2, \dots, t_n \rangle)}{l(d_{\hat{A}})}$$

As it was the case of the concept of redundancy, in general we do not know the complexity of the area  $K(\hat{A})$ , and so, in practice, it must be approximated by the complexity of the descriptions themselves  $K(\hat{d}_{\hat{A}})$ . However, in the particular case of areas, we could have also problems with the quantity  $\hat{d}_{\hat{A}}$ , since it requires to study the conditional descriptions of the topics included in the area.

**Definition 5.6.2** Let  $A \subset \mathcal{T}$  be an area with known subset  $\hat{A} = \{t_1, t_2, \dots, t_n\}$ , and let  $d_{\hat{A}}$  be a description. We define the *weak redundancy of the description*  $d_{\hat{A}}$  as:

$$\rho(d_{\hat{A}}) = 1 - \frac{K(d_{\hat{A}})}{l(d_{\hat{A}})}$$

## References

The concept of redundancy has been also investigated in the context of information theory, since we are interested on using codes with low redundancy (see for example [Abr63]).



## 6. Nescience

*There are known knowns. These are things we know that we know. There are known unknowns. That is to say, there are things that we know we don't know. But there are also unknown unknowns. There are things we don't know we don't know.*

Donald Rumsfeld

Following an initial Chapter 2 that established the foundational concepts of entity, representation, and description, and the subsequent Chapters 3, 4, and 5, which introduced novel metrics for miscoding, inaccuracy, and surfeit, we are now equipped with the essential tools to explore the profound concept of nescience in this chapter, focusing on its key characteristics.

Unlike Shannon's entropy or Kolmogorov's complexity, which quantify information, nescience assesses the absence of information, or what remains unknown. The theory of nescience quantifies our ignorance regarding a research entity through the three newly examined metrics: miscoding, inaccuracy, and surfeit. Miscoding evaluates the fidelity of an entity's representation as a sequence of symbols; inaccuracy gauges the precision with which our optimal model captures this symbolic representation; and surfeit examines the depth of our comprehension of the description, measured by the model's

length or symbol count. The challenge with these metrics lies in their inherent conflict: improving one may worsen the others. Our goal is to find a method to minimize all these elements simultaneously. This requirement underlines, according to our interpretation, that science is fundamentally a multi-objective optimization problem.

One of the most significant outcomes of our definition of nescience, grounded in the metrics of miscoding, inaccuracy, and surfeit, is its ability to categorize the domain of research topics into two distinct areas. The first area is referred to as the known unknown, which encompasses the topics we recognize as not fully understanding yet acknowledge our lack of complete comprehension. The second area is the unknown unknown, consisting of topics yet to be discovered. A crucial application of the theory of nescience lies in its use as a methodology for uncovering what resides within the unknown unknown.

Another notable consequence of the nescience concept is the highly counterintuitive notion that, for certain topics, further research may prove to be counterproductive. In these instances, the more research we conduct, the less we know, as it becomes impossible to expand our knowledge beyond a critical threshold, despite being far from possessing a comprehensive understanding.

## 6.1 Nescience

Intuitively, our understanding of an entity should be based on the quality of the model we use to describe it, namely, its ability to explain why things happen. In the theory of nescience, we propose a quantitative measure of our ignorance about a research entity using the miscoding of a string-based representation of the entity, and the inaccuracy and surfeit of the model describing this representation. Miscoding is significant because it indicates how well the representation encodes the entity, inaccuracy measures how closely our model comes to adequately describing the representation, and surfeit quantifies the extent of unnecessary effort invested in the model. We believe that the objective of Science should be to minimize these three quantities: miscoding, inaccuracy, and surfeit. Unfortunately, these metrics are conflicting, in the sense that decreasing one could increase the other two.

According to the theory of nescience, science aims to address the following multiobjective optimization problem<sup>1</sup>:

---

<sup>1</sup>Technically speaking, science is a deterministic discrete nonlinear nonconvex nondifferentiable multiobjective optimization problem with a single decision maker.

### The Science Problem

$$\begin{aligned} & \text{minimize} && \{\mu(r), \iota(d, r), \sigma(d, r)\} \\ & \text{subject to} && (r, d) \in \mathcal{B}^* \times \mathcal{D} \end{aligned}$$

A *scientific method*, detailed further in Section A.2, refers to any algorithm or computable procedure capable of addressing, or providing a closely approximate resolution to, the previously discussed minimization challenge. This encompasses a wide range of techniques and methodologies aimed at systematically reducing the metrics of miscoding, inaccuracy, and surfeit within the realm of scientific inquiry, thereby enhancing the precision and conciseness of our representations and models.

The feasible region is the Cartesian product  $\mathcal{B}^* \times \mathcal{D}$  of the set of finite binary strings  $\mathcal{B}^*$  and the set of descriptions  $\mathcal{D}$ . The decision vectors are pairs  $(r, d)$ , called *hypothesis*, composed of a representation and a description, and the objective functions to minimize are miscoding, surfeit, and inaccuracy. The objective region is the subset  $\mathbf{Z} \subset \mathbb{R}^3$ , and its elements are the objective vectors.

In our characterization of science and the scientific method, we do not involve the set  $\mathcal{E}$  of entities, since requiring knowledge of the entity  $e \in \mathcal{E}$  under study would render science an ill-defined problem for the majority of research topics. Science is essentially a problem of manipulating strings of symbols. From a practical perspective, science is about finding strings that have a meaningful interpretation in the real world and that enable us to solve practical problems. From a more theoretical perspective, the goal of science would be to understand how an unknown abstract oracle functions.

If the set  $\mathcal{R}_e$  of representations of the particular entity  $e$  in which we are interested is known, or approximately known, we could restrict the science problem to:

$$\begin{aligned} & \text{minimize} && \{\mu(r), \iota(d, r), \sigma(d, r)\} \\ & \text{subject to} && (r, d) \in \mathcal{R}_e \times \mathcal{D} \end{aligned}$$

In the theory of nescience, we are mostly interested in the decision space  $\mathcal{B}^* \times \mathcal{D}$  of representations and descriptions rather than in the objective space  $\mathbf{Z} \subset \mathbb{R}^3$  of metric values. In the following definitions, we reintroduce some of the concepts of multiobjective optimization (see Section 16.5) for the particular case of the science problem.

### Pareto Optimality

If the representation and description currently in use to describe an entity are not perfect, our goal is to find another representation, or another description,

that reduces at least one of the metrics miscoding, inaccuracy, or surfeit without deteriorating either of the other two.

**Definition 6.1.1** We say that a decision vector  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  *dominates* another decision vector  $(r', d') \in \mathcal{B}^* \times \mathcal{D}$  if it improves in one of the metrics miscoding, inaccuracy, or surfeit without deteriorating either of the other two.

We could discover a new representation that more accurately encodes the entity of interest without diminishing the quality of its descriptive model. Alternatively, we might identify a new description that addresses inaccuracies without augmenting the surfeit, or we could find a more concise description that does not compromise accuracy.

■ **Example 6.1** Consider an experiment where we've gathered several observations  $r$  and applied a mathematical model  $f_1$  from a family of models  $\mathcal{M}_1$ , resulting in an inaccuracy of  $i_1$ . Subsequently, we might fit a second model  $f_2$  from an alternative family of models  $\mathcal{M}_2$ . This second model is smaller than  $i_2$  while maintaining the same inaccuracy. In this scenario, we would say that the hypothesis  $B = (r, i_2)$  dominates the hypothesis  $A = (r, i_1)$  although they are not superior in the individual objective of inaccuracy. ■

For the majority of entities, there does not exist a single solution that simultaneously minimizes the three metrics. Instead, what we have is a collection of Pareto optimal solutions that define an optimal frontier.

**Definition 6.1.2** We say that a decision vector  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  is *Pareto optimal* if there does not exist another decision vector  $(r', d') \in \mathcal{B}^* \times \mathcal{D}$  such that  $(r', d')$  dominates  $(r, d)$ . The set of Pareto optimal solutions, denoted by  $\mathbf{P}_{\mathcal{B}^* \times \mathcal{D}}$ , is called the *Pareto frontier*.

In the realm of scientific research, the concept of the Pareto frontier, as defined by the set of Pareto optimal solutions  $\mathbf{P}_{\mathcal{B}^* \times \mathcal{D}}$ , plays a crucial role. It delineates the boundary of optimal trade-offs between the conflicting metrics of miscoding, inaccuracy, and surfeit, without one being improved at the expense of the others. This frontier represents the spectrum of best-achievable balances, guiding researchers to identify models and representations that offer the most scientifically rigorous understanding of their subject matter (see Section 6.6). However, in certain cases or specific practical applications, it might be justifiable to adopt a solution that is not Pareto optimal. For example, we might prioritize one metric significantly over others due to its relevance or the specific objectives of the research, accepting a degree of compromise in the non-prioritized metrics (see Section 6.2).

Building on the concept of Pareto optimality, where a solution is consid-

ered optimal if no other solution can improve one objective without worsening another, we introduce the notion of weakly Pareto optimality. A decision vector is deemed weakly Pareto optimal if there is no other vector that can improve all objectives simultaneously. It differs from Pareto optimality in that it includes solutions that, although they may not be superior in some individual objective, are not outperformed across all dimensions.

**Definition 6.1.3** We say that a decision vector  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  *weakly dominates* another decision vector  $(r', d') \in \mathcal{B}^* \times \mathcal{D}$  if it improves the three metrics miscoding, inaccuracy and surfeit at the same time, that is,  $\mu(r') < \mu(r)$ ,  $\iota(d', r') < \iota(d, r)$  and  $\sigma(d', r') < \sigma(d, r)$ .

A decison vector is weakly pareto optimal if it does not exist another decision vector that improves over the three metrics of miscoding, inaccuracy and surfeit.

**Definition 6.1.4** We say that a decision vector  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  is *weakly Pareto optimal* if there does not exist another decision vector  $(r', d') \in \mathcal{B}^* \times \mathcal{D}$  such that  $(r', d')$  weakly dominates  $(r, d)$ . The set of Pareto optimal solutions, denoted by  $\mathbf{P}_{\mathcal{B}^* \times \mathcal{D}}$ , is called the *weakly Pareto frontier*.

If a decision vector is weakly Pareto optimal that means we cannot find another decision vector that improves over the three metrics. However, we can still find a decison vector that improves over one of the metrics without deteriorating the other, that is, a decison vector that is Pareto optimal. The Pareto frontier is a subset of the weakly Pareto frontier. In the theory of nescience we will work mostly with the set of Pareto optimal solutions, to the detriment of set of weak Pareto solutions.

■ **Example 6.2** Based on the assumptions of Example 6.1 the hypothesis *A* still could be weakly Pareto optimal, but it cannot be Pareto optimal, since it is dominated by the hypothesis *B*, but is is not weakly dominated by the hypothesis *B*. ■

The concept of Pareto and weakly Pareto optimality can be particuralized to the case in which the set  $\mathcal{R}_e$  of representations of the particular entity *e* is known.

## Range of Solutions

As discussed in Section 16.5.1, an objective vector that achieves the minimum possible value for all objective functions is termed an ideal objective vector. For the science problem, this ideal vector is represented by the origin  $(0, 0, 0)$ , symbolizing the complete elimination of miscoding, inaccuracy, and surfeit.

**Proposition 6.1.1** The ideal objective vector for the science problem is the

origin  $(0, 0, 0)$ .

*Proof.* Proposition 3.1.1 demonstrated that miscoding is greater than or equal to zero, and Proposition 3.1.2 showed that it can be equal to zero. Similarly, Proposition 4.5.1 established that inaccuracy is greater than or equal to zero, and Proposition ?? indicated that a zero value is attainable. Finally, Proposition 6.2.1 argued for surfeit being greater than or equal to zero, and Proposition 5.1.2 confirmed that achieving a value of zero is possible. ■

A decision vector  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  is ideal if it possesses zero miscoding, zero inaccuracy, and zero surfeit. This means that the representation  $r$  is valid, the model  $d$ 's output is  $r$ , and no shorter model  $d'$  exists that also has zero inaccuracy. Intuitively, a decision vector  $(r, d)$  is ideal if there is an entity  $e \in \mathcal{E}$  such that  $r$  perfectly encodes  $e$ , and the model  $d$  for  $r$  is both accurate and minimal.

Ideal decision vectors embody the notion of perfect knowledge within the theory of nescience. Unfortunately, for most practical applications, achieving the ideal vector is not feasible due to the conflicting nature of the metrics miscoding, inaccuracy, and surfeit.

■ **Example 6.3** TODO: in practice it is impossible to reach the objective vector. ■

The upper bound of the Pareto optimal set is given by the nadir objective vector. In the theory of nescience, the nadir vector would be the  $(1, 1, 1)$  vector, in which we have the maximum miscoding, minimal accuracy, and maximum surfeit.

**Proposition 6.1.2** The nadir objective vector of the science problem is the vector  $(1, 1, 1)$ .

*Proof.* Proposition 3.1.1 demonstrated that miscoding is greater than or equal to zero, and Proposition 3.1.2 showed that it can be equal to zero. Similarly, Proposition 4.5.1 established that inaccuracy is greater than or equal to zero, and Proposition ?? indicated that a zero value is attainable. Finally, Proposition 6.2.1 argued for surfeit being greater than or equal to zero, and Proposition 5.1.2 confirmed that achieving a value of zero is possible. ■

The nadir vector represents the situation in which we have a decision vector  $(r, d)$  with zero knowledge, that is, our representation  $r$  does not contain symbols related to the entity  $e$  under study, the description  $d$  produces a string completely different from  $r$ , and it has a maximum length.

■ **Example 6.4** TODO: in practice it is impossible to reach the nadir objective vector. ■

## Trade-offs

In Section 16.5.2 we saw that since the functions we want to minimize are conflicting, we have to assume that the only way to gain a benefit in one aspect of the problem is to lose something in another aspect, what it is called a trade-off. In this section we study the different tradeoffs that can arise in the theory of nescience.

**TODO:** introduce the concept of partial miscoding - inaccuracy

**Definition 6.1.5** Let  $(r_1, d), (r_2, d) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the partial ratio of change between the functions miscoding and inaccuracy for the vectors  $(r_1, d), (r_2, d)$  as:

$$\Delta_{\mu\iota}((r_1, d), (r_2, d)) = \frac{\iota(r_1, d) - \iota(r_2, d)}{\mu(r_1) - \mu(r_2)}$$

**TODO:** study its properties

**TODO:** introduce the concept of total miscoding - inaccuracy

**Definition 6.1.6** Let  $(r_1, d_1), (r_2, d_2) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the total ratio of change between the functions miscoding and inaccuracy for the vectors  $(r_1, d_1), (r_2, d_2)$  as:

$$\Delta_{\mu\sigma}((r_1, d_1), (r_2, d_2)) = \frac{\iota(r_1, d_1) - \iota(r_2, d_2)}{\mu(r_1) - \mu(r_2)}$$

**TODO:** study its properties

**TODO:** introduce the concept of partial miscoding - surfeit

**Definition 6.1.7** Let  $(r_1, d), (r_2, d) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the partial ratio of change between the functions miscoding and surfeit for the vectors  $(r_1, d), (r_2, d)$  as:

$$\Delta_{\mu\sigma}((r_1, d), (r_2, d)) = \frac{\sigma(r_1, d) - \sigma(r_2, d)}{\mu(r_1) - \mu(r_2)}$$

**TODO:** study its properties

**TODO:** introduce the concept of total miscoding - surfeit

**Definition 6.1.8** Let  $(r_1, d_1), (r_2, d_2) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the total ratio of change between the functions miscoding and surfeit for the vectors  $(r_1, d_1), (r_2, d_2)$  as:

$$\Delta_{\mu\sigma}((r_1, d_1), (r_2, d_2)) = \frac{\sigma(r_1, d_1) - \sigma(r_2, d_2)}{\mu(r_1) - \mu(r_2)}$$

**TODO:** study its properties

TODO: introduce the concept of partial inaccuracy - surfeit

**Definition 6.1.9** Let  $(r, d_1), (r, d_2) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the partial ratio of change between the functions inaccuracy and surfeit for the vectors  $(r, d_1), (r, d_2)$  as:

$$\Delta_{\iota\sigma}((r, d_1), (r, d_2)) = \frac{\iota(r, d_1) - \iota(r, d_2)}{\sigma(r, d_1) - \sigma(r, d_2)}$$

TODO: study its properties

TODO: introduce the concept of total inaccuracy - surfeit

**Definition 6.1.10** Let  $(r_1, d_1), (r_2, d_2) \in \mathcal{B}^* \times \mathcal{D}$  be two decision vectors. We define the total ratio of change between the functions inaccuracy and surfeit for the vectors  $(r_1, d_1), (r_2, d_2)$  as:

$$\Delta_{\mu\sigma}((r_1, d_1), (r_2, d_2)) = \frac{\iota(r_1, d_1) - \iota(r_2, d_2)}{\sigma(r_1, d_1) - \sigma(r_2, d_2)}$$

TODO: study its properties

TODO: study the concept of properly Pareto optimal solutions in the context of the theory of nescience.

## 6.2 Minimizing Nescience

From a mathematical point of view, any of the solutions that compose the Pareto set would be a valid solution to the Science Problem. In fact, the problem is considered to be solved when all the solutions of the Pareto optimal set are found. However, in science, this is rarely enough, and we prefer to have a single solution. The goal of the decision maker is to provide an ordering to the optimal solutions, or equivalently, add a preference of one solution over the others.

Introduce the concept of nescience decision maker as a function of miscoding, inaccuracy and surfeit, value function

**Definition 6.2.1** Let  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  be a decision vector. A decision maker for the science problem is a multivariate function  $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ , that assigns to each triplet  $(\mu(r), \iota(d, r), \sigma(d, r))$  the real value  $f(\mu(r), \iota(d, r), \sigma(d, r))$ .

The good news are that we have designed those metrics so that they are commensurable, that is, all of them are measured using the same unit: the length of a computer program. Moreover, they are in the same scale, since they have been normalized to the interval between zero and one.

## 6.2.1 Global Criterion

The global criterion (see Section 16.5.3) is a method for solving multi-objective optimization problems in which the distance between some reference point and the feasible objective region is minimized. As reference point it is usually used the ideal vector, which in our case is the origin  $(0, 0, 0)$  of the objective space. And as distance, we could use different metrics. For example, the global criterion based on the origin and an Euclid distance requires to solve the following minimization problem:

$$\begin{aligned} \text{minimize} \quad & \sqrt{\mu(r)^2 + \iota(d, r)^2 + \sigma(d, r)^2} \\ \text{subject to} \quad & (r, d) \in \mathcal{B}^* \times \mathcal{D} \end{aligned}$$

As Proposition 16.5.1 proved, the solutions provided by the global criterion are Pareto optimal. However, not all the solutions from the Pareto frontier are considered as candidate solutions using this method.

We do not need to normalize the candidate solutions before to solve the minimization problem since the metrics we are using are already normalized.

Some examples of alternative distances than can be used with the global criterion method are the following:

**TODO: Filter out non-distances.**

- Arithmetic mean:  $\frac{\mu(r) + \iota(d, r) + \sigma(d, r)}{3}$
- Geometric mean:  $(\mu(r) \times \iota(d, r) \times \sigma(d, r))^{1/3}$
- Product:  $\mu(r) \times \iota(d, r) \times \sigma(d, r)$
- Addition:  $\mu(r) + \iota(d, r) + \sigma(d, r)$
- Harmonic mean:  $\frac{3}{\mu(r)^{-1} + \iota(d, r)^{-1} + \sigma(d, r)^{-1}}$

**TODO: Mention advantages and drawbacks of this alternative solutions.**

Geometric mean, product and harmonic mean have the problem that the nescience is zero, or not defined, if one of the three metrics (mCoding, inaccuracy or surfeit) is zero.

■ **Example 6.5 TODO: update this example** Let  $t \in \mathcal{T}$  a topic, and  $d_1$  and  $d_2$  two descriptions of  $t$ . Assume that the error of  $d_1$  is 0.1 and its redundancy 0.4, and that  $d_2$  has an error 0.2 and a redundancy of 0.2. According to Definition ??, the nescience of topic  $t$  given the description  $d_1$  would be 0.41, and the nescience given the description  $d_2$  would be 0.28. Our unknown about topic  $t$  would be smaller in case of description  $d_2$  than with description  $d_1$ . ■

**TODO: rewrite this paragraph** What Example 6.5 show us is that, given our definition of nescience, we should prefer a less redundant explanation that maybe does not describe perfectly a topic, to a highly redundant description that fits the topic much better. The Occam's razor principle states that

between two indifferent alternatives we should select the simplest one. Our theory states that even in case they are not indifferent alternatives, it might be worth to select the the simplest one. The theory of nescience has not been designed to find the true about a topic, in its philosophical sense, but to find the best theory that can be used in practice. In this sense, we borrow concepts and ideas from the area of machine learning, and in particular, from the problem of *model overfitting* when dealing with the results of an experiment (see Chapter ?? for more information about this problem).

**Proposition 6.2.1** We have that  $0 \leq \rho(d_t) \leq 1$  for all  $t \in \mathcal{T}$  and  $d_t \in \mathcal{D}$ .

*Proof.* Given that  $l(d_t) > 0$  and that  $K(t) > 0$ , since they are the lengths of non-empty strings, we only need to prove that  $l(d_t) \geq K(t)$ ; however  $l(d_t) < K(t)$  is a contradiction with the fact that  $K(t)$  is the length of the shortest possible Turing machine that prints out  $t$ . ■

### 6.2.2 Weighting Method

### 6.2.3 The other method

### 6.2.4 Evolutionary Methods

## 6.3 Joint Nescience

In Section 2.4 we introduced the concept of joint representation as the concatenation of two or more strings, and in Section 3.2 we studied how miscoding is affected when we concatenate these representations. In this section we are going to see how nescience, as a global metric, is affected when joining representations.

**Definition 6.3.1** Let  $s, t \in \mathcal{B}^*$  be two representations, and  $d \in \mathcal{D}$  a description. We call *joint nescience*, denoted by  $\mathcal{N}(st, d)$ , to the nescience of the joint representation  $st$  and the description  $d$ .

We are interested to know how the joint nescience of  $\mathcal{N}(st, d)$  is related to the individual nesciences  $\mathcal{N}(s, d)$  and  $\mathcal{N}(t, d)$ .

**TODO: rewrite paragraph.** As we have seen in Section 2.4 there is no way to gurantee that the miscoding of a joint representation will be smaller, or greater, than the miscoding of the individual representation, not even in case that  $s$  and  $t$  encode the same entity  $e$ . In the same way, we saw in Section XXX that the inaccuracy of a fixed model  $d$  for a joint representation  $st$  could be greater, or smaller, than the inaccuracy of the model for the individual representations  $s$  or  $t$ . And finally, in case of surfeit, as se saw ...

**TODO: Is there anything we can say about joint nescience? Conumtative?  
Extend to the concatenation of multiple representations.**

## 6.4 Conditional Nescience

**TODO:** Introduce the section, what we are aiming for.

**TODO:** Introduce the concept of conditional nescience

**Definition 6.4.1** Let  $(r, d) \in \mathcal{B}^* \times \mathcal{D}$  be a topic, and  $s \in \mathcal{B}^*$  be an arbitrary string. We define the *conditional nescience* of the topic  $(r, d)$  given the string  $s$ , denoted  $\mathcal{N}(r, d_{r|s})$ , as the nescience of the representation  $r$  and the conditional description  $d_{r|s}$ .

**TODO:** Explain the intuition behind this concept.

**TODO:** Provide a practical example.

**TODO:** Provide a maximum and a minimum for the concept

The next proposition states that the nescience of a topic can only decrease if we assume the background knowledge given by another topic. : **Explain the intuition behind this property.**

**TODO:** Prove something like  $N(t, s) = N(t) + N(s|t)$

**TODO:** Check the following proposition

**Proposition 6.4.1**  $N_{t|s} = N_{s|t}$  if, and only if,  $N_{t|s} = N_{s|t} = 0$ .

Finally, next proposition states the relation between nescience, conditional nescience and joint nescience.

**TODO: review**

**Proposition 6.4.2** Given any two topics  $t, s \in T$ , we have that  $N_{t|s} \leq N_t \leq N_{(s,t)}$ .

*Proof.* **TODO** ■

The concept of conditional nescience can be extended to multiple topics  $s_1, \dots, s_n$ , using the standard encoded string  $\langle s_1^*, \dots, s_n^*, a \rangle$  and the multiple conditional complexity  $K(t | s_1, \dots, s_n)$ .

**Definition 6.4.2** Let  $t, s_1, \dots, s_n \in T$  a collection of topics. The *conditional nescience* of topic  $t$  given the topics  $s_1, \dots, s_n$  and current best description  $\hat{d}_t$  is defined as:

**TODO**

## 6.5 Nescience of Areas

**TODO:** Rewrite this section.

Topics can be grouped into research areas. The concept of area is useful as long as all the topics included in the area share a common property.

The particular details of the grouping properties depend on the practical applications of the theory.

**Definition 6.5.1** An area  $A$  is a subset of topics, that is  $A \subset T$ .

Areas can overlap, that is, given two areas  $A$  and  $B$  it might happen that  $A \cap B \neq \emptyset$ . Areas can be subsets of other areas, creating an hierarchy of areas.

**Example 6.6** If the set of topics is "mathematics", examples of areas could be "calculus", "geometry" or "algebra". The areas "probability" and "statistics" largely overlap. The area "Bayesian inference" is a subarea of the area "probability". ■

**TODO:** study the properties of the research areas.

In the same way we studied the properties of individual topics, we could study the properties of areas. An *area* is a subset of topics  $A \subset T$ . The concept of area is useful as long as all the topics included in the area share a common property. What is exactly that property they share depends on the particular set  $T$ .

**Definition 6.5.2** Given an area  $A \subset T$ , we define the *average complexity of the area*  $C_A$  as  $C_A = \frac{1}{n} \sum_{t \in A} C_t$ , and the *average nescience of the area*  $N_A$  as  $N_A = \frac{1}{n} \sum_{t \in A} N_t$ , where  $n$  is the cardinality of  $A$ .

For example, in case of research topics, an area could be a knowledge area, like biology, that will contain all the topics classified under that area. In this way we could compute and compare the complexity (how difficult is to understand) and the nescience (how ignorant we are) of mathematics, physics, biology, social sciences, and other disciplines.

**TODO:** This definition can only be introduced once we have the concept of best current description.

An even easier approximation of the concept of redundancy of an area, is based on the average redundancy of the topics that compose that area.

**Definition 6.5.3** Let  $A \in \mathcal{T}$  an area, and  $d_A$  a description. We define the *average redundancy of the description*  $d_A$  as:

$$\bar{\rho}(d_A) = \sum$$

**TODO:** Study some properties of this definition

**TODO:** How these three definitions relate to each other?

## 6.6 Perfect Knowledge

**TODO:** Rewrite this section

As we have said, in our opinion, the objective of scientific research should be to reduce the nescience of topics as much as possible. When it is not possible to reduce more the nescience of a topic, we propose to say we have reached a perfect knowledge. A consequence of

**Definition 6.6.1 — Perfect Knowledge.** If the nescience of a topic  $t$  is equal to zero ( $v(t) = 0$ ), we say that we have reached a *perfect knowledge* about topic  $t$ .

If  $v(t) = 0$  we have that  $\varepsilon(t) = \rho(t) = 0$ , that is, perfect knowledge is achieved when we can fully reconstruct the original topic given its description, and the description does not contain any redundant elements. A consequence of our definition is that perfect knowledge implies randomness, that is, incompressible descriptions. The converse, in general, does not hold. The common point of view is that a random string should make nonsense, since this is what randomness is all about. However, in the theory of nescience, by random description we mean a description that contains the maximum amount of information in the least space as possible (it contains no redundant elements).

■ **Example 6.7** Aristotelian physics is an inaccurate description of our world, since it makes some predictions that do not hold in reality (for example, planets do not orbit around the earth). We could use a description of the Aristotelian physics and compress it using a standard compression program. The compressed file would be a random description (zero redundancy). However, given that description, our nescience would not be zero, that is, our knowledge would not be perfect, since the error of the description is not zero.

■

TODO: Show how nescience evolves with time

TODO: Define the concept of weak nescience

TODO: Explain how weak nescience and nescience relates to each other

TODO: Show how the weak nescience converges to nescience in the limit

**Theorem 6.6.1** Let  $t \in T$  a topic, and  $\{d_1, d_2, \dots\}$  where  $d_i \in D_t$  a set of descriptions such that  $l(d_i) < l(d_j)$  for each  $i < j$ , then

$$\lim_{x \rightarrow \infty} \hat{N}_i = N_t$$

*Proof.* To Be Done ■

■ **Example 6.8** In order to clarify how the above theorem can be applied in practice, in Figure 6.1 it is shown an hypothetical example of the typical

research process required to understand a scientific topic  $t \in T$ . In time  $t_1$  we have a description with length 12, whose compressed version has a length of 5, and so, its nescience is 1.4. In time  $t_2$ , supposedly after some intense research effort, our current description length has been reduced to 8 with a complexity of 4, and our nescience has decreased to 1. In the limit, the description length will be equal to its complexity (incompressible description), and the nescience will be 0. In this moment we could say that we have a *perfect knowledge* about that particular research topic. ■

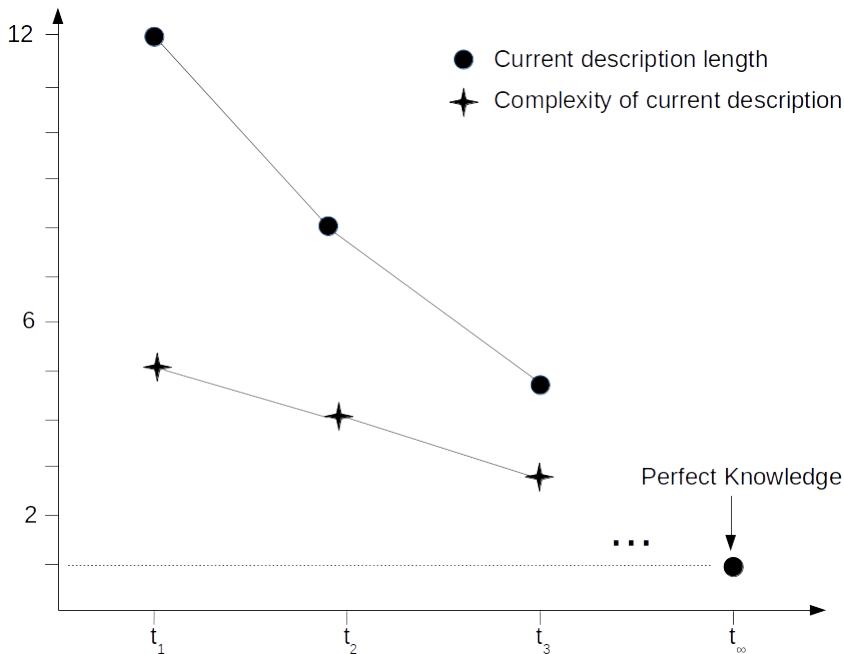


Figure 6.1: In Pursuit of Perfect Knowledge

**TODO:** Introduce the following definition.

**Definition 6.6.2** Let  $s_1, \dots, s_n \in T$  a collection of topics. The *joint nescience* of topics topics  $s_1, \dots, s_n$  given the current best descriptions  $\hat{d}_{s_1}, \dots, \hat{d}_{s_n}$  is defined as:

**TODO**

**TODO:** Mention, or prove, the properties of the generalized nescience

## 6.7 Current Best Description

**TODO:** Rewrite this section

We are also interested in our current understanding of the concatenation of two topics.

**Definition 6.7.1** Given  $t, s \in \mathcal{T}$  two different topics, and the set  $\mathcal{D}_{t,s} = \{d \in \mathcal{B}^* : d = \langle TM, a \rangle : TM(a) = \langle t, s \rangle\}$ , let  $\hat{d}_{t,s}$  be a distinguished element of  $\mathcal{D}_{t,s}$ . We call  $\hat{d}_{t,s}$  our *current best joint description* of  $t$  and  $s$ .

The concept of best joint description could be a little bit misleading, since given that the concatenation of any two topics is another topic, we could ask ourselves why do not simply use our current best description of the topic represented by that concatenation. That would make sense only in case that somebody has already studied both topics together. However, for the overwhelming majority of the possible combinations of topics, nobody has studied them yet.

■ **Example 6.9** Let  $t$  and  $s$  two different topics, and assume that nobody has studied them together before. In this case, our current best description  $\hat{d}_{t,s}$  would be  $\langle TM, \langle \hat{d}_t, \hat{d}_s \rangle \rangle$ , where  $TM$  is a Turing machine that given the input  $\langle \hat{d}_t, \hat{d}_s \rangle$  prints out the string  $ts$ . If  $\hat{d}_t = \langle TM_t, a_t \rangle$  and  $\hat{d}_s = \langle TM_s, a_s \rangle$ , the machine  $TM$  will decode  $\hat{d}_t$ , run  $TM_t(a_t)$  to print out  $t$ ; then it would do the same for  $\hat{d}_s$  to print  $s$ . ■

As we said in the preface of this chapter, in order to compute how much we do not know about a topic, first we need a way to quantify what we already know about that topic. How much we know about a topic will be given by our current best known description of that topic.

**Definition 6.7.2** Given the set of descriptions  $\mathcal{D}_t$  of a topic  $t \in \mathcal{T}$ , let  $\hat{d}_t$  be a distinguished element of  $\mathcal{D}_t$ . We call  $\hat{d}_t$  our *current best description* of  $t$ .

Which description is the current best description is something that depends on our current knowledge about the particular area in which the theory of nescience is being applied.

## 6.8 Nescience based on Datasets

**TODO: Perhaps we should remove this section**

Some topics can be described using a mathematical model that can be evaluated by a computer. For those topics we could estimate their complexity using a sample dataset, for example the result of an experiment. This property allows us, among other things, to compare how well topics are described by different models (see Chapter ??).

**Definition 6.8.1** Let  $t \in T$  be a topic,  $D = \{x_1, x_2, \dots, x_n\}$  the result of running an experiment that describes  $t$ , and  $H$  a mathematical hypothesis for  $t$ . The complexity of the topic  $t$  given the current description  $H$  and the dataset  $D$  is given by

$$\hat{C}_t = L(H) + L(D | H)$$

as it is described by the minimum description length principle.

Given the dataset  $D$  and the model  $H$  we can compute our current nescience of a topic as it is described by the next definition.

**Definition 6.8.2** Let  $t \in T$  a topic,  $D = \{x_1, x_2, \dots, x_n\}$  the result of running an experiment that describes  $t$ ,  $C$  a code that minimizes the length of  $D$ , and  $H$  a mathematical hypothesis for  $t$ . The current complexity of the topic  $t$  given the dataset  $D$  and the hypothesis  $H$  is given by

$$N_t = \frac{\hat{C}_t - l_C(D)}{l_C(D)}$$

In practice, the dataset  $D$  also allows us to approximate our current nescience of a topic  $t$ , given the model  $H$ . What we have to do is to use a near minimal encoding for  $D$ , for example, by using a Huffam encoding.

## 6.9 Unknown Unknown

TODO: pending

Know unknown

Unknown unknown

Finally, there exists a last category of unknonw, what we call the unknowable unknown unknown. This category refers to those entities we

Unknowable unknown unknonwn. In this book we are interested in the unknown unknown

## 6.10 Science vs. Pseudoscience

TODO: Provide a characterization of the difference between science and pseudoscience. In science, nescience decreases with time, in pseudoscience not.

## References

TODO: Add the paper of Chaitin about the Berry paradox

TODO: That there are numbers that are not computable can be found in the original paper of Turing

TODO: Perhaps I should provide a couple of references in epistemology and ontology





## 7. Interesting Questions

*It is not the answer that enlightens,  
but the question.*  
Eugène Ionesco

In this chapter, we propose a set of metrics for classifying research topics based on their potential as a source of interesting problems, along with a methodology for assisted discovery of new questions. The methodology can be used to identify new applications of existing tools to solve open problems and to discover new, previously unexplored research topics. The methodology is applicable to both intradisciplinary and interdisciplinary topics, but the most interesting outcomes are obtained in the latter case. In Chapters 9 and 10, we demonstrate the application of the methodology in practice and suggest new questions and research topics.

Our primary assumption is that a research question is interesting if it meets the following three criteria:

- C1** The question should be new and original, meaning that it has not been previously considered.
- C2** Upon its resolution, there should be a significant increase in our knowledge about one or more specific research topics.

**C3** It should have practical applications that could have a substantial (hopefully positive) impact on people's lives.

Some researchers may argue that the requirements presented may not be the most suitable. For instance, researchers from the so-called "hard sciences", such as pure mathematics and theoretical physics, might object that practical applications are not a crucial factor in pursuing an interesting open problem. In such situations, the metrics introduced can be redefined since they are simply mathematical abstractions, and the same methods can still be applied. The methodology described is universal and can be employed in various domains, not only in discovering new research questions. In fact, the metrics and methods outlined can be utilized in any field where there is a vast collection of interconnected describable objects, and the objective is to uncover new and previously unknown objects. The precise definition of concepts like relevance graph, applicability graph, or maturity will be contingent on the field in which the approach is being employed. Nevertheless, as in the case of Chapter 6, we prefer to present the methodology and new concepts in the specific context of scientific research because it aids in their comprehension.

It is worth mentioning the relationship between this new methodology and the areas of computational creativity and artificial intelligence. What is proposed in this chapter is an algebraic approach to the assisted discovery of potentially interesting questions. The intention is not for the computer to understand the meaning of the questions posed. Furthermore, not all the questions generated are necessarily relevant or even meaningful. It is the responsibility of the researchers to assess the proposed combinations of topics and determine if any of the questions are appropriate.

We have already explored two dimensions for classifying topics: mis-coding (Chapter 3) and mismodel (Chapter 5). These metrics enable us to quantitatively assess our understanding of a topic, which we refer to as nescience. According to criterion **C2** of our list of requirements for questions, the higher the nescience of a topic, the greater its potential as a source of interesting research questions. In this section, we will introduce two additional metrics for characterizing topics: relevance and applicability. Relevance measures the impact a topic has on people's lives and serves as a complement to nescience. Applicability measures how frequently a topic has been applied in other areas and enables us to identify new applications of already existing technologies.

## 7.1 Other Metrics

Nescience is a metric that can be used to identify the most interesting topics from the point of view of research. Since nescience is a measure of our ignorance about a topic, the higher the nescience the more opportunities for research. In this section we are going to propose other metrics that can complement nescience in the task of measuring the interest of research topics. These metrics will be helpful, not only for the classification of individual research topics, but also for the development of a methodology for the discovery of potential solutions to open problems (Section 7.2), and the discovery of new research topics (Section ??).

One of these new metrics for the classification of research topics is *relevance*. Relevance is a measure of the impact that a topic has on people's life. Intuitively, the higher the relevance of a topic, the higher its potential as a source of interesting problems, since we will be working on problems that affect many people directly. In order to measure the impact of a research topic we have to construct what we call the relevance graph, a figure that describes how people are affected by research topics (see Figure 7.1).

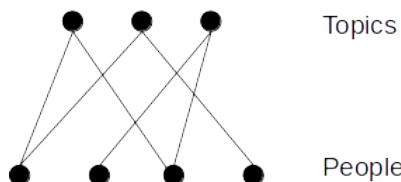


Figure 7.1: Relevance Graph

The relevance graph connects all known topics and all possible human beings. A link between a topic and a person means that this person is affected by the topic, not that he is interested in this particular topic. For example, somebody that is trying to find a cure to diabetes will not be connected to the research topic diabetes, however, somebody that actually suffer from diabetes will be. The higher the relevance of a topic, the higher its potential as a source of interesting problems to solve. In this sense, the research topic "how to cure diabetes" is more relevant than the research topic "how far dog fleas can jump", since more people are affected by the former than by the latter. The exact meaning of "*being affected by*" is an abstract concept that has to be approximated in practice. For example, we could claim that the wife of a man that has diabetes is somehow affected by the disease as well. In Part II of this book we will see some examples of how to approximate this abstract quantity.

Given these two metrics, nescience and relevance, we can provide a

quantitative measure of the interestingness of a topic as a source of interesting problems, that is, how likely is that the topic can be used as part of a new interesting research project. We define this quantity as a function of the relevance and nescience of the topic (for example, the normalized product of both quantities). Intuitively, a topic is interesting as a source of new problems if it has a large relevance (it has high impact in people's life) and a large nescience (it is not very well understood). In Figure 7.2 is graphically depicted the idea. For example, the Pythagoras' theorem has some relevance, since people life's can be indirectly affected by its implications, but since it is a very well understood theorem (our nescience is very low), it is not a very interesting research topic by itself. The World War I is very relevant, because it had a huge impact on many people's life, and also it is not very well understood topic as we have seen in Example ???. So, according to our definition, it has a huge potential as a source of new interesting research problems.

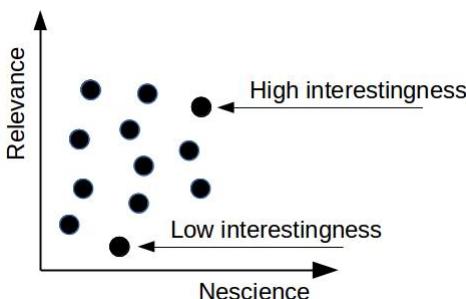


Figure 7.2: Interestingness of Topics

**■ Example 7.1** We can use the collection of Wikipedia articles to identify topics that are interesting as a source of new problems. The starting point of our analysis is the classification contained in the scientific disciplines category of Wikipedia. This category is organized<sup>1</sup> into the following main areas: "applied sciences", "behavioral sciences", "cognitive sciences", "formal sciences", "natural sciences", "physical sciences", and "social sciences". In order to evaluate the classification metrics proposed we have used the set of topics corresponding to all pages under any of the subcategories contained in the category "theory of computation" (a subcategory of the category "theoretical computer science", that belongs to the area "formal sciences"). Pages were cleaned up using the same procedure described in Example ??, and the relevance was estimated based on the number of unique visits to each

<sup>1</sup>Data from November 2014.

page (please, refer to Chapter 9 for more information about this process). Topics that fit our intuitive idea of problem, that is, not very well understood concepts with a high relevance, could include "arithmetical hierarchy" (0.72), "halting problem" (0.65), "floating point" (0.61), "quantum computer" (0.57), and "computable function" (0.55). ■

The Pythagoras' theorem is not a very interesting research topic by itself, however, it is a very important theorem, since it can be applied to solve many practical problems. A new metric is required to capture this concept of topic that is important because it can be used as a tool to solve other problems. We define the *maturity* of a topic as the inverse of nescience. Intuitively, the more mature a topic is the higher its potential applicability as a tool to solve other open problems, since we know very well how the topic works and how it can be successfully applied. In general, highly immature topics should not be applied to solve open problems, since they could provide wrong answers.

Besides maturity, we also introduce the metric of *applicability* of a topic. Applicability is based on the concept of *applicability graph*. An applicability graph is a directed graph between the research topics (see Figure 7.3). An arrow between two topics means that the first topic been successfully applied to explain or solve the second topic. For example, the topic "graph theory" has been applied to the topic "recommendation engines", since graph theory has been used to solve the problem of which products we should advertise to potential customers on Internet. Given this graph, we define the applicability of a topic as the number of problems to which the topic has been successfully applied. The higher the applicability of a topic, the higher its potential as a tool that can be applied to solve new problems.

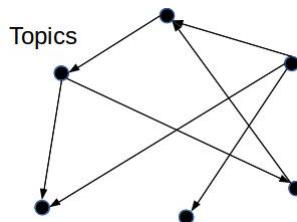


Figure 7.3: Applicability Graph

Finally, we define the concept of interestingness of a topic as a source of interesting tools, that is, how likely is that the topic can be used to solve a new problem, as a function of its maturity and applicability. Intuitively, a topic is interesting as a tool if it has been already applied to many other problems, and it is very well understood topic. For example, the Pythagoras' theorem, although not very relevant as a source of interesting problems, it is

very relevant as a source of new applications to other open problems.

■ **Example 7.2** We can use the collection of Wikipedia scientific articles to identify topics with high potential as tools. The procedure used in this example is similar to the one described in Example 7.1, except that the metric applicability has been estimated based in the collection of internal links within Wikipedia itself. The rationale is that if a page is referenced many times by other Wikipedia pages, it is highly likely that it contains useful information. Using this procedure, some examples of topics with high interest as tools in the area of theoretical computer science include "recursion" (0.42), "state space" (0.42), "abstract machine" (0.41), or "ternary numeral system" (0.48). ■

## 7.2 Interesting Research Questions

In the theory of nescience we distinguish two kinds of unknowns, the *known unknown* and the *unknown unknown*. By known unknown we mean all those already known problems for which we do not know their solutions, for example, nobody knows how to cure diabetes, but we know what diabetes is and we are aware that nobody knows how to cure it. By unknown unknown we mean the collection of unknown problems, that is, all those problems that have not been found yet, like for example the Eldermeyer's disease<sup>2</sup>. The area composed by the unknown unknown problems is a highly interesting one, since it contains those research topics that will be addressed in the future. One of the main goals of this book is to help scientists discover the topics that lay in this unknown unknown area, since that would bring to the present the research problems of the future (see Section ??). In this section we focus on how to solve the problems of the known unknown area.

An *interesting question* is a ordered pair of topics  $t$  and  $p$ , where  $t$  has a high interestingness as tool, and  $p$  has high interestingness as problem. Intuitively, the questions would be something like “can we apply the tool described by topic  $t$  to solve the problem described by topic  $p$ ?” . The interestingness of the new questions will be measured by means of a function of the interestingness of  $t$  and  $p$  themselves. In practice what we have to do is to compute all the possible combinations of tools and questions and select those with higher combined interestingness (see Figure 7.4).

An interesting question is *intradisciplinary* if it combines two topics that are studied in the framework of the same research area (e.g., computer science). An interesting question is *interdisciplinary* if it combines two topics of different research areas (e.g., computer science and philosophy). In

---

<sup>2</sup>We cannot say anything more about the Eldermeyer's disease, since Dra. Eldermeyer will born next year, and it will take her 34 years more to discover the disease named after her.

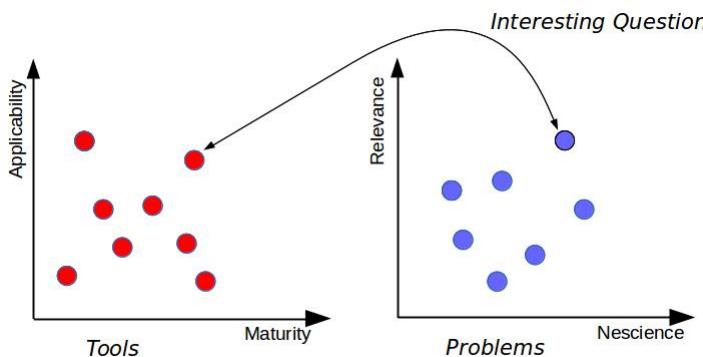


Figure 7.4: Interesting Questions

principle, the most innovative questions would be interdisciplinary questions, because the probability that somebody has thought about them is lower, since it requires specialists in both research areas working together to come up with that particular question.

**■ Example 7.3** We could combine the topics with high interestingness as tools found in the area of "computer science" with those topics with high interestingness as problems found in the area of "biochemistry" in order to find new interesting interdisciplinary questions. Some examples of the kind of questions we can find with this approach include: "can we use regular expressions to identify DNA genes?" or "can we use a recursive algorithm to characterize proteins tertiary structure?" ■

Once we have identified an interesting research question, we could use the concept of *conditional model* to see if the tool can help us to understand, or solve, the open problem. A conditional model of a topic  $t$  given a perfect model of a second topic  $s$  is also a string in the form  $\langle TM, a \rangle$ , but in this case we require that  $TM(\langle m_s^*, a \rangle) = t$ . That is, the Turin machine  $TM$  is able to print  $t$  when it has as input both, the incompressible part  $a$ , and a perfect model  $m_s^*$  for the topic  $s$ . If the conditional complexity of the open problem given the tool is smaller than its original complexity, then the tool is helpful to solve the problem. The more the tool reduce the length of the conditional model with respect to the original model, the better.

Please note that the methodology presented here is a generic one, in the sense that it can be applied to multiple domains, not only to the discovery of new interesting research questions. The metrics and methods described can be applied to any area where there is a large collection of interrelated describable objects and we are interested in discovering new, previously unconsidered, objects. The exact definition of concepts like relevance graph,

applicability graph or maturity will depend on the area in which the methodology is being applied. In Part II of this book, we will describe some examples of other applications, for example, to the identification of new software quality tests.

### 7.3 Relevance

Before to measure relevance of a topic, that is, its impact in people's life, we have to introduce the concept of *relevance graph*. The relevance graph is a graph that describes which people is affected by which research topics.

**Definition 7.3.1** We define the *relevance graph*, denoted by **RG**, as the bipartite graph  $\mathbf{RG} = (\mathcal{T}, \mathcal{P}, E)$ , where  $\mathcal{T}$  is the set of topics,  $\mathcal{P}$  the set of people, and  $E \subseteq \{(i, j) : i \in \mathcal{T}, j \in \mathcal{P}\}$  is the set of arcs between topics and people. An arc  $(i, j)$  belong to  $E$  if, and only if, person  $j$  is affected by topic  $i$ .

When we refer to the set of people  $\mathcal{P}$ , we are referring to all individuals in the world. An edge in the relevance graph indicates that someone is affected by a topic, rather than being interested in it. The exact meaning of "being affected by" is a difficult to define concept that is beyond the scope of this book. Therefore, our definition of the relevance graph is a mathematical abstraction. In Section ??, we will explore how to approximate this quantity in the case of scientific research topics. In Chapter ??, we will provide an alternative interpretation of the relevance graph in the context of measuring software quality.

■ **Example 7.4** A man who is affected by ALS (amyotrophic lateral sclerosis) disease will be connected to the ALS topic in the graph. His spouse will also be connected because she is likely to be impacted by the consequences of the disease as well. However, a researcher who is interested in ALS as a research problem, rather than someone suffering the disease, will not be connected to the ALS node in the graph. ■

Optionally, we can add a weight  $w_{ij} \in [0, 1]$  to the edges of the graph to specify the degree in which a person  $j$  is affected by a topic  $i$ . A weight of 1 could represent a life-or-death dependence, and 0 would mean that this person is not affected at all. In Figure 7.5 it is depicted an example of relevance graph.

**Definition 7.3.2** We define the *relevance* of a topic  $t \in \mathcal{T}$ , denoted by  $R(t)$ , as the degree of the node  $t$  in the relevance graph, that is,  $R(t) = \deg(t)$ .

Intuitively, the higher the relevance of a topic, the higher its potential as a source of interesting questions, since we will be working on a problem that

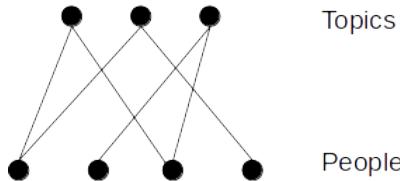


Figure 7.5: Relevance Graph

affects many people.

Sometimes it is convenient to work with a normalized version of the relevance of a topic.

**Definition 7.3.3** We define the *normalized relevance* of a topic  $t \in \mathcal{T}$ , denoted by  $\bar{R}(t)$ , as the normalized degree of the node  $t$  in the relevance graph, that is,  $\bar{R}(t) = \deg(t)/d(E)$ .

We could have computed also  $\deg(p)$ , that is, the number of edges that links to a person  $p$  in the relevance graph, as a measure of the number of topics that affects a particular person. However, this quantity is not used in the theory of nescience. The relation between  $\deg(t)$  and  $\deg(p)$  is given by the degree sum formula:

$$\sum_{t \in \mathcal{T}} \deg(t) = \sum_{p \in \mathcal{P}} \deg(p) = d(E)$$

Next proposition proves that adding more topics to a research project can only increase its relevance. Of course, a research project dealing with "life, the universe and everything" would be a highly relevant one, but very impractical as well. How to properly combine research topics will be described in Section 7.6.

**Proposition 7.3.1** Given any two topics  $t_1, t_2 \in \mathcal{T}$ , we have that  $R(t_1) + R(t_2) \geq R(t_1)$ .

*Proof.* Let  $S_1$  the set of people connected to topic  $t_1$  in the relevance graph, and  $S_2$  the set of people connected to topic  $t_2$ . Since  $d(S_1 \cup S_2) = d(S_1) + d(S_2) - d(S_1 \cap S_2)$ , and  $d(S_2) - d(S_1 \cap S_2) \geq 0$  we have that  $d(S_1 \cup S_2) \geq d(S_1)$  and thus  $R(t_1) + R(t_2) \geq R(t_1)$ . ■

Finally, we define the concept of interestingness of a topic as a source of interesting problems, that is, how likely is that the topic can be used in a new interesting research question, as a function of its nescience and relevance. We can visualize topics in a two-dimensional vector space, in which one dimension is nescience, and the other one is relevance. In this interpretation,

a natural choice of interestingness function would be the euclidean distance from the origin.

**Definition 7.3.4** Given a topic  $t \in \mathcal{T}$ , we define the *interestingness of the topic as a problem*, denoted by  $IP(t)$ , as:

$$IP(t) = \sqrt{v(t)^2 + R(t)^2}$$

Intuitively, a topic is interesting as a problem worth investigating if it has a large relevance (it has high impact in people's life) and a large nescience (it is not very well understood). In this sense, we are borrowing ideas from Popper's falsificationism: the more risky is a conjecture, the higher the advance achieved in science given its confirmation.

■ **Example 7.5** The fixed point theorem has some relevance, since people life's can be indirectly affected by its implications, but since it is a very well understood theorem (our nescience is very low), it is not a very interesting research problem by itself.

World War I is a very relevant topic, because it had a huge impact on many people's life, and also it is not very well understood topic, since it takes hundreds of pages to explain its causes, and there is no general agreement among the specialists. So, according to our definition, it is a very interesting research problem. ■

In practice, it is convenient to work with the normalized version of the relevance metric, so that we can avoid the case that our questions are always related to a small subset of extremely interesting topics.

$$IP(t) = \sqrt{v(t)^2 + \bar{R}(t)^2}$$

## 7.4 Applicability

As mentioned in Example 7.5, the fixed point theorem is not particularly compelling as a research problem on its own. However, it is an essential mathematical result since it has broad applications in proving many other theorems. In this section, we introduce the concept of *applicability*, which is a novel measure enabling us to identify which topics are important since they can serve as tools for understanding other topics. From a mathematical point of view, we say that a tool can be applied to a topic if the conditional nescience (see Section 6.4) of the topic given the tool is smaller than the unconditionally nescience of the topic.

Before to formally define the concept of applicability, first we have to introduce a new bipartite graph that describes which topics has been applied

as tools to other topics.

**Definition 7.4.1** We define the *applicability graph*, denoted by  $AG$ , as the directed graph  $AG = (\mathcal{T}, E)$ , where  $\mathcal{T}$  is the set of research topics, and  $E \subseteq \{(i, j) : i, j \in \mathcal{T}\}$ . An arc  $(i, j)$  belongs to  $E$  if  $N(t_i | t_j) < N(t_i)$ . The weight of the arc  $(i, j)$  is given by  $w_{ij} = N(t_i) - N(t_i | t_j)$ .

The applicability graph helps us identify those topics that can be potentially used as tools to understand other topics. The following definition formally introduces this idea.

**Definition 7.4.2** Given the applicability graph  $AG = (\mathcal{T}, E)$ , the *applicability* of a topic  $t_i \in \mathcal{T}$ , denoted by  $A(t_i)$ , is defined as the sum of the weights of the arcs in the outdegree of  $t_i$ , that is:

$$A(t_i) = \sum_{(i,j) \in E} w_{ij}$$

where  $E$  is the set of arcs in the applicability graph, and  $w_{ij}$  is the weight of the arc  $(i, j)$ .

A topic with higher applicability will have a greater overall impact on the understanding of other topics, and intuitively, the higher the applicability of a topic, the higher its potential as a tool that can be applied to solve open problems. If a tool has been successfully applied multiple times in the past to address open problems, it is more likely that it can be effectively used to solve other open problems as well.

Next proposition proves that the combination of two topics can only increase their applicability. That is, the more tools we have at our disposal, the more problems we could solve in principle.

**Proposition 7.4.1** Given any two topics  $t_1, t_2 \in \mathcal{T}$ , we have that  $A(t_1) + A(t_2) \geq A(t_1)$ .

*Proof.* Use the same argument than in Proposition 7.3.1. ■

In order to better compare the applicability of different topics and understand their relative importance in a standardized way, it is useful to introduce a normalized version of the applicability measure.

**Definition 7.4.3** Given the applicability graph  $AG = (\mathcal{T}, E)$ , the *normalized applicability* of a topic  $t_i \in \mathcal{T}$ , denoted by  $A_n(t_i)$ , is defined as the ratio of the applicability of  $t_i$  to the maximum possible applicability value,

expressed as:

$$\tilde{A}(t_i) = \frac{A(t_i)}{\max_{t_k \in \mathcal{T}} A(t_k)}$$

where  $A(t_i)$  is the applicability of topic  $t_i$  as defined in Definition 7.4.2, and  $\max_{t_k \in \mathcal{T}} A(t_k)$  is the maximum applicability value among all topics in  $\mathcal{T}$ .

Normalized applicability scales the original applicability value of a topic to a range between 0 and 1, allowing for easier comparison across different topics.

In practice, it is very difficult to compute the applicability graph, since for the majority of the topics, the quantity  $N(t_i | t_j)$  has not been estimated. In order to solve this problem, we can use an approximation to the applicability graph, in which the arcs have no weight, and an edge  $(i, j)$  represents that the topic  $j$  has been applied to solve the problem  $i$ . For example, there would be a direct link between the topics "graph theory" and "recommendation engines", since graph theory has been successfully applied to the problem of how to recommend purchase items to customers over Internet.

**Definition 7.4.4** We define the *simplified applicability graph*, denoted by SAG, as the directed graph  $SAG = (\mathcal{T}, E)$ , where  $\mathcal{T}$  is the set of research topics, and  $E \subseteq \{(i, j) : i, j \in \mathcal{T}\}$ . An arc  $(i, j)$  belongs to  $E$  if the topic  $j$  has been used to understand topic  $i$ .

Using this simplified version of the applicability graph, the concept of applicability becomes:

**Definition 7.4.5** We define the *simplified applicability* of a topic  $t \in \mathcal{T}$ , denoted by  $SA(t)$ , as the outdegree of that node in the applicability graph, that is:

$$SA(t) = \text{outdeg}(t)$$

And the corresponding normalized version becomes:

**Definition 7.4.6** Given the simplified applicability graph  $SAG = (\mathcal{T}, E)$ , the *simplified normalized applicability* of a topic  $t_i \in \mathcal{T}$ , denoted by  $SA_n(t_i)$ , is defined as the ratio of the simplified applicability of  $t_i$  to the maximum possible simplified applicability value, expressed as:

$$SA_n(t_i) = \frac{SA(t_i)}{\max_{t_k \in \mathcal{T}} SA(t_k)}$$

where  $SA(t_i)$  is the simplified applicability of topic  $t_i$  as defined in Definition 7.4.2, and  $\max_{t_k \in \mathcal{T}} SA(t_k)$  is the maximum applicability value among all topics in  $\mathcal{T}$ .

When using topics as tools, we are primarily interested in those topics that are better understood. Generally, relying on a background knowledge that is poorly understood is not advisable, even if it significantly reduces the (conditional) nescience of our problem. In the following definition, we will introduce the concept of the maturity of a topic as one minus its nescience.

**Definition 7.4.7** Given a topic  $t \in \mathcal{T}$ , we define the *maturity* of topic  $t$ , denoted as  $M(t)$ , as:

$$M(t) = v(t)^{-1}$$

Intuitively, the more mature a topic is, the greater its utility in solving other open problems. Highly immature topics should not be applied as tools to address open problems, as doing so would merely transfer our lack of understanding from one topic to another.

■ **Example 7.6** Linear regression is a highly mature topic, since its nescience is very small. ■

Finally, we define the concept of interestingness of a topic as a source of interesting tools, that is, how likely is that the topic can be used to solve a new problem, as a function of its maturity and applicability. Similar to the definition of relevance introduced in the previous section, topics can be visualized within a two-dimensional vector space, where one dimension represents maturity and the other represents applicability. The interestingness of a topic will be determined by its Euclidean distance from the origin.

**Definition 7.4.8** Given a topic  $t \in \mathcal{T}$ , we define the *interestingness of the topic as a tool*, denoted by  $IT(t)$ , as:

$$IT(t) = \sqrt{M(t)^2 + A(t)^2}$$

Intuitively, a topic is considered interesting as a tool if it is thoroughly understood and has already been applied to numerous other problems.

■ **Example 7.7** The Pythagorean theorem (in a right-angled triangle, the square of the length of the hypotenuse is equal to the sum of the squares of the other two sides) is undoubtedly one of the most widely used and applied theorems in various fields and practical situations, including but not limited to: engineering (calculating distances, angles, and forces in structures and mechanical systems), architecture (determining lengths and angles in building design and construction projects), land surveying (measuring

distances and calculating areas of land parcels), physics (analyzing problems in mechanics, optics, and electromagnetism), computer graphics and game development (calculating distances and angles in 2D and 3D spaces) or trigonometry(serving as a foundation for the study of trigonometric functions and their applications). The widespread application of the Pythagorean theorem to so many fields is due to its simplicity and its ability to describe complex geometric relationships. ■

## 7.5 Interesting Questions

In the quest to uncover novel research questions, combining existing topics from various fields can yield fascinating results. By applying tools and methodologies from one topic to solve problems in another, researchers can explore innovative avenues and potentially make groundbreaking discoveries. In the following section, we delve into the methodology of combining topics to generate questions, along with the concept of interestingness and interdisciplinary research.

In our methodology, an interesting question emerges from the combination of two pre-existing topics. Given a pair of topics in  $t_1, t_2 \in \mathcal{T}$ , the question can be framed as "*can we apply the tool described by topic  $t_1$  to solve the problem described by topic  $t_2$ ?*".

**Definition 7.5.1** Given two topics  $t_1, t_2 \in \mathcal{T}$ , a *question*, denoted by  $Q_{t_1 \rightarrow t_2}$ , is the ordered pair  $(t_1, t_2)$ .

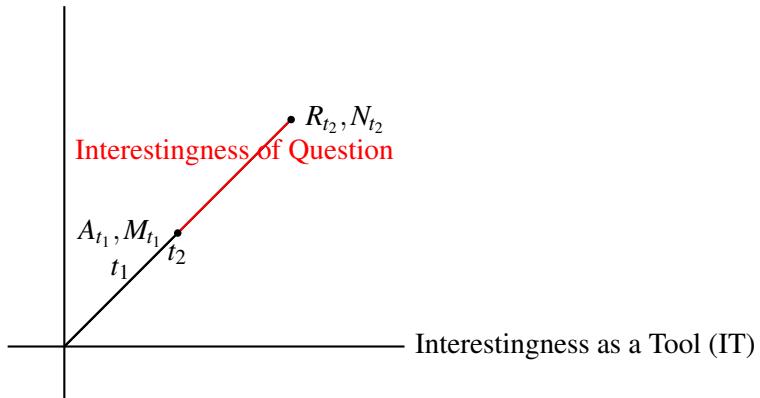
The most intriguing questions emerge when topic  $t_1$  exhibits high interestingness as a tool, and topic  $t_2$  demonstrates high interestingness as a problem. We define the interestingness of a question using the Euclidean distance, taking into account the interestingness of topics  $t_1$  and  $t_2$  as points in a two-dimensional space, with coordinates  $(A_{t_1}, M_{t_1})$  and  $(R_{t_2}, N_{t_2})$ , respectively. The coordinates of the resulting vector represent the interestingness of a question with  $t_1$  as a tool and  $t_2$  as a problem.

**Definition 7.5.2** Let  $t_1$  and  $t_2 \in \mathcal{T}$  be two topics. The *interestingness* of the question  $Q_{t_1 \rightarrow t_2}$ , denoted by  $IQ_{t_1 \rightarrow t_2}$ , is given by:

$$IQ_{t_1 \rightarrow t_2} = \sqrt{(A_{t_1} + R_{t_2})^2 + (M_{t_1} + N_{t_2})^2}$$

Employing the Euclidean distance in this way enables a geometric interpretation of the interestingness of a question based on the vector representation of topics in the interestingness space. The larger the magnitude of the vector representation, the higher the interestingness of the resulting question.

### Interestingness as a Problem (IP)



In practice, we must calculate all possible combinations of topics with high interestingness as tools and those with high interestingness as problems. We then select the combinations with the highest interestingness as questions. Naturally, most questions generated using this approach will be meaningless, much like those arising during brainstorming sessions when researchers attempt to identify new tools for tackling difficult problems.

This methodology can be applied in other scenarios as well. For instance, a researcher familiar with problem  $p$  might be interested in finding applicable tools to solve it. Similarly, a researcher specializing in tool  $t$  may be interested in discovering open problems where his expertise can be applied.

The above procedure can be easily generalized to encompass multiple tools and possibly multiple problems. This leads to the application of two tools to a given problem ( $t_1 + t_2 \rightarrow p$ ), the application of a single tool to the combination of two problems ( $t \rightarrow p_1 + p_2$ ), and so on. The exact meaning of these tool and problem combinations depends on the topics themselves and is left to the researcher's creative interpretation.

At times, it is beneficial to limit our search to specific areas of knowledge to identify interesting questions.

**Definition 7.5.3** Let  $\mathcal{A} \subset \mathcal{T}$  be a research area, and  $t_1, t_2 \in \mathcal{T}$  be two topics. If both topics belong to the same area, meaning  $t_1, t_2 \in \mathcal{A}$ , we say that the question  $Q_{t_1 \rightarrow t_2}$  is *intradisciplinary*, otherwise, we say that the question is *interdisciplinary*.

The most innovative questions tend to be interdisciplinary, as they have a lower likelihood of having been considered previously. This is because they require collaboration between specialists from different research areas. Interdisciplinary questions address our requirement **C1** for interesting questions, which dictates that the question must be new and original.

## 7.6 New Research Topics

**TODO:** Review this section

In the previous section our focus was in how to find new interesting research questions. In this section we will go one step beyond, and we will show how to identify new, previously unknown, research topics.

**Definition 7.6.1** In the two-dimensional space defined by relevance and nescience, the *unknown frontier*, denoted by  $\mathbb{F}$ , is defined as the following arc:

$$\mathbb{F} = \{(x, y) \mid x^2 + y^2 = \max(\{N_t^2 + R_t^2, t \in T'\}), x > 0, y > 0\}$$

If we plot all the known research topics according to their relevance and nescience, the unknown frontier will cover them. Intuitively the *unknown frontier* marks the frontier between what we do not know and we are aware that we do not know (we do not fully understand those topics), and what we do not know and we are not yet aware that we do not know those topics. This intuitive property is in general terms, since it may happen that some unknown topics lie under the unknown frontier as well.

**Definition 7.6.2** Lets  $T'$  the set of known research topics. The *new topics area*, denoted by  $\mathbb{S}$ , is defined by:

$$\mathbb{S} = \{(x, y) \mid x^2 + y^2 > \max(\{N_t^2 + R_t^2, t \in T'\}), x > 0, y > 0\}$$

The new topics area contains all those unknown topics that we are not aware we do not know them (unknown unknown). The big issue is how to reach this new topics area if we do not know anything about the topics included in that area.

**Proposition 7.6.1** Let  $r \in T$  be a topic, if  $N_r^2 + R_r^2 > \max(\{N_t^2 + R_t^2, t \in T'\})$  then  $t \in T \setminus T'$ .

*Proof.* Let  $N_r^2 + R_r^2 > \max(\{N_t^2 + R_t^2, t \in T'\})$  and suppose that  $r \in T'$ , then have that  $\max(\{N_t^2 + R_t^2, t \in T'\}) \geq N_r^2 + R_r^2$  that it is a contradiction, and so  $t \in T \setminus T'$ . ■

Proposition 7.6.1 together with the fact that the combined nescience of two topics is higher than the nescience of any of them isolated (Proposition XXX), and that their combined relevance is higher than the relevance of any of them (Proposition 7.3.1), a possible approach to identify new topics could be by means of combining already existing interesting problems. In Figure 7.6 is depicted graphically the idea.

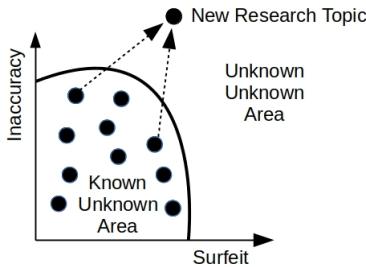


Figure 7.6: The discovery of new research topics

**Definition 7.6.3** Given two topics  $t_1, t_2 \in T'$ , a *new topic*, denoted by  $S_{\{t_1, t_2\}}$ , is the unordered pair  $\{t_1, t_2\}$ .

The exact meaning of the new topic that results as the combination of topics  $t_1$  and  $t_2$  is left to the creative interpretation of the researcher.

**Definition 7.6.4** The *interestingness* of the new topic, denoted by  $IS_{\{t_1, t_2\}}$ , is given by:

$$IS_{\{t_1, t_2\}} = R_{t_1}R_{t_2} + N_{t_1}N_{t_2}$$

In practice, what we have to do is to compute all possible combination of those topics with very large interestingness as problems  $IP_t$  with themselves, and select the combinations with higher  $IS$ . Of course, some of the combinations generated would be totally meaningless. Advanced techniques from the area of natural language processing or machine learning could be used to try filter out those nonsense combinations.

**Definition 7.6.5** Let  $A \subset T$  a research area, and  $t_1, t_2 \in T$  two topics. If both topics belongs to the same area, that is  $t_1, t_2 \in A$ , we say that the new topic  $S_{\{t_1, t_2\}}$  is *intradisciplinary*, otherwise, we say that the topic is *interdisciplinary*.

Again, the most innovative new topics would be by the combination of interdisciplinary topics, because the probability that somebody has already thought about them is lower.

## 7.7 Classification of Research Areas

In the same way we studied the nescience of research areas (see Section 6.5), we could also study the interestingness of research areas.

**Definition 7.7.1** Given a research area  $A \subset T$ , we define the *average interestingness of the area as a source of interesting tools* by

$$IT_A = \frac{1}{n} \sum_{t \in A} IT_t$$

and the *average interestingness of the area as a source of interesting problems* by

$$IP_A = \frac{1}{n} \sum_{t \in A} IP_t$$

where  $n$  is the cardinality of  $A$ .

In this way we could compute the interestingness of mathematics, physics, biology, social sciences, and other disciplines as a source of interesting tools and problems. Other alternative measures of centrality and dispersion could be used for the characterization of research areas as well.

As I will show in Chapter ??, the interestingness of mathematics as a source of tools is higher than the interest of social sciences, since mathematics is composed of topics with a high applicability that are very well understood, and that is not the case, in general, for social sciences. On the other hand, the interestingness of social sciences as a source of problems is higher than the interest of mathematics, since the topics studied by the social sciences are more relevant to humankind <sup>3</sup> and, in general, not very well understood.

■ **Example 7.8** We could use the interestingness of an area to identify research areas in decay. A knowledge area is in decay (from the research point of view) if it has no enough interesting research problems. For example, although the aerodynamics of zeppelins is not fully understood (still some nescience), it is not longer useful (low relevance), since people does not use zeppelins to travel anymore, and so, the average interestingness is very low. Another example of area in decay is classical geometry: although it is relevant, our understanding of this subject is nearly perfect, since there are almost no unsolved problems, and so, its average nescience is very low. However, on the contrary to what happens in case of the aerodynamics of zeppelins, classical geometry is still very interesting as a source of tools. ■

It is worth to mention that we could add other metrics to provide a finer, or even alternative, characterization of the unknown unknown area. For example, we could add to nescience and relevance a third dimension with

---

<sup>3</sup>Please mind that I am not saying that the topics addressed by mathematics are not relevant to humankind, what I am saying is that, in relative terms, the problems addressed by social sciences have a higher relevance.

the probability that a topic description is true. However, these extended or alternative characterizations will be not considered in this book. Fortunately, the idea of how to reach the unknown unknown area is the same, regardless of the number and the metrics used (as long as these metrics satisfy some minimal mathematical properties, described in Chapters 6 and 7).

## References

Popper and falsificationism

**Review the following references**

- Freitas, A. A. (1999). On Objective Measures of Rule Surprisingness. In Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98), pp. 1-9.
- Gureckis, T. M., & Goldstone, R. L. (2009). How You Named Your Child: Understanding The Relationship Between Individual Decision Making and Collective Outcomes. *Topics in Cognitive Science*, 1(4), 651-674.
- Klein, J. T. (1990). Interdisciplinarity: History, Theory, and Practice. Wayne State University Press.
- Kuhn, T. S. (1962). The Structure of Scientific Revolutions. University of Chicago Press.
- Newell, A., & Simon, H. A. (1972). Human Problem Solving. Prentice-Hall.
- Page, S. E. (2007). The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies. Princeton University Press.
- Rescher, N. (1986). The Riddle of Existence: An Essay in Idealistic Metaphysics. University Press of America.
- Schelling, T. C. (1978). Micromotives and Macrobbehavior. W. W. Norton & Company.



# Part 3: Applications

## 8 Machine Learning ..... 159

- 8.1 Nescience Python Library
- 8.2 A Note About Compression
- 8.3 Miscoding
- 8.4 Inaccuracy
- 8.5 Surfeit
- 8.6 Nescience
- 8.7 Auto Machine Classification
- 8.8 Auto Machine Regression
- 8.9 Time Series
- 8.10 Anomaly Detection
- 8.11 Decision Trees
- 8.12 Algebraic Model Selection
- 8.13 The Analysis of the Incompressible

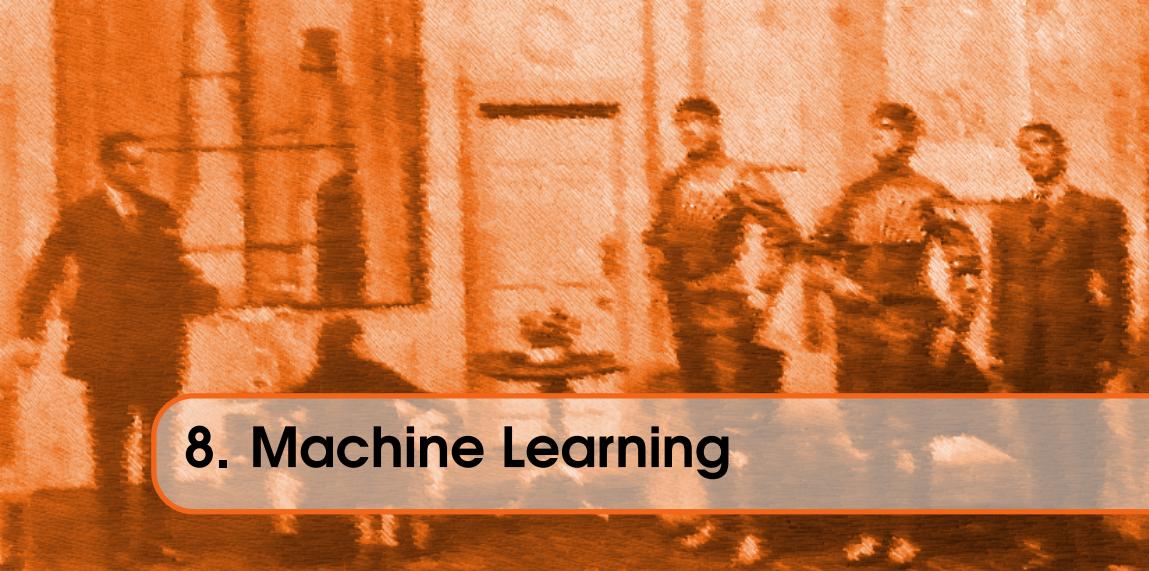
## 9 Analysis of Science ..... 211

- 9.1 Describing Current Knowledge
- 9.2 Measuring Knowledge
- 9.3 Measuring Research Areas
- 9.4 The Evolution of Knowledge
- 9.5 The Demarcation Problem
- 9.6 Perfect knowledge
- 9.7 Unknown-unknown

## 10 Computational Creativity ..... 243

- 10.1 Maturity
- 10.2 Interesting Research Questions
- 10.3 New Research Topics
- 10.4 References
- 10.5 Future Work





## 8. Machine Learning

*There are no difficult problems,  
only lack of imagination.*

Antonio García

We have seen that the most difficult problems to which we can apply the results of the theory of nescience arise when set of entities  $\mathcal{E}$  under study is composed by abstract elements. The difficulty with abstract entities is that it does not exits a way to encode them as strings of symbols so we can effectively reconstruct them. In practice, a possible approach to deal with this problem is to run an experiment and collect the results, as we do in case of physics. An alternative approach would be to take a collection of measurements, like for example, by means of observing the behavior of users in an online social network.

This chapter is devoted to how to apply the concept of minimum nescience to the area of machine learning. We assume that the entities under study are encoded as a dataset  $\mathbb{X}$  composed by  $n$  training vectors of  $p$  predictors and a response variable  $y$  (see Section 16.2).

We will start by providing practical approximations for the concepts of miscoding, inaccuracy and surfeit when the entities are encoded as datasets,

and then we will show how to combine them in the single quantity of nescience. These approximations will allow us to introduce the *minimum nescience principle*, a technique designed to automate the process of finding optimal models in machine learning (auto-machine learning).

Besides introducing these approximations, we will show how to apply them to solve practical problems. The examples will be based on the `nescience`<sup>1</sup> library, an open source python library that provides an implementation of the ideas included in this chapter.

## 8.1 Nescience Python Library

The `nescience` library is an open source Python library that provides an implementation of the ideas included in this book applied to the area of machine learning. The library follows the API and conventions of the highly popular `scikit-learn` machine learning tool suite, and so, it can be combined with the methods provided by this package.

The `nescience` library can be installed with the `pip` utility:

```
pip install nescience
```

In the web page that accompanies this book, the reader can find a collection of notebooks for the `jupyter-lab` environment describing how the library works. For each subsection of this chapter, there is a notebook that implements all the examples included, so that the reader can repeat and play with them. Additional information about the `nescience` library, and a reference of the API provided, can be also found in the web page of the book.

## 8.2 A Note About Compression

As it is customary, the Kolmogorov complexity  $K(s)$  of a string  $s$  will be approximated by the length of the compressed version of that string using a standard compressor, that is, we will use the normalized compression distance:

$$E_Z(\mathbf{x}_j, \mathbf{y}) = \frac{\max\{\hat{K}_Z(\mathbf{x}_j | \mathbf{y}), \hat{K}_Z(\mathbf{y} | \mathbf{x}_j)\}}{\max\{\hat{K}_Z(\mathbf{x}_j), \hat{K}_Z(\mathbf{y})\}}$$

where  $\hat{K}_Z(s)$  denotes the length of the compressed version of the string  $s$  using the compressor  $Z$ . In the particular case of having a vector  $\mathbf{x} = \{x_1, \dots, x_n\}$  of measurements, the string  $s$  to be compressed will be the concatenation of the

---

<sup>1</sup><https://github.com/rleiva/nescience>

encoded values  $s = \langle x_1, \dots, x_n \rangle$ . We prefer the following equivalent definition of normalized compression dinstance, since in practice it is easier to compute the joint distribution of two vectors than the conditional distribution:

$$E_Z(\mathbf{x}_j, \mathbf{y}) = \frac{\hat{K}_Z(\mathbf{x}_j, \mathbf{y}) - \min\{\hat{K}_Z(\mathbf{x}_j), \hat{K}_Z(\mathbf{y})\}}{\max\{\hat{K}_Z(\mathbf{x}_j), \hat{K}_Z(\mathbf{y})\}}$$

As compression technique we will use a code  $C$  with minimal length, given the relative frequencies of the different observed values (see Section 14.3). If  $\mathbf{x}$  is a qualitative vector (either a feature or the target variable) taking values from a set labels  $\mathcal{G} = \{g_1, \dots, g_\ell\}$ , that is  $\mathbf{x} \in \mathcal{G}^n$ , the quantity  $\hat{K}_C(\mathbf{x})$  can be computed as:

$$\hat{K}_C(\mathbf{x}) = - \sum_{i=1}^l \log_2 \frac{\sum_{j=1}^n I(x_j = g_i)}{n}$$

If the case of  $\mathbf{x}$  being based on a continuous random variable, we cannot calculate the probability of  $x_j$  since, in general, the underline probability distribution of  $\mathbf{x}$  is unknown. Moreover, we have that  $P(x_j) = 0$  for all  $j$ . In order to approximate the value  $K(\mathbf{x})$  using a minimal length code  $C$ , we have to discretize first the vector  $\mathbf{x}$  into a collection of intervals.

A discretization algorithm is a mapping between a (possibly huge) number of numeric values and a reduced set of discrete values, and so, it is a process in which some information is potentially lost. The choice of discretization algorithm is something that could have a high impact in the practical computation of the nescience. We are interested in a discretization algorithm that produces a large number of intervals (low bias), with a large number of number of observations per interval (low variance). Common techniques include *equal width discretization*, *equal frequency discretization* and *fixed frequency discretization*. However, these techniques require the optimization of an hyperparameter, and so, they are not suitable for our purposes.

In the nescience library we use a proportional discretization approach (see Section ??), where the number of intervals  $m$  and the number of obser-vations per interval  $s$  are equally proportional to the number of observations  $n$ . In particular, in the nescience library we set  $s = m = \sqrt{n}$ .

Using this discretization procedure, we can approximate the Kolmogorov complexity of a vector  $\mathbf{x}$  by:

$$\hat{K}_C(\mathbf{x}) = - \sum_{i=1}^m \log_2 \frac{\sum_{j=1}^n I(x_j \in D_i)}{n}$$

where  $D_i$  is the interval defined by the end points  $(i-1, i)$ .

The quantity  $\hat{K}_C$  can be generalized to the an arbitrary number of  $m$  vectors  $\hat{K}_C(\mathbf{x}_1, \dots, \mathbf{x}_m)$  composed by  $n$  samples each, by considering the joint encoded vector

$$\langle \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m \rangle = \{\langle x_{11}, x_{12}, \dots, x_{1m} \rangle, \dots, \langle x_{n1}, x_{n2}, \dots, x_{nm} \rangle\}$$

**Proposition 8.2.1** The normalized compression distance of two vectors  $x$  and  $y$  computed using a compressor based on optimal codes is equivalent to the normalized mutual information of these two vectors, that is:

$$NCD_C(\mathbf{x}, \mathbf{y}) = 1 - \frac{I(\mathbf{x}; \mathbf{y})}{\max\{H(\mathbf{x}), H(\mathbf{y})\}}$$

*Proof.* We have to prove that

$$\begin{aligned} NCD(\mathbf{x}, \mathbf{y}) &= \frac{C(\mathbf{x}, \mathbf{y}) - \min\{C(\mathbf{x}), C(\mathbf{y})\}}{\max\{C(\mathbf{x}), C(\mathbf{y})\}} = \frac{nH(\mathbf{x}, \mathbf{y}) - \min\{nH(\mathbf{x}), nH(\mathbf{y})\}}{\max\{nH(\mathbf{x}), nH(\mathbf{y})\}} \\ &= \frac{H(\mathbf{x}, \mathbf{y}) - \min\{H(\mathbf{x}), H(\mathbf{y})\}}{\max\{H(\mathbf{x}), H(\mathbf{y})\}} \end{aligned}$$

We have to consider two cases. In case 1 we assume that  $H(\mathbf{x}) > H(\mathbf{y})$ , and so

$$\begin{aligned} NCD(\mathbf{x}, \mathbf{y}) &= \frac{H(\mathbf{x}, \mathbf{y}) - H(\mathbf{y})}{H(\mathbf{x})} = \frac{H(\mathbf{y}) + H(\mathbf{x} | \mathbf{y}) - H(\mathbf{y})}{H(\mathbf{x})} = \frac{H(\mathbf{x} | \mathbf{y})}{H(\mathbf{x})} \\ &= \frac{H(\mathbf{x}) - (H(\mathbf{x}) - H(\mathbf{x} | \mathbf{y}))}{H(\mathbf{x})} = 1 - \frac{I(\mathbf{x}; \mathbf{y})}{H(\mathbf{x})} \end{aligned}$$

and in case of  $H(\mathbf{y}) > H(\mathbf{x})$  we have that

$$\begin{aligned} NCD(\mathbf{x}, \mathbf{y}) &= \frac{H(\mathbf{x}, \mathbf{y}) - H(\mathbf{x})}{H(\mathbf{y})} = \frac{H(\mathbf{x}) + H(\mathbf{y} | \mathbf{x}) - H(\mathbf{x})}{H(\mathbf{y})} = \frac{H(\mathbf{y} | \mathbf{x})}{H(\mathbf{y})} = \frac{H(\mathbf{y}) - (H(\mathbf{y}) - H(\mathbf{x} | \mathbf{y}))}{H(\mathbf{y})} \\ &= 1 - \frac{I(\mathbf{x}; \mathbf{y})}{H(\mathbf{y})} \end{aligned}$$

and so

$$NCD_c(\mathbf{x}, \mathbf{y}) = 1 - \frac{I(\mathbf{x}; \mathbf{y})}{\max\{H(\mathbf{x}), H(\mathbf{y})\}}$$

■

The quantity  $1 - \frac{I(\mathbf{x}; \mathbf{y})}{\max\{H(\mathbf{x}), H(\mathbf{y})\}}$  has been already proposed in [ferri2009experimental] as a candidate definition of the concept of Normalized Mutual Information. However, to the best of our knowledge, it has not been used in practice.

■ **Example 8.1**

R

In order to properly approximate the complexity of a continuous feature, we have to use a discretization algorithm that does not change the distribution of the observed values. For example, the kmeans algorithm, given that the optimization criteria is to minimize

In the nescience we use a uniform discretization, in which all the intervals have the same length. However, this approach is not optimal, especially when we are in a two dimensional space, in which we have to discretize the joint distribution of two attributes  $x$  and  $y$ .

For example, in the next figure, we can see a cloud of points, and how a uniform distribution has many intervals without points.

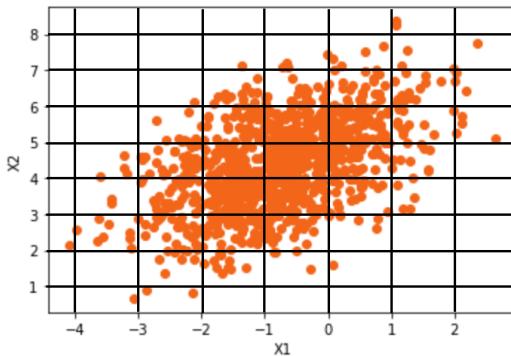


Figure 8.1: Uniform discretization.

A better approach would be to compute the convex hull around the actual cloud of points, and then divide the space into sqrt intervals of the same length, by for example, computing the centroid using a kmeans algorithm and then using voronoi polygons. However, up to the best of our knowledge, a discretization algorithm based on a constrained version of the kmeans in which the distribution of the points is not changes, has not been developed.

## 8.3 Miscoding

In Section 3.1.1 we introduced the concept of miscoding as a quantitative measure of how well a string based encoding  $r \in \mathcal{R}$  represents a research entity from  $\mathcal{E}$ . The miscoding of a representation  $r$  was defined as:

$$\mu(r) = \min_{s \in \mathcal{R}_{\mathcal{E}}} \frac{\max\{K(s | r), K(r | s)\}}{\max\{K(s), K(r)\}}$$

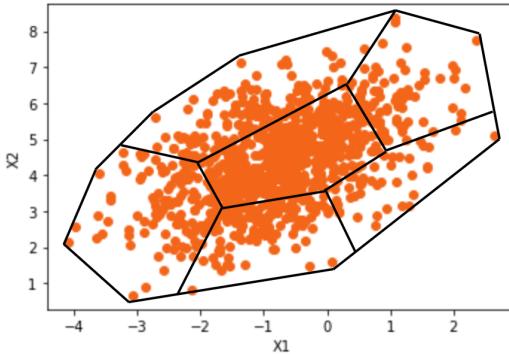


Figure 8.2: Convex hull and kmeans discretization.

and we saw that this quantity cannot be computed in practice for the general case. First of all because it requires a computation from an abstract oracle machine, second because it is based on the uncomputable Kolmogorov complexity, and third because it does not take into account the entity  $e$  in which we are interested.

In this section we are going to see how this concept can be adapted in practice to compute the error made by using a dataset  $\mathbf{X}$  as a representation of a response variable  $\mathbf{y}$  (see Section 16.2). Our goal is double, in one hand we are interested in measuring the quality of the dataset  $\mathbf{X}$  as a predictor of the variable  $\mathbf{y}$ , and in the other we want to identify those features  $\mathbf{x}_j$  of  $\mathbf{X}$  that have the higher predictive power for  $\mathbf{y}$ . This is the problem addressed by discriminative models (see Section 16.2.5) in which we want to estimate the conditional distribution  $P(\mathbf{y} | \mathbf{X})$ .

Given a training dataset  $\mathbf{X}$ , we can approximate the miscoding of a feature  $\mathbf{x}_j$  for the target variable  $\mathbf{y}$  by computing the normalized information distance between  $\mathbf{x}_j$  and  $\mathbf{y}$  (see Section 15.5):

$$E(\mathbf{x}_j, \mathbf{y}) = \frac{\max\{K(\mathbf{x}_j | \mathbf{y}), K(\mathbf{y} | \mathbf{x}_j)\}}{\max\{K(\mathbf{x}_j), K(\mathbf{y})\}}$$

The Kolmogorov complexity  $K(\mathbf{v})$  of a vector  $\mathbf{v}$  will be approximated by the length of the compressed version of that vector  $\hat{K}_C(\mathbf{v})$  using as compressor a minimal length code  $C$  (see Section 8.2).

**Definition 8.3.1** Let  $\mathbf{y}$  be a response variable,  $\mathbf{X}$  a dataset composed by  $p$  features, and  $\mathbf{x}_j$  the  $j$ -th feature. We define the regular feature miscoding

of  $\mathbf{x}_j$  as a representation of  $\mathbf{y}$ , denoted by  $\hat{\mu}(\mathbf{x}_j, \mathbf{y})$ , as:

$$\hat{\mu}(\mathbf{x}_j, \mathbf{y}) = \frac{\hat{K}_C(\mathbf{x}_j, \mathbf{y}) - \min\{\hat{K}_C(\mathbf{x}_j), \hat{K}_C(\mathbf{y})\}}{\max\{\hat{K}_C(\mathbf{x}_j), \hat{K}_C(\mathbf{y})\}}$$

Intuitively, the quantity  $\hat{\mu}(\mathbf{x}_j, \mathbf{y})$  is a measure of the effort, as the length of a computer program and in relative terms, required to fully encode  $\mathbf{y}$  assuming a knowledge of  $\mathbf{x}_j$ , and the other way around. The lower this value, the better would be the quality of  $\mathbf{x}_j$  as a predictor for  $\mathbf{y}$ .

**Example 8.2** Let's  $\mathbf{y}$  be a target variable composed by 1.000 random samples that follows a normal distribution  $N(3, 1)$  with mean  $\mu = 3$  and standard deviation  $\sigma = 1$ ,  $\mathbf{x}_1$  be a predictor feature that is equal to  $\mathbf{y}$  with some random noise, that is  $\mathbf{x}_1 = \mathbf{y} + N(3, 1)/10$ , and  $\mathbf{x}_2$  be a second predictor based on random samples from a exponential distribution with a rate of  $\lambda = 1$ .

```
from scipy.stats import norm, expon

y = norm.rvs(loc=3, scale=1, size=10000)
x1 = y + norm.rvs(loc=3, scale=1, size=10000) / 10
x2 = expon.rvs(size=10000)
```

We can use the Nescience library to compute the miscoding of the features  $\mathbf{x}_1$  and  $\mathbf{x}_2$  when they encode the target variable  $\mathbf{y}$ .

```
from fastautoml.miscoding import Miscoding
import numpy as np

X = np.column_stack((x1, x2))

mCoding = Miscoding()
mCoding.fit(X, y)
mCoding.miscoding_features(mode="regular")
```

The output of the library would be something similar to the following<sup>2</sup>:

```
array([0.27445364, 0.9934222])
```

As it was expected the miscoding of  $\hat{\mu}(\mathbf{x}_1, \mathbf{y})$  is much smaller than the miscoding of  $\hat{\mu}(\mathbf{x}_2, \mathbf{y})$ . In this case, we should prefer  $\mathbf{x}_1$  over  $\mathbf{x}_2$  as a predictor of  $\mathbf{y}$ .

Sometimes we will use the normalized version of the complements of the individual miscodings, that is  $\frac{1-\hat{\mu}(\mathbf{x}_i, \mathbf{y})}{\sum_{i=j}^p 1-\hat{\mu}(\mathbf{x}_i, \mathbf{y})}$ , instead of the regular ones

<sup>2</sup>Since we are generating a list of 1.000 random samples, the reader could get a slightly different result when running this example.

$\hat{\mu}(\mathbf{x}_i, \mathbf{y})$ , because they are easier to compare with other feature selection techniques, and because they have a visually appealing interpretation. We call this version of miscoding the *adjusted* feature miscoding.

■ **Example 8.3** In this example we are going to generate a synthetic dataset where the target variable  $\mathbf{y}$  is a collection of normally-distributed clusters of points, and the training set  $\mathbf{X}$  is composed by both, relevant and irrelevant predictors. In particular we will generate 1.000 samples composed by 20 features that describe 10 clusters; only 4 of the features are relevant for prediction, and the other remaining 6 are just random values.

In Figure 8.3 we can see a two-dimensional projection of this dataset, along the hyperplane composed by features 8 and 10.

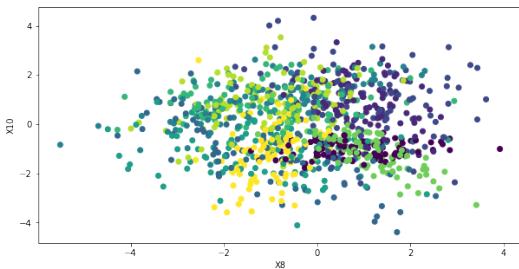


Figure 8.3: Gaussian Blob Cluster.

```
from fastautoml.miscoding import Miscoding
from sklearn.datasets.samples_generator import make_classification

X, y = make_classification(n_samples=1000, n_features=20, n_informative=4,
                           n_redundant=0, n_classes=10, n_clusters_per_class=1, flip_y=0)

miscoding = Miscoding()
miscoding.fit(X, y)
msd = miscoding.miscoding_features(mode='adjusted')
```

We will use the adjusted version of the miscoding for an easier comparison with other feature selection techniques. If we plot the results (see Figure 8.4) we will see that the library has successfully identified the four relevant predictors ( $\mathbf{x}_3$ ,  $\mathbf{x}_8$ ,  $\mathbf{x}_{10}$  and  $\mathbf{x}_{16}$ ). Since we are using the adjusted version of miscodings, the higher the value the better, and mind that actual values have to be interpreted in relative terms.

We can compare miscoding with correlation, a common technique used in machine learning to identify the most relevant features of a dataset. In Figure 8.5 is shown correlation between the individual features that compose  $\mathbf{X}$  and the target variable  $\mathbf{y}$ . As we can observe, correlation fails to properly identify one of the relevant features ( $\mathbf{x}_3$ ).

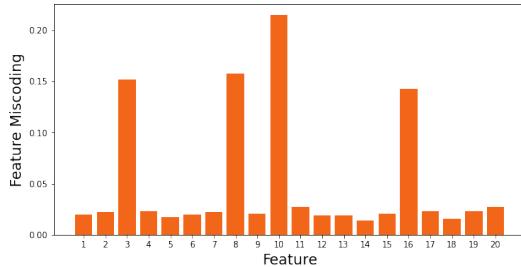


Figure 8.4: Miscoding of a Synthetic Dataset.

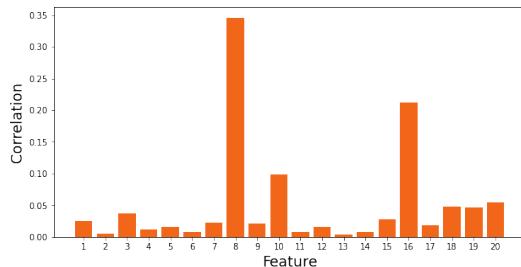


Figure 8.5: Correlation of a Synthetic Dataset.

Feature miscoding allow us to identify the most relevant features of a training dataset  $\mathbf{X}$ , but it cannot be used to compute the miscoding of the dataset itself. If we start with a miscoding of 1 (full unknown), and subtract the miscodings of the individual features, we will end up with a negative miscoding, something that it is not allowed by our theory. If we use the adjusted version, the dataset miscoding will be 0 for all datasets, which is against our intuition that not all possible datasets  $\mathbf{X}$  represent equally well a target variable  $\mathbf{y}$ . According to the theory of nescience, we expect that non-relevant features add, instead of subtract, to the global miscoding of the dataset.

In order to address this problem, we have to introduce the concept of partial miscoding of a feature, as the difference between the adjusted and normalized miscodings.

**Definition 8.3.2** Let  $\mathbf{y}$  be a target variable,  $\mathbf{X}$  a dataset composed by  $p$  features, and  $\mathbf{x}_j$  the  $j - th$  feature. We define the partial miscoding of  $\mathbf{x}_j$

as a representation of  $\mathbf{y}$ , denoted by  $\tilde{\mu}(\mathbf{x}_j, \mathbf{y})$ , as:

$$\tilde{\mu}(\mathbf{x}_i, \mathbf{y}) = \frac{1 - \hat{\mu}(\mathbf{x}_i, \mathbf{y})}{\sum_{j=1}^p 1 - \hat{\mu}(\mathbf{x}_j, \mathbf{y})} - \frac{\hat{\mu}(\mathbf{x}_i, \mathbf{y})}{\sum_{j=1}^p \hat{\mu}(\mathbf{x}_j, \mathbf{y})}$$

A positive partial miscoding means that the feature contributes to describe the target variable, meanwhile a negative value means that the feature is not relevant.

**■ Example 8.4** We will use again the synthetic dataset of Example 8.3, but we will increase the number of relevant features from 4 to 14. Then, we will compute the list of partial miscodings.

```
from fastautoml.miscoding import Miscoding
from sklearn.datasets.samples_generator import make_classification

X, y = make_classification(n_samples=1000, n_features=20, n_informative=14,
                           n_redundant=0, n_classes=10, n_clusters_per_class=1, flip_y=0)

miscoding = Miscoding()
miscoding.fit(X, y)
msd = miscoding.miscoding_features(mode="partial")
```

As we can see in Figure 8.6, not only the library has been able to correctly identify the relevant features, but also, non relevant features have now a negative contribution to the global miscoding.

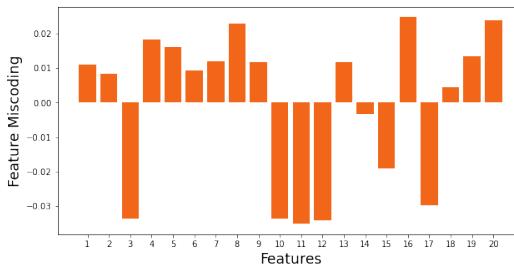


Figure 8.6: Partial Feature Miscoding.

Given the definition of partial feature miscoding we can provide a definition of the concept of miscoding of a target variable given a subset of predictors that it is closer to the original concept of miscoding defined by the theory of nescience.

**Definition 8.3.3** Let  $\mathbf{y}$  be a target variable,  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_p\}$  a dataset composed by  $p$  features, and  $\mathbf{Z} = \{\mathbf{z}_1, \dots, \mathbf{z}_k\}$  a subset of features, that is,

$\{\mathbf{z}_1, \dots, \mathbf{z}_k\} \subseteq \{\mathbf{x}_1, \dots, \mathbf{x}_p\}$ . We define the miscoding of  $\mathbf{Z}$  as a representation of  $\mathbf{y}$ , denoted by  $\hat{\mu}(\mathbf{Z}, \mathbf{y})$ , as:

$$\hat{\mu}(\mathbf{Z}, \mathbf{y}) = \sum_{i=1}^k \tilde{\mu}(\mathbf{z}_i, \mathbf{y})$$

■ **Example 8.5** Based on the dataset and the partial features miscoding computed in Example 8.4, in Figure 8.7 we can see the evolution of the miscoding of the training subset  $\mathbf{Z}$  as we add more features to the study.

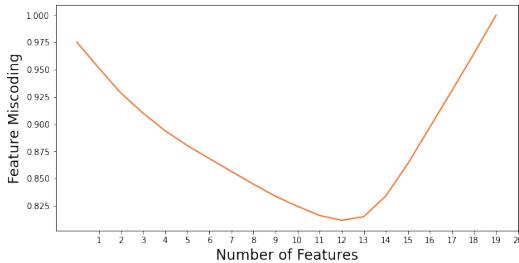


Figure 8.7: Accumulated Partial Feature Misclassification.

In the following example we are going to compare the performance of a machine learning classifier when using a full dataset and a reduced version of the same dataset using only those features identified as relevant, i.e., with positive partial miscoding.

■ **Example 8.6** Let's train a neural network with the standard MNIST dataset in order to classify hand written digits. The evaluation criteria will be the score of the classifier, that is, the percentage of digits correctly classified, applied over a test dataset different from the dataset used for training. The neural network will be trained and evaluated using all the features that compose the dataset, and with a reduced version of the dataset composed by only those features with a positive partial miscoding.

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import load_digits
from fastautoml.fastautoml import Miscoding

data = load_digits()
X_raw = data.data
y_raw = data.target

miscoding = Miscoding()
```

```

mCoding = miscoding(X_raw, y_raw)
mscd = mCoding.mcoding_features(miscoding='partial')
X_red = X_raw[:, np.where(mscd > 0)[0]]
y_red = y_raw

X_raw_train, X_raw_test, y_raw_train, y_raw_test = train_test_split(X_raw,
    y_raw, test_size=.3)
X_red_train, X_red_test, y_red_train, y_red_test = train_test_split(X_red,
    y_red, test_size=.3)

clf = MLPClassifier(alpha=1, max_iter=1000)

clf.fit(X_raw_train, y_raw_train)
score_raw = clf.score(X_raw_test, y_raw_test)

clf.fit(X_red_train, y_red_train)
score_red = clf.score(X_red_test, y_red_test)

reduction = 1 - X_red_train.shape[1] / X_raw_train.shape[1]

print("Score raw:", score_raw, " Score Miscoding:", score_red,
      " Reduction:", reduction)

Score raw: 0.9833333333333333  Score Miscoding: 0.9814814814814815  Data Reduction: 0.468

```

If we run the above source code, we will see that the score of the neural network classifier is about the same for the two datasets, 98% of the digits are correctly classified using the test data. However, the reduced dataset used for training based on the optimal miscoding is 43% smaller than the original dataset. This size reduction could have a big impact in the training time of the neural network. Smaller datasets are also relevant when working with ensembles of models, like random forests, where hundreds or thousands of models have to be trained.

Intuitively, as Example 8.5 shows, we should prefer the subset  $\mathbf{Z}$  of  $\mathbf{X}$  composed by all those features whose partial miscoding are greater than zero. However, as we will see in the following sections of this chapter, this might not be the case. Feature selection is only one of the criteria used in the process of finding an optimal model for an entity represented by a dataset. It might happen that other elements, like inaccuracy or surfeit, suggest to use a different subset of predictors. The global optimization criteria we should use is the concept of nescience. A sensible approach to use partial miscoding would be to incrementally add to our model those features with higher miscoding, until all features with a positive value have been added, or an optimality criterion has been reached.

In case of having a generative model (see Section 16.2.5), that is, a machine learning algorithm designed to find the joint probability  $P(\mathbf{X}, \mathbf{y})$ , we could use miscoding to compute how the different features relate to each

other, that is, the quantity  $\hat{\mu}(\mathbf{x}_i, \mathbf{x}_j)$  for each pair of values  $i, j \leq p$ . The result would be a miscoding matrix (see Example 8.7).

**Example 8.7** The Boston dataset included in `scikit-learn` library contains a collection of variables that (potentially) could explain the price of houses in the area of Boston. In this example, instead of computing which are the factors that contribute the most to the price of houses, we are going to study the inter-dependence between these factors, using a miscoding matrix.

```
from fastautoml.fastautoml import Miscoding
from sklearn.datasets import load_boston

data = load_boston()

miscoding = Miscoding(X_type="numeric", y_type="numeric")
miscoding.fit(data.data, data.target)
mscd_matrix = miscoding.features_matrix(mode='regular')
```

In Figure 8.8 we can see a graphical representation using a heatmap of the miscoding matrix computed over the features. The darker values represent a lower miscoding (mind we are using the regular version of the concept of miscoding). In particular, the values of the main diagonal are equal to zero.

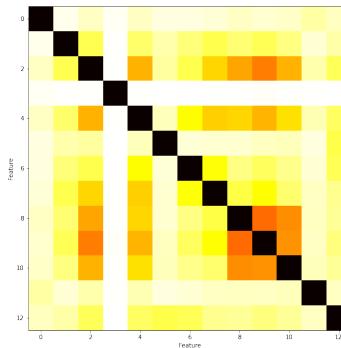


Figure 8.8: Regular Miscoding Matrix.

The minimum value of 0.52 is obtained with the pair (8, 9) that correspond to the features "index of accessibility to radial highways" and "full-value property-tax rate per \$10,000". These features are good candidates to evaluate in a predictive model, since they contain non-redundant information. The maximum value of 0.99 is achieved with the pair (3, 12) with the features "Charles River dummy variable" and "% lower status of the population". These features contains almost the same information, and so, including both in a model does not add nothing new, but increases the complexity of the model and the risk of over-fitting.

## 8.4 Inaccuracy

In Section 4.1 we defined the inaccuracy of a description  $d \in \mathcal{D}$  for a representation  $r \in \mathcal{R}$  as the normalized information distance between the representation  $r$  and the string  $\Gamma(d)$  printed out by a universal Turing machine when given the description as input:

$$\iota(d, r) = \frac{\max\{K(r | \Gamma(d)), K(\Gamma(d) | r)\}}{\max\{K(r), K(\Gamma(d))\}}$$

Inaccuracy, being based in Kolmogorov complexity, is not computable for the general case, and so, it has to be approximated in practice. In this section we are going to see how this concept can be estimated in case of a model trained using a dataset. The approach will be similar to the one used in case of miscoding (see Section ?? for more information).

**TODO:** This definition correspond to the discriminative case. Introduce the generative case as well.

**Definition 8.4.1** Let  $\mathbb{X}$  be a dataset,  $\mathbf{y}$  a response variable,  $m$  a model, and  $\hat{\mathbf{y}} = m(\mathbb{X})$  the predicted values by  $m$  given  $\mathbb{X}$ . We define the *inaccuracy* of the model  $m$  for the target values  $\mathbf{y}$ , denoted by  $\hat{\iota}(\hat{\mathbf{y}}, \mathbf{y})$ , as:

$$\hat{\iota}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\hat{K}_C(\hat{\mathbf{y}}, \mathbf{y}) - \min\{\hat{K}_C(\hat{\mathbf{y}}), \hat{K}_C(\mathbf{y})\}}{\max\{\hat{K}_C(\hat{\mathbf{y}}), \hat{K}_C(\mathbf{y})\}}$$

Intuitively, the quantity  $\hat{\iota}(\hat{\mathbf{y}}, \mathbf{y})$  is a measure of how far are the predicted values from real values. The lower this quantity, the better is the quality of  $m$  as a predictor for  $\mathbf{y}$ . With our new inaccuracy metric we are measuring not only how difficult is to reconstruct the original target vector  $\mathbf{y}$  given the predicted values  $\hat{\mathbf{y}}$ , but also how much additional information  $\hat{\mathbf{y}}$  contains that is not related to  $\mathbf{y}$ , being the latter a novelty with respect to other metrics used in machine learning to measure the accuracy of a model.

**■ Example 8.8** Inaccuracy, according to the minimum nescience principle, is given by the normalized compression distance between the actual targets  $\mathbf{y}$  and the predicted targets  $\hat{\mathbf{y}}$  by the model. In the following example we are going to compare the behavior of our new inaccuracy metric with a classical score metric. The experiment will be based on the MNIST dataset (hand written digits recognition) provided by scikit-learn.

```
from fastautoml.fastautoml import Inaccuracy
from sklearn.datasets import load_digits

X, y = load_digits(return_X_y=True)

inacc = Inaccuracy()
```

```
inacc.fit(X, y)
```

For this example we will train a decision tree classifier up to a predetermined tree depth of  $i$ , where  $i$  goes from 1 to 20.

```
from sklearn.tree import DecisionTreeClassifier

scores      = list()
inaccuracies = list()

for i in range(20):

    tree = DecisionTreeClassifier(max_depth=i, random_state=42)
    tree.fit(X, y)

    scores.append(1 - tree.score(X, y))
    inaccuracies.append(inacc.inaccuracy_model(tree))
```

We are interested to compare the behavior of score (actually we are comparing against one minus score) and inaccuracy metrics. As we can see in Figure 8.9, both metrics present a similar behavior, having inaccuracy a larger value, due to a stronger emphasis in incorrectly predicted values.

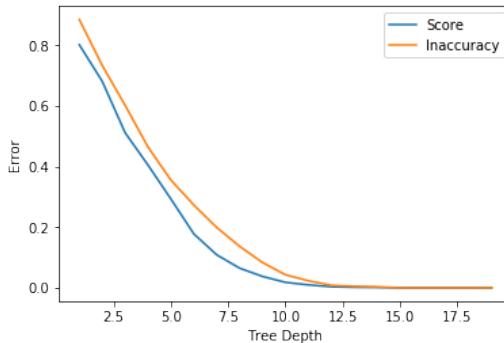


Figure 8.9: Inaccuracy vs. Score of Decision Trees

In Example 8.8 we have seen that the deeper the tree, the smaller is the training error. Of course, the higher the value of  $i$ , the higher the risk of overfitting the data. However, in case of inaccuracy we are not interested in avoiding overfitting, since overfitting is controlled by the metric of surfeit (see Section ??).

We can see inaccuracy as the effort, measured as the length of a computer program, required to fix the predictions made by a model. In this sense, according to the minimum nescience principle, it is not the same a model that makes one hundred times the same error than a model that makes one

hundred different errors, since it should be easier to fix the former than the later (see Example 8.9).

**■ Example 8.9** In this example we are going to use again a decision tree classifier, but this time it will be trained with the hyperparameter minimum number of samples per leaf node set to 5 (a common approach used in practice to avoid decision trees to overfit).

```
tree = DecisionTreeClassifier(min_samples_leaf=5)
tree.fit(X, y)
```

The inaccuracy of this new trained model is 0.17, and its score 0.08. Next we will artificially introduce one hundred errors in the dataset, simulating the case that the tree is not able to model correctly these data points. In this particular case all the errors are exactly the same.

```
X2 = X.copy()
y2 = y.copy()
for i in range(100):
    X2 = np.append(X2, [X[0]], axis=0)
    y2 = np.append(y2, (y[0]+1) % 10)
```

The inaccuracy of the decision tree, given this new dataset, has increased<sup>3</sup> from 0.17 to 0.21.

```
inacc.fit(X2, y2)
inacc.inaccuracy_predictions(pred)
```

Score has also increased, in this case from 0.08 to 0.13.

```
1 - tree.score(X2, y2)
```

Finally, we are going to repeat exactly the same experiment, but this time instead of adding one hundred times the same error, adding one hundred different errors.

```
X3 = X.copy()
y3 = y.copy()
for i in range(100):
    index = np.random.randint(X.shape[0])
    X3 = np.append(X3, [X[index]], axis=0)
    y3 = np.append(y3, (y[index]+1) % 10)
```

In this last case the inaccuracy of the model has increased up to 0.25, meanwhile score remained the same. ■

---

<sup>3</sup>Note that we had to `fit()` again the class `Inaccuracy` in order to use the new dataset. Normally this is not the way we use this class; instead what we should do is to fit once a dataset, and then compute the inaccuracy of different models. We are doing here in this way to demonstrate an interesting property of the concept of inaccuracy.

In line with Example 8.9, an extreme case would be a model for a target binary variable (True and False) that always fails with its predictions, that is, if the value of the target is True, the model will predict False, and if it is False, it will predict True. The classical evaluation metrics would say that this model is the worst possible model, but our inaccuracy would claim that the model is perfect. We might be wondering what it is the value of a model that always fails to predict the correct target. But if we are the managers of an edge fund investing in the stock market, we will very happy to pay a huge amount of money for a model that predicts that the shares of IBM will go down whenever they go up, and the other way around.

In case of having a highly unbalanced dataset, that is, when some categories have a lot of more training data than others, the classical score metric can provide a misleading result, since a good score does not necessarily mean a good model, it might happen that the model is simply properly classifying the samples of the category with the higher number of training samples, and misclassifying the others. In practice, we solve this problem by using metrics specifically designed to deal with unbalanced datasets. In case of the new metric of inaccuracy, as Example 8.10 shows, a model that can not properly classify one of the categories is considered a bad model, even if this category has only a few points in the training dataset.

■ **Example 8.10** For this example, we will create a synthetic dataset using the `make_classification` utility of scikit-learn, with two classes in which one of them has 95% of the samples, and the other 5%.

```
from sklearn.datasets import make_classification

depth = list()
score = list()
inacc = list()

inaccuracy = Inaccuracy()

for i in np.arange(1, 100):

    X, y = make_classification(n_samples=1000, n_features=2,
                               n_informative=2, n_redundant=0,
                               class_sep=2, flip_y=0, weights=[0.95,0.05])

    inaccuracy.fit(X, y)

    tree = DecisionTreeClassifier(min_samples_leaf=i)
    tree.fit(X, y)

    depth.append(i)
    score.append(1 - tree.score(X, y))
    inacc.append(inaccuracy.inaccuracy_model(tree))
```

The experiment consists in training a decision tree classifier with a

minimum number of samples per leaf of  $i$ , where  $i$  goes from 1 to 100. In Figure 8.10 we can see the behavior of inaccuracy and score. In case of large values of  $i$ , the score metric tell us that no more than a 5% of the samples is misclassified, however, the inaccuracy says that even if the total number of misclassified points is low, the inaccuracy of the model is very bad.

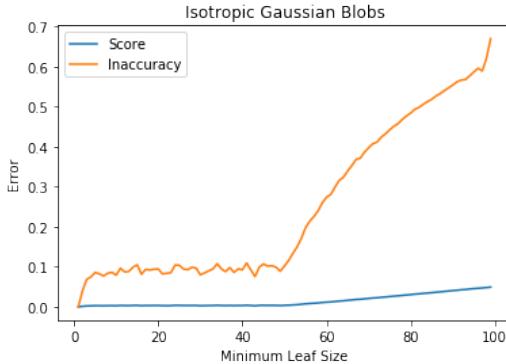


Figure 8.10: Inaccuracy of Decision Tree.

■

## 8.5 Surfeit

In Section 5.2 we defined the surfeit of the model  $m \in \mathcal{M}$  for the representation  $r \in \mathcal{R}$  as:

$$\sigma(m, r) = 1 - \frac{K(r)}{l(m)}$$

Since the length  $K(r)$  of shortest possible model for the representation  $r$  is in general unknown, we have to approximate this concept in practice. In case of having a training dataset  $\mathbb{X}$  and a target variable  $\mathbf{y}$ , we can approximate the surfeit of a model  $m$  for the representation  $\mathbf{y}$  by means of computing:

$$\hat{\sigma}(m, y) = 1 - \frac{\hat{K}_C(\mathbf{y})}{l(m)}$$

Where  $\hat{K}_C(\mathbf{y})$  is the length of the compressed version of the vector  $\mathbf{y}$  using as compressor a minimal length code  $C$ , computed given the relative frequencies of the values observed in  $\mathbf{y}$  (see Section ??).

**Definition 8.5.1** Let  $\mathbf{y}$  be a response variable,  $\mathbb{X}$  a dataset composed by  $p$  features and  $n$  samples. We define the surfeit of the model  $m \in \mathcal{M}$  as a representation of  $\mathbf{y}$ , denoted by  $\hat{\sigma}(m, \mathbf{y})$ , as:

$$\hat{\sigma}(m, \mathbf{y}) = 1 - \frac{\hat{K}_C(\mathbf{y})}{l(m)}$$

The definition of surfeit requires a method of encoding the models as a string of symbols, so we can compute their length. Ideally, we should use as encodings Turing machines, and agree upon an universal Turing machine to interpret those models. However, that would make very difficult to add new models to the nescience library. Instead, we have used for the encoding of models a simplified version of the Python language, where not all the constructions are allowed, and we do not allow the use of libraries.

Surfeit is a metric that can help us to avoid overfitted models. The higher is the surfeit of a model, the higher is the probability that the model is an overfit of the training dataset, as Example 8.11 shows.

■ **Example 8.11** In this example we are going to generate a dataset composed by 900 samples of a sinusoidal curve, and we will fit the data using a  $n$  degree polynomial, where  $n$  goes from 1 to 15.

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

from Nescience.Nescience import Surfeit
from Nescience.Nescience import Inaccuracy

n_samples = 900
degrees = np.arange(1, 15)

X = np.sort(np.random.rand(n_samples) * 3)
y = np.cos(1.5 * np.pi * X)

linacc = list()
lsurfeit = list()

for i in degrees:

    poly = PolynomialFeatures(degree=i, include_bias=False)
    newX = poly.fit_transform(X[:, np.newaxis])

    linear_regression = LinearRegression()
    linear_regression.fit(newX, y)

    inacc.fit(newX, y)
    inaccuracy = inacc.inaccuracy_model(linear_regression)

    sft.fit(newX, y)
    surfeit = sft.surfeit_model(linear_regression)
```

```
linacc.append(inaccuracy)
lsurfeit.append(surfeit)
```

In figure 8.11 we can see the results of this experiment. As it was expected, the higher the degree of the polynomial, the smaller is the error of the model. However, at the same time we see that the higher the polynomial, the higher the surfeit of the model. The ideal model is that one that has a low inaccuracy and a low surfeit.

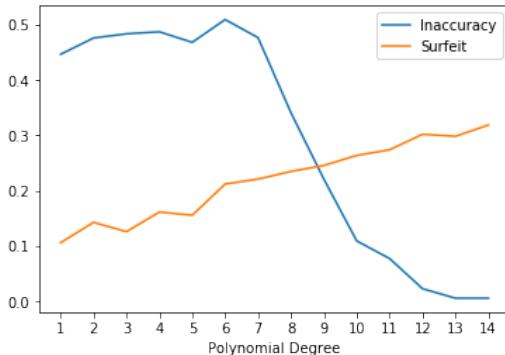


Figure 8.11: Surfeit vs Inaccuracy

Another advantage of the concept of surfeit is that it allows us to compare and decide between models that belong to different families. For example, in case of models having the same accuracy, shall we prefer a decision tree over a neural network, or a naive Bayes classifier over a support vector machine? Next example shows how we can decide about those questions.

■ **Example 8.12** In this example we are going to compare a decision tree with a neural network. We will use a synthetic dataset composed by two isotropic Gaussian blobs, and we will train our models to split them apart. In the first part of the example we will use a standard deviation of 1 and only two dimensions, so the two clusters are easy to classify (see figure on Table 8.1, left side).

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from Nescience.Nescience import Surfeit
from Nescience.Nescience import Inaccuracy
from sklearn.datasets.samples_generator import make_blobs

X, y = make_blobs(n_samples=1000, centers=2, n_features=2, cluster_std=1)
```

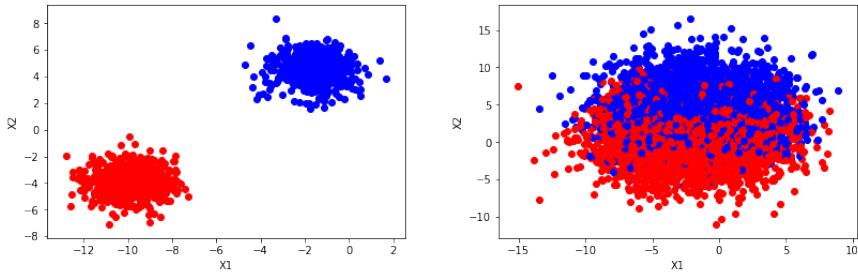


Table 8.1: Isotropic Gaussian blobs.

```

tree = DecisionTreeClassifier()
tree.fit(X, y)
tree.score(X, y)

nn = MLPClassifier()
nn.fit(X, y)
nn.score(X, y)

sft = Surfeit()
sft.fit(X, y)

sft.surfeit_model(tree)
sft.surfeit_model(nn)

```

If we ran the above code we will see that both models have exactly the same accuracy of 1, that is, they are perfect classifiers. However the surfeit of the decision tree is 0.25, meanwhile the surfeit of the neural network is 0.73. In this particular case we should prefer the decision tree over the neural network.

If we perform the same experiment using a standard deviation of 3, so two clusters that are more difficult to split (see Table 8.1, right side), the situation will change.

```
X, y = make_blobs(n_samples=10000, centers=2, n_features=8, cluster_std=3)
```

In this second case, again, both models have the same accuracy (we have increased the number of samples, and the number of dimensions, so the models can still perform a perfect classification), but the surfeit of the decision tree has increased to 0.82, and the surfeit of the neural network is almost the same, 0.76. For this second dataset we should prefer the neural network over the decision tree.

In example 8.12 we have assumed that both models, decision tree and

neural networks, have the same accuracy. When this is not the case, when the models do not have the same accuracy, we have to apply to the concept of nescience in order to decide between them.



**TODO:** Mention the problem of the stability of the signal.

## 8.6 Nescience

In Chapter 6 we defined the concept of nescience as the solution to a non-linear multi-objective optimization problem, where we had to minimize the miscoding, inaccuracy and surfeit of representations and models. The solution to this problem is, in general, not unique, in the sense that we can find multiple pairs of representations and models that have the property that we can not improve one of these quantities without degrading the others (Pareto optimality). However, in practice, we expect that a machine learning library should provide a single solution when training a model over a dataset. In order to provide this unique solution, we have to resort to a utility function that selects one from the available solutions. The nescience library provide different alternatives of utility functions, being the default one the arithmetic mean the tree metrics.



The nescience library implements the following utility functions to approximate the concept of nescience, that is, to compute  $\hat{V}(\mathbb{Z}, m, \mathbf{y})$ :

- Euclid distance:  $(\hat{\mu}(\mathbb{Z}, \mathbf{y})^2 + \hat{i}(\hat{\mathbf{y}}, \mathbf{y})^2 + \hat{\sigma}(m, \mathbf{y})^2)^{1/2}$
- Arithmetic mean:  $\frac{\hat{\mu}(\mathbb{Z}, \mathbf{y}) + \hat{i}(\hat{\mathbf{y}}, \mathbf{y}) + \hat{\sigma}(m, \mathbf{y})}{3}$
- Geometric mean:  $(\hat{\mu}(\mathbb{Z}, \mathbf{y}) \times \hat{i}(\hat{\mathbf{y}}, \mathbf{y}) \times \hat{\sigma}(m, \mathbf{y}))^{1/3}$
- Product:  $\hat{\mu}(\mathbb{Z}, \mathbf{y}) \times \hat{i}(\hat{\mathbf{y}}, \mathbf{y}) \times \hat{\sigma}(m, \mathbf{y})$
- Addition:  $\hat{\mu}(\mathbb{Z}, \mathbf{y}) + \hat{i}(\hat{\mathbf{y}}, \mathbf{y}) + \hat{\sigma}(m, \mathbf{y})$
- Weighted mean:  $w_\mu \hat{\mu}(\mathbb{Z}, \mathbf{y}) + w_i \hat{i}(\hat{\mathbf{y}}, \mathbf{y}) + w_\sigma \hat{\sigma}(m, \mathbf{y})$
- Harmonic mean:  $\frac{3}{\hat{\mu}(\mathbb{Z}, \mathbf{y})^{-1} + \hat{i}(\hat{\mathbf{y}}, \mathbf{y})^{-1} + \hat{\sigma}(m, \mathbf{y})^{-1}}$

Euclid distance and addition have the drawback that they produce nescience values greater than one, something that it is against our theory. Geometric mean, product and harmonic mean have the problem that the nescience is zero, or not defined, if one of the three metrics (miscoding, inaccuracy or surfeit) is zero. And the weighted mean introduce three new hyperparameters that have to be optimized. It is still an open question which one is the best utility function to compute the nescience of a dataset and a model.

Example 8.13 shows how we can use the nescience library to compute the nescience of a dataset and a model.

■ **Example 8.13** This example shows how to compute in practice the nescience of a dataset and a model. In particular, we are going to compute the nescience of a decision tree classifier applied over the dataset digits (MNIST hand written digits classification problem) included in the `sklearn` library.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.datasets import load_digits
from Nescience.Nescience import Nescience

data = load_digits()

tree = DecisionTreeClassifier()
tree.fit(data.data, data.target)
tree.score(data.data, data.target)
[ ] 1

nescience = Nescience()
nescience.fit(data.data, data.target)

nescience.nescience(tree)
[ ] 0.5895603819965907
```

The score of the decision tree model is 1, meaning that all the samples have been properly classified. Of course, what happened is that the decision tree is overfitting the dataset. In order to avoid this kind of problems we usually split the data in separate training and testing subsets, or we perform a more advanced cross-validation. However, if we compute the nescience, we will get a value of 0.59, rising the flag that something is wrong with the model or the training dataset. ■

In Example 8.13 we have shown that one of the advantages of the concept of nescience is that we can evaluate the quality of a model without applying computationally expensive procedures like cross-validation, and without requiring to save part of the data as a test subset. Another advantage of the metric nescience is that it allows us to decide between competing models from different families of models, as it is shown in Example 8.14.

■ **Example 8.14** In this example we are going to compare two models from two different families of models: decision trees and neural networks. Both models will be trained with the breast cancer dataset provided by the `sklearn` library.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import load_breast_cancer
from Nescience.Nescience import Nescience

data = load_breast_cancer()
```

```
X = data.data
y = data.target

tree = DecisionTreeClassifier(max_depth=3)
tree.fit(X, y)
tree.score(X, y)
[ ] 0.9789103690685413

nescience = Nescience()
nescience.fit(X, y)
nescience.nescience(tree)
[ ] 0.5945936419010083

nn = MLPClassifier()
nn.fit(X, y)
nn.score(X, y)
[ ] 0.9261862917398945

nescience.nescience(nn)
[ ] 0.7860523786210711
```

Both models have a similar score. In this case, not only the decision tree provide a better score, but also, the nescience is much lower than in case of the multi-layer perceptron, and so, we should prefer the former over the latter.



Nescience is a metric that can be used to optimize the hyperparameters that define a (parametric) family of models. The advantage of nescience is that we can use a greeedy approach to select the best value for an hypesparameter, saving a lot of computational time and resources during the search. That is, if we have a model controlled by an hyperparameter such that the higher the value the better the score, we should select that value in which the nescience stops decreasing and starts to increase, since this is the point in which we are not longer learning anything new from that dataset (see Example 8.15).

■ **Example 8.15** For this example we will use again a decision tree classifier with the breast cancer dataset. We will train 10 different trees, setting the hyperparameter `max_depth` with values from 1 to 10. The `max_depth` hyperparameter controls how deep we allow the tree to grow in order to classify the samples of the dataset. The deeper the tree the higher the score of the model, but also, the higher the risk of overfitting the training data. For each tree we will compute the nescience of the model, and we will compare it with a cross validation score. The results are shown in Figure 8.12. As we can see in the figure, both, nescience and cross validation score, decrease as we increase the depth of the tree, until we reach a point in which it starts to increase. This inflection point is where the model begins to overfit the data. The nescience library suggests to use a tree with a maximum depth of

7, meanwhile with the cross validation we got an optimal level of 6.

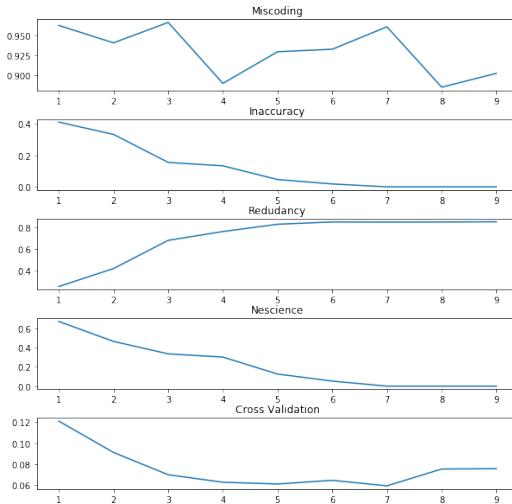


Figure 8.12: Evolution of Nescience with Tree Depth. ■

It is interesting to note the behavior of the three metrics that define the concept of nescience in Figure 8.12. As it is expected the the deeper the tree the smaller is the inaccuracy of the model and the higher the surfeit. However, in case of miscoding, we have a sort of random evolution. This behavior is due to the fact that each candidate tree uses a different subset of features at the decision nodes. I would be very nice to have an decision tree building algorithm that takes into account miscoding in order to decide the best features for new branches. Such an algorithm is described in Section 8.11.

Finally, we are going to see how to use nescience in case of hyperparameter searches where we cannot apply a greedy approach, for example when the search is performed over a collection of (usually conflicting) hyperparameters. Hyperparameter search is a computationally expensive approach, since the number of possible combinations to test could be very large. Moreover, if for each candidate set we have to cross-validate the result, the search becomes prohibitive. As we have seen, nescience do not requires the use of crossvalidation to detect a situation of overfitting, and so, it can significantly speed up the process of searching for optimal hyperparameters. In Example 8.16 it is show how we can do that with the nescience library.

■ **Example 8.16** In this example we are going to see how we can use the nescience library to find the optimal hyperparameters for a model using a

grid search. In particular, we are going to select the best hyperparameters for a multilayer perceptron classifier, including the number of hidden layers, and the size of those layers (what it is called Neural Architecture Search). The procedure will be demonstrated using the digits dataset.

```
from Nescience.Nescience import Nescience
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
```

First of all we have to provide a custom loss function based on the concept of nescience to be integrated with the search procedure. The next code shows how to implement such a function.

```
def my_custom_loss_func(estimator, X, y):
    nsc = Nescience()
    nsc.fit(X, y)
    nescience = nsc.nescience(estimator)

    # scikit-learn expect that higher numbers are better
    score = -nescience

    return score
```

Second, we have to define the grid of hyperparameters over which we are going to do the search. The larger the grid, the better the result, but also, the more computer time is required to evaluate all possible combinations.

```
parameters = {'solver': ['lbfgs'],
              'max_iter': [1000, 1500, 2000],
              'alpha': 10.0 ** -np.arange(1, 10, 3),
              'hidden_layer_sizes': [(60,), (100,), (60, 60,), (100, 100,),
                                     (60, 60, 60,), (100, 100, 100,)]}
```

Next code show how to do a classical grid search using the score of the models. The search will be evaluated using a train/test split of the dataset.

```
clf_std = GridSearchCV(estimator=MLPClassifier(), param_grid=parameters,
                       cv=3, iid=True, n_jobs=-1)
clf_std.fit(X_train, y_train)
clf_std.best_params_

[] {'alpha': 0.1,
[]  'hidden_layer_sizes': (100,),
[]  'max_iter': 1000,
[]  'solver': 'lbfgs'}

y_true, y_pred = y_test, clf_std.predict(X_test)
print(classification_report(y_true, y_pred))

[] precision    recall   f1-score   support
```

|                |      |      |      |     |
|----------------|------|------|------|-----|
| [] avg / total | 0.98 | 0.97 | 0.97 | 540 |
|----------------|------|------|------|-----|

Next code show how to perform exactly the same search, but using the concept of nescience instead of the metric score.

```
clf_nsc = GridSearchCV(estimator=MLPClassifier(), param_grid=parameters,
                       cv=3, scoring=my_custom_loss_func, iid=True)
clf_nsc.fit(X_train, y_train)
clf_nsc.best_params_

{'alpha': 0.1,
 'hidden_layer_sizes': (60,),
 'max_iter': 1500,
 'solver': 'lbfgs'}

y_true, y_pred = y_test, clf_nsc.predict(X_test)
print(classification_report(y_true, y_pred))

avg / total      precision      recall      f1-score      support
               0.98        0.98        0.98        540
```

As we can see, the results provided by the nescience library are slightly better in terms of train/test evaluations. However, what it is important is the library has opted for a smaller model (one layer of 60 neurons instead of one layer of 100 neurons) that provides a better result by increasing the maximum number of iterations (from 1000 to 1500). Nescience always select the smallest model that provides the best possible accuracy that does not overfit the training data.

■

## 8.7 Auto Machine Classification

The nescience library also includes a module for auto-machine learning (both for classification and regression problems). The auto-machine learning module returns the model, from a collection of families of models, that provides the smalles nescience. For each family of models, the class perform a greedy search over the hyperparameters required for each family. In Appendix XX is described the detail for each family of models.

Next example shows how to apply the automachine learning tools.

■ **Example 8.17** In this example we are going to see how to apply the nescience library to find the best model that describes the digits dataset.

```
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split

from Nescience.Nescience import AutoClassifier

(X, y) = load_digits(return_X_y=True)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)

model = AutoClassifier()
model.fit(X_train, y_train)

model.score(X_test, y_test)
[] 0.9622222222222222
```

If we write `type(model.model)` we will see that the library has selected a linear support vector machine as the best model for this dataset.

**TODO:** Compare with other automl tools.

### 8.7.1 Surfeit of Algorithms

#### Decision Trees

For the representation of a tree as a string we use the following template:

```
def tree{[attrs]}:
    if [attr] <= [thresh]:
        return [label] || [subtree]
    else:
        return [label] || [subtree]
```

Where `[attrs]` is the list of attributes used, and only those used in the model,<sup>4</sup> `[attr]` is a single attribute represented by the letter X followed by a number (e.g. `X1`), `[thresh]` is the threshold used for the split, `[label]` is one of the valid labels from the set  $\mathcal{G}$ , and `|| [subtree]` means that the `return` statement can be replaced by another level of `[if - else]` conditions. We could have used a much shorter description of trees by replacing word tokens with symbols, e.g., by the ternary conditional operators `? and :` used in modern programming languages, or by dropping the `return` statement. This would produce shorter trees, but the complexity of the models would remain the same, up to an additive constant that does not depend on the model itself. Since the harmonic mean compares relative values instead of absolute ones, this additive constant can be safely ignored.

## 8.8 Auto Machine Regression

### 8.9 Time Series

**TODO:** Introduce this section

---

<sup>4</sup>If the dataset contains many attributes, listing all of them when dealing with very short models would make the length of the model's header greater than the length of the body.

## 8.9.1 Automiscoding, Crossmicing and Partial Automiscoding

In this section we are going to study the application of the concept of miscoding to a time series and a delayed version of itself, or to a delayed version of a second time series.

Automiscoding is the application of miscoding to a time series and a delayed version of itself, as a function of this delay. Automiscoding is intended to estimate up to what extend previous observations of the time series can explain (or can be used to forecast) future observations. In this sense, automiscoding has a similar objective than autocorrelation in classical time series analysis (see Section 16.2.9).

**Definition 8.9.1** Let  $\{\mathbf{x}_t\}$  be a time series composed by  $n$  samples. We define lag  $k$  *regular automiscoding* of  $\{\mathbf{x}_t\}$  as  $\hat{\mu}(\mathbf{x}_{x_{k+1}, x_{k+2}, \dots, x_n}, \mathbf{x}_{x_0, x_1, \dots, x_{(n-k)}})$ . We define in the same way the concepts of *adjusted automiscoding* and *partial automiscoding*.

On the contrary of what happened with the concept of autocorrelation, automiscoding is defined for all time series, including time series with a trend. Moreover, automiscoding is interpretable even in case of having a trending time series, for example, for the identification of seasonal components without requiring to apply a decomposition.

■ **Example 8.18** In this example we are going to study the presence of cycles in the number of passengers of a US airline. In Figure 8.13 is depicted a time series of monthly passengers from 1949 to 1960 (AirPassenger dataset, see References section bellow). As we can observe, there is a clear cycle that repeats every twelve months. We can apply the concept of auto-miscoding to validate analytically that this is the case.

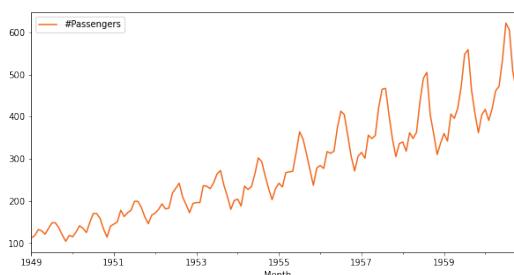


Figure 8.13: Air Passengers.

```
from nescience.timeseries import TimeSeries
data = pd.read_csv("data/AirPassengers.csv", index_col=["Month"], parse_dates=True)
```

```
X = np.array(data["#Passengers"]).reshape(-1, 1)

ts = TimeSeries(auto=False)
ts.fit(data)
mscd = ts.auto_miscoding(max_lag=36)
```

As we can see in Figure 8.14 there is a peak on the value of adjusted automiscoding every twelve months (the distance between both time series is minimal when we the lag is a multiple of a year).

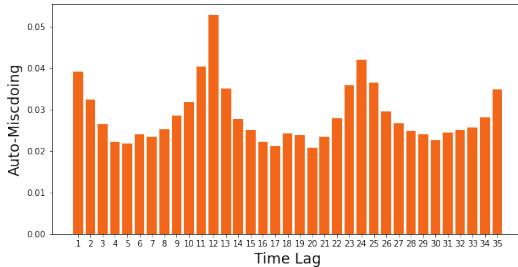


Figure 8.14: Auto-miscoding of Air Passengers.

Crossmiscoding computes the inter-relation between a time series and a lagged version of a second time series. The objective is to detect if the first time series has a temporal predictive power over the second.

**Definition 8.9.2** Let  $\{\mathbf{x}_t\}$  and  $\{\mathbf{y}_t\}$  be two time series composed by  $n$  samples each. We define lag  $k$  *regular crossmiscoding* of  $\{\mathbf{x}_t\}$  and  $\{\mathbf{y}_t\}$  as  $\hat{\mu}(\mathbf{x}_{k+1}, \mathbf{x}_{k+2}, \dots, \mathbf{x}_n, \mathbf{y}_{x_0}, \mathbf{x}_1, \dots, \mathbf{x}_{(n-k)})$ . We define in the same way the concepts of *adjusted crossmiscoding* and *partial crossmiscoding*.

**Example 8.19** We are interested to determine if it possible to predict the energy consumption of the appliances of a house. The data set to study (appliances energy prediction dataset, see References below) is composed by the temperature and humidity conditions measured in the different rooms of the house every ten minutes, and the energy consumption of the appliances. The dataset also includes some information about the current weather, from a nearby weather station.

For every feature we will compute the optimal lag at which the features has the best prediction capabilities:

```
from fastautoml.fastautoml import Miscoding

X = pd.read_csv("../data/energydata_complete.csv", parse_dates=["date"], index_col="date")
y = X["Appliances"]
```

```
X = X.drop(["Appliances", "lights"], axis=1)

miscoding = Miscoding()
mCoding.fit(X, y)

best_lag = list()
for i in np.arange(X.shape[1]):
    mscd = miscoding.cross_misCoding(attribute1=i, min_lag=1, max_lag=30)
    best_lag.append(np.where(mscd == np.max(mscd))[0][0] + 1)
```

In Figure 8.15 we can see a plot of the results. As we can observe, in general, for the in-house measurements we should use small lag values. However, in case of the weather conditions, bigger lags provide better results.

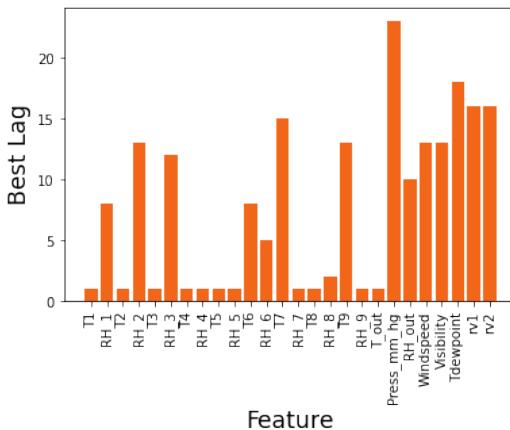


Figure 8.15: Cross MisCoding Lag



The approximation to the concept of miscoding introduced in this chapter estimates the quality of individual features as predictors of a target value, and the quality of the training dataset as a whole. However, the current approximation does not take into account the existing redundancy among the features themselves. For example, it might happen that two features  $x_i$  and  $x_j$  have very low miscoding with respect to the target variable  $y$ , but at the same time they are redundant, in the sense that they contain almost the same information. It is still an open question how to extend the concept of miscoding to take into account feature redundancy, in such a way that it is close to the theoretical definition, it can be computed efficiently, and it does not require a huge number of samples.

## 8.9.2 Auto Time Series

TODO: Introduce the auto-time series models, and clearly state what it can be expect from such models (short story: nothing)

TODO: Introduce structural time series models, the state space representation, and the Kalman filter

structural approach [...] different unobserved components or building blocks responsible for the dynamics of the series such as trend, seasonal, cycle, and the effects of explanatory and intervention variables are identified separately before being put together in a state space model.

State space methods originated in the

eld of control engineering, starting with the groundbreaking paper of Kalman (1960). They were initially (and still are) deployed for the purpose of accurately tracking the position and velocity of moving objects such as ships, airplanes, missiles, and rockets [...] these ideas could well be applied to time series analysis generally as well.

In a state space analysis the time series observations are assumed to depend linearly on a state vector that is unobserved and is generated by a stochastically time-varying process (a dynamic system). The observations are further assumed to be subject to measurement error that is independent of the state vector. The state vector can be estimated or identified once a sufficient set of observations becomes available.

**Definition 8.9.3** A linear Gaussian state space model for the multivariate time series  $\mathbf{y} = \mathbf{y}_1, \dots, \mathbf{y}_n$ , where each observation is a  $p$  dimensional vector  $\mathbf{y}_i = \{y_{i1}, \dots, y_{ip}\}$ , is given by

$$\mathbf{y}_t = \mathbf{Z}_t \boldsymbol{\alpha}_t + \mathbf{d}_t + \boldsymbol{\varepsilon}_t \quad \boldsymbol{\varepsilon}_t \sim N(0, \mathbf{H}_t) \quad (8.1)$$

called **space? or** *observation or measurement equation*, and

$$\boldsymbol{\alpha}_t = \mathbf{T}_t \boldsymbol{\alpha}_{t-1} + \mathbf{c}_t + \mathbf{R}_t \boldsymbol{\eta}_t \quad \boldsymbol{\eta}_t \sim N(0, \mathbf{Q}_t) \quad (8.2)$$

called *state or transition equation*, where the individual summands corre-

spond to:

- $\mathbf{y}_t$  observed or measured values,
- $\mathbf{Z}_t$  design matrix,
- $\alpha_t$  unobserved state,
- $\mathbf{d}_t$  observation intercept,
- $\varepsilon_t$  observational disturbance,
- $\mathbf{H}_t$  observational disturbance covariance matrix,
- $\mathbf{T}_t$  transition matrix,
- $\mathbf{c}_t$  state intercept,
- $\mathbf{R}_t$  selection matrix,
- $\eta_t$  state disturbance, and
- $\mathbf{Q}_t$  state disturbance covariance matrix

The  $p \times m$  matrix  $Z_t$  links the observation vector  $y_t$  with the unobservable state vector  $\alpha_t$  and may consist of regression variables. The  $m \times m$  transition matrix  $T_t$  determines the dynamic evolution of the state vector [...] the observation and state disturbances  $\varepsilon_t$  and  $\eta_t$  are assumed to be serially independent and independent of each other at all time points [...] matrix  $R_t$  is an  $m \times r$  selection matrix with  $r < m$ .

The initial state vector  $\alpha_1$  is assumed to be generated as  $\alpha_1 \sim NID(a_1, P_1)$ , independent of the observation and state disturbances  $\varepsilon_t$  and  $\eta_t$ . Mean  $a_1$  and variance  $P_1$  can be treated as given known.

**Talk about initialization?**

For example, if the time series  $y$  is unidimensional and the state space model is time invariant (only  $y_t$  and  $\alpha_t$  depends on  $t$ , being the rest of the summands constant), a model with  $m$  unobserved states will be given by

$$y_t = [z_1 \dots z_m] \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} + d_t + \varepsilon_t$$

Some of the most common time series models are particular cases of the state-space model (see Example XX).

### ■ Example 8.20

**TODO: Explain the Kalman filter**

The *Kalman filter* is a recursive formula that provides an optimal estimate for the unknown state in a state space model. At each time step  $t$ , the Kalman filter computes the predicted state conditional to the observations up to time  $t - 1$ .

Kalman filter can be used for filtering, prediction and smoothing. Here we are only interested in prediction [...] forward pass [...] recursive formulas

#### Definition 8.9.4

$$\begin{aligned}\mathbf{a}_{t+1} &= \mathbf{T}_t \mathbf{a}_t + \mathbf{K}_t \mathbf{v}_t \\ \mathbf{K}_t &= \mathbf{T}_t \mathbf{P}_t \mathbf{Z}_t^T \mathbf{F}_t^1 \\ \mathbf{v}_t &= \mathbf{y}_t - \mathbf{Z}_t \mathbf{a}_t\end{aligned}\tag{8.3}$$

called *prediction equations*, and

$$\begin{aligned}\mathbf{F}_t &= \mathbf{Z}_t \mathbf{P}_t \mathbf{Z}_t^T + \mathbf{H}_t \\ \mathbf{L}_t &= \mathbf{T}_t - \mathbf{K}_t \mathbf{Z}_t \\ \mathbf{P}_{t+1} &= \mathbf{T}_t \mathbf{P}_t \mathbf{L}_t^T + \mathbf{R}_t \mathbf{Q}_t \mathbf{R}_t^T\end{aligned}\tag{8.4}$$

called *updating equations*.

■ **Example 8.21** TODO: Provide an example where we can see how the Kalman filter integrates the predicted probability distribution and observed probability distribution to make a prediction ■

TODO: Explain the space search algorithm

■ **Example 8.22** TODO: Provide an example of auto-time series ■

## 8.10 Anomaly Detection

As we have seen in Section XXX, the main problem in the area of anomalies detection is that we do not have a precise mathematical definition of what an anomaly is. Given a dataset, in this book we propose to equate the concept of abnormal samples with that of incompressible samples, and study its consequences. The essence of this chapter is that learning is about finding regularities in a dataset, and finding regularities is what data compression is about. We have also seen that the best model is the one that minimizes the sum of the length of the model plus the length of the data given the model. This optimal model divides our dataset into two disjoint subsets, the compressible part, and the incompressible part. It is the former in which we are interested in this section, since being incompressible means that they cannot be explained by the model, that is, they are model-based anomalies given the best possible model.

Fix the following definition.

■ **Definition 8.10.1** Let  $X$  be a dataset composed by  $p$  features and  $n$  sam-

ples,  $y$  the target variable, and  $M$  a model such that the nescience  $N(\mathbf{X}, M)$  is minimal. Let  $\hat{y} = M(\mathbf{X})$  be the predictions made by the model  $M$  over the vectors of  $\mathbf{X}$ . We define the *anomaly subset* of  $\mathbf{X}$ , denoted by  $\mathcal{A}_M^{\mathbf{X}}$ , to the set of  $X$  such that  $y \neq \hat{y}$ .

The `nescience.anomalies` class allow us to identify the anomaly subset, that is, the collection of samples that do not match the regularity patterns found in the rest of the dataset. In Example 8.23 we will see how to apply this class to indentify houses with abnormal low prices, and to explain why they are cheaper.

■ **Example 8.23** In this example we will use the Boston House Price dataset provided by the `scikit-learn` toolkit. The dataset contains 13 predictive features (both, numeric and categorical) measuring different characteristics of the houses, such as number of rooms, age, etc., and the target is the median value of owner-occupied homes. The dataset is composed by five hundred samples. We are interested in to identify those house that have an abnormally low price, that is, houses that given their characteristics (number of rooms, size, etc) should have a higher price.

```
from sklearn.datasets import load_boston  
  
data = load_boston()  
X = data.data  
y = data.target
```

We have to train a "knowledge model", that is, find the best model that explains the target variable given the predictors, without overfitting. By default, the `anomalies` class uses the AutoML capabilities provided by the `nescience` library.

```
from nescience.anomalies import Anomalies  
  
model = Anomalies(X_type="mixed", y_type="numeric")  
model.fit(X, y)
```

Finally, we have to select those samples for which the actual price is smaller than the one predicted by the model.

```
anomalies = model.get_anomalies("smaller")  
X.shape, anomalies.shape  
((506, 13), (25,))
```

As we can see, there are 25 houses with abnormally low price. ■

The `anomalies` class also allow us to classify the identified anomalies according to the characteristics they share.

Let's see which are the best attributes that describe those abnormal houses.

```
model.get_classes(n_dims=1, an_type="smaller", filter_balancedness=True,
                  filter_redundancy=False, filter_repeated_attrs=False)
```

| Attribute1 | Attribute2 | Inertia     | N  | Class 0 | N | Class 1 | Ratio |
|------------|------------|-------------|----|---------|---|---------|-------|
| 2          | None       | 154.953975  | 9  | 16      |   | 0.36    |       |
| 4          | None       | 0.090593    | 6  | 19      |   | 0.24    |       |
| 5          | None       | 5.002437    | 17 | 8       |   | 0.68    |       |
| 7          | None       | 20.352117   | 6  | 19      |   | 0.24    |       |
| 8          | None       | 77.611111   | 18 | 7       |   | 0.72    |       |
| 9          | None       | 41737.11111 | 7  | 18      |   | 0.28    |       |
| 10         | None       | 26.577436   | 13 | 12      |   | 0.52    |       |
| 12         | None       | 430.294828  | 18 | 7       |   | 0.72    |       |

According to the inertia, the best attribute that allow us to classify the anomalies is the number 4 (nitric oxides concentration). This attribute divides the abnormal houses into two clusters of size 6 and 19. Let's see how the house's price is affected by this dimension.

```
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

lr = LinearRegression()
lr.fit(X[:,4].reshape(-1, 1), y)

plt.scatter(X[:,4].reshape(-1, 1), y)
plt.plot(X[:,4], lr.intercept_ + lr.coef_ * X[:,4], color="red")
plt.xlabel(data.feature_names[4])
plt.ylabel("Price")
```

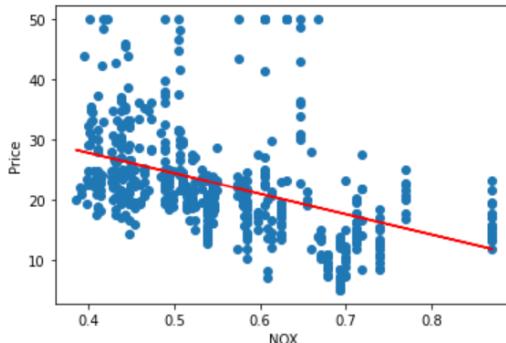


Figure 8.16: Price as a function of NOX.

The regression line suggest that the price of the houses is smaller in those areas with higher levels of nitric oxides concentration. Let's see how the anomalies are classified according to this dimension.

```
class0, class1 = model.get_class_points(attribute1=4, attribute2=None, an_type="smaller")
```

```
plt.hist(class0)
plt.hist(class1)
plt.ylabel("Count")
plt.xlabel(data.feature_names[4])
plt.show()
```

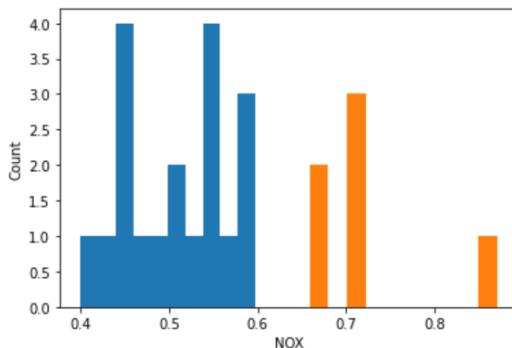


Figure 8.17: Histogram of anomalies NOX.

The analysis suggests that six of the houses have an abnormally low price because they are in an area with high levels of contamination. We can repeat the same analysis with the other attributes, although they have a higher inertia, so the class separation is not that evident. Please mind that there could be more than one reason why the price of a house is abnormally low.

## 8.11 Decision Trees

In the last sections we have seen how to use the concepts of miscoding, inaccuracy, surfeit and nescience to evaluate the quality of datasets and models, and to automatically select a family of models and search over its hyperparameters to find the best possible description of a topic. In particular, we have studied in detail the family of binary decision trees. The procedure used in the `fastautoml` library with trees was a mix between a classical approach (a CART algorithm combined with a cost-complexity pruning), and an evaluation of candidate trees using the minimum nescience principle. In this section we are going to see a new algorithm to derive optimal trees, both for classification and regression problems, that is entirely based on the theory of nescience. The new algorithm, by design, avoids the overfitting of the training dataset without losing accuracy, it does not require the optimization of hyperparameters, thus significantly reducing the training time, and it produces much smaller and more shallow trees than traditional algorithms, facilitating the interpretability of the results.

### 8.11.1 Algorithm Description

The following pseudocode shows the proposed algorithm to build a decision tree given a training dataset ( $\mathbf{X}, \mathbf{y}$ ). The procedure is based on a breadth first traversal of trees combined with a greedy approach. It requires a function called `BESTSPLIT()` that returns the best split of a given subset of the data into two subsets; and a second function, called `NESCIENCE()` that provides an estimation of the nescience of the current tree. The algorithm is based on two nested loops: the external **while** loop keeps a list of the candidate nodes to grow, whereas the internal **for** loop finds the best node to grow the tree. The latter operation requires to check all possible growing options and select the one that minimizes the nescience. The exit point of the algorithm is when there are no more branches to grow. We keep track of the best nescience achieved during the building process and return the associated tree.

```
def BUILD_TREE(data)

    nodesList <- list()
    tree <- BESTSPLIT(data)
    bestNescience <- NESCIENCE(tree)
    nodesList.append(tree)

    while not nodesList.empty()

        nescience <- bestNescience
        bestNode <- None
        childNode <- None

        for i <- 1, nodesList.length()

            node <- nodesList[i]

            node.child <- BESTSPLIT(node.ldata)
            tmp <- NESCIENCE(tree)
            if tmp < nescience
                nescience <- tmp
                bestNode <- i

            node.left <- None

            if nescience < bestNescience
                node <- nodeList[bestNode]
                bestNescience <- nescience
                nodesList.append(node.left)

            if not node.left.empty() and not node.right.empty()
                nodesList.remove(bestNode)

    return tree
```

The main difference of our algorithm from other decision tree building algorithms is in the way the tree is evaluated. Instead of using only accuracy

as most of the algorithms do, in addition, we take into account the complexity of the tree (surfeit) to avoid overfitting, and the quality of the subset of data used during the training process (miscoding).

### Nescience

The calculation of the nescience implemented in the algorithm is based on a Euclidean distance utility function (see Section 8.6), because that one was the one that produced the best results in the tests we have performed. For the computation of miscoding and inaccuracy, we use the same techniques that the one used in the `fastautoml` library, described in Section 8.3 and Section 8.4 respectively. For the implementation of surfeit, we use the same template to describe trees that was used in the `DecisionTreeClassifier` of the `AutoClassifier` class, and that was described in Section 8.7.1. The only difference is that we also allow equalities in the nodes (`if [attr] = [thresh]`), something not supported by the `DecisionTreeClassifier` algorithm of the `scikit-learn` library.

The generic problem of the instability of inaccuracy due to very short models, also applies to this algorithm (see Section 8.5), and the particular problem of the algorithms to build decision trees, in which the best local split might not be that one that minimizes the error (see Section ??) is also relevant in this case.

The concept of nescience is used in two different ways in our algorithm. For every iteration of the `for` loop we have to decide which one of the candidate branches of the tree we should develop. Recall that the order in which we develop the branches is important, since it might happens that one branch does not get developed because that would mean increase the sufeit without a sufficiently large decrease of the inaccuracy. The second place is at the end of the `while` loop, where we keep track of the nescience of the different building steps, to decide at the end of the algorithm which tree we return.

We treat regression problems as classification problems in which we discretizes the continuous target variable  $y$  into  $n$  intervals given the number of samples, and using a uniform discretization (see Section ). Once the target variable has been discretized, we train a regular classification tree.

### Splitting Criteria

Given a subset  $\mathbf{Q} \subseteq \mathbf{X}$  we have to find a split for  $\mathbf{Q}$  such that the values of  $y$  are grouped together. Recall that a split is a pair  $\theta = (j, w)$ , where  $1 \leq j \leq p$  is a feature index and  $w$  is the partition point (see Section 16.2.8). A split divides the set  $\mathbf{Q}$  into two disjoint subsets  $\mathbf{Q}_l$  and  $\mathbf{Q}_r = \mathbf{Q} \setminus \mathbf{Q}_l$ . In case of a continuous variable we have that  $\mathbf{Q}_l = \{\mathbf{x}_i \in \mathbf{Q} : x_{ij} \leq w\}$ , and if the feature

is categorical we define  $\mathbf{Q}_l = \{\mathbf{x}_i \in \mathbf{Q} : x_{ij} = w\}$ <sup>5</sup>.

In Section 16.2.8 we saw that a common splitting criteria used in practice is to minimize the weighted entropy  $\tilde{H}$  of the subsets  $\mathbf{Q}_l$  and  $\mathbf{Q}_r$ , that is, to find an split that it is minimal  $\theta^* = \arg \min_{\theta} \tilde{H}(\mathbf{Q}, \theta)$ . More explicitly, if  $\mathbf{y}$  is a target vector taking values from a set of  $k$  labels  $\mathcal{G} = \{g_1, \dots, g_k\}$  (either because is a categorical target or a continuous target that has been discretized into  $k$  intervals), and denoting the subsets of  $\mathbf{y}$  as  $\mathbf{y}^l = \{y_i : \mathbf{x}_i \in \mathbf{Q}_l\}$  and  $\mathbf{y}^r = \{y_i : \mathbf{x}_i \in \mathbf{Q}_r\}$ , and  $n_l$  and  $n_r$  are the number of elements of  $\mathbf{y}^l$  and  $\mathbf{y}^r$  respectively, we have that

$$\begin{aligned}\tilde{H}(\mathbf{Q}, \theta) &= \frac{n_l}{n} \left( - \sum_{i=1}^k \frac{\sum_{j=1}^{n_l} I(y_j^l = g_i)}{n_l} \log_2 \frac{\sum_{j=1}^{n_l} I(y_j^l = g_i)}{n_l} \right) \\ &\quad + \frac{n_r}{n} \left( - \sum_{i=1}^k \frac{\sum_{j=1}^{n_r} I(y_j^r = g_i)}{n_r} \log_2 \frac{\sum_{j=1}^{n_r} I(y_j^r = g_i)}{n_r} \right)\end{aligned}\quad (8.5)$$

In our nescience based decision tree algorithm, the splitting criteria is to minimize the total length of encoding the subsets  $\mathbf{Q}_l$  and  $\mathbf{Q}_r$  using optimal codes. We have to find the optimal split  $\theta^* = \arg \min_{\theta} \hat{K}_C(\mathbf{Q} | \theta) = \arg \min_{\theta} \{\hat{K}_{C_l}(\mathbf{Q}_l) + \hat{K}_{C_r}(\mathbf{Q}_r)\}$  where  $C_l$  and  $C_r$  are the optimal codes given the relative frequencies of the observed values of  $\mathbf{y}^l$  and  $\mathbf{y}^r$  respectively. The quantity  $\hat{K}_C(\mathbf{Q} | \theta)$  is computed as:

$$\hat{K}_C(\mathbf{Q} | \theta) = \hat{K}_{C_l}(\mathbf{Q}_l) + \hat{K}_{C_r}(\mathbf{Q}_r) = - \sum_{i=1}^k \log_2 \frac{\sum_{j=1}^{n_l} I(y_j^l = g_i)}{n_l} - \sum_{i=1}^k \log_2 \frac{\sum_{j=1}^{n_r} I(y_j^r = g_i)}{n_r}$$

In this particular case (if we use as compression algorithm a code with optimal lengths, and continuous variables have been discretized) it turns out that both expressions are equivalent given the following relation:

$$\tilde{H}(\mathbf{Q}, \theta) = \frac{1}{n} \hat{K}_C(\mathbf{Q} | \theta)$$

We prefer to talk of encoding length instead of weighted entropy because it has an easier interpretation in the context of the theory of nescience.



Strictly speaking, if we want to implement a decision trees search algorithm fully compliant with the minimum nescience principle, instead of using a total length encoding as splitting criteria, we

---

<sup>5</sup>Ideally, for the categorical case, instead of a single feature  $w$  we should search over all the elements of the power set of the set of features  $\mathcal{P}\{1, 2, \dots, p\}$ . Unfortunately, that would imply to check  $2^p$  cases, something that is time-expensive from the computational point of view.

should have computed the nescience at each split and select that one that makes it minimal. However, early experiments have shown that at local level it works better to group the values of  $y$  than to reduce the nescience. Further research is required to confirm and explain this point.

### Practical Implementation

In the web page that accompanies this book<sup>6</sup> we provide an open-source implementation of our algorithm in Python. Our software can be used together with other machine learning tools from the `scikit-learn` library, since we adhere to their API guidelines. For example, our algorithm can be used as part of an ensemble of classifiers, like the `BaggingClassifier` meta-estimator, or the results of the classification could be cross-validated with tools like `cross_val_score`. As an example, to provide a model for the breast cancer dataset, we could do something like the following:

```
from NescienceDecisionTree import NescienceDecisionTreeClassifier
from sklearn.datasets import load_breast_cancer

data = load_breast_cancer()

model = NescienceDecisionTreeClassifier()
model.fit(data.data, data.target)
print("Score: ", model.score(data.data, data.target))
```

### 8.11.2 Algorithm Evaluation

In this section we are going to evaluate our new algorithm, and compare its performance against the well-known algorithm CART. CART, *Classification and Regression Trees*, is the de-facto standard algorithm used in the machine learning industry for the derivation of decision trees. For this particular experiment we have used the CART implementation provided by `scikit-learn`.

Figure 8.18 shows a synthetic dataset consisting of 1000 random points lying on a two dimensional plane, where all the points with an  $X_1$  attribute less than 50 are colored blue, and the rest as red. We have artificially introduced a red point, simulating a measurement error, in the blue area. The black lines correspond to the decisions performed by CART. Since the CART algorithm will not stop until all the points have been properly classified, we have to specify an expected count condition to limit the number of splits. The figure corresponds to the tree generated by CART setting the `min_samples_leaf` hyperparameter to 5.

The tree obtained by applying our algorithm to the dataset of Figure 8.18 can be seen in Figure 8.19. The nescience based algorithm does not try

<sup>6</sup><http://www.mathematicsunknown.com>

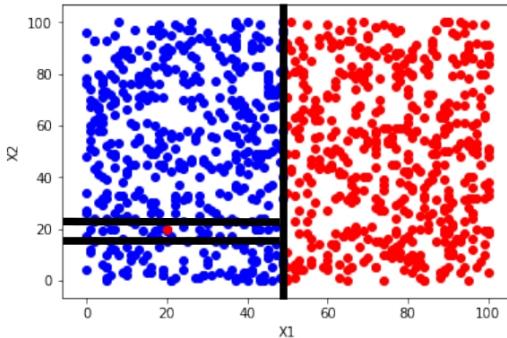


Figure 8.18: Synthetic dataset with CART algorithm splits.

to model the error point, since the gain due to an increment in the accuracy does not compensate the surfeit introduced in the model. Recall that the algorithm stops when the total nescience of the tree, based on the measures of miscoding, inaccuracy and surfeit, does not decrease when adding new nodes to the tree. Our algorithm presents a lower sensitivity to the errors found in datasets, at least if the number of errors is small compared with the number of valid points.

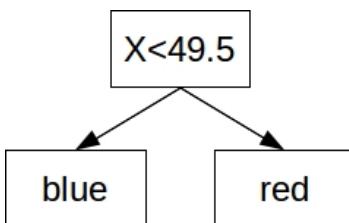


Figure 8.19: Decision tree obtained by the nescience algorithm.

Our second experiment, again with synthetic dataset, is depicted in Figure 8.20. There, we create two isotropic Gaussian blobs that partially overlap. We start with a standard deviation of 2.5 for each cluster, so they are easy to separate, and we increase the standard deviation in increments of 0.01, until we reach 4.5, which causes significant overlaps. For each value of the standard deviation, we run the experiment 100 times and we compute the average accuracy for the two algorithms using different datasets for training (70% of the data) and testing (30% of the data). The results of this experiment are shown in Figure 8.21.

As we can see, the performance of both algorithms, in terms of accuracy, is similar. However we should note that the hyperparameter `minimum_leaf_size`

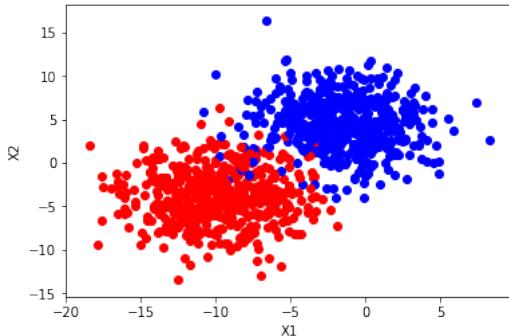


Figure 8.20: Isotropic Gaussian Blobs.

of the CART algorithm has been optimized to achieve the best accuracy. For this particular experiment, the best value was achieved with a minimum leaf size of 26 points. By definition, given the fact that CART has one degree of freedom more than the nescience algorithm, it should produce better accuracy; something that it is not observed (both algorithms have a mean accuracy of 0.87).

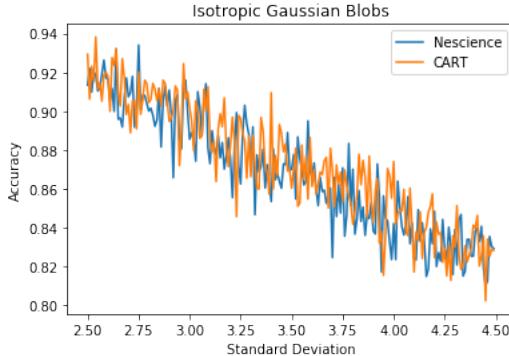


Figure 8.21: Accuracy of Isotropic Gaussian Blobs.

For each iteration of the experiment, we have also computed the average number of nodes, including internal and leaf nodes, required by the models to properly classify the clouds in the dataset. The results of this measurement are shown in Figure 8.22. Our algorithm requires an average of 4 nodes compared to 23 nodes for the CART algorithm. Moreover, our algorithm is more stable than CART, in the sense that it produces models of similar complexity when it gets similar input datasets (a standard deviation of 0.31 compared to 3.77 for CART).

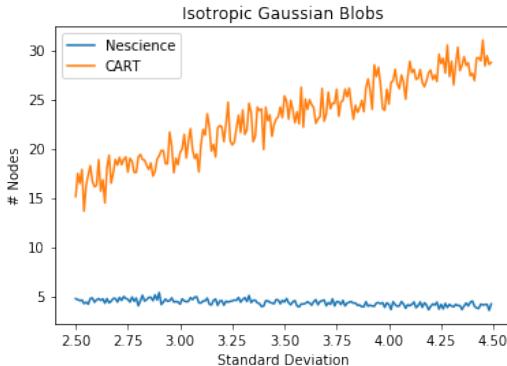


Figure 8.22: Number of Nodes.

In Figure 8.23 we show the maximum depth of the tree, defined as the longest path from the root of the tree to any of its leaves. The maximum depth of the tree is a good measure of the average time it will require for the model to provide a classification. The nescience algorithm has an average depth of 1.6 nodes, whereas the average depth yielded by the CART algorithm is 4.8 nodes.

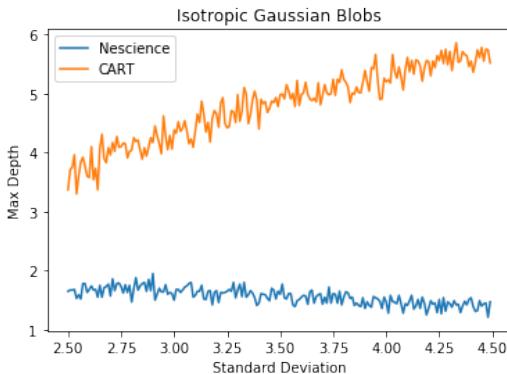


Figure 8.23: Maximum depth of the model.

The last part of the evaluation consists in comparing the performance of our algorithm and CART with a collection real datasets. More specifically, we have selected 12 well known datasets from the UCI Machine Learning Repository. The selected datasets are: diagnosis of breast cancer (`cancer`), optical recognition of handwritten digits (`digits`), predicting protein localization sites in gram-negative bacteria (`yeast`), classification of NASA space shuttle data (`shuttle`), classification of blocks in web pages (`page`),

segmentation of outdoor images (`image`), predicting the age of abalones from physical measurements (`abalone`), predicting the quality of red and white variants of Portuguese wine (`wine`), filter spam emails (`spam`), wall-following robot navigation (`wall`), classification of land use based on Landsat satellite images (`landsat`), and distinguishing signals from background noise in the MAGIC gamma telescope images (`magic`). For each dataset, we have repeated the experiment 100 times, by randomly selecting the training (70%) and testing (30%) subsets at each iteration.

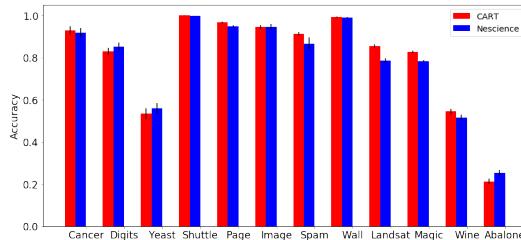


Figure 8.24: Maximum depth of the model.

In Figure 8.24 we compare the accuracy of the resulting models obtained by applying the CART algorithm and the nescience algorithm to the above datasets. In 4 of the 12 datasets, our algorithm provides better accuracy than CART. In the remaining 8 cases, the accuracy is, on average, less than 1% smaller.

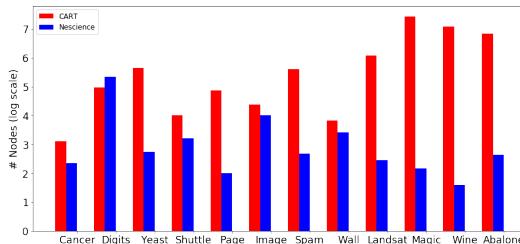


Figure 8.25: Maximum depth of the model.

In Figure 8.25 it is shown a comparison of the total number nodes (internal nodes plus leaf nodes) of the resulting models. Only for one of the datasets (`digits`), our model produces a slightly more complex tree than those generated by CART. In the rest of the cases, the number of nodes in the trees generated by the nescience algorithm have between two and three orders of magnitude fewer nodes (in this figure the y axis is in logarithmic scale).

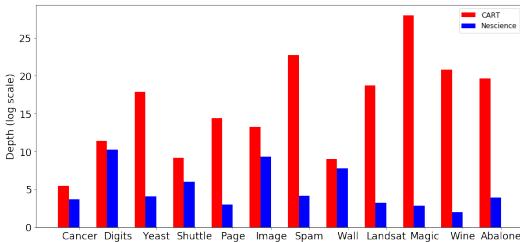


Figure 8.26: Maximum depth of the model.

Finally, in Figure 8.26 we provide a comparison of the depth of the tree of the resulting models. Our algorithm always yields a shallower tree than the CART algorithm.

We would like to mention that the nescience algorithm is highly robust with respect to the compressor selected or the nescience function implemented. In Table 8.11.2, we have applied the nescience algorithm to the datasets described above, and evaluate different alternatives for the definition of the nescience function  $N(X, M)$ : arithmetic mean  $(\mu(M, D) + \iota(X, M) + \sigma(M, D))/3$ , geometric mean  $(\mu(M, D) + \iota(X, M) + \sigma(M, D))^{1/3}$ , harmonic mean  $3/(\mu(M, D) + \iota(X, M) + \sigma(M, D)) - 1$ , Euclidean distance  $(\mu(M, D) + \iota(X, M) + \sigma(M, D))^{1/2}$ , sum  $\mu(M, D) + \iota(X, M) + \sigma(M, D)$ , and product  $\mu(M, D) + \iota(X, M) + \sigma(M, D)$ . The table shows limited difference between the different functions.

|          | Euclid | Arithmetic | Geometric | Product | Addition | Harmonic |
|----------|--------|------------|-----------|---------|----------|----------|
| Accuracy | 0.758  | 0.784      | 0.803     | 0.803   | 0.784    | 0.81     |
| Stdev    | 0.051  | 0.041      | 0.033     | 0.033   | 0.041    | 0.038    |

Table 8.2: Comparison of nescience functions

Similarly, Table 8.11.2 shows the performance of our algorithm when using the LZMA, zlib, and bz2 compressors. We observe that all of them yield similar performance. The above results suggest that the performance of our algorithm is independent of the specific choice made for either implementation aspect.

We emphasize that the CART algorithm requires to optimize a configuration hyperparameter in order to obtain good results, whereas the algorithm proposed in this book does not require from this optimization.

Shallower trees means faster forecasting times when the models used in production, since the number of if-else conditions to be evaluated is smaller. Moreover, smaller trees makes easier to interpret the results by

|          | <b>bz2</b> | <b>lzma</b> | <b>zlib</b> |
|----------|------------|-------------|-------------|
| Accuracy | 0.813      | 0.804       | 0.81        |
| Stdev    | 0.03       | 0.045       | 0.038       |

Table 8.3: Comparison of compressors

human analysts, and much shorter training times, something very relevant in case of training ensembles of trees, like random forest or boosted trees (although the use of ensembles of models is highly discouraged by the theory of nescience, given their high surfeit).

## 8.12 Algebraic Model Selection

As it was the case for the definition of nescience based on the encyclopedic description of research topics, the nescience of structured datasets can be used to evaluate alternative descriptions of research topics (mathematical models), and to identify how far these descriptions are from an ideal perfect knowledge. This evaluation could be used to identify those topics which require further research. Moreover, the same methodology could be applied to collections of datasets to identify our current knowledge of research areas (collections of topics).

If we combine the concept of nescience of a model, with our concepts of relevance and applicability of research topics, we could apply our methodology for the assisted discovery of interesting questions to collections of datasets; a very useful methodology now that big datasets are becoming widely available.

In order to evaluate the methodology developed, we are going to apply it to a particular research topic: *Multipath Wave Propagation and Fading*. The problem at hand is to understand the effect of a propagation environment on a radio signal, such as the one used by wireless devices. The signals reaching the receiving antenna could follow multiple paths, due to atmospheric reflection and refraction, and reflection from water and objects such as buildings. The effects of these multiple wave paths include constructive and destructive interference (fading), and phase shifting of the original signal, resulting a highly complex received signal (see Figure 8.27).

In many circumstances, it is too complicated to describe all reflection, diffraction and scattering processes that determine the different paths the signal will follow. Rather, it is preferable to describe the probability (stochastic model) that the received signal attains a certain value. We are interested in to analyze how well these stochastic models (our current knowledge) are able

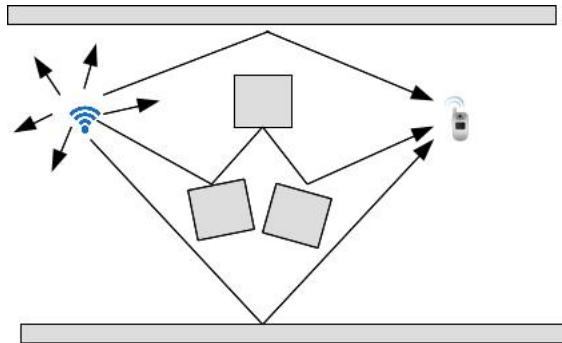


Figure 8.27: Multipath Signal Propagation

to describe what happens in reality.

The *Rayleigh fading model* assumes that the magnitude of a signal will vary randomly, or fade, according to a Rayleigh distribution (the radial component of the sum of two uncorrelated Gaussian random variables). The Rayleigh probability density function of the power signal is given by:

$$P_\sigma(x) = \frac{1}{\sigma} \exp\left[-\frac{x}{\sigma}\right]$$

where  $\sigma$  is the mean of the received signals. Rayleigh fading is viewed as a reasonable model for the effect of heavily built-up urban environments, when there is no dominant propagation along a line of sight between the transmitter and receiver.

The Rice or *Rician distribution* describes the power of the received signal when the target consists in many small scatterers of approximately equal strength, plus one dominant scatterer whose individual received signal equals all that of all the small scatterers combined (there is a dominant line of sight). The probability density function of the power of the received signal is given by:

$$P(x) = \frac{1}{\bar{\sigma}} (1 + a^2) \exp\left[-a^2 - \frac{x}{\bar{\sigma}} (1 + a^2)\right] I_0\left[2a\sqrt{(1 + a^2)} \frac{x}{\bar{\sigma}}\right]$$

where  $\bar{\sigma}$  is the mean of the received signals, and it is equal to  $\bar{\sigma} = (1 + a^2) \bar{\sigma}_R$ , being  $a^2 \bar{\sigma}_R$  the power of the dominant scatterer, and  $I_0$  is the modified zeroth order Bessel function of the first kind.

An experiment (see Figure 8.28) was set up to collect a real dataset to analyze. The experiment was run on a  $135m^2$  office full of obstacles

(interacting objects). The transmitter was an Odroid C1 Linux computer with a Ralink RT5370 USB Wifi adapter. The receiver was a (fixed in space) Motorola Moto G mobile phone. Data was collected using the Kismet<sup>7</sup> platform (an 802.11 layer2 wireless network detector, sniffer, and intrusion detection system), with some ad hoc, home made, software extensions, mostly for data aggregation. A total of 3,177 samples (power level measured in dBm) were collected during one hour experiment.

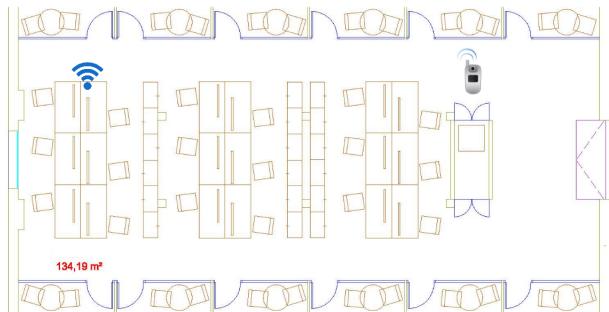


Figure 8.28: Experimental Set Up

Next table summarizes the results of applying the three considered models (uniform, Rayleigh and Rice) and the optimal encoding using a Huffman code:

| Model    | LDM    | Nescience |
|----------|--------|-----------|
| Uniform  | 17,351 | 1.30      |
| Rayleigh | 13,229 | 0.75      |
| Rice     | 11,118 | 0.47      |
| Huffman  | 7,541  | -         |

Table 8.4: Nescience of Models

The uniform model, that is, assuming zero knowledge about the topic covered by the dataset, has a nescience of 1.30. This value is a kind of upper level for the nescience associated with that particular topic and dataset; any model with a higher nescience should be classified as zero knowledge model. If we introduce the knowledge that in a environment with multiple obstacles the signal propagation can be described as a Gaussian process (Rayleigh distribution), we are able to decrease our nescience to 0.75, that is, there were a 43% improvement in our understanding of the topic. If we add the

<sup>7</sup><https://www.kismetwireless.net/index.shtml>

knowledge that there is usually a strongly dominant signal seen at the receiver caused by a line of sight between the antenna and the mobile phone (Rician distribution), the nescience decreases to 0.47, and so, we have achieved an additional 23% gain in our understanding. Given that numbers we can conclude that the Rayleigh model increases our knowledge with respect to the uniform model, and that the Rice model does so with respect to Rayleigh. However, the nescience of this last model is 0.47. That means that there still patterns in the dataset that are not explained by the Rice model, or what it is equivalent according to our methodology, there is still some knowledge to discover and learn.

The methodology has been applied to a dataset gathered in a single experiment under a controlled environment, since the goal of this Chapter was to provide a methodology to quantify the nescience of structured datasets, not to evaluate models for signal propagation and fading. In order to conclude that, in general, the Rice model is an improvement over Rayleigh, a more realistic experiment is required, with multiple datasets gathered in real environments.

### 8.13 The Analysis of the Incompressible

As we have said in Chapter chap:Introduction, one of the reasons to understand how things work is to understand the cause-effect relation in systems. We are interested in this cause/effect relation in two ways. That is, if we want to see an effect in a system, we want to understand which causes trigger that effect. Also, and perhaps more interesting, if we have observed an (probably undesired effect) in a system we would like to discover what has caused that effect, so we can fix it, and revert the normal situation.

We could use the theory of nescience to model, and modify, those uncommon effect, by means of training a model and looking at the incompressible part of the data.

A model  $\mathcal{M}$  for a dataset  $\mathcal{D} = (X, y)$  is a compressed version of that dataset, since the length of the dataset given the model  $l(\mathcal{D} | \mathcal{M})$  is smaller than the length of the original dataset  $l(\mathcal{D})$ . The model  $\mathcal{M}$  is composed by the regularities found in the dataset (subject to the algorithm used and the families of models considered). What is left,  $\mathcal{D} | \mathcal{M}$  is the incompressible part of the dataset, that is, those samples that have no regularity at all, or present a regularity that requires a description longer than the length of the raw data.

In this section we are going to show the practical applications of analyzing what is left, that is, the incompressible samples of a dataset. An element that is incompressible represents a very unlikely, or uncommon,

situation of the entity being studied. A incompressible element does not necessarily means a problem, since if a problem is sufficiently common, it can be compressed. An incompressible element is something than cannot be explained given the normal behaviour of the system. Of course, all of this is assuming that our dataset has no errors.

Once we have found a model that has the lowest possible nescience for a dataset, we could separate those elements that have not been compresed, denoted by  $XX$ , and fit a second model. We could argue that it does not make any sense to model the incompressible part, since, it is incompressible. However, the incompressible part is incompressible with respect to the original entity under study, that is, the global system. And in this new case, we are studying a different entity, namely, the uncommon parts of an entity. It might happen that we can find regularities in this new entity.

## References

A insightful description of the differences between explanation models and predictive models, how these models are used in different scientific disciplines, and what are the implications for the process of statistical modeling can be found in [Shm+10].

The application of the minimum description length principle to the identification of optimal decision trees have been proposed in [QR89], further refined and clarified in [WP93]; however the coding method proposed by those authors is different from the one used in this book.

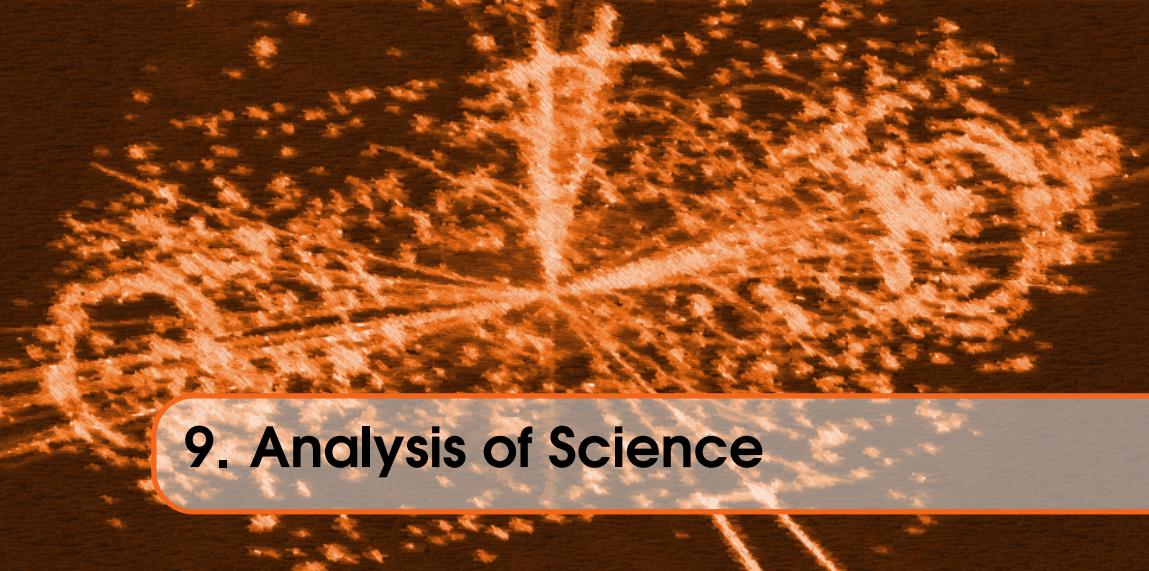
In our web page can be found an implementation of the decision tree following the guidelines defined in [].

The Minimum Description Length [Grü07] and the Minimum Message Length [Wal05] techniques have been applied to the problem of inferring decision trees in [QR89], later on clarified and extended in [WP93], in [MRA+95] as a technique for pruning, and in [RS98], among others. Although the underlining concepts behind the cost function proposed in this chapter are the same (namely, that learning is equivalent to the capability to compress), our approach is very different from the ones described in these works.

An excellent survey of the available discretization methods can be found in [Gar+13]; in the paper the authors also propose a taxonomy to classify existing methods based in their properties and they conduct an extensive comparative experimental study. The proportional discretization method used to compute miscoding is introduced in [YW09], where there is also a theoretical justification of why this method reduces the bias and the variance of the discretized variable.

TODO: Find the original reference of the AirPassengers dataset.

TODO: Find the original reference of the Appliance energy consumption dataset. <https://archive.ics.uci.edu/ml/datasets/Appliances+energy+prediction>



## 9. Analysis of Science

*Science may be regarded as the art of data compression.*

Li & Vitányi

### **TODO: Write this introduction**

In this chapter we will use our theory of nescience to study our current scientific knowledge

We will see how we can approximate the concept of nescience in practice. The concept of surfeit will be based, as it was described in Chapter XX, on the concept of redundancy, and redundancy will be computed in practice based on standard text compressors.

A historical analysis of the evolution of nescience is also covered in the chapter.

At the end of the chapter we will extend the analysis from individual research topic to research areas, in order to understand

### **9.1 Describing Current Knowledge**

According to the theory of nescience, evaluating our current knowledge begins by selecting a specific set of entities we want to understand. Within this framework, universal sets that include every possible entity, whether

known or unknown, are not permitted. This restriction arises from the logical paradoxes associated with such sets, including Cantor's theorem, which demonstrates that the set of all subsets has a strictly greater cardinality than the set itself, and Russell's paradox, which involves sets that are not members of themselves (see Section 2.1). Therefore, in this chapter, since our objective is to assess the current state of human knowledge, we focus on a finite set composed of entities that are already known and have been studied by science. These entities are suitable for analysis because humanity possesses at least some information about them.

The next step involves identifying the best available encoding of the selected entities as representations, typically composed of data or facts, followed by determining the most accurate descriptions we have, such as models, theories, and laws. In the theory of nescience, we maintain a clear distinction between representations and descriptions because this separation facilitates the discovery of new knowledge: either by improving the representation or by refining the description. However, since our aim here is to evaluate rather than expand our knowledge, we simplify the process by using only descriptions. These descriptions serve both as the representation and the description of an entity (see Section XX, which explains why descriptions can fulfill both roles). Moreover, we will drop the requirement that these descriptions be computable, since very few scientific descriptions in practice are actually computable. Requiring computability would severely limit the scope of our analysis and exclude a significant portion of scientific knowledge.

For our analysis, we use descriptions derived from the collection of scientific pages from Wikipedia. Wikipedia is a free, collaborative online encyclopedia launched in 2001 that allows anyone with internet access to create, edit, and update articles on a wide array of topics. It is maintained by a global community of volunteer contributors and aims to provide reliable, neutral, and verifiable information to the public. Several features make Wikipedia particularly suitable for analyzing scientific knowledge: it maintains a transparent version history that tracks content changes over time; supports collaborative validation that reduces individual bias; enforces strict citation requirements to ensure information is verifiable and based on authoritative sources; offers broad and consistent coverage across scientific disciplines; and is dynamically updated to reflect new findings and corrections.

Wikipedia is fundamentally a tertiary source. It compiles and synthesizes information from primary sources, such as firsthand accounts of events or discoveries, and secondary sources, which interpret and evaluate those primary materials. Tertiary sources play a key role in academia by summarizing information that has already been thoroughly discussed and vetted. High-

quality Wikipedia articles strive to be comprehensive, covering all major aspects of a topic in appropriate detail while avoiding undue emphasis on minor or peripheral information.

Furthermore, Wikipedia is inherently citational. Every statement included in a Wikipedia article must be supported by a published, reliable, and verifiable source. Original research is not allowed, meaning that novel claims, hypotheses, or breakthroughs must already be published in credible sources such as academic journals or books before being included. Wikipedia also employs a form of peer review, where one or more editors review and suggest improvements to an article. While this process is not anonymous or uniformly applied like traditional academic peer review, it still acts as a quality control mechanism that enhances the coverage, clarity, and accuracy of articles through community oversight.

Wikipedia pages are written in the MediaWiki Markup Language, a simplified system for formatting text that allows users without technical knowledge of XHTML or CSS to easily edit articles. Before we can analyze the content of a scientific page, it is essential to remove all markup tags and formatting elements to isolate only the meaningful textual information. To achieve this, we used the Python library `wikibetxparser`, which allows us to parse and process the raw wikitext of Wikipedia articles to extract the relevant content. In addition to stripping out the markup, the library was configured to eliminate other non-relevant elements, such as images, references, and lists, which do not contribute directly to our analysis of knowledge.

Wikipedia articles are written in the MediaWiki Markup Language, a simplified formatting system that allows users without technical expertise to contribute easily. Before we can analyze this content, it is necessary to remove all markup tags and formatting elements to isolate only the meaningful text. To do this, we used the Python library `wikibetxparser`, which parses raw wikitext and extracts relevant content. In addition to removing markup, the library was configured to exclude other non-essential elements such as images, references, and lists, which do not directly support our knowledge analysis.

Wikipedia hosts a vast collection of articles covering disciplines such as history, literature, politics, entertainment, and science. Since our goal is to evaluate scientific knowledge specifically, we limited our analysis to articles categorized under the "Science" category. This subset includes topics related to the natural and formal sciences—such as physics, biology, chemistry, and mathematics—ensuring that our study remains aligned with the goal of assessing scientific understanding. We further refined this selection by excluding articles focused on individuals, organizations, awards, lists, portals,

and similar metadata-oriented content. This filtering ensures that our analysis includes only those articles centered on scientific concepts, processes, or phenomena.

Both the list of articles and the full content of each article were retrieved directly from Wikipedia using the MediaWiki API. This API provides programmatic access to Wikipedia's content, allowing automated queries for page metadata, article text in wikitext format, category information, and revision history. It ensures we are working with the most current and accurate versions of each article while offering a reproducible and efficient method for gathering large-scale data<sup>1</sup>. A detailed explanation of the entire extraction and preparation process can be found in Appendix ??, allowing any interested reader to reproduce the steps and ensure the consistency of the analysis.

## 9.2 Measuring Knowledge

As mentioned in the previous section, to evaluate our current knowledge in practice, we used the collection of scientific pages from Wikipedia, which we pre-processed to extract the relevant information for our analysis. We focused exclusively on the topic descriptions, treating descriptions and representations as equivalent. In this context, since descriptions and representations are considered the same, the inaccuracy of a topic is effectively zero, leaving miscoding and surfeit as our primary metrics of interest.

To demonstrate how our theory can be practically applied to assess the current state of knowledge, we selected a collection of topics listed under the Theoretical Computer Science category on Wikipedia. This includes all pages within the main category, as well as those in its subcategories and their respective subcategories, up to a depth of five levels. Figure 9.1 illustrates the Theoretical Computer Science category (Level 1) alongside all its Level 2 subcategories, such as Theory of Computation, Graph Theory, and Logic in Computer Science.

Only pages classified as "GA" have been considered. Articles in the GA class are considered complete, and they have been examined by one or more impartial reviewers. From these list, we have removed pages related to non-relevant topics, such as journals, symposiums, associations, or awards. Additionally, we have removed articles too generic pages (e.g. parallel computing) or pages describing multiple algorithms (e.g. Tropical cyclone forecasting). For a detailed description of the Wikipedia page preprocessing, please refer to Appendix ??.

---

<sup>1</sup>The articles analyzed in this chapter correspond to April 2025.



Figure 9.1: Categories in Theoretical Computer Science

Only pages classified as "GA" (Good Article) have been considered. GA articles are regarded as complete and have been reviewed by one or more impartial evaluators. From this list, we excluded pages related to non-relevant topics, such as journals, symposiums, associations, or awards. We also removed overly generic articles (e.g., Parallel Computing) and pages describing multiple algorithms (e.g., Tropical Cyclone Forecasting). For a detailed description of the Wikipedia page preprocessing, please refer to Appendix ??.

The topics analyzed cover a wide range of areas within theoretical computer science and mathematics. In graph theory, we explored topics such as graph coloring (Greedy coloring), structural properties and special classes

(Snark, Perfect graph, Halin graph, Laves graph, Rook's graph, Rado graph, Well-covered graph, Pseudoforest, Component, Unit distance graph, Cop-win graph), graph transformations (Graph homomorphism, Logic of graphs), and graph-based theorems (Steinitz's theorem, De Bruijn–Erdős theorem, Turán's brick factory problem). In algorithms and data structures, we considered classical techniques like Binary search, Selection algorithm, Euclidean algorithm, Fast inverse square root, Farthest-first traversal, Linear probing, Trie, as well as more specialized algorithms such as the Gale–Shapley algorithm, Widest path problem, and data structures like the Cartesian tree. Topics in logic and computational foundations included 2-satisfiability, the Rule of inference, and the BIT predicate. In mathematical methods and theorems, we covered results such as Pick's theorem, Viète's formula, Sylvester–Gallai theorem, Sylvester's sequence, Shapley–Folkman lemma, and the Handshaking lemma. The list also touches on applied mathematics and modeling, through topics like the Dirac delta function, Earth–Moon problem, Tropical cyclone forecast model, Finite subdivision rule, and Network synthesis. Furthermore, in machine learning and computational complexity, we included Reinforcement learning from human feedback and the Small set expansion hypothesis. Finally, some topics address computer science system design and principles, such as the Allocator (C++), Book embedding, Three utilities problem, the Commutative property, and visualization techniques like the Arc diagram. Overall, the list covers a broad spectrum of topics within the area of theoretical computer science, highlighting its diversity and interdisciplinary connections.

### 9.2.1 Surfeit

In Section 5.2, we introduced the concept of surfeit as a relative measure to quantify the unnecessary effort involved in explaining an entity using a particular description. The surfeit of a description  $d$  for a representation  $r$  is defined as:

$$\sigma(d, r) = \frac{|l(d) - K(r)|}{l(d)}$$

where  $r \in \mathcal{B}^*$  and  $d \in \mathcal{D}$  is a description of  $r$ . Intuitively, the less we know about an entity, the longer our description tends to be. As our understanding of the entity improves, we should be able to remove all redundant elements from its description.

In practice, when descriptions and representations are considered equivalent, the concept of surfeit simplifies to:

$$\rho(d) = 1 - \frac{K(d)}{l(d)}$$

| Algorithm                     | Analysis of Boolean functions   | Automated reasoning     |
|-------------------------------|---------------------------------|-------------------------|
| Farthest-first traversal      | Gale–Shapley algorithm          | Widest path problem     |
| Cartesian tree                | Viète’s formula                 | Dirac delta function    |
| Euclidean algorithm           | Fast inverse square root        | Rule of thumb           |
| Binary search                 | Linear probing                  | Selection algorithm     |
| 2-satisfiability              | Graph homomorphism              | Logic of games          |
| Steinitz’s theorem            | De Bruijn–Erdős theorem         | Earth–Moon problem      |
| Turán’s brick factory problem | Laves graph                     | Rook’s problem          |
| Snark                         | Perfect graph                   | Clique problem          |
| Feedback arc set              | Euclidean minimum spanning tree | Combinatorial game      |
| Unit distance graph           | Halin graph                     | Cop-win strategy        |
| Pseudoforest                  | Well-covered graph              | Rado graph              |
| Network synthesis             | Telephone number (mathematics)  | Book embedding          |
| Three utilities problem       | BIT predicate                   | Pick’s theorem          |
| Sylvester–Gallai theorem      | RL from human feedback          | Finite subdivision rule |
| Sylvester’s sequence          | Tropical cyclone forecast model | Shapley–Folklore        |
| Theil–Sen estimator           | Trie                            | Rule of inference       |
| Commutative property          | Allocator (C++)                 | Handshaking lemma       |

Table 9.1: Topics in Theoretical Computer Science

This is referred to as the redundancy of a description.

The Kolmogorov complexity of the Wikipedia pages was estimated by compressing the raw text. For compression, we used bzip2, a free and open-source program based on the Burrows–Wheeler algorithm. bzip2 is commonly used in practice to estimate Kolmogorov complexity because it employs a very large compression buffer. It is well known that if the buffer size is too small, the estimation of a text’s Kolmogorov complexity can be significantly distorted. For example, the gzip compressor uses a 32 KB buffer, which is too small for our purposes. In contrast, bzip2 offers a buffer size of 900 KB at its highest compression setting (compression level of 9), which is more than sufficient for compressing Wikipedia pages.

Figure 9.2 (left) depicts an histogram of the redundancy for the 50 theoretical computer science articles studied. The histogram shows that, once markup and boiler-plate are stripped, redundancy of articles is tightly concentrated between  $\approx 0.75$  and 0.82, peaking near 0.80. In practical terms, even the clean prose of these pages can typically be shrunk to about three-quarters of its original size, indicating a common stock of recurring phrases, technical jargon, and definitional patterns across the corpus. The scatter plot in figure 9.2 (right) adds a systematic dimension: redundancy rises by roughly 2–3

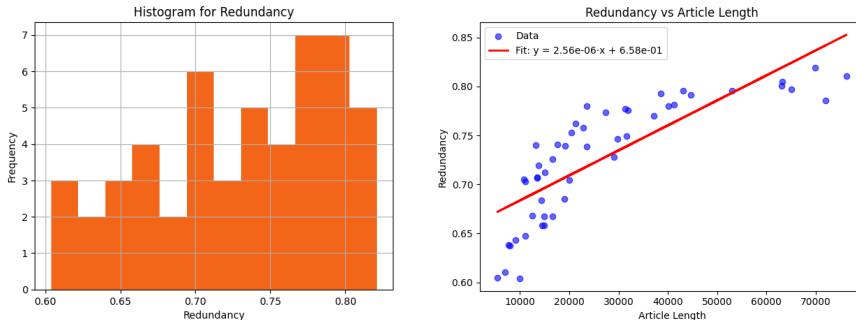


Table 9.2: Redundancy of Topics

percentage points for every additional 10000 characters. Because every file fits inside the single 900 kB block used at `bzip2`'s highest setting (-9), block-size effects are neutralised and the only compressor artefacts left—headers and model warm-up—explain at most a few tenths of that increase. The bulk of the slope therefore reflects the prose itself: longer articles increasingly reuse terminology, theorem–proof templates, and explanatory scaffolding, making them intrinsically more compressible. In short, both figures together show that theoretical-computer-science pages share a fairly uniform baseline of linguistic redundancy, and that this redundancy grows gradually with article length because the writing becomes more internally self-similar rather than because of any quirk of the compression tool.

Table 9.3 lists the ten topics from Wikipedia within the area of Theoretical Computer Science that exhibit higher redundancy. Each page revolves around a single, well-defined object—a classic algorithm (binary search, Euclidean algorithm, clique problem), a mathematical construct (Dirac delta, Rado graph), or a concise metatheorem (2-SAT, Shapley–Folkman lemma). Pages of this kind inevitably iterate the same symbol palette, definitions, and step-by-step explanations: every subsection restates the object, walks through an example, then re-expresses the same idea in a different formalism (pseudocode, recurrence, proof sketch, code snippet). That self-similar scaffolding gives compressors a wealth of repeated n-grams to exploit, driving redundancy up to 0.79 – 0.82 even after boiler-plate has been stripped. While the list includes some of the longer articles in the corpus (e.g. Dirac delta at 117 kB), it also contains mid-sized pages such as Rado graph (38 kB). In other words, high redundancy is not just a by-product of length; rather, it emerges when a compact conceptual core is expanded through multiple parallel presentations (history, intuition, formal statement, variations, applications, proof, pseudocode).

Table 9.4 lists the ten topics that exhibit lower redundancy. These articles

| Topic                  | Redundancy | Page Length | Length Compressed |
|------------------------|------------|-------------|-------------------|
| Dirac delta function   | 0.821      | 117121      | 20969             |
| Binary search          | 0.819      | 69838       | 12634             |
| Euclidean algorithm    | 0.811      | 113242      | 21455             |
| 2-satisfiability       | 0.811      | 76225       | 14442             |
| Shapley–Folkman lemma  | 0.805      | 63172       | 12323             |
| Clique problem         | 0.801      | 63125       | 12574             |
| RL from human feedback | 0.797      | 64981       | 13160             |
| Book embedding         | 0.796      | 53046       | 10831             |
| Rule of inference      | 0.795      | 43158       | 8826              |
| Rado graph             | 0.793      | 38626       | 7989              |

Table 9.3: Topics with higher redundancy

sit in the low-redundancy tail (between 0.60 and 0.67) differ from the high-redundancy ones in both size and content mix. (i) Every entry is under 17 kB—well below the corpus median. Because bzip2’s fixed header and model “warm-up” overhead account for a larger share of such small files, their redundancy starts a few percentage points lower even before content is considered. This matches the positive slope in the scatter-plot: shorter pages naturally cluster toward the left-hand foot of the trend line. (ii) Many of these topics centre on a single formula, sequence, or code idiom (e.g. Viète’s formula, Sylvester’s sequence, Allocator). They pack unique symbols, numeric constants and one-off identifiers that appear only once or twice, so compressors see far fewer repeated n-grams than in algorithm-walk-through articles. Others, such as “Tropical cyclone forecast model” and “Finite subdivision rule”, read more like concise surveys, stringing together disparate subtopics rather than iterating one core definition from multiple angles; that heterogeneity also suppresses repetition.

Together, the list illustrates the floor of the redundancy spectrum: pages that are (i) short enough for compression overhead to matter and (ii) information-dense or eclectic enough to avoid recycling the same prose patterns never reach the 0.70–0.80 plateau seen elsewhere. Their position reinforces the earlier finding that rising redundancy with length is not merely a compressor artefact; it also depends on how much authors reuse terminology and explanatory scaffolding as an article grows.

### 9.2.2 Inaccuracy

Inaccuracy serves as the second metric in assessing our understanding of a research entity. The underlying idea is that the more accurate our model, the

| Topic                           | Redundancy | Page Length | Length Compressed |
|---------------------------------|------------|-------------|-------------------|
| Tropical cyclone forecast model | 0.667      | 16675       | 5550              |
| Allocator (C++)                 | 0.658      | 14887       | 5085              |
| Finite subdivision rule         | 0.658      | 14416       | 4933              |
| Snark (graph theory)            | 0.648      | 11020       | 3882              |
| Theil–Sen estimator             | 0.643      | 9171        | 3275              |
| Earth–Moon problem              | 0.638      | 7640        | 2762              |
| Halin graph                     | 0.637      | 7953        | 2885              |
| Viète’s formula                 | 0.61       | 7016        | 2733              |
| Turán’s brick factory problem   | 0.605      | 5427        | 2146              |
| Sylvester’s sequence            | 0.604      | 9994        | 3960              |

Table 9.4: Topics with Lower redundancy

better our understanding of the entity. Formally, we calculate the inaccuracy of a description  $d$  as the normalized information distance between the original representation  $r$  and the output representation  $r'$  generated by our description  $d$ . That is, inaccuracy is quantified as the length of the smallest computer program capable of correcting the erroneous output of our model.

Inaccuracy, serving as the second gauge to measure our comprehension of a research entity, is based on the principle that the more precise our model, the better our grasp of the entity. Formally, the inaccuracy of a description  $d$  is computed as the normalized information distance between the original representation  $r$  and the output representation  $r'$  generated by the description  $d$ . Thus, inaccuracy is assessed as the extent of the smallest computer program that can rectify the incorrect output of our model.

Inaccuracy evaluates how well the output of our description aligns with the selected representation encoding the entity. However, this representation could be flawed itself, as discussed in the preceding chapter. Inaccuracy focuses solely on the description  $d$ , neglecting the potential miscoding within the representation  $r$ . Furthermore, even though it doesn’t require an oracle, inaccuracy cannot be calculated for every case, so it needs to be estimated in practical situations, as we will explore in Part III of this book.

Let us consider  $r \in \mathcal{B}^*$  as a representation, and  $d \in \mathcal{D}$  as a description, where  $d = \langle TM, a \rangle$ . We then define the *inaccuracy* of the description  $d$  with respect to the representation  $r$ , denoted as  $\iota(d, r)$ , according to the following formula:

$$\iota(d, r) = \frac{\max\{K(r | \delta(d)), K(\delta(d) | r)\}}{\max\{K(r), K(\delta(d))\}}$$

Intuitively, accuracy measures the difficulty in converting an incorrect rep-

resentation  $r'$  produced by a description  $d$  into the original representation  $r$ . In essence, this involves the computation of the normalized information distance between  $r'$  and  $r$ .

In practice, and in the particular case of descriptions based on the scientific articles of Wikipedia, we cannot apply this definition, because an article of Wikipedia is not a Turing machine that produces a string based output.

Figure 9.8 left depicts a histogram showing the inaccuracy of the analyzed topics, and right side shows a scatterplot of the article lengths vs. the length of the correspondign talk pages, toghether with a regression line fitted to the data. The two plots suggest that, within this sample of good-quality theoretical computer science pages, talk pages chatter is generally modest and becomes proportionally smaller as articles grow. Inaccuracy, approximated as  $\text{talk} / (\text{article} + \text{talk})$ , is highly right-skewed. Roughly half the pages fall below 0.05 and more than three-quarters below 0.15, meaning their talk pages contain at most one word of discussion for every six to twenty words of main-text content. Only a small tail of articles pushes beyond 0.40, indicating a handful of topics that still attract extensive debate or revision despite their “good” label. In the scatter plot we can observe that the best-fit line ( $y \approx -2.44 \times 10^{-6} \text{length} + 0.271$ ) slopes downward, so predicted inaccuracy drops from about 0.27 for a 0-length stub to  $\approx 0.10$  at 70 kB. Empirically, short articles (< 15 kB) show the widest spread—from virtually no talk to talk pages two-thirds the size of the article—whereas long articles (> 50 kB) cluster below 0.15 with only rare outliers. In other words, once an article expands to tens of thousands of characters, the relative volume of unresolved discussion shrinks, suggesting that length correlates with maturity and consensus. Taken together, the figures imply that most good theoretical-CS pages are comparatively settled, that the few contentious ones are disproportionately short, and that growing an article tends to absorb or resolve the issues reflected on its talk page rather than magnifying them.

Table 9.9 lists topics exhibiting the lowest levels of nescience. The topics with the lowest inaccuracy values ( $\leq 0.04$ ) look like settled science. They cover classical, mathematically-rigorous results—matching algorithms (Gale–Shapley), structural graph properties (perfect graphs, Steinitz’s theorem), and well-studied data-structures or routing primitives (Cartesian tree, widest-path problem). For such topics there is little room for interpretive dispute: statements are either formally correct or not, and once a clean exposition is in place subsequent editors rarely need lengthy back-and-forth. That is why some talk pages are only a few dozen words long (25 for Gale–Shapley) even when the article itself spans tens of kilobytes. Length alone does not guarantee a quiet talk page—the list includes both mid-sized texts ( $\approx 20$  kB) and the 72 kB Network synthesis article—yet all of them keep the ratio of

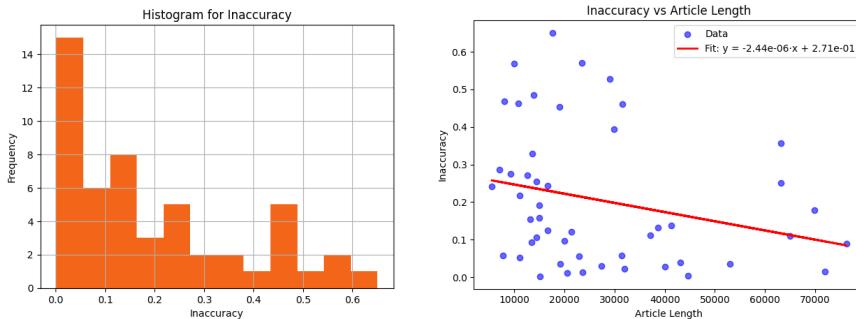


Table 9.5: Inaccuracy of Topics

| Topic                  | Inaccuracy | Article length | Talk length |
|------------------------|------------|----------------|-------------|
| Gale–Shapley algorithm | 0.001656   | 15072          | 25          |
| Perfect graph          | 0.004412   | 44683          | 198         |
| Unit distance graph    | 0.012203   | 20479          | 253         |
| Rook's graph           | 0.013267   | 23576          | 317         |
| Network synthesis      | 0.014493   | 71945          | 1058        |
| Logic of graphs        | 0.022260   | 31933          | 727         |
| Steinitz's theorem     | 0.027765   | 40094          | 1145        |
| Cartesian tree         | 0.030849   | 27395          | 872         |
| Widest path problem    | 0.035644   | 19128          | 707         |
| Book embedding         | 0.036141   | 53046          | 1989        |

Table 9.6: Topics with lower inaccuracy

discussion to content below 4%. Where talk does appear (e.g. 1 989 words for Book embedding or 1 145 for Steinitz's theorem) it is still dwarfed by the main text, implying that open issues tend to be minor wording tweaks, sourcing details, or peripheral expansions rather than fundamental disagreements. In short, low-inaccuracy articles are those whose subject matter is uncontroversial, formally locked-down, and already presented in a stable, comprehensive fashion, leaving editors with little need for ongoing debate.

Finally, Table 9.7 lists the topics with the highest nescience values. The articles with the highest inaccuracy scores are not necessarily the most error-ridden; rather, they are the ones whose talk pages have become arenas for protracted discussion because their subjects invite disagreement or continual refinement—foundational concepts like the commutative property and tries that draw many first-time editors, folklore-tinged algorithms such as the fast inverse square root, or theorems and graph classes (e.g., Sylvester–Gallai, pseudoforests, Halin graphs) that have multiple equivalent statements, proofs,

| Topic                    | Inaccuracy | Article length | Talk length |
|--------------------------|------------|----------------|-------------|
| Sylvester–Gallai theorem | 0.394610   | 29810          | 19431       |
| Pseudoforest             | 0.454130   | 18999          | 15806       |
| Selection algorithm      | 0.461374   | 31578          | 27049       |
| BIT predicate            | 0.463016   | 10817          | 9327        |
| Halin graph              | 0.467671   | 7953           | 6987        |
| Three utilities problem  | 0.485948   | 13846          | 13089       |
| Fast inverse square root | 0.528723   | 29001          | 32536       |
| Sylvester’s sequence     | 0.567808   | 9994           | 13130       |
| Trie                     | 0.570980   | 23503          | 31280       |
| Commutative property     | 0.650826   | 17583          | 32773       |

Table 9.7: Topics with higher inaccuracy

or naming conventions. In every case the talk page approaches or even exceeds the size of the main text, so anywhere from 40% to 65% of the combined bytes are debate rather than exposition; this ratio is amplified by the fact that most of these articles are relatively short ( $\approx 10\text{--}30$  kB), meaning even modest amounts of conversation loom large. Together they illustrate that a high inaccuracy score chiefly signals ongoing editorial contention—about scope, presentation order, historical credit, or performance claims—rather than a simple lack of factual correctness, and that such contention is most pronounced where definitions are ambiguous, folklore collides with formalism, or broad audiences repeatedly revisit basic material.

### 9.2.3 Nescience

According to the theory of nescience, our understanding of an entity should be based on the quality of the description used to explain it. In Chapter 6, we introduced a quantitative measure of our ignorance regarding a research entity. This measure depends on the miscoding of a string-based representation of the entity, as well as the inaccuracy and surfeit of the model describing this representation. Ideally, we seek representations and descriptions that simultaneously minimize these three aspects.

In practical terms, particularly when entities correspond to topics covered by scientific Wikipedia pages, we consider only the surfeit and inaccuracy of a description. In this scenario, the representation of a topic coincides with its description, rendering the computation of miscoding irrelevant.

To address the multi-objective optimization problem posed by nescience, we apply a global criterion approach (see Section 16.5.3). This method solves multi-objective optimization problems by minimizing the distance between

a chosen reference point and the feasible region of the objective space. We select the origin vector  $(0, 0)$  as our reference point. The distance metric used will be the harmonic mean of surfeit and inaccuracy, defined as:

$$\frac{2}{\iota(d, r)^{-1} + \sigma(d, r)^{-1}}$$

As described in previous sections, surfeit is approximated by the redundancy of the topic's description (the Wikipedia page), while inaccuracy is estimated by the length of the corresponding Talk page.

As it was explained in Section XX, since we are using a decision maker to compute nescience based on redundancy and inaccuracy, and since these quantities do not have the same scale, it is highly convenient to apply the following additional transformation to them:

$$\mu_t = \frac{\mu_t - \min(\mu)}{\max(\mu) - \min(\mu)}$$

where  $\mu_t$  refers to the considered metric (nescience, relevance, ...).

Figure 9.8 left depicts a histogram showing the nescience levels of the analyzed topics, and right side shows a scatterplot of the article lengths vs. the nescience, together with a regression line fitted to the data. The two panels paint a mixed picture of how much “unknown-ness” remains in these 50 “good” theoretical-CS articles once both redundancy (compressibility of the prose) and inaccuracy (relative size of the talk page) are factored in. Histogram — a long, uneven tail. Roughly one-third of the articles cluster below nescience  $\approx 0.10$ , meaning they are simultaneously concise (low redundancy) and largely uncontested (little talk). Frequency then stays fairly flat through the  $0.15 - 0.35$  band and drops again, before a second bump appears between  $0.60$  and  $0.75$ . That small peak corresponds to pages that are both highly repetitive and heavily debated—the same ones that topped the earlier redundancy and inaccuracy lists. In short, while many good-class pages are well understood, a non-trivial minority still exhibits a pronounced knowledge gap. Scatter plot — almost length-neutral. The best-fit line rises only 0.007 points for every additional 10 000 characters (slope  $\approx 7.5 \times 10^{-7}$ ), so article length alone is not a strong predictor of nescience. Short pages can swing from near-zero to 0.7, depending on how contentious they are, whereas long pages ( $> 50$  kB) compress more (pushing nescience up) but also attract proportionally less talk (pushing it down), leaving them parked in a mid-range around 0.25–0.35. The gentle upward tilt therefore reflects the fact that redundancy grows with length a bit faster than inaccuracy falls, but the wide vertical spread shows that editorial stability—rather than size—is the decisive factor. Overall, “good” status keeps most theoretical-CS articles

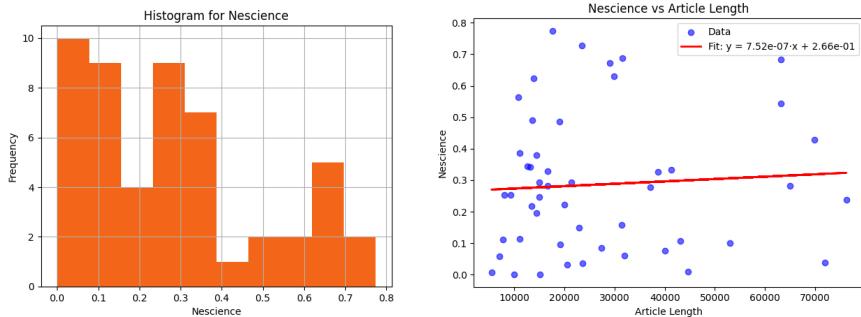


Table 9.8: Nescience of Topics

in a low-to-moderate nescience zone, yet about a quarter of them still suffer from enough repetition and unresolved discussion to signal that our coverage of those topics remains less mature than the label might suggest.

Table 9.9 lists topics exhibiting the lowest levels of nescience. The articles that score best on the nescience metric fall into two complementary archetypes, each driving the harmonic mean down by making one of the two ingredients—redundancy or inaccuracy—almost vanish. The first archetype is epitomised by "Sylvester's sequence" and "Turán's brick-factory problem": they are short, information-dense pages whose prose compresses poorly (normalized redundancy  $\approx 0$ ), so even a rather lively talk page cannot lift their nescience above zero. The second archetype—illustrated by "Gale–Shapley algorithm", "Perfect graph", "Network synthesis", and several other graph-theoretic topics—shows the opposite pattern: their text is highly repetitive (normalized redundancy  $\approx 0.8^{\sim}0.86$ ), but the talk pages are almost silent, so inaccuracy is near zero and the harmonic mean again collapses. Taken together, the list reveals that low lack-of-knowledge can arise either from concise, debate-heavy but non-redundant expositions or from long, internally repetitive but well-settled treatments; what matters for nescience is that at least one dimension of uncertainty is squeezed to the floor.

Finally, Table Finally, Table 9.10 lists the topics with the highest nescience values. The upper tail of the nescience scale is populated by pages that score high on both axes at once—they are repetitive and heavily debated—so the harmonic mean refuses to average the trouble away. Half of the list ("BIT predicate", "Three-utilities problem", "Fast inverse square root", "Selection algorithm", "Trie", "Commutative property") shows normalized inaccuracy  $\geq 0.70$ , signalling talk pages that rival the article in size; the same set attracts a steady stream of newcomers or folklore-laden claims, which keeps discussion alive. The other half ("Shapley–Folkman lemma", "Clique problem", "Sylvester–Gallai theorem", "Farthest-first traversal") earns its

| Topic                         | Nescience | Norm. Inaccuracy | Norm. Redundancy |
|-------------------------------|-----------|------------------|------------------|
| Sylvester's sequence          | 0.000000  | 0.872117         | 0.000000         |
| Gale–Shapley algorithm        | 0.000000  | 0.000000         | 0.499166         |
| Turán's brick factory problem | 0.007362  | 0.370945         | 0.003718         |
| Perfect graph                 | 0.008448  | 0.004245         | 0.864084         |
| Unit distance graph           | 0.031744  | 0.016248         | 0.686717         |
| Rook's graph                  | 0.035003  | 0.017887         | 0.812327         |
| Network synthesis             | 0.038635  | 0.019774         | 0.836612         |
| Viète's formula               | 0.057626  | 0.437415         | 0.030845         |
| Logic of graphs               | 0.061030  | 0.031738         | 0.791546         |
| Steinitz's theorem            | 0.076638  | 0.040219         | 0.811139         |

Table 9.9: Topics with lower nescience

| Topic                    | Nescience | Norm. Inaccuracy | Norm. Redundancy |
|--------------------------|-----------|------------------|------------------|
| Farthest-first traversal | 0.490109  | 0.505215         | 0.475880         |
| Shapley–Folkman lemma    | 0.543853  | 0.384947         | 0.926181         |
| BIT predicate            | 0.562673  | 0.710692         | 0.465683         |
| Three utilities problem  | 0.622535  | 0.746017         | 0.534125         |
| Sylvester–Gallai theorem | 0.629670  | 0.605318         | 0.656063         |
| Fast inverse square root | 0.671295  | 0.811908         | 0.572197         |
| Clique problem           | 0.683333  | 0.548081         | 0.907206         |
| Selection algorithm      | 0.688247  | 0.708163         | 0.669420         |
| Trie                     | 0.727080  | 0.877003         | 0.620932         |
| Commutative property     | 0.774591  | 1.000000         | 0.632109         |

Table 9.10: Topics with higher nescience

place chiefly through very high redundancy ( $\approx 0.90$  in the case of the clique problem), reflecting articles that restate definitions, proofs and examples in multiple near-duplicate forms. In every case the other dimension is still uncomfortably large ( $\geq 0.38$ ), so neither trimming repetition nor settling open talk-page issues alone would be enough to pull these topics out of the danger zone. In short, high-nescience entries are those where unresolved editorial contention co-exists with a self-similar writing style—pages that simultaneously need consensus-building and structural tightening before they can be considered well understood.

### 9.2.4 Conclusion

Across the 50 “good-class” Wikipedia pages in theoretical computer science, our indicators tell a nuanced story about what the encyclopaedia can reveal—and obscure—about collective understanding. Talk-page activity, captured by our inaccuracy ratio, does a credible job of flagging articles whose content is still contested: pages with minimal discussion almost always cover settled, formally unambiguous material, whereas those whose talk pages rival the main text correspond to concepts riddled with naming ambiguity, folklore or pedagogical disagreement. Redundancy, distilled from compression ratios after stripping boiler-plate, complements this view by signalling how tightly authors have distilled the topic: low values align with concise, information-dense expositions, while high values mark articles that re-iterate definitions, proofs and code snippets in multiple guises. Each metric thus illuminates a different facet—editorial consensus on one side, stylistic efficiency on the other—yet neither by itself captures “knowledge quality” outright.

Their harmonic mean, which we dubbed nescience, succeeds as a triage tool because it punishes a topic the moment either dimension becomes extreme. Articles that are both verbose and hotly debated—such as those on the fast inverse square root or the clique problem—cluster at the high end, clearly pointing to areas where Wikipedia’s coverage remains unsettled. Conversely, pages with either crisp, irredundant prose or near-empty talk pages sink to the bottom, reflecting mature, well-understood subjects like the Gale–Shapley algorithm or perfect graphs. In that sense, nescience offers a practical snapshot of our residual ignorance as reflected in the encyclopaedia: it reliably highlights where clarification, consolidation or further research are still needed, even if it cannot, on its own, disentangle whether the root cause is factual dispute, pedagogical complexity, or simple editorial neglect.

## 9.3 Measuring Research Areas

In this section, we evaluate the nescience of entire research areas rather than individual topics, measuring how much knowledge each discipline encompasses. We examined all English-language “good-class” articles in six broad disciplines (biology, chemistry, mathematics, philosophy, physics, and psychology) comprising roughly 600 pages (see Table 9.11). For each discipline, we assessed three metrics: inaccuracy, redundancy, and nescience.

Wikipedia organizes articles into categories by topic to simplify navigation and discovery. Rather than a strict hierarchy, categories form an interconnected network: each category can have multiple subcategories, and a subcategory may belong to multiple parent categories. Editors avoid

| Area        | Num. Topics |
|-------------|-------------|
| Biology     | 214         |
| Philosophy  | 33          |
| Psychology  | 44          |
| Mathematics | 127         |
| Chemistry   | 138         |
| Physics     | 52          |

Table 9.11: Number of topics analyzed per area

circular links, where a category would indirectly include itself through its descendants. Conceptually, this structure resembles a partially ordered set in mathematics and computer science, and it allows diverse classification schemes to coexist within a unified framework. When one category clearly falls under another, it is designated as a subcategory to preserve a logical “is-a” relationship. Articles are typically placed only in their most specific relevant category to avoid redundancy. For example, the article on Claude Shannon appears in “Category:American information theorists,” not directly under the broader “Category:Mathematicians.”

The top level of Wikipedia’s category system is “Category:Contents,” which has no parent. From there, the path we have used proceeds as: Contents → Articles → Main topic classification → Academic disciplines → Science → Branches of Science (Applied Sciences, Formal Sciences, Natural Sciences, Social Sciences). Our analysis focuses on five scientific areas: Mathematics, which falls under Formal Sciences; Biology, classified within Life Sciences under Natural Sciences; Physics and Chemistry, both part of Physical Sciences in the Natural Sciences; and Psychology, included in Social Sciences. We also analyze Philosophy, categorized under Humanities within Academic disciplines. These selections ensure comprehensive coverage of both theoretical and empirical fields.

First, we looked at redundancy (see Figure 9.2), which shows how much an article repeats itself. We took out templates, lists, and other markup, then compressed each page using bzip2 at its highest setting. Pages with more repeated text stay larger after compression. We found that most good articles in every field fall in a redundancy range of about 0.70 to 0.80, so they repeat text at similar rates. Biology had the widest range, from about 0.52 up to over 0.90, meaning some articles are very concise while others include many standard sections. Chemistry was the most consistent group, clustering around 0.72, possibly because of regular naming rules and reaction sections. Mathematics, philosophy, physics, and psychology were all in the middle, with medians near 0.75 but small differences in spread. For example,

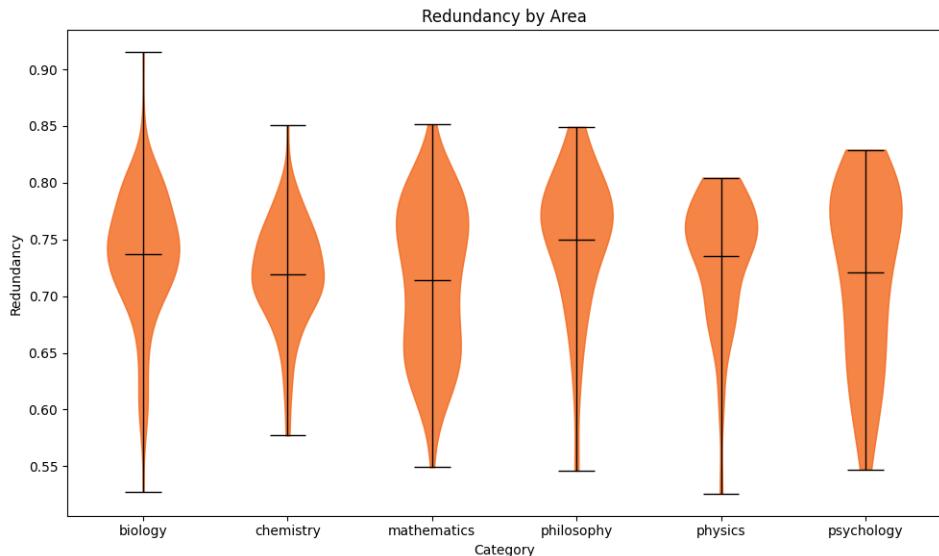


Figure 9.2: Redundancy by Area

math and philosophy pages sometimes repeat proofs or arguments, raising their upper range, while psychology has some short experiment-focused articles. Physics looks like psychology but with a slightly lower median. In short, redundancy is similar across fields: most good articles keep about three-quarters of their text after compression, showing that writing style and citation rules has a higher impact than the topic itself.

Next, we estimated inaccuracy by looking at editorial debate rather than factual error (see Figure 9.3). We looked at each article's talk page size and compared it to the total size of both the talk page and the article. A larger talk page means more discussion or disagreement. The violin plot shows that most articles in every field spend only a small part of their bytes on talk pages, but a few pages have much longer debates. In chemistry, most articles keep their talk pages below about 0.1 of the total bytes, and the median is just above that. Only a few articles reach up to 0.80. Biology, philosophy, and psychology also have most pages under roughly 0.15, but these fields have taller plots in the upper half. That means a few topics, like controversial medical issues in biology or big theoretical arguments in philosophy and psychology, spark long discussions. Mathematics and physics have the highest and widest range of debate. Their median sits around 0.25, and many pages go up to 0.5 or even 0.9, showing that some articles have as much discussion as main content. Overall, most scientific topics settle into low debate, but math and physics pages often have more back-and-forth about

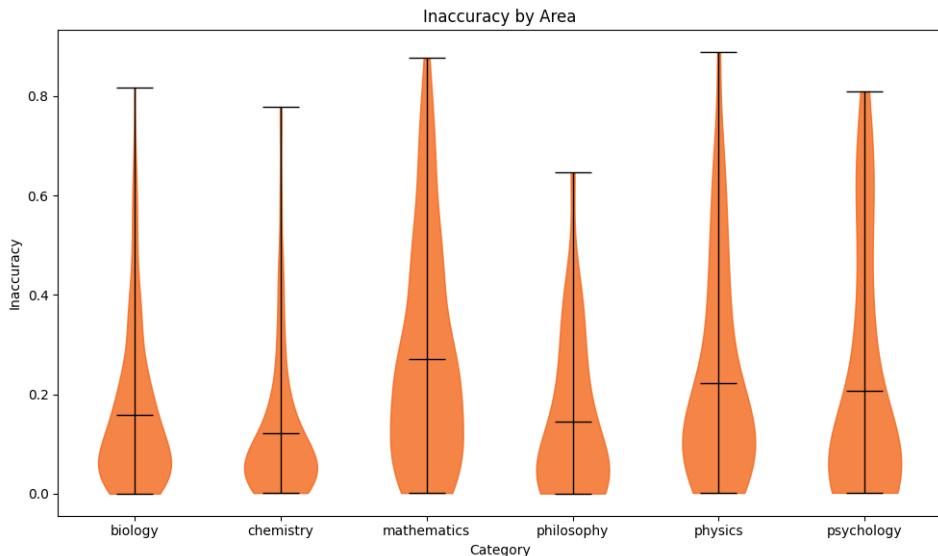


Figure 9.3: Inaccuracy by Area

definitions, notation, and sources, while chemistry articles are more stable.

Finally, we combined redundancy and inaccuracy into one score, nescience (see Figure 9.4). We first scaled each measure to range from 0 to 1, then took their harmonic mean. This score highlights pages that are either very repetitive, heavily debated, or both. In the violin plot, chemistry articles have low nescience (mostly under 0.20), showing they are concise and rarely debated. Biology is similar but has a few articles with higher scores, reflecting some ongoing talk-page discussions. Mathematics has the highest median nescience (around 0.40) and reaches up to 0.90, because many pages repeat concepts and have active debates over definitions. Philosophy and physics each show two peaks. one for well-settled articles near zero and one above 0.50 for more contested topics. Psychology mainly has low nescience, but some articles still face long debates. Overall, while every field has many well-covered pages, our nescience score shows that gaps in knowledge are biggest in mathematics and, to a lesser degree, in philosophy, physics, and psychology.

Table 9.12 shows the average scores for each discipline. The numbers show that redundancy is almost the same across all six fields, ranging from 0.71 to 0.75, but inaccuracy varies more than a factor of two, from 0.12 in chemistry to 0.27 in mathematics. Because redundancy stays nearly constant, the differences in the combined nescience score come mostly from inaccuracy. In other words, after removing repeated boilerplate, the

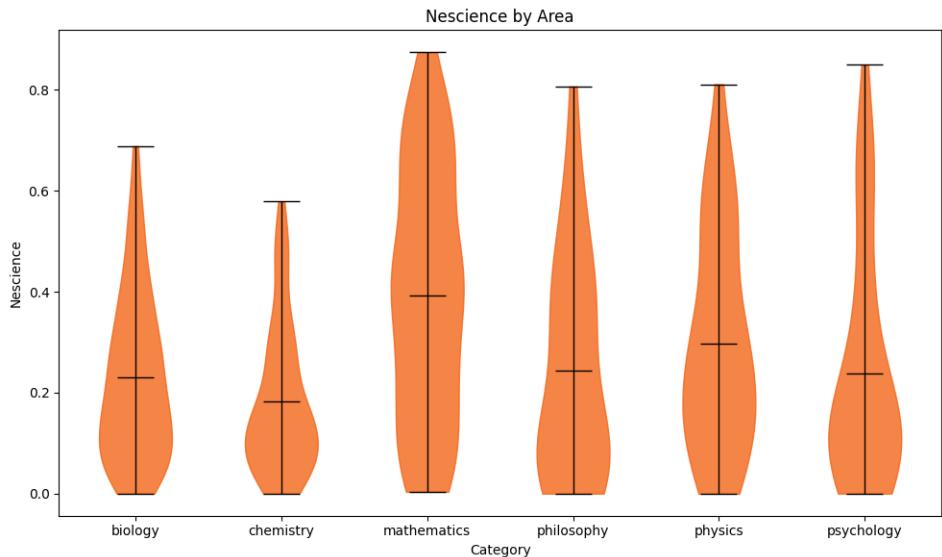


Figure 9.4: Nescience by Area

| Category    | Inaccuracy | Redundancy | Nescience |
|-------------|------------|------------|-----------|
| Biology     | 0.157773   | 0.736704   | 0.229765  |
| Chemistry   | 0.122821   | 0.719500   | 0.183116  |
| Mathematics | 0.271307   | 0.714365   | 0.392008  |
| Philosophy  | 0.144690   | 0.750007   | 0.244622  |
| Physics     | 0.223447   | 0.735314   | 0.297802  |
| Psychology  | 0.207355   | 0.721208   | 0.237835  |

Table 9.12: Average metrics by area

writing style is similar across subjects; what really changes is how much editors discuss or disagree on the talk pages. For example, chemistry's low inaccuracy (0.12) gives it the smallest nescience (0.18), even with average redundancy. In contrast, mathematics has the highest inaccuracy (0.27) and therefore the highest nescience (0.39), despite slightly lower redundancy. Physics and psychology have similar redundancy (around 0.72–0.74), but physics sees more talk-page debate (0.22 vs. 0.21), so its nescience is higher than psychology's. Philosophy shows the highest redundancy (0.75) but only a medium nescience (0.24) because its talk pages are quieter. Overall, this table tells us that for high-quality articles, the main difference between fields is how much editors argue, not how the text is written.

Putting together the redundancy and inaccuracy measures gives a single

nescience score. This shows that all fields have similar writing patterns, most high-quality articles compress to about 70–80% of their original length, but they differ in how much they spark online discussions. Chemistry and biology have the quietest talk pages, so they score lowest on nescience. Mathematics and physics have more debates, so they score highest, even though their writing density is similar. Philosophy and psychology fall in between. These gaps happen because each field has many settled articles and a smaller group that need more discussion. In short, Wikipedia’s top science articles are usually concise and well written, but the talk pages reveal where editors still disagree most. The biggest knowledge gaps remain in math topics heavy on definitions and in physics areas open to different interpretations.

## 9.4 The Evolution of Knowledge

Our understanding of a scientific topic typically improves through sustained research efforts over time, resulting in a reduction of the topic’s nescience, our lack of knowledge. Specifically, this improvement occurs through reductions in redundancy (repeated, unnecessary information) and inaccuracy (incorrect or misleading information), without significant increases in either component offsetting these improvements. New explanatory frameworks may emerge from innovative theories, refinements of existing theories, or simplified models. Ideally, each successive explanation will become more concise and accurate, removing previously redundant details and errors. Occasionally, new descriptions may temporarily lengthen as additional factual information is incorporated, causing a temporary increase in nescience. Nevertheless, the overall trend in scientific research should consistently be a progressive decrease in nescience as our knowledge advances.

To demonstrate how our theory can be used to characterize the evolution of our understanding of a scientific topic, that is, how nescience decreases with time, we have selected three highly relevant research topics as illustrative examples: graphene, CRISPR, and deep learning. Graphene is a single layer of carbon atoms arranged in a two-dimensional honeycomb lattice, known for its exceptional strength, conductivity, and versatility. CRISPR (Clustered Regularly Interspaced Short Palindromic Repeats) is a revolutionary gene-editing technology that allows for precise, targeted modifications to DNA. Deep learning is a subset of machine learning based on artificial neural networks, enabling complex pattern recognition and predictive modeling. These topics were chosen for their scientific importance and the richness of their associated descriptions, allowing us to apply our methodology and illustrate its effectiveness.

Figure 9.13 illustrates the evolution over time of redundancy, inaccuracy,

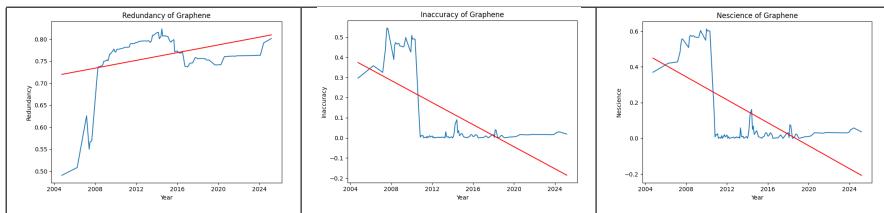


Table 9.13: Knowledge evolution of graphene

and nescience metrics for the topic of Graphene. Each point on the graph corresponds to the analysis of a cleaned-up version of the Wikipedia page at a specific point in time. The early history of the graphene article shows the classic growth-and-turmoil phase of a fast-moving research topic. Between 2005 and 2010 redundancy climbed from  $\approx 0.5$  to just over 0.7 as the page ballooned with additional sections, figures and repeated explanations, while inaccuracy, our proxy for debate, spiked above 0.5. That surge coincides with the period when graphene research exploded in the literature, patents, and the popular press, so editors were actively negotiating scope, terminology and sourcing. The high redundancy plus high inaccuracy pushed the nescience curve to its peak near 0.9 in 2009-2010, signalling that the encyclopaedia's coverage lagged behind the field's rapid advances.

A sharp inflection occurs in 2011. The talk-page ratio collapses almost to zero-like because extensive discussion was archived or split into sub-pages after the article stabilised-driving inaccuracy, and therefore nescience, down to negligible levels. From that point on the page continues to lengthen: redundancy drifts upward into the 0.78-0.82 band as definitions, production methods and applications are reiterated for different audiences. Yet talk-page activity remains a small fraction of total bytes, so nescience stays close to zero with only minor blips when new findings (e.g., commercial production techniques around 2014 or 2019) briefly rekindle debate. The negative regression slope on both the inaccuracy and nescience plots confirms a long-term convergence toward consensus, while the positive slope on redundancy simply reflects a mature, template-rich article that keeps accumulating detail without reopening fundamental disputes.

CRISPR's Wikipedia trajectory (see Figure 9.14) mirrors the arc of a scientific whirlwind maturing into mainstream knowledge. When the page first took shape (2010 - 2013) redundancy sat in the low-to-mid 0.60, reflecting a compact article that still compressed poorly, while inaccuracy-driven by an energetic talk page-hovered around 0.40. Those years coincide with the burst of laboratory discoveries that turned CRISPR-Cas9 from an obscure bacterial defence into a headline-making gene-editing tool, so

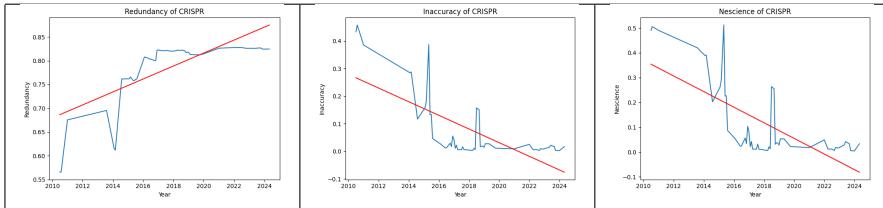


Table 9.14: Knowledge evolution of CRISPR

editors were constantly renegotiating scope and sourcing. Both factors pushed nescience above 0.70, signalling a large knowledge gap between fast-moving science and the encyclopaedia's ability to consolidate it.

A decisive shift occurs in 2015-2016. The talk-page share collapses, dropping inaccuracy to near zero almost overnight; the page was substantially rewritten and much of the debate was archived once standard terminology, mechanism diagrams and milestone experiments had stabilised. Redundancy, meanwhile, jumps above 0.75 and keeps inching up as new sections on ethics, patents and clinical trials are added—material that inevitably repeats the core biology and acronyms. With one input shrinking and the other growing only slowly, nescience plummets and stays near the floor, apart from brief spikes that map neatly onto external flashpoints (the 2018 gene-edited-babies scandal, for example).

The overall downward regression slopes for both inaccuracy and nescience confirm a steady convergence toward consensus, while the upward slope in redundancy simply records the article's swelling, template-rich structure. By 2020 the CRISPR page looks much like that of graphene in its mature phase: highly compressible, rarely contested, and updated incrementally rather than rewritten from scratch each time the field advances.

The deep-learning article (see 9.15) traces a pattern typical of a field that shifted from niche research to global headline status almost overnight. In its early years (2011-2014) the page was compact and still being drafted: redundancy climbed from  $\approx 0.40$  to the low 0.70 as basic definitions, algorithm lists and seminal breakthroughs were added and often restated, while inaccuracy oscillated but stayed below 0.30, reflecting moderate talk-page traffic. The real turbulence began in 2017: the talk-page share abruptly jumped above 0.40 and stayed there for nearly five years, mirroring the community's debates over benchmark claims, ethics, and the hype surrounding "AI booms." Because redundancy had already settled in the mid-0.70s, this burst of discussion drove nescience to a sustained plateau above 0.85, marking the article as one of Wikipedia's most actively contested science pages during the height of deep-learning publicity.

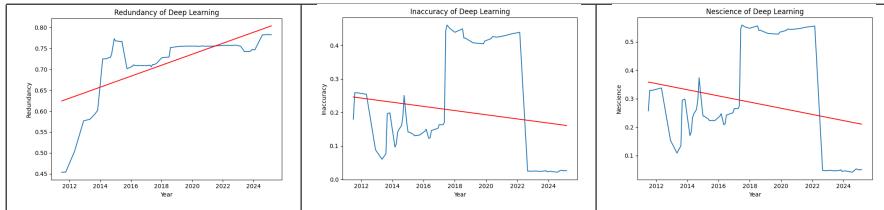


Table 9.15: Knowledge evolution of deep learning

A dramatic correction occurs in late 2022. Large chunks of deliberation were apparently archived or spun off, slashing inaccuracy to near zero almost overnight and dropping nescience with it. Since then the page has remained highly compressible ( $\text{redundancy} \approx 0.75 - 0.78$ ) but largely uncontested; only small upticks appear as new foundation-model milestones are folded in. The negative regression slopes for inaccuracy and nescience therefore chart a long-run movement toward consensus, even though the five-year plateau of high nescience serves as a reminder that Wikipedia's knowledge gap can stay wide for a protracted period when a discipline's methods, jargon and social implications are evolving faster than editors can lock down a stable narrative.

Across three emblematic case studies (graphene, CRISPR and deep learning) the time-series confirm that Wikipedia's nescience reliably chronicles the lifecycle of a scientific topic: initial growth brings a surge of redundancy as material is layered on, and a spike of inaccuracy as editors negotiate scope and sourcing; once consensus forms, talk-page activity collapses, redundancy stabilises in the high-0.70s, and nescience falls toward zero. Graphene reached this mature phase around 2011, CRISPR around 2016, while deep learning lingered in a high-gap state from 2017 to 2022 before a mass archiving of debate produced an abrupt convergence. The trajectory is thus quantifiable: rising redundancy without rising inaccuracy signals steady accretion of detail, whereas persistent high nescience marks periods when the science itself, or its social framing, is still in flux.

## 9.5 The Demarcation Problem

In this section, we propose a practical method, based on the theory of nescience, to address the demarcation problem, specifically, the challenge of distinguishing scientific from non-scientific knowledge in real-world contexts. Although demarcation is a longstanding philosophical issue, our aim is not to conclusively solve this complex problem but rather to provide insights into its nature and outline potential paths toward future solutions through practical and operational methods.

For our experiments and analysis, we have selected six scientific topics and six pseudoscientific topics to evaluate our approach to the demarcation problem. Our analysis is based on the descriptions of these topics provided by Wikipedia and their associated Talk pages. As in previous analyses conducted in this chapter, we have preprocessed the Wikipedia pages by using the `wikitextparser` Python library to remove Wikimedia tags and other irrelevant elements such as tables and images.

The six scientific topics selected are: Climate Change (the long-term alteration of temperature and weather patterns), Graphene (a single-layer carbon material with extraordinary physical properties), Dark Matter (a hypothesized form of matter making up approximately 85% of the universe), Deep Learning (a subset of machine learning using neural networks with multiple layers), Lithium-ion Battery (a type of rechargeable battery widely used for portable electronics and electric vehicles), and Brain-Computer Interface (a technology enabling direct communication between the brain and external devices). These scientific topics have been selected due to their intensive research activity over the past 20 years.

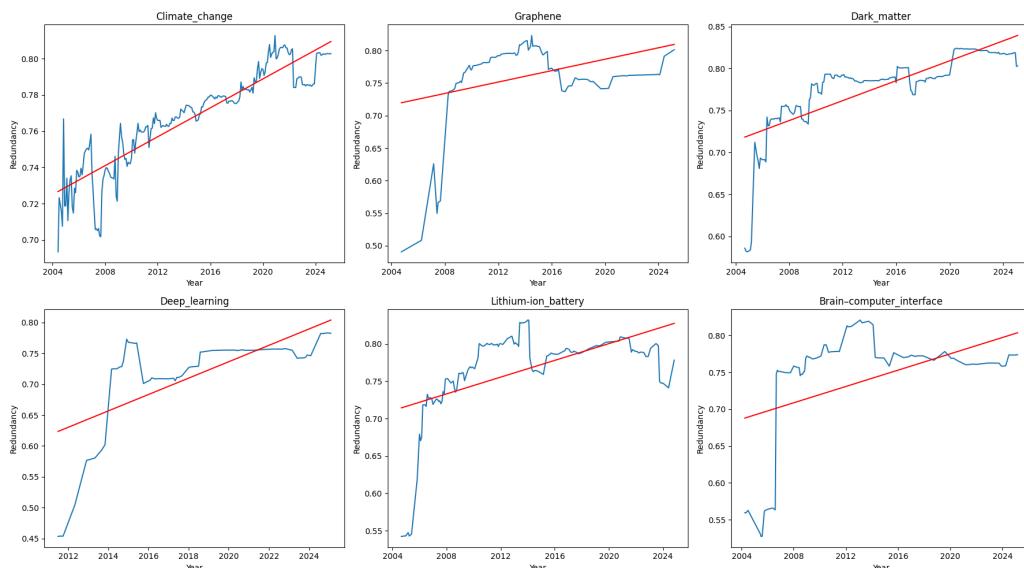


Figure 9.5: Evolution of Redundancy in Scientific Topics

Redundancy, as an approximation of the concept of surfeit, is computed, as in the previous case, by comparing the ratio of the length of a text to its compressed version. Figure 9.5 shows the evolution of redundancy for these selected scientific topics over the past 20 years. As observed, redundancy for

these topics demonstrates an increasing trend, as confirmed by the computed regression line. Although one might generally expect redundancy to decrease as our understanding improves, new discoveries and emerging knowledge frequently necessitate additional details and explanations, thus increasing redundancy in descriptions.

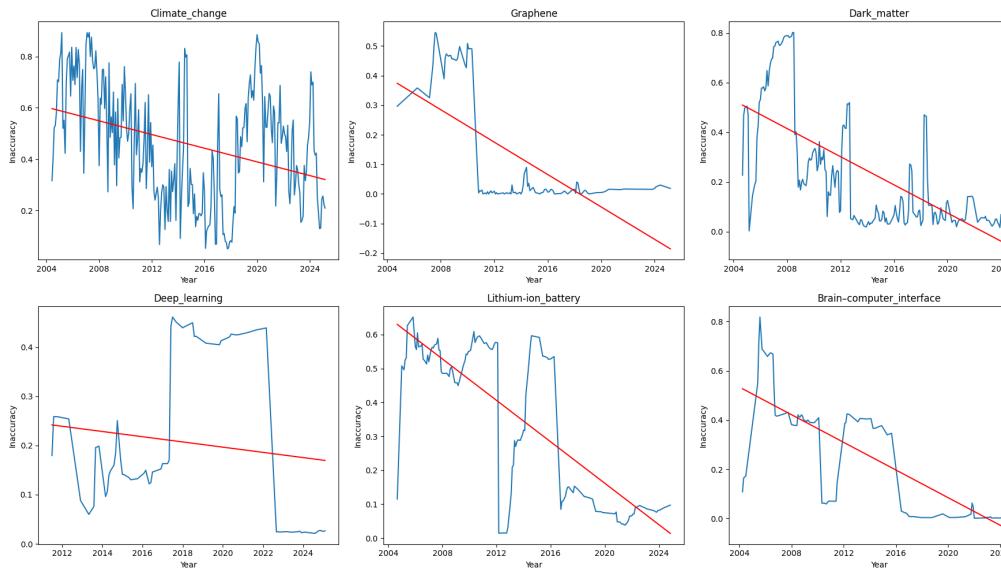


Figure 9.6: Evolution of Inaccuracy in Scientific Topics

Inaccuracy is computed based on the ratio  $\text{length\_talk} / (\text{length\_talk} + \text{length\_article})$ , where `length_talk` represents the length of the Talk page for each topic, and `length_article` is the length of the corresponding Wikipedia article, as done in the previous sections. Utilizing Talk pages is an effective method for approximating article inaccuracy since these pages typically contain discussions, disputes, and clarifications regarding inaccuracies or controversies in the articles. Figure 9.6 illustrates the evolution of inaccuracies for the selected scientific topics, showing a clear decreasing trend across all topics.

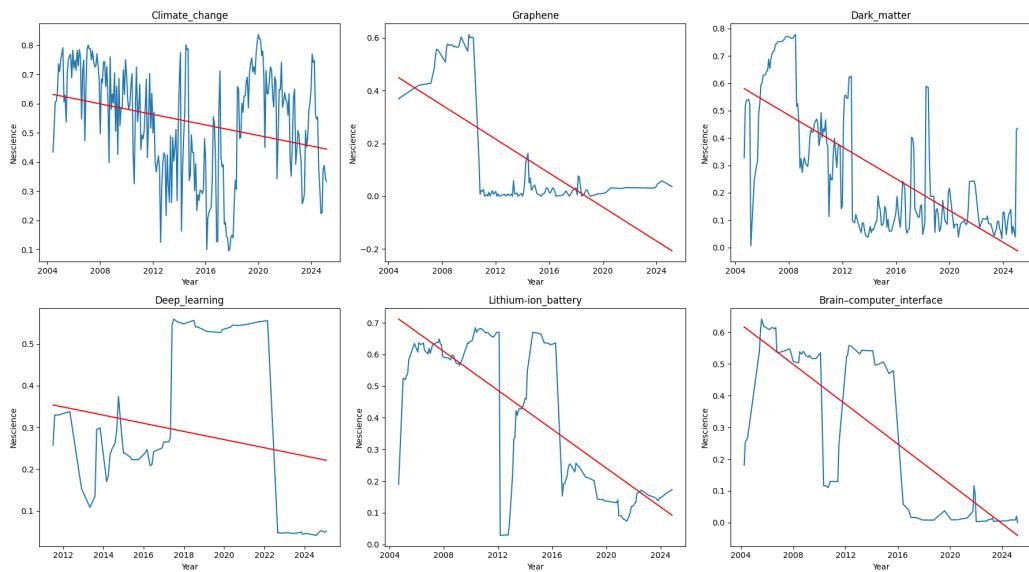


Figure 9.7: Evolution of Nescience in Scientific Topics

Finally, Figure 9.7 shows the evolution of nescience for all selected scientific topics, where nescience is estimated as the harmonic mean of the metrics of redundancy and inaccuracy. As depicted, despite the positive trend in redundancy, nescience exhibits a decreasing trend, suggesting our overall understanding of these topics improves over time.

Additionally, we have selected six pseudoscientific topics to further evaluate our demarcation method: Lunar Effect (the belief that lunar cycles influence human behavior), Water Memory (the claim that water retains a memory of substances previously dissolved in it), Astral Projection (the claimed ability of consciousness to leave the physical body and travel in the astral plane), Enneagram of Personality (a model describing personality types based on a geometric figure with nine interconnected points), Perpetual Motion (the hypothetical concept of a machine that operates indefinitely without energy input), and Dowsing (a technique claiming the ability to locate water, minerals, or other hidden substances through intuitive means). These topics represent different pseudoscientific categories—Lunar Effect (Astrology), Water Memory (Homeopathy), Astral Projection (Parapsychology), Enneagram of Personality (Numerology), Perpetual Motion (Physics-related pseudoscience), and Dowsing (Divination)—and were selected based on classifications from Wikipedia itself as pseudoscience.

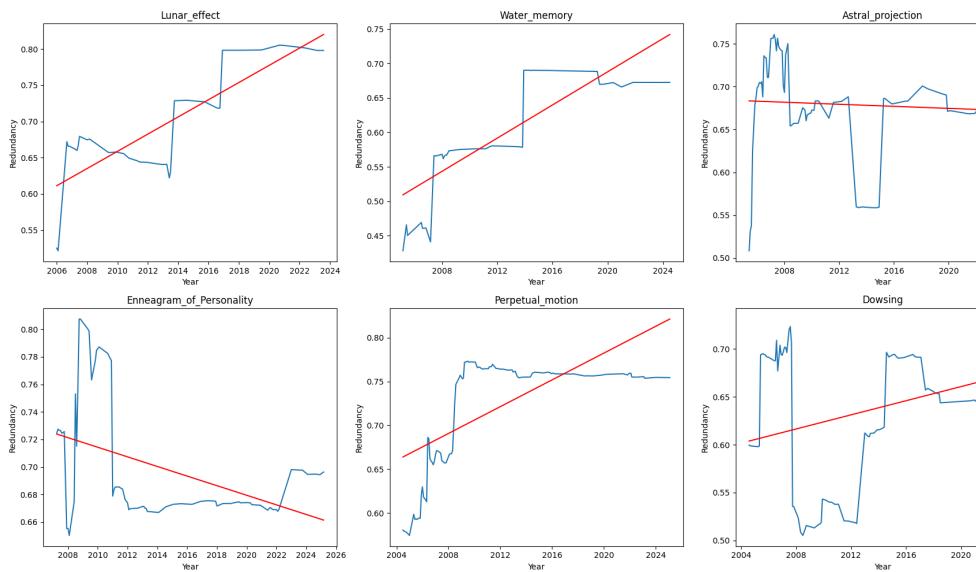


Figure 9.8: Evolution of Redundancy in Pseudoscientific Topics

Figure 9.8 shows the evolution of redundancy for these selected pseudoscientific topics over the past 20 years. As it was the case of scientific topics, redundancy for these topics demonstrates an increasing trend for "Lunar Effect", "Water Memory", "Perpetual Motion", and "Dowsing", and a rather surprising non-increasing trend for "Astral Projection" and "Enneagram of Personality". This decrease in redundancy for the latter two topics may indicate that their Wikipedia articles have undergone substantial editing aimed at streamlining or simplifying the content.

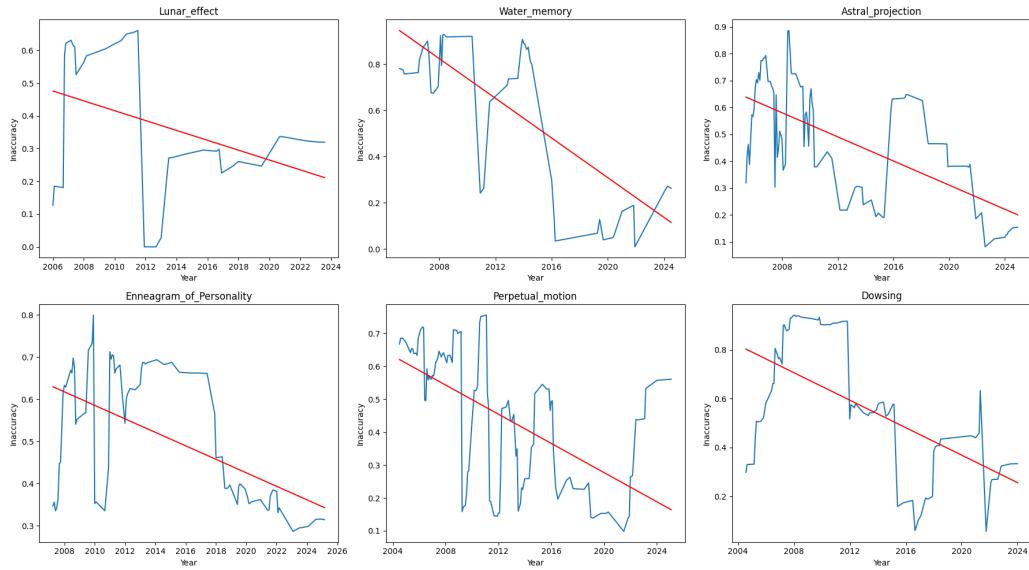


Figure 9.9: Evolution of Inaccuracy in Pseudoscientific Topics

Figure 9.9 illustrates the evolution of the estimated inaccuracies for the same pseudoscientific topics. A clear decreasing trend in inaccuracies is observable across most topics, suggesting, somewhat unexpectedly, that controversies regarding these topics have started to settle down.

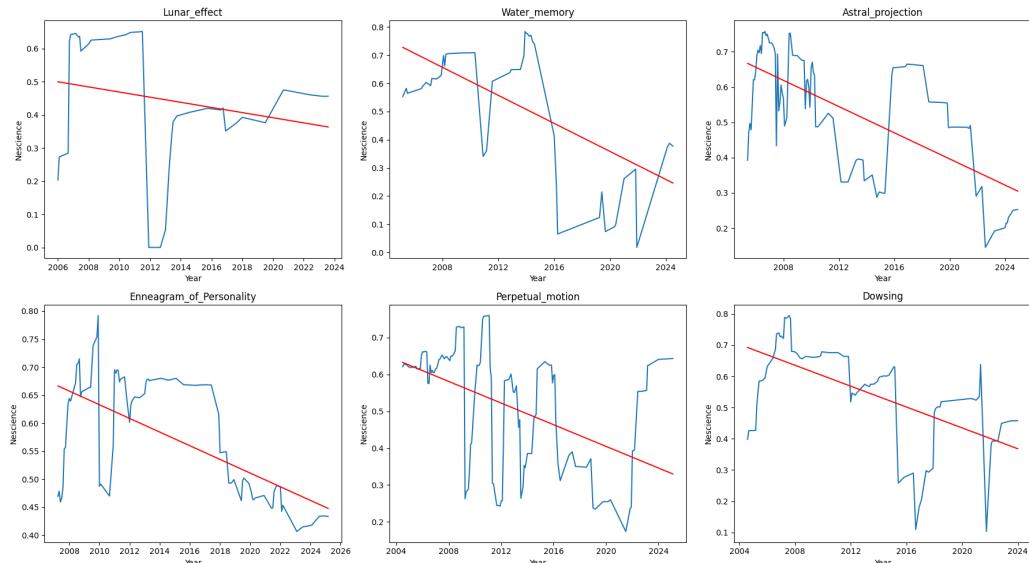


Figure 9.10: Evolution of Nescience in Pseudoscientific Topics

Finally, Figure 9.10 shows the evolution of the estimated nescience for all selected pseudoscientific topics. As expected, given the previous figures for redundancy and inaccuracy, a decreasing trend is observed for most topics, suggesting that our knowledge about these topics has, in fact, decreased with time.

Our preliminary analysis is too superficial to draw definitive conclusions. A more detailed analysis, employing randomized controlled experiments and rigorous hypothesis testing, needs to be conducted. Additionally, the approximations for redundancy and inaccuracy could be refined further (e.g., citation analysis, scientific community surveys, bibliometric analysis). Based on our initial findings, there appear to be no significant practical differences between science and pseudoscience—both communities seem capable of increasing our knowledge about their respective topics. The essential distinction between science and pseudoscience may lie in the validity and truthfulness of claims, rather than in their methodologies. Alternatively, demarcation could also be based on science’s demonstrated capability to transform theoretical results into practical, real-world applications—a capability generally lacking in pseudoscience.

## 9.6 Perfect knowledge

Philosophers of science deal with the problem of how knowledge about our world is gathered through our senses, and if we can trust our perceptions. Also, they address the difficult issue of how knowledge is derived from facts (for example, by means of applying the principle of induction), and if it is sound, from a logical point of view, to make those derivations. Finally, philosophers are interested in how scientific theories are generated based in this knowledge. Any of these problems is covered by the theory of nescience, since we assume that theories (or descriptions in our own terminology) are already known. We do not provide any method to create those theories. What the theory of nescience provides is a set of metrics to quantitatively evaluate, and compare, existing scientific theories.

It might appear that the descriptions in which the theory of nescience is based are truly objective, in the sense that they must be so clear and well stated that even a computer can reconstruct the original topic given its description. Although this point is true, the problem that prevents the theory to provide an absolute knowledge about our world is the way we choose the entities to study, and how we encode as strings those entities. As we have seen (see Chapter 2), the accuracy of our descriptions depend on how good is our encoding of the abstract entities we are studying. Unless the entities are strings themselves, we must assume that our encoding could not be perfect.

Moreover, we could be wrong about our assumption that the selected set of entities covers all possible entities of that kind. That is, the set of entities are subject to change as our scientific understanding about them develops. **The same might happen in case of encodings.**

Although the theory of nescience does not say anything about how we can reach an absolute knowledge about an entity, it can tell us if we have reached a perfect description (that we make equal to a perfect knowledge). That is, the theory of nescience can answer the question if we have reached a perfect knowledge about a topic, subject that the entity under study has been properly identified, and the encoding of this entity has no errors.

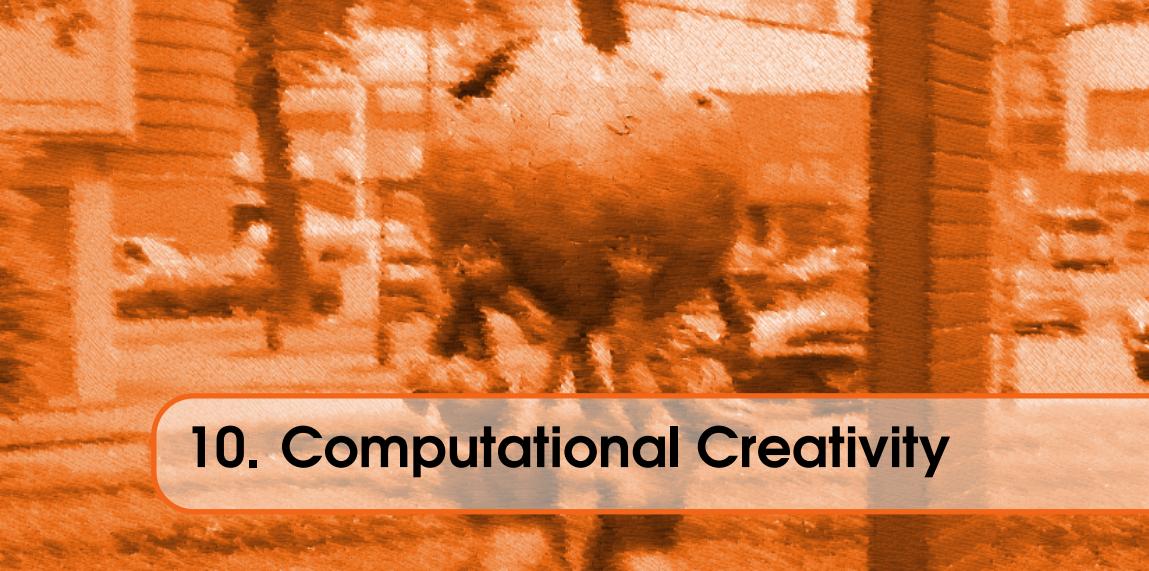
## **9.7 Unknown-unknown**

### **References**

Provide a reference to Wikipedia

The behavior of compressors depending of the size of objects and window (buffer) size is studied in detail in [CAO+05] with applications to the normalized compression distance (a measure of similarity between objects proposed in [Li+04]).

Add a reference to the Burrows–Wheeler algorithm and to bzip2, and gzip. Perhaps a comparison, or more bib in NCD in practice.



## 10. Computational Creativity

*To be surprised, to wonder,  
is to begin to understand.*

José Ortega y Gasset

In this Chapter we are going to see how to apply in practice our methodology for the assisted discovery of interesting research questions. As it was the case of previous chapter, in which we studied the concept of nescience from a practical point of view, ...

In the first part of this chapter we will see how to approximate the new metrics introduced: relevance and applicability. The relevance of a topic will be based on the number of web pages on Internet that link to the topic's page on Wikipedia (external links), and applicability will be estimated by the number of links from the Wikipedia's scientific pages to themselves (internal links). We will provide some practical examples of both quantities for the set of topics that compose the research area of theoretical computer science. Then, we will describe how to apply in practice our methodology for the discovery of interesting questions, and we will come up with some examples of new research questions that, in principle, could be addressed by science. The new questions proposed will be both, intradisciplinary, coming from the area of theoretical computer science, and interdisciplinary,

by means of combining the area of theoretical computer science with the area of philosophy and the area of biochemistry. Finally, we will derive some new interesting research topics, according to our subjective interpretation of the combinations found, that are enough interesting to deserve to be the subject of new research activities. We will also evaluate if the proposed topics fulfill the requirements that we proposed in Chapter 7 for a question to be classified as interesting.

In the last part of the chapter, we will apply the set of metrics defined for the classification of individual research topics to full research areas. In this way, we will compute the interestingness of the different research disciplines as source of new problems, and their interestingness as a source of useful tools to solve open problems. These metrics will allow us to compare the relative merits of different knowledge disciplines. Some examples of research areas in decay will be shown as well.

## 10.1 Maturity

The maturity of a topic is estimated based on the length of the Wikipedia article (only the text), and the length of the compressed version. Figure 10.1 shows a plot of the maturity of the selected set of topics after the normalization process.

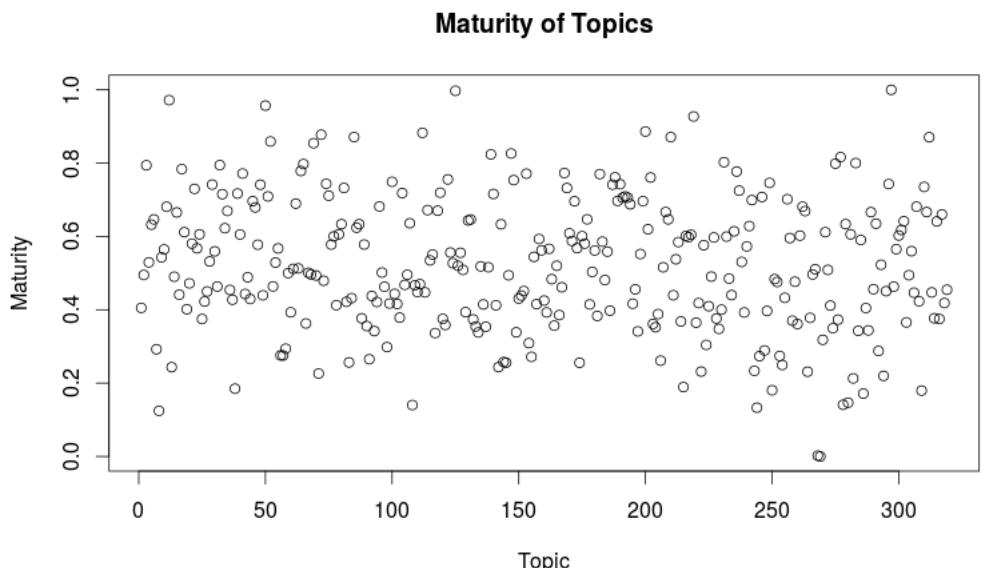


Figure 10.1: Maturity of topics

| Topic                   | Maturity | Norm. |
|-------------------------|----------|-------|
| Carry operator          | 5.34     | 1.00  |
| Binade                  | 4.54     | 0.99  |
| Comm. X-Machine         | 3.01     | 0.97  |
| PowerDEVS               | 2.35     | 0.94  |
| MPIR                    | 2.00     | 0.92  |
| Constraint automaton    | 1.84     | 0.90  |
| RO right moving TM      | 1.73     | 0.89  |
| P”                      | 1.71     | 0.89  |
| Crossing sequence (TM)  | 1.63     | 0.88  |
| Microsoft Binary Format | 1.53     | 0.86  |

Table 10.1: Maturity of topics

Table 10.1 contains the ten most relevant topics according to its maturity. For each topic it is shown the maturity and the normalized version of this number. Well classified topics, that is, topics that our intuition tell us that are well understood, could include *Read-only right moving Turing machines*, *Crossing sequence (Turing machines)*, and perhaps the *P” language*. Other topics that perhaps are misclassified include *communication X-Machine*, *Power DEVS*, *MIPR*, and *constraint automaton*.

### 10.1.1 Applicability

Applicability measures how likely is that a research topic can be applied to solve open problems. If a tool has been already applied to solve multiple problems, then there is a high probability that it can be used again to solve new problems. The number of problems in which a tool has been applied is computed with the aid of the applicability graph (see Definiton 7.4.1), and applicability is formally defined as the out-degree of the topic in this graph (see Definiton 7.4.2). We have approximated the applicability graph by means of using the graph of internal links between the scientific pages of Wikipedia. That is, we approximate the applicability of a topic by counting the number of pages from Wikipedia domain that links to the topic’s page (we have used the “*What links here*” facility from Wikipedia, a tool to see the list of the pages that link to, but not redirect to, the current page).

TODO: Perhaps we could include a nice picture of the graph of Wikipedia internal links

The applicability of a topic Figure 10.2 shows an histogram of the applicability of the selected set of topics. The histogram’s shape is dominated by an enormous spike in the very first bin and a sparsely populated, elongated

right-hand tail. Roughly three-quarters of the theoretical-CS topics attract fewer than 50 incoming Wikipedia links, while only a sprinkling reach the triple-digit range and just one or two break past 300. That steep drop-off is the signature of a power-law (or at least heavy-tailed) distribution: applicability, as measured here, is concentrated in a tiny set of “super-connectors,” with the median topic enjoying only modest reuse. In practical terms, the average link count is a misleadingly rosy figure—pulled upward by a few giants—whereas the typical topic remains niche.

Such inequality has several implications. First, it reinforces the idea that a small core of foundational ideas underpins a large share of problem-solving across the field; investing effort in those hubs yields the greatest leverage. Second, the emptiness of the mid-range hints that rising topics face a kind of applicability “valley of death”: they must clear a substantial gap before joining the elite club of widely referenced concepts. Finally, the long tail highlights opportunity—numerous specialised notions wait at low link counts, potentially poised for breakout if new cross-disciplinary applications emerge.

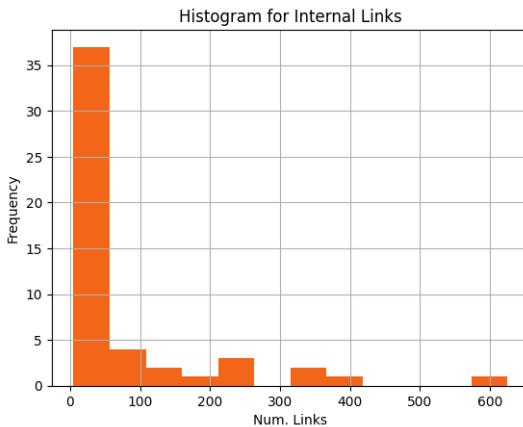


Figure 10.2: Applicability of topics

Table 10.2 contains the ten most relevant topics according to its applicability. The backlink data confirm the long-tailed picture hinted at by the histogram: applicability—at least as proxied by Wikipedia’s “What links here” counts—is highly unequal. A single super-hub (the \*\*Dirac delta function\*\*, 625 links) dwarfs the rest, while only a handful of topics even cross the 200-link mark. This means that when researchers look for broadly reusable tools, the “return on attention” is greatest in a very small set of concepts that already function as connective tissue across many areas.

| Topic                                      | Internal Links |
|--|----------------|
| Dirac delta function                       | 625            |
| Rule of inference                          | 409            |
| Reinforcement learning from human feedback | 351            |
| Euclidean algorithm                        | 343            |
| Trie                                       | 244            |
| Telephone number (mathematics)             | 228            |
| Commutative property                       | 214            |
| Tropical cyclone forecast model            | 195            |
| Cartesian tree                             | 155            |
| Binary search                              | 116            |

Table 10.2: Applicability of topics

Equally striking is \*who\* those hubs are. Alongside classic algorithmic staples such as the \*\*Euclidean algorithm\*\*, \*\*Trie\*\*, and \*\*Binary search\*\*, we see foundational math/logic notions (e.g., \*\*Rule of inference\*\*), \*\*Commutative property\*\*), a fast-rising AI methodology (\*\*Reinforcement learning from human feedback\*\*), and even a domain-specific meteorology model. In other words, high applicability favors breadth rather than disciplinary purity: ideas that spill into multiple conversations—whether introductory, theoretical, or applied—accumulate the most links. For anyone mapping future research bets, this suggests monitoring backlink growth over time (to catch newcomers like RLHF early) and normalizing counts by sub-field size to filter out topics whose popularity is driven by narrow, self-referential clusters.

### 10.1.2 Relevance

In Definition 7.3.2 we introduced the concept of relevance of a research topic as a measure of the impact that this topic could have in people's life. The idea was that the higher the relevance, the higher its potential as a source of interesting questions, since we would be addressing a problem that affects many people. Relevance was defined as the degree of the research topic in the relevance graph, a bipartite graph connecting topics and people (see Definition 7.3.1). Of course, this relevance graph is a mathematical abstraction that it is very difficult to compute in practice, since we do not have information about how people is affected by each topic.

As an approximation of the relevance of a topic we have used the number of links (URLs) from external web pages on the whole Internet that point to the topic's web page on Wikipedia. The rationale is that the more relevant

is a topic, the more people will be talking about it on Internet, and the more URLs there will be linking to Wikipedia, since Wikipedia is a well known source of information to which many people refer. In fact, we are not interested in knowing the absolute relevance of research topics, since what we need is a measure of relative relevance between different topics. An underestimate of the relevance is not harmful as long this underestimation is equally distributed among all the topics. It requires further research to fully understand how well the theoretical concept of relevance is approximated by this URLs counting procedure.

Another problem is that we do not know the number of links to a web page on Internet, and so, we have to apply a second approximation. In this case, the number of links was estimated using Google's link: facility in searches. Google's link: lists the links that Googlebot discovered during its crawling and indexing process of Internet. For example, the number of external pages that link to the *Computer Science* article in Wikipedia is given by:

```
link:http://en.wikipedia.org/wiki/Computer_science
```

As Google recognizes in its web page, not all links to a site may be listed. The number of links could vary due to redirections, crawling errors, and other problems. How these errors affect to the accuracy of the links counting is not clear, since the details of the Google's crawling algorithm are not public.

Figure 10.3 shows a plot of the relevance of the set of topics after the application of the normalization process described in XXX. A practical problem is that there are too many topics with a very low number of links (as indexed by Google). That should not be a problem since we are not using at any moment those topics with very low relevance.

Table 10.3 contains the ten most relevant topics according to its relevance. For each topic it is shown its relevance (number of external links) and the normalized version of this number. The list includes basic concepts (*binary number*, *floating point*), advanced concepts (*Turing machine*, *finite-state machine*, *cellular automaton*, *lambda calculus*, *Turing completeness*), highly popular tools (*recursion*, *regular expression*) and classical problems (*halting problem*). All those topics could fit into our intuitive idea of highly relevant, although some authors could perfectly disagree that some of them are the most relevant ones in the area of theory of computation (for example, *floating point*).

## 10.2 Interesting Research Questions

Before to compute the new interesting questions, it is highly convenient to normalize the metrics of the topics involved in the study, otherwise, a very

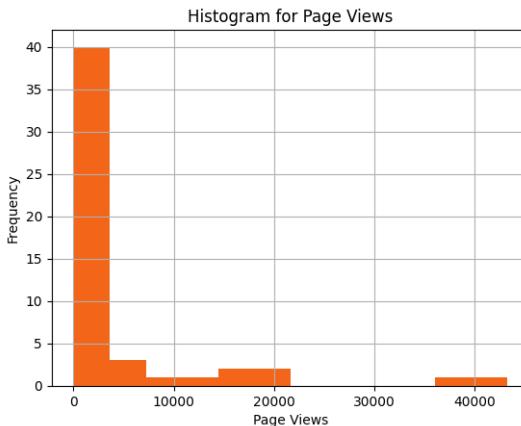


Figure 10.3: Relevance of topics

| Topic                | Relevance | Norm. |
|----------------------|-----------|-------|
| Regular expression   | 409       | 1.00  |
| Turing machine       | 159       | 0.90  |
| Binary number        | 141       | 0.89  |
| Recursion            | 133       | 0.88  |
| Finite-state machine | 118       | 0.86  |
| Halting problem      | 108       | 0.85  |
| Cellular automaton   | 104       | 0.85  |
| Floating point       | 99        | 0.84  |
| Lambda calculus      | 95        | 0.84  |
| Turing completeness  | 93        | 0.83  |

Table 10.3: Relevance of topics

reduced set of topics could dominate all the questions. For the normalization process we have used the BoxCox method, that it is based on the identification of the best transformation from a family of power transformations.

### TODO: Explain the BoxCox method

The most difficult part of the identification of topics as tools, that is, topics with very high maturity (or very low nescience), is to distinguish when the description of a topic is short because it is well understood (for example, a mathematical theorem), or when it is short because it is a unfinished or poorly written article. Our work is based on the classification of Wikipedia articles as stubs, however, this classification is not very reliable, since many stubs articles are not classified as such (many of the misclassified topics suffer from this problem). How to automatically distinguish between a well-understood

| Tool                   | Problem                | Interestingness |
|------------------------|------------------------|-----------------|
| Ternary numeral system | Regular expression     | 1.21            |
| GNU MPAL               | Arithmetical hierarchy | 1.19            |
| IEEE 854-1987          | Arithmetical hierarchy | 1.17            |
| Quantum computer       | Regular expression     | 1.17            |
| Ternary numeral system | Arithmetical hierarchy | 1.15            |
| Division by zero       | Regular expression     | 1.15            |
| Turing machine         | Regular expression     | 1.14            |
| GNU MPAL               | Regular expression     | 1.13            |
| Ternary numeral system | Halting problem        | 1.13            |
| Recursion              | Halting_problem        | 1.13            |

Table 10.4: Interesting Intradisciplinary Questions

topic and a poorly written description is still an open question.

By combining the elements of Table 10.8 and Table 10.9 we could come up with new interesting ideas of how to apply existing tools to open problems. As it was said above, the goal of the approach described in this article is to identify highly potential interesting applications, but is up to the researcher to decide if certain combination of topics make sense or not, and if they deserve the effort to pursue them. The results of the combination is in Table 10.4.

Most of the interesting questions identified have very low quality. As it was said before, the problem is that it is very difficult to distinguish (automatically and unsupervised) between a poorly written article from a very well understood topic. In this section we review some on the interesting intradisciplinary questions identified with the aim to clarify what we mean as interesting question and how interesting questions should be interpreted. Some combinations worth examining could be:

- Interesting Question 7: *Can we apply* Turing machines *to* regular expressions? The answer to this question is yes, since it is a very well known question. Regular expressions are recognized by finite automata, and finite automata can be simulated by Turing machines.
- Interesting Question 10: *Can we apply* recursion *to the* halting problem? Again the answer is yes, since the proof of the halting problem is based on a machine that calls itself.

Note that both questions have well known answers, and so, we have failed to provide original questions.

The most interesting questions arise when we combine topics from two different disciplines. However, the probability that the identified questions are meaningful is lower than in the case of intradisciplinary analysis.

| Tool                   | Problem            | Interestingness |
|------------------------|--------------------|-----------------|
| State space            | Action potential   | 1.17            |
| Turing machine         | Action potential   | 1.16            |
| Quantum computer       | Action potential   | 1.16            |
| Abstract machine       | Action potential   | 1.14            |
| Computational model    | Action potential   | 1.13            |
| State space            | Membrane potential | 1.12            |
| State space            | Meiosis            | 1.11            |
| Arithmetic logic unit  | Meiosis            | 1.11            |
| GNU MPAL               | Flashbulb memory   | 1.11            |
| Ternary numeral system | Working memory     | 1.10            |

Table 10.5: Interesting Interdisciplinary Questions

For the interdisciplinary analysis we have used the collection of pages from the theory of computation already used in the intradisciplinary analysis, and a new collection of topics from the area of bioinformatics. The topics were selected using the Wikipedia category *natural sciences, biology, biological processes*. In total, there were more than  $10^5$  combinations analyzed. Table 10.5 contains the list of the most relevant intradisciplinary applications.

The set of interdisciplinary questions also suffer from the problem of the stub articles, and so, the quality of the results is low. Some interdisciplinary applications could be:

- Interesting question 1: *Can we apply state space to action potential?* Questions 1, 2, 3, 4 and 5, all of them, suggest the same idea, that is, if it is possible to formalize the concept of action potential, in such a way that can be reproduced by a computer.
- Interesting question 7: *Can we apply state space to meiosis?* Question 7 is similar to question 1, and it asks about the possibility of formalize the concept of meiosis using a computer.

## 10.2.1 Intradisciplinary Questions

The interest of a topic as a tool measures how likely is that this tool can be applied to other problems. Figure 10.4 shows a plot of the interestingness of the selected set of topics after the normalization process.

Table 10.6 shows the average applicability and average maturity of each of the selected areas, and the average interestingness of each area as a source of interesting tools. The table largely fits our intuitive idea of which areas are more important as a source of tools: computer science is the area of highest

### Interestingness as Tools

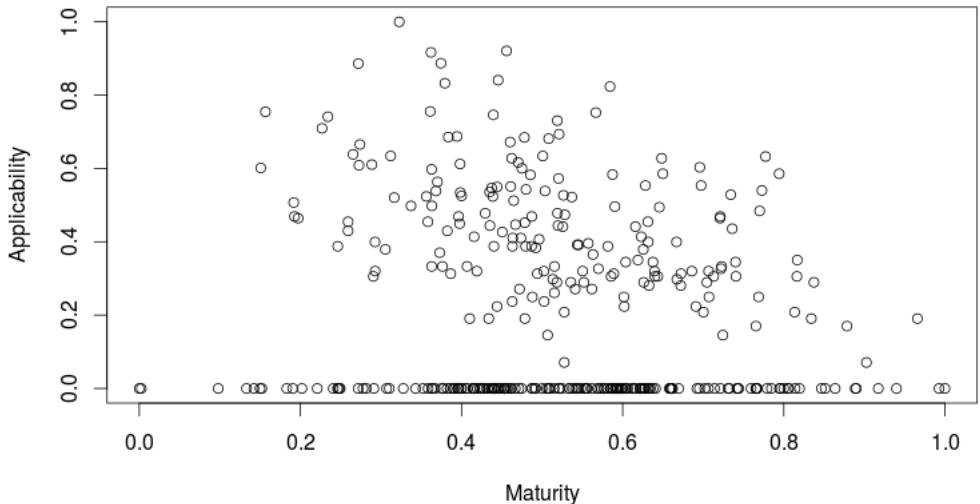


Figure 10.4: Interestingness of Tools

interest, and sociology is the area with less interest. The only strange elements is that epistemology appears as a source of very interesting tools, even more interesting, on average, than topics from mathematics (probably because it contains a large ratio of poorly written articles).

Finally, Table 10.7 shows the relevance and nescience of the selected areas, and their interest as a source of interesting problems. Again, the results largely match our intuitive idea of which areas are less understood: sociology is the area with the highest number of interesting problems, and mathematics is the area with the lower number of problems.

Table 10.8 contains the ten most relevant topics according to its interestingness as a source of interesting tools. Out of the ten topics, only two (*ternary numeral system* and *recursion*) appear in the list of top ten mature topics or top ten applicable topics; the rest of topics are new. In the list we can find topics like the *GNU Multiple Precision Arithmetic Library* and the *standard for radix-independent floating-point arithmetic* (IEEE 854-1987) that are definitely tools, but not in the sense of tool that we are looking for our methodology. Some other topics are not clear that can be considered as tools, like *arithmetic logic unit*, *barrel shifter*, or *arithmetic overflow*. Topics that match or intuitive idea of tool include *recursion*, *state space*, *abstract machine*, and *ternary numeral system*. There are also some topics,

| Research Area    | Applicability         | Maturity              | Tools                 |
|------------------|-----------------------|-----------------------|-----------------------|
| Sociology        | $1.00 \times 10^{-3}$ | $2.93 \times 10^{-3}$ | $3.09 \times 10^{-3}$ |
| Biology          | $9.20 \times 10^{-4}$ | $4.65 \times 10^{-3}$ | $4.74 \times 10^{-3}$ |
| Chemistry        | $3.11 \times 10^{-3}$ | $5.01 \times 10^{-3}$ | $5.90 \times 10^{-3}$ |
| Psychology       | $1.14 \times 10^{-3}$ | $6.91 \times 10^{-3}$ | $7.00 \times 10^{-3}$ |
| Mathematics      | $9.32 \times 10^{-3}$ | $9.47 \times 10^{-3}$ | $1.32 \times 10^{-2}$ |
| Epistemology     | $1.55 \times 10^{-3}$ | $1.75 \times 10^{-2}$ | $1.76 \times 10^{-2}$ |
| Computer_science | $9.93 \times 10^{-3}$ | $1.90 \times 10^{-2}$ | $2.15 \times 10^{-2}$ |

Table 10.6: Interestingness of Areas as Tools

|                  | Relevance             | Nescience             | Problems              |
|------------------|-----------------------|-----------------------|-----------------------|
| Mathematics      | $4.22 \times 10^{-2}$ | $3.51 \times 10^{-1}$ | $3.53 \times 10^{-1}$ |
| Computer_science | $2.35 \times 10^{-2}$ | $4.43 \times 10^{-1}$ | $4.44 \times 10^{-1}$ |
| Chemistry        | $5.95 \times 10^{-2}$ | $4.66 \times 10^{-1}$ | $4.70 \times 10^{-1}$ |
| Biology          | $3.85 \times 10^{-2}$ | $4.75 \times 10^{-1}$ | $4.77 \times 10^{-1}$ |
| Psychology       | $5.06 \times 10^{-2}$ | $5.28 \times 10^{-1}$ | $5.31 \times 10^{-1}$ |
| Epistemology     | $4.54 \times 10^{-2}$ | $5.30 \times 10^{-1}$ | $5.32 \times 10^{-1}$ |
| Sociology        | $4.21 \times 10^{-2}$ | $5.43 \times 10^{-1}$ | $5.44 \times 10^{-1}$ |

Table 10.7: Interestingness of Areas as Problems

like *computational model*, too broad to be considered in a question.

Finally, Figure 10.5 contains a plot of the interestingness of the topics considered as potential interesting problems.

Table 10.9 shows the ten most interesting topics as interesting problems. Topics that fit our intuitive idea of problem, that is, not very well understood concepts with a high relevance, could include *arithmetical theory*, *halting problem*, *floating point*, *quantum computer*, and *computable function*. The topic *recursion* appears both as a tool and as a problem. However in case of tools it refers to the concept of recursion in general, and in case of problems it refers to the implementation of the concept of recursion in the particular case of computer science. The case of *regular expression*, a topic that intuitively should be classified as a tool and not as a problem, can be explained due to the length of the article in Wikipedia, that provides a detailed description of the language used for regular expressions. This problem rises the question of how to distinguish in Wikipedia between introductory articles and reference articles. Finally, there are topics like *computability theory*, *lambda calculus* and *computability* that are too broad to be analyzed as problems.



Based on the given definition of "Interestingness of a topic as a tool",

| Topic                  | Interestingness |
|------------------------|-----------------|
| GNU MPAL               | 0.49            |
| Ternary numeral system | 0.48            |
| IEEE 854-1987          | 0.47            |
| Arithmetic logic unit  | 0.43            |
| Recursion              | 0.42            |
| Barrel shifter         | 0.42            |
| State space            | 0.42            |
| Abstract machine       | 0.41            |
| Computational model    | 0.39            |
| Arithmetic overflow    | 0.39            |

Table 10.8: Interestingness of Tools

| Topic                  | Interestingness |
|------------------------|-----------------|
| Arithmetical hierarchy | 0.72            |
| Regular expression     | 0.68            |
| Computability theory   | 0.65            |
| Halting problem        | 0.65            |
| Recursion (CS)         | 0.64            |
| Lambda calculus        | 0.63            |
| Floating point         | 0.61            |
| Quantum computer       | 0.57            |
| Computability          | 0.55            |
| Computable function    | 0.55            |

Table 10.9: Interestingness of Problems

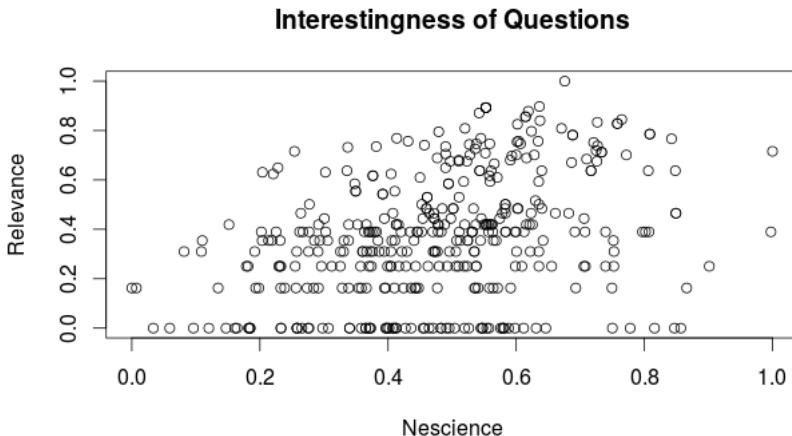


Figure 10.5: Interestingness of Questions

we can explore various mathematical properties and concepts derived from this idea. Some possibilities include:

- Normalization: To compare different topics fairly, we can normalize their interestingness values by scaling them to a specific range, e.g., [0, 1]. This normalization can help compare the relative interestingness of various topics.
- Weighted Interestingness: We can introduce weights to the maturity and applicability dimensions to emphasize one over the other depending on the specific context or application. This would allow us to fine-tune the interestingness measure for different scenarios.
- Correlation: Studying the correlation between maturity and applicability could provide insights into how these dimensions are related and possibly reveal trends across different research topics.
- Cluster Analysis: By examining topics in the two-dimensional vector space defined by maturity and applicability, we can perform cluster analysis to identify groups of topics with similar levels of interestingness. This can help identify areas of research that share characteristics and possibly suggest interdisciplinary research opportunities.
- Rate of Change: Investigating the rate of change of interestingness over time can provide insights into the evolving landscape of a research field. This analysis could reveal emerging topics or those that are becoming less relevant.
- Optimization: Using the interestingness metric, we can explore optimization techniques to find the most interesting topics given certain constraints or within specific domains. This could be useful for research prioritization and resource allocation.

These derived mathematical properties and concepts can provide a deeper understanding of the interestingness of research topics and their potential application as tools for solving problems.

### 10.2.2 Interdisciplinary Questions

## 10.3 New Research Topics

If we combine the list of highly relevant and not very well understood problems with themselves, it might happen that we come up with a new topic that lies in the new unknown unknown area.

In Figure 10.6 it is shown a plot<sup>1</sup> of the interestingness of the (potential) new topics compared with the interestingness of the topics that generated them.

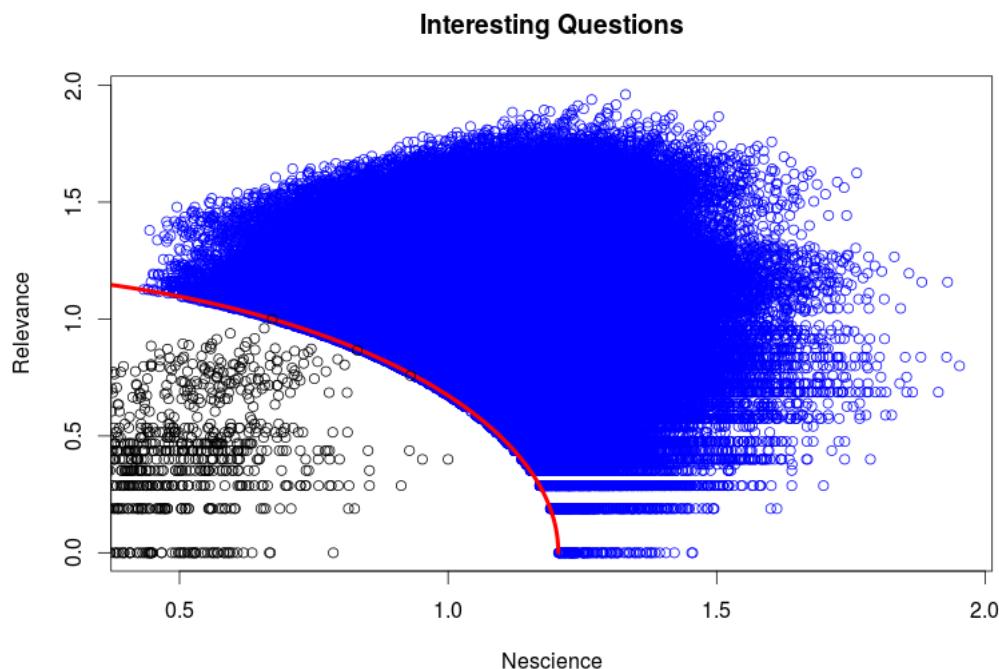


Figure 10.6: Interesting Intradisciplinary Questions

Table 10.10 contains a list of the top 25 candidates to become new topics according to their interestingness. In this analysis we have included all the topics from all the knowledge areas. Most of the questions deal with

<sup>1</sup>With the aim to make the figure clear, only a reduced set of the topics is depicted.

the concept of intellectual property (*copyright, open access, public domain*, and perhaps, *wiki*), suggesting that this is an area where there are still a lot of things to discover, much more than we are aware of. Perhaps, it could be also a problem of a certain bias of Wikipedia to these, and related, topics. Further investigation is required to clarify this point.

In order to understand how new topics are generated, we have selected the following two examples:

- New topic 17: *Public domain + Earth*. This question rises the issue if the Earth should be considered as a public resource; it touches the very concept of private property. The methodology suggest that this is not a very well understood topic.

- New topic 18: *Public domain + Internet*. Raises the same issue that Question 17, but in this case restricted to Internet and its governance.

Unfortunately, in both cases we fail to provide a well defined, innovative, and previously unseen, research topic.

We could also restrict our search of new topics to a reduced number of knowledge categories. For example, in Table 10.11 contains the ten most interesting new topics corresponding to the already studied areas of *theory of computation* and a new area of *phenomenology* (from Level 2 *philosophy of mind*, and Level 1 *cognitive science*). Given the list of topics contained in the table, we could come up with, for example, the following potential new topics:

- New topic 2: *Turing machine + synesthesia*: this new topic could be about a new kind of Turing machine that incorporates synesthetic properties. These new *synesthetic Turing machines* could be defined as the union of a group of Turing machines that are linked together in such a way that when one machines read a symbol from its tape, it produces an automatic change in the state of another machine. The property of synesthesia could be also extended to the case of non-deterministic Turing machines.
- New topic 4: *Kolmogorov Complexity + Self-awareness*: This topic could be interpreted as investigating the minimum complexity required for a computer program to have the capacity of self-awareness.

## 10.4 References

Papers about the BoxCox method ...

## 10.5 Future Work

| Problem       | Problem                  | Interestingness |
|---------------|--------------------------|-----------------|
| Public domain | Open access              | 1.71            |
| Public domain | REST                     | 1.70            |
| Public domain | Wiki                     | 1.70            |
| Open access   | REST                     | 1.70            |
| Copyright     | Public domain            | 1.69            |
| Open access   | Wiki                     | 1.69            |
| Public domain | QR code                  | 1.69            |
| Copyright     | Open access              | 1.68            |
| Wiki          | REST                     | 1.68            |
| Open access   | QR code                  | 1.68            |
| Public domain | Transport Layer Security | 1.68            |
| Copyright     | REST                     | 1.68            |
| QR code       | REST                     | 1.67            |
| Open access   | Transport Layer Security | 1.67            |
| Copyright     | Wiki                     | 1.67            |
| Wiki          | QR code                  | 1.67            |
| Public domain | Earth                    | 1.67            |
| Public domain | Internet                 | 1.67            |
| REST          | Transport Layer Security | 1.66            |
| Copyright     | QR code                  | 1.66            |
| Earth         | Open access              | 1.66            |
| Internet      | Open access              | 1.66            |
| Public domain | Open source              | 1.66            |
| Public domain | Web 2.0                  | 1.66            |
| Wiki          | Transport Layer Security | 1.66            |

Table 10.10: New Topics

| Question              | Question         | Interestingness |
|-----------------------|------------------|-----------------|
| Kolmogorov complexity | Change blindness | 1.24            |
| Turing machine        | Synesthesia      | 1.23            |
| Kolmogorov complexity | Qualia           | 1.23            |
| Kolmogorov complexity | Self-awareness   | 1.22            |
| Turing machine        | Qualia           | 1.22            |
| Kolmogorov complexity | Synesthesia      | 1.21            |
| Turing completeness   | Synesthesia      | 1.20            |
| Turing machine        | Self-awareness   | 1.20            |
| Turing completeness   | Qualia           | 1.20            |
| Turing completeness   | Self-awareness   | 1.18            |

Table 10.11: Restricted New Topics



# Part 3: Mathematical Prerequisites

|           |                                   |     |
|-----------|-----------------------------------|-----|
| <b>12</b> | <b>Discrete Probability</b>       | 281 |
| 12.1      | Foundations of Probability Theory |     |
| 12.2      | Conditional Probability           |     |
| 12.3      | Random Variables                  |     |
| 12.4      | Characterizing Distributions      |     |
| 12.5      | Common Distributions              |     |
| 12.6      | Large Random Samples              |     |
| <b>13</b> | <b>Computability</b>              | 335 |
| 13.1      | Turing Machines                   |     |
| 13.2      | Universal Turing Machines         |     |
| 13.3      | Non-Computable Problems           |     |
| 13.4      | Computable Functions and Sets     |     |
| 13.5      | Oracle Turing Machine             |     |
| 13.6      | Computational Complexity          |     |
| <b>14</b> | <b>Coding</b>                     | 355 |
| 14.1      | Coding                            |     |
| 14.2      | Kraft Inequality                  |     |
| 14.3      | Optimal Codes                     |     |
| 14.4      | Entropy                           |     |
| 14.5      | Huffman Algorithm                 |     |
| <b>15</b> | <b>Complexity</b>                 | 379 |
| 15.1      | Strings Complexity                |     |
| 15.2      | Properties of Complexity          |     |
| 15.3      | Joint Kolmogorov Complexity       |     |
| 15.4      | Conditional Kolmogorov complexity |     |
| 15.5      | Information Distance              |     |
| 15.6      | Incompressibility and Randomness  |     |
| <b>16</b> | <b>Learning</b>                   | 397 |
| 16.1      | Statistical Inference             |     |
| 16.2      | Machine Learning                  |     |
| 16.3      | Minimum Message Length            |     |
| 16.4      | Minimum Description Length        |     |
| 16.5      | Multiobjective Optimization       |     |
| <b>17</b> | <b>Philosophy of Science</b>      | 437 |
| 17.1      | What is Science                   |     |
| 17.2      | What is an Entity                 |     |
| 17.3      | Observation                       |     |
| 17.4      | Scientific Representation         |     |
| 17.5      | Scientific Discovery              |     |
| 17.6      | Scientific Explanation            |     |
| 17.7      | Scientific Justification          |     |
| 17.8      | The Limits of Science             |     |



# 11. Discrete Mathematics

*Mathematics may be defined as the subject in which  
we never know what we are talking about,  
nor whether what we are saying is true.*

Bertrand Russell

The majority of mathematical concepts employed throughout this book fall within the realm of *discrete mathematics*. This field of study focuses on mathematical objects that possess distinct or individual values, as opposed to continuous ones. This book utilizes a variety of discrete objects, including integers, strings, graphs, and computer programs. A defining characteristic of discrete sets is their countability, meaning they can be enumerated with natural numbers. In contrast, we will scarcely apply continuous mathematics, such as calculus, to the theoretical formulations within the theory of nescience.

Our primary interest in discrete mathematics stems from its relevance to computers. The theory of nescience draws upon various facets of computer science, such as algorithms, coding, and string complexity. Computers function in discrete stages and process data stored in discrete memory units. We are captivated by computers due to our ambition to apply our theoretical explorations to a wide range of real-world objects. We believe computers

provide the most appropriate means of modeling our world. While pure mathematics often delves into abstract objects without concern for their representations, the theory of nescience places great importance on the representation (or encoding) of objects.

This chapter serves as a brief overview of the fundamental concepts of discrete mathematics, and as such, does not provide formal definitions or prove theorems. Discrete mathematics is a highly diverse and vast field of study. We will only focus on those components essential for understanding the theory of nescience. Certain involved theories (such as computability, information, complexity, and so forth) demand a more extensive coverage and are therefore explored in individual chapters. The References section includes a list of recommended books that delve into the topics addressed in this chapter in greater detail.

This chapter serves as a brief overview of the fundamental concepts of discrete mathematics, introducing topics such as sets, strings and languages, counting methods, matrices, and graphs. Though we do not provide formal definitions or prove theorems in this overview, these subjects offer a foundation for the theories and concepts discussed in subsequent sections. Discrete mathematics is a highly diverse and vast field of study. Our focus will be on those components essential for understanding the theory of nescience. Certain involved theories (such as computability, information, complexity, and so forth) demand a more extensive coverage and are therefore explored in individual chapters.

The References section includes a list of recommended books that delve into the topics addressed in this chapter in greater detail. By using this list, readers interested in deepening their understanding of these topics can explore each in more detail, complementing the fundamental overview presented in this chapter.

## 11.1 Sets, Relations and Functions

The sets of *natural*, *integers*, *rational*, and *real* numbers are represented by  $\mathbb{N}$ ,  $\mathbb{Z}$ ,  $\mathbb{Q}$ , and  $\mathbb{R}$  respectively, which includes the number 0. The *positive integers* are denoted by  $\mathbb{Z}^+$ , and the *positive reals* by  $\mathbb{R}^+$ , where both of these sets incorporate the number 0. Let  $A$  be a *set*. We signify that  $x$  is an *element* of  $A$  by the notation  $x \in A$ , and that  $x$  is not an element of  $A$  by  $x \notin A$ . Elements of a set can be enumerated using braces, such as  $A = \{0, 1, 2, 3\}$ , or they can be defined by conditions via the *set-builder* notation, for instance  $A = \{x \in \mathbb{N} : x < 4\}$ , with the stipulation that the *universe* of the set be clearly established.

Suppose  $A$  and  $B$  are two sets. We use the notation  $A = B$  to denote that

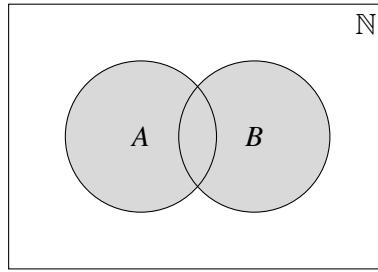


Figure 11.1: Representation of  $A \cup B$  as a Venn Diagram

the sets are *equal*. The expression  $A \subseteq B$  signifies that  $A$  is a *subset or equal* to  $B$ , and  $A \subset B$  is used to indicate that  $A$  is a *proper subset* of  $B$  (implying  $A$  is not equivalent to  $B$ ). The condition  $A = B$  holds true if, and only if, both  $A \subseteq B$  and  $B \subseteq A$  are satisfied concurrently. The symbol  $\emptyset$  represents the *empty set*, a set that contains no elements.

■ **Example 11.1** For every set  $A$  we have that  $\emptyset \subseteq A$  and  $A \subseteq A$ . ■

The term *cardinality* refers to the number of elements within a finite set  $A$ , denoted as  $d(A)$ . Accordingly, the cardinality of the empty set  $\emptyset$  is 0, as it contains no elements. For any two sets  $A$  and  $B$ , the notation  $A \cup B$  signifies the *union* of  $A$  and  $B$ , whereas  $A \cap B$  designates the *intersection* of  $A$  and  $B$ . When dealing with  $n$  sets, say  $A_1, A_2, \dots, A_n$ , their union and intersection are denoted as  $\cup_{i=1}^n A_i$  and  $\cap_{i=1}^n A_i$  respectively. For an arbitrary collection of sets indexed by  $I$ , we employ  $\cup_{i \in I} A_i$  and  $\cap_{i \in I} A_i$ . In the context of an infinite collection of sets, the notations  $\cup_i^\infty A_i$  and  $\cap_i^\infty A_i$  are adopted.

On occasion, we will resort to the use of *Venn diagrams* as a visual means to represent sets, as exemplified in Figure 11.1.

Given the sets  $A$  and  $B$ ,  $A \setminus B$  is the *set difference*, and  $A^c$  is the *complement* set of  $A$ . The *De Morgan's laws* state that for every two sets  $A$  and  $B$  we have that  $(A \cup B)^c = A^c \cap B^c$  and  $(A \cap B)^c = A^c \cup B^c$ .

Two sets  $A$  and  $B$  are *disjoint* if  $A \cap B = \emptyset$ . The sets  $A_1, A_2, \dots, A_n$  are disjoint if for every  $i$  and  $j$  such that  $i \neq j$  we have that  $A_i \cap A_j = \emptyset$ . A *partition* of a set  $A$  is a collection of nonempty disjoint subsets of  $A_1, A_2, \dots, A_n$  of  $A$  such that  $A = \cup_{i=1}^n A_i$ . The *power set*  $\mathcal{P}(A)$  is the set whose members are all possible subsets of  $A$ . If  $d(A) = n$  then  $d(\mathcal{P}(A)) = 2^n$ .

Given any two sets  $A$  and  $B$ , we denote the *set difference* as  $A \setminus B$ , and the *complement* of the set  $A$  as  $A^c$ . *De Morgan's laws* articulate that for any pair of sets  $A$  and  $B$ , we have  $(A \cup B)^c = A^c \cap B^c$  and  $(A \cap B)^c = A^c \cup B^c$ .

The term *disjoint* is applied to two sets  $A$  and  $B$  when their intersection  $A \cap B = \emptyset$ . We say that the sets  $A_1, A_2, \dots, A_n$  are disjoint if for every distinct

pair  $i$  and  $j$  ( $i \neq j$ ), we find  $A_i \cap A_j = \emptyset$ . A *partition* of a set  $A$  is an assembly of nonempty disjoint subsets  $A_1, A_2, \dots, A_n$  of  $A$  satisfying  $A = \cup_{i=1}^n A_i$ . The *power set* of  $A$ , denoted as  $\mathcal{P}(A)$ , is the collection of all conceivable subsets of  $A$ . If the cardinality of  $A$  is  $n$ , i.e.,  $d(A) = n$ , then the cardinality of the power set of  $A$  is  $2^n$ , thus expressed as  $d(\mathcal{P}(A)) = 2^n$ .

■ **Example 11.2** Given the set  $A = \{1, 2, 3\}$ , its power set is:

$$\mathcal{P}(A) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, A\}$$

■

Consider a non-empty set  $A$  and a collection  $\mathcal{F}$  of subsets of  $A$ . The pair  $(A, \mathcal{F})$  is designated as an *field* over  $A$  if it fulfills the following conditions: it includes the empty set, denoted as  $\emptyset \in \mathcal{F}$ , it is closed under the operation of complementation, meaning for each  $F \in \mathcal{F}$ , the complement  $F^c \in \mathcal{F}$ , and it is closed under finite unions, which indicates  $F_1 \cup \dots \cup F_n \in \mathcal{F}$  for all subsets  $F_1, \dots, F_n \in \mathcal{F}$ . Additionally, it can be demonstrated that a field also fulfills two further criteria:  $A \in \mathcal{F}$ , and it is closed under finite intersections, expressed as  $F_1 \cap \dots \cap F_n \in \mathcal{F}$  for all subsets  $F_1, \dots, F_n \in \mathcal{F}$ .

Consider two elements,  $x$  and  $y$ . An *ordered pair*, symbolized as  $(x, y)$ , is a pairing of  $x$  and  $y$  in that order. By expanding this concept, an *n-tuple* — which can be visualized as an *n*-element ordered pair — is written as  $(x_1, \dots, x_n)$ . We define the *Cartesian product* of two sets  $A$  and  $B$ , which is symbolized as  $A \times B$ . This product is a set consisting of all possible ordered pairs  $(x, y)$ , where  $x$  belongs to set  $A$  and  $y$  belongs to set  $B$ . The Cartesian product can be extended to  $n$  sets,  $A_1, A_2, \dots, A_n$ , and is expressed as  $A_1 \times A_2 \times \dots \times A_n$ . Moreover, the *n-fold Cartesian product* of a set  $A$  with itself is represented as  $A^n$ .

Let  $R$  be a subset of the Cartesian product of set  $A$  with itself, i.e.,  $R \subseteq A \times A$ . Such a subset is referred to as a *binary relation*, which can be represented as  $aRb$ , meaning that the ordered pair  $(a, b)$  is a member of  $R$ . A binary relation is deemed *reflexive* if for any element  $a$  in  $A$ , it holds that  $aRa$ . It is termed *symmetric* if for all  $a, b$  in  $A$ , the presence of  $aRb$  automatically implies  $bRa$ . The property of *antisymmetric* is attributed to a binary relation when the coexistence of  $aRb$  and  $bRa$  leads to the conclusion that  $a = b$ . It is considered *transitive* if for any  $a, b, c$  in  $A$ , the occurrence of  $aRb$  and  $bRc$  infers  $aRc$ . A binary relation is identified as *total* if for all  $a, b$  in  $A$ , either  $aRb$  or  $bRa$  holds. Binary relations can be extended to include two different sets  $A$  and  $B$  as a subset  $R \subseteq A \times B$ , and can also be adapted to *n-ary* relations represented as  $R \subseteq A_1 \times A_2 \times \dots \times A_n$ .

Let  $R$  be a binary relation that is a subset of the Cartesian product  $A \times A$ , i.e.,  $R \subseteq A \times A$ . If this relation is reflexive, symmetric, and transitive, it is

designated as an *equivalence relation*, typically symbolized by  $\sim$ . Within the context of an equivalence relation, two elements  $a, b$  from set  $A$  are deemed *equivalent* if the relation  $a \sim b$  holds. The notion of an *equivalence class*, denoted as  $[a]$ , refers to the set of all elements in  $A$  that are equivalent to a particular element  $a$ . In other words, the equivalence class of  $a$  is defined as  $[a] := \{b \in A : a \sim b\}$ . An equivalence relation serves to partition the base set into what is known as the *quotient set*. Represented as  $A/\sim$ , the quotient set consists of all equivalence classes stemming from elements of  $A$ , that is,  $A/\sim := \{[a] : a \in A\}$ .

A binary relation that is reflexive, transitive, and antisymmetric is known as a *partial order*, often represented by the symbol  $\preceq$ . A set equipped with a partial order is referred to as a *partially ordered set*, also abbreviated as *poset*. In the context of a poset, an element  $a$  from set  $A$  is considered *minimal* if there is no other element  $b$  in  $A$  for which  $b \preceq a$ . Similarly, an element  $a$  is labeled as *maximal* if no other element  $b$  in  $A$  exists such that  $a \preceq b$ . A relation that embodies reflexivity, transitivity, antisymmetry, and totality is designated as a *total order*, typically symbolized by  $\leq$ . A set that is coupled with a total order is identified as a *totally ordered set*. For any totally ordered set  $A$ , the *maximum* element is represented by  $\max(A)$ , meaning that  $\max(A) \geq x$  holds for all  $x$  in  $A$ . Similarly, the *minimum*, represented by  $\min(A)$ , is defined as an element such that  $\min(A) \leq x$  for all  $x$  in  $A$ .

■ **Example 11.3** Let  $R$  be a relation that is a subset of the Cartesian product of the set of natural numbers  $\mathbb{N}$  with itself, i.e.,  $R \subset \mathbb{N} \times \mathbb{N}$ . In this relation, an ordered pair  $(a, b)$  belongs to  $R$  if  $a$  is a divisor of  $b$ . The set  $\mathbb{N}$ , when paired with relation  $R$ , forms a partially ordered set. In this context, the number 11 serves as a minimal element of  $R$ . This is because 11 is a prime number, which, by definition, can only be divided evenly by 1 and itself. ■

A *function* is defined as a binary relation  $f \subseteq A \times B$ , where for each element  $x \in A$ , there exists at most one  $y \in B$  satisfying the condition  $(x, y) \in f$ . In this context, elements  $(x, y) \in f$  are denoted by  $f(x) = y$ , with the function represented by  $f : A \rightarrow B$ . The set  $A$  is referred to as the *domain* of  $f$ , and  $B$  is the *codomain*. The set  $\{y \in B : \exists x \in A, f(x) = y\}$  constitutes the *range* of  $f$ . If the relation is not defined for all  $x \in A$ , the function is termed *partial*, denoted by  $f(x) \uparrow$  for undefined  $x$  in the function  $f$ .

■ **Example 11.4** In Section 13.4, we will discuss an alternate interpretation of a function as a procedure, or a series of steps, that assigns an element of  $B$  to each element of  $A$ . For instance, the subsequent C code delineates a partial function from  $\mathbb{R}$  to  $\mathbb{R}$ , characterized as partial due to  $\text{inv}(0) \uparrow$ :

```
double inv(double x) {
```

```

    return 1 / x;
}

```

■

A function is said to be *injective* if, for all elements  $x$  and  $y$ ,  $f(x) = f(y)$  implies  $x = y$ . A function is *surjective* when for every  $y$ , there exists at least one  $x$  such that  $f(x) = y$ . A function is described as *bijective* if it exhibits both injective and surjective properties. The *identity* function  $I_A : A \rightarrow A$ , determined by  $f(a) = a$  for all  $a \in A$ , is bijective. These concepts of function, partial function, injective, surjective, and bijective can be extended to  $n$ -ary functions, represented as  $f : A_1 \times A_2 \times \dots \times A_n \rightarrow B$ .

The *inverse* function of a function  $f$ , represented by  $f^{-1}$ , is identified by  $f(f^{-1}(x)) = f^{-1}(f(x)) = x$ . Given two functions  $f$  and  $g$ , where the domain of  $f$  coincides with the range of  $g$ , we define the *composition* of  $f$  with  $g$ , represented by  $f \circ g$ , as  $f \circ g = f(g(x))$ .

An infinite set  $A$  is designated as *countable* if there is a bijective function that maps the elements of  $A$  onto the set of natural numbers  $\mathbb{N}$ . In contrast, a set is deemed *uncountable* if it is not finite and also not countable. We refer to a set as having *countably many* elements if it is either finite or countable.

■ **Example 11.5** Sets like  $\mathbb{N}$ ,  $\mathbb{Z}$  and  $\mathbb{Q}$  are countable, while  $\mathbb{R}$  is not. ■

The *characteristic function* of a set  $A$  is denoted as  $1_A : A \rightarrow \{1, 0\}$ , where  $1_A(x) = 1$  if  $x \in A$  and 0 otherwise.

Considering a real number  $x \in \mathbb{R}$ , its *absolute value*, represented as  $|x|$ , is defined to be  $x$  if  $x \geq 0$  and  $-x$  if  $x < 0$ . The *ceil* function of  $x$ , denoted as  $\lceil x \rceil$ , is the smallest integer that is greater than or equal to  $x$ . The *floor* function of  $x$ , represented by  $\lfloor x \rfloor$ , is the largest integer that is less than or equal to  $x$ . Given two positive integers  $a$  and  $b$ , the *modulo* operation of  $a$  and  $b$ , symbolized by  $a \bmod b$ , gives the remainder of the integer division of  $a$  by  $b$ .

For two functions  $f$  and  $g$ , defined as  $f, g : \mathbb{N} \rightarrow \mathbb{R}^+$ , we assert that  $f(n)$  is of *order of*  $g(n)$ , symbolized by  $f(n) = O(g(n))$ , if there exist positive integers  $c$  and  $m$  such that  $f(n) \leq cg(n)$  for every integer  $n \geq m$ . If  $f(n) = O(g(n))$ , we denote  $g$  as an *upper bound* for  $f$ .

## 11.2 Strings and Languages

Consider a non-empty finite set  $\mathcal{S} = \{s_1, s_2, \dots, s_q\}$ , which we will refer to as the *alphabet*. The elements of this set are termed *symbols*. A *sequence* over  $\mathcal{S}$  is defined as an ordered arrangement of symbols  $x_1 x_2 \dots x_n$  taken from  $\mathcal{S}$ . In the special case where the alphabet is the set  $\mathcal{B} = \{0, 1\}$ , such

sequences are known as *binary sequences*. We use the term *string* to denote a finite sequence. This book predominantly focuses on binary strings.

The *length* of a string  $s$ , represented as  $l(s)$ , refers to the total number of symbols present in  $s$ . We use the notation  $\lambda$  to represent the *empty string*, which is defined as the unique string over  $\mathcal{S}$  that has a length of 0. Given a symbol  $x \in \mathcal{S}$ , the string comprising  $x$  repeated  $n$  times is denoted by  $x^n$ . If  $s = x_1x_2\dots x_n$  constitutes a string, its *reverse*, designated as  $s^R$ , is  $x_nx_{n-1}\dots x_1$ .

The set of all strings  $s_1s_2\dots s_n$  of length  $n$  over the alphabet  $\mathcal{S}$  is denoted by  $\mathcal{S}^n$ <sup>1</sup>. We denote by  $\mathcal{S}^+$  the union of all  $\mathcal{S}^n$  for  $n \geq 1$ , and by  $\mathcal{S}^*$  the set  $\mathcal{S}^+ \cup \{\lambda\}$ . Note that all strings in  $\mathcal{S}^*$  have finite lengths. The term *Kleene closure* refers to  $\mathcal{S}^*$ .

■ **Example 11.6** The following relations hold: the cardinality of the set of binary strings of length  $n$  is  $d(\{s \in \mathcal{B}^* : l(s) = n\}) = 2^n$ , and the cardinality of the set of binary strings of length up to  $n$  is  $d(\{s \in \mathcal{B}^* : l(s) \leq n\}) = 2^{n+1} - 1$ . ■

Given two strings  $s$  and  $t$  from  $\mathcal{S}^*$ , the *concatenation* of  $s$  and  $t$ , symbolized as  $st$ , is the sequence that results from placing the sequence of symbols in  $t$  immediately after the sequence of symbols in  $s$ . Consequently, the length of the concatenated string,  $l(st)$ , is the sum of the lengths of  $s$  and  $t$ . This suggests that  $\mathcal{S}^*$  is closed under the concatenation operation. Furthermore, the set  $\mathcal{S}^*$ , together with the operation of concatenation, forms a *free monoid*. This implies that concatenation is associative  $(s(tr)) = ((st)r)$ , and that there is an identity element, specifically, the empty string  $\lambda$ , for which  $\lambda a = a\lambda = a$  holds.

A string  $s$  is termed a *substring* of  $t$  if there exist strings  $u$  and  $v$  (which may be empty) such that  $t = usv$ . If there exists a string  $u$  such that  $t = su$ ,  $s$  is considered a *prefix* of  $t$ , which is notated as  $s <_p t$ . A subset  $S \subset \mathcal{S}^*$  is described as *prefix-free* if, for any  $s, t \in S$ ,  $s = t$  whenever  $s <_p t$ . Let  $S, T \subset \mathcal{S}^*$  be two sets of strings, the (left) *quotient*  $S^{-1}T$  is defined as the residual words obtained from  $T$  by removing some prefix in  $S$ ; formally,  $S^{-1}T = \{t \mid st \in T \wedge s \in S\}$ .

We denote the *self-delimited* form of a string  $s \in \mathcal{S}^*$  by  $\bar{s}$ , and define it as  $\bar{s} = 1^{l(s)}0s$ . Consequently, the length of  $\bar{s}$ ,  $l(\bar{s})$ , is twice the length of  $s$  plus one, i.e.,  $l(\bar{s}) = 2l(s) + 1$ .

■ **Example 11.7** The set  $\bar{\mathcal{S}}^*$ , which comprises all the self-delimited strings from  $\mathcal{S}^*$ , is prefix-free. ■

---

<sup>1</sup>It is important to avoid confusing the set of strings of length  $n$  over an alphabet  $\mathcal{S}^n$  with the  $n$ -fold Cartesian product of a set  $S^n$ . The use of calligraphic fonts helps distinguish between alphabets and other sets.

In cases where  $\mathcal{S}$  is a totally ordered set, we are able to define a total order on  $\mathcal{S}^*$ . This ordering, termed *shortlex ordering*, arranges sequences primarily by length, positioning the shortest sequences first. Sequences of identical length are further sorted according to lexicographical order.

■ **Example 11.8** Given  $S = \{a, b, c\}$  with  $a < b < c$ , the shortlex order on  $\mathcal{S}^*$  generates the relations  $\lambda < a < b < c < aa < ab < \dots < cc < aaa < aab < \dots < ccc < \dots$  ■

For any arbitrary object  $O$ , we utilize the notation  $\langle O \rangle$  to represent its string representation, presupposing the existence of a standard encoding method. For objects  $O_1, O_2, \dots, O_k$ , the notation  $\langle O_1 O_2 \dots O_k \rangle$  denotes the concatenation of the string representations of these objects:  $\langle O_1 \rangle \langle O_2 \rangle \dots \langle O_k \rangle$ . We employ the notation  $\langle O_1, O_2, \dots, O_k \rangle$  to indicate the concatenation of the representations of these objects in a manner that allows for decoding and unique identification of all objects. As an example,  $\langle O_1, O_2, \dots, O_k \rangle$  might be represented as  $\langle \bar{O}_1 \rangle \langle \bar{O}_2 \rangle \dots \langle \bar{O}_k \rangle$ .

■ **Example 11.9** Natural numbers can be represented by binary strings via the following encoding method:  $\langle 0 \rangle = \lambda$ ,  $\langle 1 \rangle \rightarrow 0$ ,  $\langle 2 \rangle \rightarrow 1$ ,  $\langle 3 \rangle \rightarrow 00$ ,  $\langle 4 \rangle \rightarrow 01$ ,  $\langle 5 \rangle \rightarrow 10$ ,  $\langle 6 \rangle \rightarrow 11$ ,  $7 \rightarrow 000$ , and so on. Therefore, the pair of numbers  $\langle 3, 7 \rangle$  would be represented as 110001110000. Given this particular encoding, it follows that  $l(\langle n \rangle) = \lfloor \log_2(n+1) \rfloor$ . ■

A *language*  $L$  over an alphabet  $\mathcal{S}$  is a set of strings  $L \subset \mathcal{S}^*$ . The elements of  $L$  are called *words*. The *empty language* is the language that contains no words  $L = \emptyset$ .

A *language*, denoted by  $L$ , over an alphabet  $\mathcal{S}$ , is defined as a subset of strings such that  $L \subseteq \mathcal{S}^*$ . The individual elements of  $L$  are termed as *words*. The unique language that does not contain any words is referred to as the *empty language*, and is expressed as  $L = \emptyset$ .

Consider two languages  $L_1$  and  $L_2$  over a common alphabet  $\mathcal{S}$ . There are several standard operations that can be applied to these languages, including: language union  $L_1 \cup L_2 = \{w \in \mathcal{S}^* \mid w \in L_1 \text{ or } w \in L_2\}$ , language intersection  $L_1 \cap L_2 = \{w \in \mathcal{S}^* \mid w \in L_1 \text{ and } w \in L_2\}$ , language complement  $\overline{L_1} = \{w \in \mathcal{S}^* \mid w \notin L_1\}$ , and Kleene closure  $L_1^* = \{\lambda\} \cup \{wz \mid w \in L_1 \text{ and } z \in L_1^*\}$ .

Languages can be systematically generated using a finite set of string rewriting rules known as grammars. A *grammar*, denoted as  $G$ , is a 4-tuple  $(N, \Sigma, P, S)$ , where:  $N \subseteq \mathcal{S}$  is a finite set of *nonterminal symbols*;  $\Sigma \subseteq \mathcal{S}$  is a finite set of *terminal symbols*;  $P$  is a finite set of *production rules* in the form  $(\Sigma \cup N)^* N (\Sigma \cup N)^* \rightarrow (\Sigma \cup N)^*$ ; and  $S \in N$  is a special *start symbol*. Each production rule transforms one string of symbols into another, starting with the designated start symbol.

■ **Example 11.10** Let us consider the alphabet  $\mathcal{S} = \{S, a, b\}$ , and define the grammar  $(N, \Sigma, P, S)$  where  $N = \{S\}$ ,  $\Sigma = \{a, b\}$ ,  $P = \{S \rightarrow aSb, S \rightarrow ba\}$ , and the start symbol is  $S \in N$ . This grammar generates the language  $L = \{a^n bab^n \mid n \geq 0\} = \{ba, abab, aababb, aaababbb, \dots\}$ . ■

The *Chomsky hierarchy* serves as a categorization system for grammars, based on their expressive capacity or the range of languages they can generate. The hierarchy, listed from the most to the least restrictive grammars, is defined as follows (where  $a$  represents a terminal symbol,  $A, B$  are non-terminal symbols, and  $\alpha, \beta$ , and  $\gamma$  denote strings composed of either terminal or non-terminal symbols):

**Type-3** Also known as *regular grammars*. For these grammars, the left-hand side of every production rule consists solely of a single nonterminal symbol. The right-hand side may either be an empty string, a single terminal symbol, or a single terminal symbol followed by a nonterminal symbol. Formally, the rules are  $(A \rightarrow \lambda, A \rightarrow a, A \rightarrow aB)$ .

**Type-2** These are referred to as *context-free grammars*. Here, the left-hand side of each production rule is comprised of only a single nonterminal symbol, symbolically expressed as  $(A \rightarrow \alpha)$ .

**Type-1** These are *context-sensitive grammars*. In this case, the left and right sides of the production rules may be embedded within a context of terminal and nonterminal symbols. This can be written as  $(\alpha A \beta \rightarrow \alpha \gamma \beta)$ .

**Type-0** The *recursively enumerable grammars* fall in this category, and they do not impose any constraints on the production rules ( $\gamma \rightarrow \alpha$ ).

■ **Example 11.11** The grammar discussed in Example 11.10 is classified as a Type-2, or context-free grammar. ■

The *Backus–Naur form* is a notation system specifically designed for context-free grammars. This notation is frequently employed in the field of computer science to formally outline the syntax of programming languages and communication protocols. A Backus–Naur form consists of a set of production rules structured in the following way:

```
<symbol> ::= __expression__
```

Here, `<symbol>` stands for a non-terminal symbol, `__expression__` refers to a string of terminal or non-terminal symbols, and `::=` indicates that the symbol on the left-hand side should be substituted with the expression on the right. Several `__expression__`s can be integrated into a single production rule by separating them with a vertical bar `|`, suggesting that any one of them can be selected for substitution. Symbols that are not found

on a left-hand side of a production rule are considered terminal symbols. Conversely, those that appear on a left-hand side are non-terminal symbols and are invariably enclosed within the pair <>. The non-terminal symbol of the first production rule is identified as the start symbol.

■ **Example 11.12** The grammar discussed in Example 11.10 can be reformulated in Backus-Naur form using the subsequent production rule:

`<string> ::= a <string> b | ba`

■

### 11.3 Counting Methods

|           | Without replacement | With replacement   |
|-----------|---------------------|--------------------|
| Ordered   | $\frac{n!}{(n-r)!}$ | $n^r$              |
| Unordered | $\binom{n}{r}$      | $\binom{n+r-1}{r}$ |

*Combinatorics*, a specialized branch of mathematics, primarily focuses on the investigation of discrete objects and their mutual relationships. The core aspects of combinatorics include the counting, arranging, and selection of these objects, supplemented by the methodologies employed for achieving these objectives. It presents an array of robust tools essential for handling extensive collections of objects exhibiting certain characteristics. In this section, we shall focus on revisiting the significant outcomes of combinatorics from the perspective of sets and ordered lists.

The *multiplication rule* is a fundamental theorem that specifies the number of potential outcomes in the Cartesian product of sets. According to this rule, if there are  $k$  sets  $A_1, A_2, \dots, A_k$ , and each set contains  $n_i$  elements ( $i = 1, \dots, k$ ), then the Cartesian product  $A_1 \times A_2 \times \dots \times A_k$  encapsulates a total of  $n_1 n_2 \dots n_k$  elements. Particularly, if a set  $A$  consists of  $n$  elements, then  $A^k$  includes  $n^k$  elements.

The *inclusion-exclusion principle* provides the count of the union of several sets, given the individual size of each set, and the size of every feasible intersection of these sets. Given  $k$  sets  $A_1, A_2, \dots, A_k$ , the expression is as follows:

$$d\left(\bigcup_{i=1}^k A_i\right) = \sum_{i=1}^k d(A_i) - \sum_{i < j} d(A_i \cap A_j) + \sum_{i < j < l} d(A_i \cap A_j \cap A_l) - \dots - \sum_{i < j < l < m} d(A_i \cap A_j \cap A_l \cap A_m) + \dots + (-1)^{k+1} d(A_1 \cap A_2 \cap \dots \cap A_k)$$

*Permutations* denote the number of ways in which a set's elements can be arranged. Suppose  $A$  is a set with  $n$  elements, the number of permutations of  $n$  elements taken  $k$  at a time, represented as  $P_{n,k}$ , is  $P_{n,k} =$

$n(n-1)\dots(n-k+1)$ . When  $k = n$ , the permutations are calculated as  $P_{n,n} = n(n-1)\dots 1 = n!$ , where  $n!$  is known as  $n$  factorial.

The *pigeonhole principle* is a simple yet powerful concept that states if you have more pigeons than pigeonholes, then there must be at least one pigeonhole with more than one pigeon. In a more mathematical sense, if you have  $n$  items to place into  $m$  containers and  $n > m$ , then at least one container must contain more than one item.

For performing vast computations, an approximation of  $n!$ , commonly known as *Stirling's formula*, proves highly effective:

$$\log(n!) \approx \frac{1}{2} \log(2\pi) + \left(n + \frac{1}{2}\right) \log(n) - n$$

■ **Example 11.13** Consider the set  $\{a, b, c\}$ . There are six unique permutations of its elements:  $[a, b, c], [a, c, b], [b, a, c], [b, c, a], [c, a, b], [c, b, a]$ . Each permutation signifies a distinct order of the elements in the original set. ■

Numerous counting problems revolve around determining the quantity of subsets of a certain size present within a given set. Given a set with  $n$  elements, the total possible subsets amount to  $2^n$ , inclusive of the empty set and the set itself. The quantity of subsets of size  $k$ , also known as the number of *combinations* of  $k$  elements from a set of  $n$ , denoted by  $C_{n,k}$ , is computed using the formula  $C_{n,k} = \frac{P_{n,k}}{k!} = \frac{n!}{k!(n-k)!}$ . The symbol  $\binom{n}{k}$  also represents the number  $C_{n,k}$ , which is known as the *binomial coefficient*. We know that for all  $n$ ,  $\binom{n}{0} = \binom{n}{n} = 1$ , and for all  $n$  and  $k = 0, 1, \dots, n$ ,  $\binom{n}{k} = \binom{n}{n-k}$ . Moreover,  $\binom{n}{k} = 0$  for  $k > n$ .

■ **Example 11.14** Take for instance the set  $\{a, b, c, d\}$ . There exist 4 combinations of size 3:  $[a, b, c], [a, b, d], [a, c, d]$ , and  $[b, c, d]$ . Combinations do not consider order, hence  $[a, c, d]$  and  $[d, c, a]$  are deemed the same combination.

■ The *multinomial coefficient*, an extension of the binomial coefficient to more than two categories or types, signifies the number of ways a set of objects can be partitioned into a fixed number of subsets, each of which contains a specific number of objects. Assuming we have a set with  $n$  elements, which can be partitioned into  $k$  subsets of sizes  $n_1, n_2, \dots, n_k$ , respectively. The multinomial coefficient, symbolized as  $\binom{n}{n_1, n_2, \dots, n_k}$ , represents the number of feasible partitions, and is computed as  $\binom{n}{n_1, n_2, \dots, n_k} = \frac{n!}{n_1! n_2! \dots n_k!}$ .

## 11.4 Matrices

A *matrix*, denoted by  $A$ , of order  $m \times n$  is composed of a sequence of  $mn$  scalars. These scalars are arranged in a rectangular array consisting of  $m$

rows and  $n$  columns, as depicted below:

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

The entry  $a_{ij}$  denotes the element of  $A$  found at the  $i$ -th row and  $j$ -th column. The set of all matrices of order  $m \times n$  is symbolized as  $\mathcal{M}_{m \times n}$ . A *row matrix* belongs to the set  $\mathcal{M}_{1 \times n}$ , whereas a *column matrix* is a member of  $\mathcal{M}_{m \times 1}$ . A *square matrix* is any matrix within the set  $\mathcal{M}_{n \times n}$ . The entries  $a_{ii}$  compose the *main diagonal* of a square matrix. A *diagonal matrix* is distinguished by having all of its entries outside the main diagonal as zero. The *identity matrix*, denoted by  $I$ , is a diagonal square matrix with all entries on the main diagonal equal to 1.

The *transpose* of a matrix  $A \in \mathcal{M}_{m \times n}$  is defined as the matrix  $A^T \in \mathcal{M}_{n \times m}$ , with entries at position  $(i, j)$  mirroring the entries at position  $(j, i)$  in  $A$ . If  $A = A^T$ , then the matrix  $A$  is classified as a *symmetric matrix*. A *submatrix* of a matrix is obtained by eliminating any selection of rows and/or columns.

■ **Example 11.15** Take for instance the square matrix  $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ . Here, the entry located at position  $(2, 3)$  has the value 6, the main diagonal is made up of the numbers 1, 5, and 9. The transpose of the matrix is  $A^T = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$ , and the matrix  $B = \begin{pmatrix} 1 & 3 \\ 4 & 6 \end{pmatrix}$  is identified as a submatrix of  $A$ . ■

The addition of two matrices  $A$  and  $B$  of identical size results in a new matrix  $A + B$ . Each entry  $(i, j)$  of this matrix is given by  $(A + B)_{ij} = a_{ij} + b_{ij}$ . The operation of matrix addition exhibits associativity, i.e.,  $(A + B) + C = A + (B + C)$ , and commutativity, i.e.,  $A + B = B + A$ . It has a neutral element, such that  $A + 0_{m \times n} = A$ , and an inverse element,  $A + (-A) = 0_{m \times n}$ .

The product of a scalar  $\lambda$  and a matrix  $A$  yields another matrix, denoted  $\lambda A$ , where each entry  $(i, j)$  is  $(\lambda A)_{ij} = \lambda a_{ij}$ . This scalar multiplication operation is distributive relative to the addition of matrices, as  $(\alpha(A + B)) = \alpha A + \alpha B$ , and to the addition of scalars, as  $(\alpha + \beta)A = \alpha A + \beta B$ . It is also associative with scalar multiplication, such that  $(\alpha\beta)A = \alpha(\beta A)$ , and has a unit element, as  $1A = A$ .

The product of two matrices  $A_{m \times n}$  and  $B_{n \times p}$  results in a matrix  $AB_{m \times p}$ , where each entry  $(i, j)$  is given by  $(AB)_{ij} = \sum_{k=1}^n a_{ik}b_{kj}$ . The operation of matrix multiplication is associative, as  $(AB)D = A(BD)$ , and it possesses a left neutral element  $AI_n = A$  and a right neutral element  $I_mA = A$ . It is associative with respect to scalar multiplication, such that  $\alpha(AB) = (\alpha A)B =$

$A(\alpha B)$ , and is distributive with respect to matrix addition, both from the right  $A(B + C) = AB + AC$  and from the left  $(B + C)D = BD + CD$ .

Additionally, the transpose operation satisfies the following properties:  $(A + B)^T = A^T + B^T$ ,  $(\lambda A)^T = \lambda A^T$ , and  $(AB)^T = B^T A^T$ .

■ **Example 11.16** Given the matrices  $A = \begin{pmatrix} 1 & 2 \\ 3 & 5 \end{pmatrix}$  and  $B = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$  we have that  $A + B = \begin{pmatrix} 6 & 8 \\ 10 & 13 \end{pmatrix}$ ,  $2A = \begin{pmatrix} 2 & 4 \\ 6 & 10 \end{pmatrix}$  and  $AB = \begin{pmatrix} 19 & 22 \\ 50 & 58 \end{pmatrix}$ . ■

A square matrix  $A$  is *invertible* or *non-singular* if there exists a matrix  $B$  such that  $AB = BA = I$ . If  $A$  is non-singular,  $B$  is unique and is called the *inverse matrix* of  $A$ , denoted by  $A^{-1}$ . A matrix  $A$  is *orthogonal* if its transpose is equal to its inverse  $A^T = A^{-1}$ ; the columns and rows of a orthogonal matrix are called *orthonormal vectors*.

The *determinant* is a special scalar value that can be computed from the elements of a square matrix. The determinant is denoted as  $\det(A)$  for a matrix  $A$ . The determinant of a square matrix can be computed using the Leibniz formula, which is given as:

$$\det(A) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \cdot a_{1,\sigma(1)} \cdot a_{2,\sigma(2)} \cdots a_{n,\sigma(n)}$$

where  $S_n$  denotes the set of all permutations of the numbers 1 to  $n$ , and  $\text{sgn}(\sigma)$  is the signature of the permutation  $\sigma$ , being +1 for even permutations and -1 for odd permutations. The determinant is nonzero if and only if the matrix is invertible

■ **Example 11.17** The computation of the determinant of a  $3 \times 3$  matrix  $A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$  is given by:

$$\det(A) = aei + bfg + cdh - ceg - bdi - afh$$

■

For a given matrix  $A$ , the *rank*, denoted as  $\text{rank}(A)$ , is the maximum number of linearly independent rows or columns within the matrix.

A number  $\lambda$ , and a non-zero vector  $\mathbf{v}$  satisfying  $A\mathbf{v} = \lambda \mathbf{v}$  are called an *eigenvalue* and an *eigenvector* of  $A$ , respectively.

Matrix decomposition is the process of rendering a matrix into a more easily accessible form, meanwhile certain properties, like the determinant or the rank, are preserved. The *singular value decomposition* of a matrix  $A$  of order  $m \times n$  is a factorization of the form  $A = U\Sigma V^T$  where  $U$  is an  $m \times m$  orthogonal matrix,  $\Sigma$  is an  $m \times n$  non-negative diagonal matrix of rectangular shape, and  $V^T$  is the transpose of an  $n \times n$  orthogonal matrix.

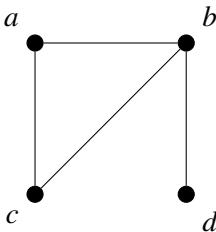


Figure 11.2: An Example of Graph

## 11.5 Graphs

A *graph*<sup>2</sup>  $G$  is represented as an ordered pair  $(V, E)$ , comprising a set  $V$ , referred to as *vertices*, and a set  $E$ , denoted as *edges*. The members of  $E$  take the form of unordered pairs  $\{u, v\}$ , constituting distinct vertices  $u, v \in V$  (loops are not permitted). Vertices  $u$  and  $v$  are described as *adjacent* when an edge  $\{u, v\} \in E$  exists, consequently, they are termed the *endpoints* of that edge. If the set  $V$  is infinite, the graph is classified as an *infinite graph*. This book, however, focuses exclusively on finite graphs. Given that  $G = (V, E)$  is a graph, its *adjacency matrix* is a square  $d(V) \times d(V)$  matrix  $A$  whereby element  $A_{uv}$  equals 1 if  $\{u, v\} \in E$  and 0 otherwise.

Graphs are typically illustrated as a collection of dots representing vertices, linked by lines denoting edges.

■ **Example 11.18** If  $V = \{a, b, c, d\}$  and  $E = \{\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}\}$ , the graph  $G = (V, E)$  is represented in Figure 11.2. ■

When a vertex  $v$  serves as an endpoint of an edge  $e$ , it is said that  $e$  is *incident* upon  $v$ . The *degree* of a vertex  $v$ , denoted as  $\deg(v)$ , corresponds to the count of edges incident on  $v$ . A vertex with degree zero is labeled as *isolated*, while a vertex with degree one is termed *pendant*. The *neighborhood* of a vertex  $v$ , signified by  $N(v)$ , includes all vertices that are adjacent to  $v$ . If  $A \subset V$ , the neighborhood of  $A$  is given by  $N(A) = \cup_{v \in A} N(v)$ . A *path* in a graph consists of a series of unique vertices  $\{v_0, v_1, \dots, v_k\}$  such that  $v_i$  and  $v_{i+1}$  are adjacent for each  $1 \leq i < k$ . A path is known as a *simple path* if no vertex is repeated. A graph is considered *connected* when a path exists between any two vertices. If  $v_0 = v_k$ , the path is defined as a *cycle*. A cycle is called a *simple cycle* if it consists of at least three vertices and only the first and last vertices are repeated.

■ **Example 11.19** In a given graph  $G = (V, E)$ , the *handshaking theorem*

---

<sup>2</sup>The definition of a graph stated here corresponds to the definition of a *simple graph* as found in typical discrete mathematics literature.

posits that  $\sum_{v \in V} \deg(v) = 2m$ , where  $m = d(E)$ , given that each edge has two endpoints. ■

If the vertex pairs  $u, v$  are arranged in ordered pairs, the graph is termed a *directed graph*. In this case,  $u$  is referred to as the *initial vertex*, while  $v$  is known as the *terminal vertex*. Given that  $G$  is a directed graph, the *in-degree* of a vertex  $v$ , signified by  $\text{indeg}(v)$ , corresponds to the count of edges in which  $v$  serves as a terminal vertex. The *out-degree*, denoted by  $\text{outdeg}(v)$ , refers to the number of edges where  $v$  is the initial vertex. A directed graph is *strongly connected* if a directed path connects each pair of vertices. Directed graphs are typically illustrated with arrows, rather than lines, to represent edges.

A graph  $G$  is classified as *bipartite* if the vertex set  $V$  can be partitioned into two subsets  $V_1$  and  $V_2$  such that every edge of  $G$  links a vertex from  $V_1$  to one from  $V_2$ . Bipartite graphs are usually denoted as  $G = (V_1, V_2, E)$ . The degree of vertices in a bipartite graph adheres to the *degree sum formula*,  $\sum_{u \in V_1} \deg(u) = \sum_{v \in V_2} \deg(v) = d(E)$ .

A graph  $G(V', E')$  is a *subgraph* of  $G(V, E)$  if  $V'$  is a subset of  $V$  and  $E'$  is a subset of  $E$  with endpoints that belong to  $V'$ . A graph  $G$  is termed a *labeled graph* if its edges and/or vertices are assigned specific data. Specifically, if each edge  $e$  of  $G$  is allocated a nonnegative number  $w(e)$ , then  $w(e)$  is referred to as the *weight* of  $e$ .

A specific type of graph that plays a fundamental role in this book is the *tree*. Defined as a non-empty graph, a tree ensures any pair of vertices are interconnected by a singular, unique path. A tree always includes a specially designated vertex termed the *root*, and every edge of the tree is oriented away from the root.

■ **Example 11.20** An alternative definition of trees, based on set theory, posits that a tree is a partially ordered set  $(T, <)$  such that for each  $t \in T$ , the set  $S = \{s \in T : s < t\}$  possesses a least element – an element smaller than all other elements in  $S$ . ■

Given a tree  $T$ , for any vertex  $v$  other than the root, the *parent* of  $v$  is the single, unique vertex  $u$  such that an edge directly links  $u$  to  $v$ . Conversely, if  $u$  is the parent of  $v$ ,  $v$  is then identified as a *child* of  $u$ . Any other vertex within the tree sharing the same parent as  $v$  is considered a *sibling* of  $v$ . The *ancestors* of a vertex encompass all vertices along the path from the root to the given vertex, excluding the vertex itself but including the root. The *descendants* of a vertex  $v$  include those vertices that count  $v$  as an ancestor. A vertex with no children is labeled as a *leaf*, whereas vertices possessing children are referred to as *branches*. The *depth* of a vertex  $v$  is determined by the length of the unique path leading from the root to  $v$ . The *height* of a

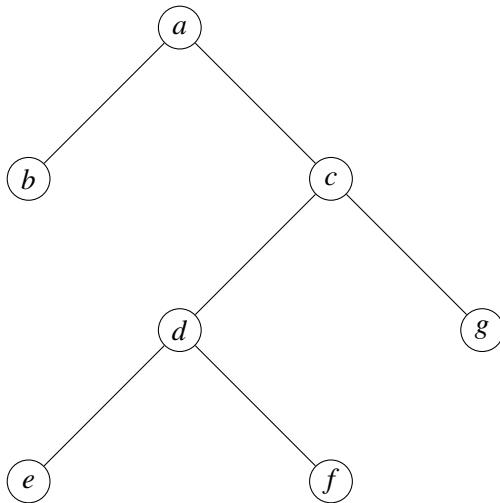


Figure 11.3: An Example of a Tree

tree is the maximum among the depths of all its vertices.

■ **Example 11.21** For the tree illustrated in Figure 11.3, the root vertex is  $a$ ;  $c$  serves as a parent of  $d$ , making  $d$  a child of  $c$ ;  $d$  and  $g$  are siblings; the ancestors of  $d$  include  $a$  and  $c$ ; the descendants of  $c$  comprise  $d$ ,  $e$ , and  $f$ ; leaf nodes in the tree are  $b$ ,  $e$ ,  $f$ , and  $g$ ;  $a$  and  $c$  are branches; the depth of  $d$  is 3; the height of the tree is 4. ■

Given a vertex  $v$  in a tree, the *subtree* with  $v$  as its root is the subgraph within the tree that includes  $v$ , its descendants, and all edges connected to these descendants. A tree is termed a *k-ary tree* if each branch houses no more than  $k$  children. If every branch comprises exactly  $k$  children, the tree is then labeled a *full k-ary tree*. A *k-ary tree* where  $k = 2$  is specifically referred to as a *binary tree*. A *k-ary tree* of height  $h$  is deemed *balanced* if all its leaves are located at a depth of  $h$  or  $h - 1$ .

■ **Example 11.22** A tree composed of  $n$  vertices includes  $n - 1$  edges. A full *k-ary tree* featuring  $i$  branches hosts  $m = ki + 1$  vertices. ■

The procedure of visiting each node in a tree exactly once is defined as *tree traversal*. The classifications of tree traversals are determined by the order of node visits, namely, *depth-first* and *breadth-first* order. In a depth-first traversal, the algorithm initiates at the root node and ventures as far as possible along each branch before transitioning to the next sibling. There are three standard strategies for processing nodes within the tree: *in-order*, *pre-order*, and *post-order*.

The following snippet of code, resembling C language syntax, demonstrates the usage of a recursive pre-order depth-first traversal algorithm to print a binary tree:

```
void print_tree(binary_tree *tree) {  
    if (!is_empty(tree)) {  
        printf("%c\n", tree->node); /* print node */  
        print_tree(tree->left_branch); /* process left branch */  
        print_tree(tree->right_branch); /* process right branch */  
    }  
}
```

Conversely, in a breadth-first traversal, the algorithm commences at the tree root and explores all nodes at the current depth before progressing to nodes at the subsequent depth level. Implementing depth-first tree traversal algorithms necessitates the employment of sophisticated data structures. For an example of such algorithms, please consult the references section.

■ **Example 11.23** In the case of the tree delineated in Example 11.18, a pre-order depth-first traversal would yield the string "abcdefg". Conversely, a pre-order breadth-first traversal would generate the string "abcdgef". ■

## References

The book "Discrete Mathematics" by Johnsonbaugh [Joh09] is tailored for undergraduate students taking a one- or two-semester course in discrete mathematics, thoroughly covering key topics in the field. "Introduction to the Theory of Computation" by Sipser [Sip12] offers a comprehensive, clear, and student-centric introduction to the latest computational theory topics and methodologies. It is highly acclaimed for its in-depth exploration of automata, formal languages, and complexity theory. "Introduction to Algorithms" by Cormen et al. [Cor+90], commonly known by the acronym "CLRS" derived from the authors' initials, is a seminal work in the algorithms field, encompassing a broad spectrum of topics, including graph algorithms. This book is both extensive and rigorous. Finally, "Matrix Computations" by Golub and Van Loan [GV13] delves into various topics related to matrices, with an emphasis on matrix computations - an area of particular interest to those engaged in computational studies.



## 12. Discrete Probability

*The purpose of models is not to fit the data  
but to sharpen the questions.*

Samuel Karlin

Probability theory is the branch of mathematics that studies random experiments and random phenomena. Probability assigns a numerical description to all the possible outcomes of an experiment based on how likely these outcomes are to occur. Even if the outcome of an experiment cannot be determined in advance, we can study its properties with probability theory and draw relevant results and conclusions. For example, while we cannot predict the next number that will appear in a lottery game, probability theory can help us understand why it is not a good strategy to spend all our savings on lottery tickets with the goal of becoming rich.

The importance of probability theory extends far beyond games of chance. It provides the mathematical foundation for statistical inference, enabling us to draw conclusions from data, and for machine learning, where it underpins algorithms used for classification and forecasting based on large datasets. Additionally, probability theory plays a crucial role in fields such as finance, risk management, and the natural sciences, where understanding uncertainty and variability is essential.

In this chapter, we are going to focus on the area of discrete probability. In this version of the theory, the possible outcomes of an event are finite or, at most, countably infinite. We are interested in discrete probability, first because of its practical applications in the area of learning from data, and second, because discrete probability has some very interesting connections to theories used in this book: the length of optimal codes, the probability that a random machine will halt, and the derivation of universal distributions based on Kolmogorov complexity. All of these connections are relevant to our theory of nescience.

On a philosophical level, it might be argued that there can only be countable sample spaces, since measurements cannot be made with infinite accuracy. [...] While in practice this is true, probabilistic and statistical methods associated with uncountable sample spaces are, in general, less cumbersome than those for countable spaces, and provide a close approximation to the true (countable) situation.

We will cover only the most important concepts and results of probability theory. The contents have been selected based on their applicability to the theory of nescience. For example, moment-generating functions are not covered. For a more comprehensive introduction to probability theory refer to the references section of this chapter.

We are going to study probability theory from a formal, axiomatic point of view. We will start by formulating a basic collection of fundamental axioms and then derive the major results and properties from them. Axioms are crucial in mathematical theory because they provide the foundation for constructing a consistent, universal, and rigorous framework. In probability theory, they allow us to define and manipulate the elusive concept of probability in a way that is both precise and broadly applicable.

## 12.1 Foundations of Probability Theory

The concept of *probability* represents a profound intellectual challenge. Let us consider an instance where a dice is rolled and the objective is to compute the probability of an even number being the outcome. The dice comprises six distinct outcomes, and given that half of these are even numbers, we posit that the probability of yielding an even number is  $3/6$  or equivalently  $1/2$ . This embodies the *classical interpretation* of probability which asserts that in an experiment where all finite potential outcomes possess an equal likelihood of occurrence, the probability of an event equates to the count of favorable instances over the total number of instances. This interpretation, however, confronts the issue of circularity in its definition as "equally likely" essentially amounts to "possessing the same probability". An alternative

method for probability assignment might be the implementation of the *principle of indifference*, which postulates that in the absence of any relevant evidence, all potential outcomes should possess identical probability. This principle, however, encounters a predicament when there exists evidence that contradicts the presumption of equivalence amongst all outcomes. For instance, how should probability be assigned when knowledge of a loaded dice suggests that not all sides possess an equal likelihood of occurrence?

The *frequentist interpretation* of probability posits that one should roll the dice multiple times and contrast the frequency of even numbers with the total number of rolls. The fundamental notion is to execute the experiments repeatedly under similar conditions and assign the relative frequency of each outcome as its probability. This interpretation, however, faces two primary limitations. Firstly, the definition of repeating an experiment under "similar conditions" remains vague; if the conditions were truly identical, the results across all trials would invariably be the same. Secondly, the notion of a "large number of times" is undefined (technically, the experiment should be executed an infinite number of times). From a practical standpoint, implementing the frequentist interpretation is fraught with challenges. Some experiments, such as predicting the probability of a candidate winning an election, cannot be repeated multiple times. Furthermore, probability is defined in the context of a sequence of experiments, hence precluding the computation of the probability of a solitary outcome. Lastly, this interpretation necessitates the existence of a relative frequency limit, a condition which is not always satisfied, as illustrated by financial time series.

The *subjective interpretation*, representing a third approach to the concept of probability, suggests assigning probabilities to each event that reflect our degree of belief: the higher our conviction in the event's occurrence, the greater its assigned probability. However, it's important to note that not all potential probability allocations are viable; certain coherence rules need to be satisfied. For instance, when placing bets on the outcome of a dice roll, an assignment of probabilities that ensures a complete loss of money - a scenario known as a *dutch book* - would contravene the conditions of the subjective interpretation. It transpires that the conditions both necessary and sufficient to ensure a fair bet align with the axioms of probability introduced subsequently. Hence, we are free to assign any probabilities we desire to events, provided they remain consistent with the axioms of probability. A key drawback of the subjective interpretation is the inherent variability in individuals' degrees of belief. The *Bayesian interpretation* of probability offers a solution: we commence with a provisional assignment of probabilities and, upon accruing further evidence, adjust our degree of belief or probability accordingly. With the accumulation of more evidence, estimated probabilities will converge to

the true probabilities. Regardless, the task of assigning probabilities to an infinite number of events is typically unattainable for humans in general.

### Probability theory builds upon set theory

We do not define probabilities in terms of frequencies but instead take the mathematically simpler axiomatic approach. [...] The axiomatic approach is not concerned with the interpretation of probabilities, but is concerned only that the probabilities are defined by a function satisfying the axioms.

Currently, the notion of probability is defined axiomatically via the *axiomatic interpretation*. This implies that we abandon attempts to explicitly define the concept of probability and instead accept some of its properties as inherently true. Intuitively, a probability should be a value between 0 and 1, wherein an event with a zero probability is deemed impossible, and an event with a probability of one is certain to occur. Additional properties are required of probabilities. For instance, should two events  $A$  and  $B$  with probabilities  $P(A)$  and  $P(B)$  respectively, be disjoint, the probability of either  $A$  or  $B$  occurring should be  $P(A) + P(B)$ . If  $A$  and  $B$  could occur simultaneously and are independent (however that is defined), the probability of both events occurring concurrently should be  $P(A)P(B)$ . Moreover, the probability of  $A$  occurring given that  $B$  has already happened should be the fraction of the probability of  $A$  that intersects with  $B$ . Anything that satisfies these properties could be considered a probability.

Probability theory is fundamentally concerned with the task of assigning a numerical value to specific events drawn from a sample space. The term "event" in this context may be somewhat counterintuitive, as it suggest the occurrence of something, which is not always applicable. For instance, consider the sample space of all possible outcomes when tossing a fair coin. A subset of this sample space could be the empty set, which represents no coin toss happening at all. In the conventional understanding of an "event", this scenario may confuse people who aren't familiar with the mathematical meaning, as it does not correspond to something "happening". Nonetheless, for the sake of clarity, we will continue to employ the term "events" to designate what essentially are subsets.

**Definition 12.1.1** Given  $(\Omega, \mathcal{A})$  as a field over a non-empty discrete set,  $\Omega$  is referred to as the *sample space*, its constituents are termed *outcomes*, and the components of  $\mathcal{A}$  are referred to as *events*. Specifically,  $\Omega$  is designated the *certain event*, while the empty set  $\emptyset$  is deemed the *impossible event*.

As previously discussed in Section 11.1, given that  $(\Omega, \mathcal{A})$  is a field, we can deduce that  $\Omega \in \mathcal{A}$  and that  $\emptyset \in \mathcal{A}$ . Additionally, the union of a finite collection of events constitutes an event  $A_1 \cup A_2 \cup \dots \cup A_n \in \mathcal{A}$ , and

the intersection of a finite collection of events is likewise deemed an event  $A_1 \cap A_2 \cap \dots \cap A_n \in \mathcal{A}$ .

As alluded to in the introduction of this chapter, our principal interest lies in discrete mathematics, and hence, we will largely focus on probabilities pertaining to discrete sets (be they finite or countably infinite). An extension of the concept of probability to continuous sets would necessitate the utilization of  $\sigma$ -algebras of sets instead of fields and the application of measure theory. For example, consider a scenario where we measure the electric current in a circuit, which can take any value between -5 and 5 volts. Unlike a discrete set of outcomes, such as specific whole numbers, here we have infinitely many possible values within this interval. In this case, it doesn't make sense to talk about the probability of the current being exactly 2.5 volts (or any other specific real number), as there are infinitely many possible outcomes. Instead, we would talk about the probability of the current falling within a certain range, such as between 2 and 3 volts.

The prevailing axiomatization utilized in the realm of probability theory is encapsulated within the framework of the *Kolmogorov axioms*<sup>1</sup>.

**Definition 12.1.2** (*Kolmogorov's Axioms*) A *probability* is a real number  $P(A) \in \mathbb{R}$  allocated to each event  $A \in \mathcal{A}$  in the field  $(\Omega, \mathcal{A})$ . This allocation adheres to the following axioms:

**Axiom 1** Each probability is nonnegative:  $P(A) \geq 0$ .

**Axiom 2** The probability of the certain event is one:  $P(\Omega) = 1$ .

**Axiom 3** For any finite sequence of disjoint events  $A_1, A_2, \dots, A_n$ , the probability of the union of these events is the sum of their probabilities:  $P(\bigcup_{i=1}^n A_i) = \sum_{i=1}^n P(A_i)$ .

The triplet  $(\Omega, \mathcal{A}, P)$  constitutes what is known as a *probability space*.

Despite their significance, the Kolmogorov axioms encounter certain complexities. While they provide the foundational principles that probabilities must conform to, such as non-negativity, normalization, and additivity, they do not offer explicit guidance on how to assign probabilities to specific events. Essentially, the axioms define the rules that probability must follow but leave the determination of those probabilities open. This stems from the fact that Kolmogorov's axioms are highly general and abstract, designed to apply to any measure-theoretic structure. This level of generality means they can accommodate a wide range of constructs beyond probability, such as normalized mass, volume, or other physical quantities. Finally, the connec-

<sup>1</sup>In discrete probability theory, the sample space consists of a finite or countably infinite set of distinct outcomes, which means events are typically composed of individual, separable outcomes. Since probabilities are assigned directly to these discrete events, only finite unions of disjoint events need to be considered in Axiom 3 to cover all practical cases.

tion between model theory (as discussed in Appendix ??) and probability theory through the Kolmogorov axioms is not straightforward. Model theory primarily deals with first-order logic, which can express basic properties like non-negativity and finite additivity of sets. However, when it comes to formally defining a continuous quantity, such as probability as a real number, it fails because first-order logic cannot fully capture the complexity of real numbers and continuous structures.

■ **Example 12.1** Consider a sample space  $\Omega$  composed of  $n$  equally probable elements. If we have an event  $A \subset \Omega$  comprised of  $d(A) = m$  elements, then the probability of event  $A$  can be represented as  $P(A) = m/n$ . ■

We need general methods of defining probability functions that we know will always satisfy Kolmogorov's Axioms.

**Proposition 12.1.1** Let  $\mathcal{S} = \{s_1, \dots, s_n\}$  be a finite set. Let  $\mathcal{B}$  be any sigma algebra of subsets of  $\mathcal{S}$ . Let  $p_1, \dots, p_n$  be nonnegative numbers that sum to 1. For any  $A \in \mathcal{B}$ , define  $P(A)$  by:

$$P(A) = \sum_{\{i : s_i \in A\}} p_i$$

(The sum over an empty set is defined to be 0.) Then  $P$  is a probability function on  $\mathcal{B}$ . This remains true if  $\mathcal{S} = \{s_1, s_2, \dots\}$  is a countable set.

*Proof.*

We now venture to establish certain fundamental theorems concerning probabilities, beginning with the calculation of the complement of an event, which represents the probability of an event not occurring.

**Proposition 12.1.2** For each event  $A$ , it holds true that  $P(A^c) = 1 - P(A)$ .

*Proof.* Sets  $A$  and  $A^c$  are disjoint, and their union  $A \cup A^c$  equals  $\Omega$ . By applying Axiom 3, we infer that  $P(A \cup A^c) = P(A) + P(A^c)$ , and by applying Axiom 2, we conclude that  $P(A \cup A^c) = P(\Omega) = 1$ . Therefore, we can assert that  $P(A) + P(A^c) = 1$ . ■

As a direct consequence of the aforementioned proposition, we can deduce the probability of the impossible event.

**Proposition 12.1.3** The probability of the impossible event equals zero, that is,  $P(\emptyset) = 0$ .

*Proof.* Since  $P(\emptyset) = 1 - P(\Omega) = 0$ . ■

As anticipated, sub-events (subsets) are associated with smaller probabilities than their corresponding events.

**Proposition 12.1.4** Given that  $A \subset B$ , it follows that  $P(A) \leq P(B)$ .

*Proof.* The event  $B$  can be dissected into the union of two disjoint events  $A$  and  $A^c \cap B$ . Consequently,  $P(B) = P(A) + P(A^c \cap B)$ , which combined with the notion that  $P(A^c \cap B) \geq 0$ , substantiates the proposition. ■

With these fundamental elements established, we can demonstrate that probabilities range between zero and one.

**Proposition 12.1.5** For each event  $A$ ,  $0 \leq P(A) \leq 1$ .

*Proof.* By virtue of Axiom 1 and the consideration that  $A \subset \Omega$  and consequently  $P(A) \leq P(\Omega) = 1$ . ■

Axiom 3 provides the means to compute the probability of the union of disjoint events, however, it does not extend to scenarios involving non-disjoint events. The succeeding proposition illustrates the method for computing the probability of the union of non-disjoint events.

**Proposition 12.1.6** For any two events  $A$  and  $B$ , it follows that  $P(A \cup B) = Pr(A) + Pr(B) - Pr(A \cap B)$ .

*Proof.* The union of sets  $A$  and  $B$  can be represented as the union of two disjoint sets  $A \cup B = B \cup (A \cap B^c)$ . Given Axiom 3, we ascertain that

$$P(A \cup B) = P(B) + P(A \cap B^c)$$

Similarly, the set  $A$  can be deconstructed as the union of the disjoint sets  $A = (A \cap B) \cup (A \cap B^c)$ . As a result, we get

$$P(A \cup B^c) = P(A) - P(A \cap B)$$

The combination of both expressions yields the desired result. ■

The following equation extends to the scenario of  $n$  events  $A_1, \dots, A_n$ , employing the principle of inclusion-exclusion (see Section 11.3):

$$\begin{aligned} P\left(\bigcup_{i=1}^n A_i\right) &= \sum_{i=1}^n P(A_i) - \sum_{i < j} P(A_i \cap A_j) + \sum_{i < j < k} P(A_i \cap A_j \cap A_k) - \\ &\quad - \sum_{i < j < k < l} P(A_i \cap A_j \cap A_k \cap A_l) + \dots + (-1)^{n+1} P(A_1 \cap A_2 \cap \dots \cap A_n) \end{aligned}$$

A probability mass function is characterized as a function that assigns to every possible event within a sample space its corresponding probability.

**Definition 12.1.3** Let  $(\Omega, \mathcal{A}, P)$  be a discrete probability space. A *probability mass function* is a real-valued function  $f : \mathcal{A} \rightarrow [0, 1]$  that it assigns  $f(A) = P(A)$  to every  $A \in \mathcal{A}$ .

In Example 12.1, we introduced a discrete probability space  $(\Omega, \mathcal{A}, P)$  comprising  $n$  equally probable elements. The probability mass function associated with this experiment is defined as  $f : \mathcal{A} \rightarrow [0, 1]$ , such that  $f(A) = d(A)/n$  for all  $A \in \mathcal{A}$ .

## 12.2 Conditional Probability

The principle of conditional probability is a cornerstone within the discipline of statistical learning. Conditional probability allow us refine the probability of an event based on new information or conditions. Under the axiomatization prescribed by Kolmogorov, conditional probability is introduced as a definition.

**Definition 12.2.1** Let  $A$  and  $B$  be two events such that  $P(B) > 0$ . The *conditional probability* of  $A$  given  $B$ , denoted by  $P(A | B)$ , is defined as:

$$P(A | B) = \frac{P(A \cap B)}{P(B)}$$

By virtue of satisfying the axioms, a conditional probability is, in itself, a probability. The conditional probability  $P(A | B)$  is undefined in instances where  $P(B) = 0$ .

Certain scholars posit that, given its pivotal role within probability theory, conditional probability ought to be an attribute that is logically deduced from the foundational axioms rather to be a definition. This perspective, naturally, necessitates an augmentation of Definition 12.1.2 with supplementary properties. Regrettably, there exists no agreed-upon method among mathematicians and philosophers regarding the manner in which this augmentation should be conducted.

The conventional interpretation of conditional probability posits it as the recalibrated probability of event  $A$  following the occurrence of event  $B$ . This perspective, however, potentially implies a sequential or even causative linkage between events  $B$  and  $A$ , a suggestion which may not necessarily hold validity.

■ **Example 12.2** Suppose we are playing a game with a standard deck of 52 cards, and we draw two cards. Let event  $A$  be "drawing at least one heart" and event  $B$  be "drawing at least one queen". These two events are dependent since the occurrence of event  $B$  affects the probability of event  $A$ . However, these two events are not temporally related because the draw of the card

happens at the same time - one event does not occur before the other. This example showcases the essence of dependency in probability theory without any temporal association between the events involved. ■

The probability of two events transpiring concurrently (although not necessarily contemporaneously, as previously discussed in Example 12.2), given their respective conditional probabilities, is encapsulated by the formula  $P(A \cap B) = P(A | B)P(B)$ . This equation offers perhaps a more intuitive comprehension of the conditional probability concept. Indeed, there exist a number of authors who advocate for this interpretation to form the basis of the definition of conditional probability, as opposed to the quotient method.

The extension of this formula to accommodate  $n$  events, termed the *multiplication rule*, is expressed as follows:

$$P(A_1 \cap A_2 \cap \dots \cap A_n) = P(A_1)P(A_2 | A_1) \dots P(A_n | A_1 \cap A_2 \cap \dots \cap A_{n-1}) \quad (12.1)$$

The notion of event independence holds significant importance in the realm of probability theory and statistical learning.

**Definition 12.2.2** Two events  $A$  and  $B$  are declared to be *independent* if  $P(A \cap B) = P(A)P(B)$ .

From an intuitive perspective, the events  $A$  and  $B$  are considered independent if witnessing the occurrence of event  $B$  does not influence the probability of event  $A$ . This characteristic can be logically inferred from the definition of independence.

I think it is better start with this intuition, and then provide the defintion of independence. See Casella.

**Proposition 12.2.1** Given two events  $A$  and  $B$  such that  $P(A) > 0$  and  $P(B) > 0$ ,  $A$  and  $B$  are independent if and only if  $P(A | B) = P(A)$ .

*Proof.* Assume  $A$  and  $B$  are independent, implying that  $P(A \cap B) = P(A)P(B)$ . Then,

$$P(A | B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A)P(B)}{P(B)} = P(A)$$

Proceeding from the assumption that  $P(A | B) = P(A)$ , and utilizing the multiplication rule, it follows that

$$P(A \cap B) = P(A | B)P(B) = P(A)P(B)$$



If  $P(A | B) = P(A)$  we have also that  $P(B | A) = P(B)$  which follow from the fact that the joint probability  $P(A \cap B)$  can be expressed as  $P(A)P(B)$ .

Similar to the case of conditional probability, certain authors posit that independence, as a foundational concept in probability theory, ought to be a logical extension of the axioms, rather than being imposed as a definition.

The principle of independence can be expanded to accommodate multiple events: the events  $A_1, \dots, A_n$  are deemed to be independent (or mutually independent) if for every subset  $A_{i_1}, \dots, A_{i_j}$  comprising  $j$  events ( $j = 2, 3, \dots, n$ ), it holds true that  $P(A_{i_1} \cap \dots \cap A_{i_j}) = P(A_{i_1}) \dots P(A_{i_j})$ .

Perhaps mention that it is not sufficient with pairwise independence.

**■ Example 12.3** A degree of confusion often arises regarding the distinction between mutually exclusive (or disjoint) events and independent events. For two mutually exclusive events  $A$  and  $B$ , the computation of the probability that  $A$  will transpire given  $B$  is somewhat nonsensical, since if  $B$  occurs,  $A$  is inherently impossible. Analogously, discussing the conditional probability that  $A$  will occur given  $B$  when the probability of  $B$  is zero is likewise flawed. However, as Definition 12.2.2 does not explicitly exclude the instance of  $A$  and  $B$  being mutually exclusive, we are compelled to conclude that two mutually exclusive events are independent if, and only if, the probability of at least one (or both) of them is zero. ■

An intriguing scenario arises when events  $A$  and  $B$  are not independent, yet attain independence contingent on the occurrence of another event  $C$ .

**■ Definition 12.2.3** Consider  $A$ ,  $B$  and  $C$  as events such that  $P(B \cap C) > 0$ .

$A$  and  $B$  are considered *conditionally independent* given  $C$  if  $P(A | B \cap C) = P(A | C)$ .

**■ Example 12.4** Consider the act of rolling two dice; it is reasonable to assert that the outcomes of the two dice are independent from each other. That is, observing the outcome of one die provides no insight into the outcome of the other die. However, suppose the first die results in a four, and a third event is introduced - that the sum of the outcomes is an odd number - then this additional piece of information narrows the potential outcomes for the second die to only odd numbers. This illustrates the point that two events can be independent, yet fail to maintain conditional independence. ■

The following theorem presents Bayes' rule, a fundamental principle underpinning a significant statistical learning technique known as Bayesian inference (see Section 16.1.2).

**Theorem 12.2.2 — Bayes' Theorem.** Let  $A$  and  $B$  be two events such

that  $P(B) \neq 0$ . Then we have that

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

In this context,  $P(A)$  is referred to as the *prior probability*, while  $P(A | B)$  is deemed the *posterior probability*.

*Proof.* As per the definition of conditional probability,  $P(A | B) = P(A \cap B) / P(B)$  (given  $P(B) \neq 0$ ) and  $P(B | A) = P(A \cap B) / P(A)$  (provided  $P(A) \neq 0$ ). By solving for  $P(A \cap B)$  and substituting into the previous expressions for  $P(A | B)$ , we arrive at the theorem. ■

As shown in the proof, Bayes' theorem is directly derived from the definition of conditional probability. However, despite its practical usefulness, this raises important philosophical concerns about its foundational basis. If conditional probability is accepted as a definition rather than something derived from a collection of axioms, Bayes' theorem inherits this uncertainty. It functions within the limits of this definition, underscoring the idea that it is a tool built on a potentially arbitrary foundation rather than one grounded in more fundamental principles.

Bayesian inference facilitates the computation of how our degree of certainty about event  $A$  (the prior probability  $P(A)$ ) evolves when we acquire supplementary evidence via the occurrence of event  $B$  (transforming into the posterior probability  $P(A | B)$ ).

■ **Example 12.5** Consider  $E$  to be a disease affecting one in every million people,  $P(E) = 1 \times 10^{-6}$ , and let  $+$  represent a test devised to detect the disease, with a failure rate of one in every thousand applications,  $P(+) | E) = 999/1000$ . We aim to determine the probability of disease presence if the test is positive  $P(E | +)$ . Upon employing Bayes' theorem, we find that:

$$P(E | +) = \frac{P(+) | E)P(E)}{P(+)} = \frac{P(+) | E)P(E)}{P(+) | E)P(E) + P(+) | E^c)P(E^c)} = 0.001$$

This implies that despite the test only failing once per thousand applications, it remains highly improbable that we have the disease following a positive result. This paradoxical outcome can be attributed to the higher probability of test failure  $10^{-3}$  compared to the likelihood of disease occurrence  $10^{-6}$ . Practically, this issue is circumvented by applying a second test to individuals who received a positive result, as the probability of disease presence following two positive results is 0.5 (under the assumption that the successive test repetitions are independent). ■

Bayes' theorem is most useful when the events involved are dependent and when new information about one event can update our understanding of the other event's probability.

■ **Example 12.6** Suppose you're drawing a single card from a standard deck of 52 playing cards. Let Event  $A$  be "drawing a red card" and Event  $B$  be "drawing a queen". In this context, the use of Bayes' theorem to compute  $P(A | B)$ , the probability of drawing a red card given that a queen has been drawn, would not yield a meaningful result because the event  $B$  provides no new information that would affect the probability of event  $A$ . ■

Bayes' theorem can indeed be extended to accommodate multiple events. Consider a set of events  $A_1, \dots, A_k$  such that  $P(A_j) > 0$  for all  $j$  in the range of 1 to  $k$ . Suppose these events constitute a partition of the sample space  $\Omega$ . Now, let  $B$  denote an event with the property that  $P(B) > 0$ . In such a context, it can be deduced for each  $i$  in the range of 1 to  $k$  that the conditional probability  $P(A_i | B)$  is given by the formula

$$P(A_i | B) = \frac{P(B | A_i) P(A_i)}{\sum_{j=1}^k P(B | A_j) P(A_j)}$$

This illustrates the capacity of Bayes' theorem to apply to a broader set of scenarios involving multiple events.

## 12.3 Random Variables

A random variable is a function that assigns a real number to each possible outcome of an experiment, providing a quantitative representation of the results. Unlike the framework defined by Kolmogorov axioms, where probabilities are assigned to events, random variables simplify the process by directly associating numerical values with outcomes. This alleviates the limitations of assigning probabilities to events, offering a more intuitive and manageable approach, and facilitating the discovery of their analytical properties. Such is the efficacy of random variables that a majority of statisticians primarily consider their investigations within the framework of random variables as opposed to probability spaces.

**Definition 12.3.1** Let  $(\Omega, \mathcal{A}, P)$  be a discrete probability space. A *random variable* is a function  $X : \Omega \rightarrow \mathbb{R}$  mapping from the set of outcomes  $\Omega$  to the real numbers  $\mathbb{R}$ . A random variable is discrete if its range  $\{x_1, x_2, \dots, x_i, \dots\}$  is finite or countably infinite.

Our primary focus here is on discrete random variables in discrete probability spaces. While it's possible to define a discrete random variable on a

non-discrete probability space by giving it a discrete range, this particular case is not covered in this book.

The terminology "random variable" might potentially lead to some confusion. Firstly, these are not variables in the conventional algebraic sense, but rather, they are functions. Secondly, they are not inherently random; it is the experiment that they represent which possesses randomness. Despite these points of potential confusion, we adhere to the established terminology.

Random variables are more useful when they represent characteristics of the experiment. For instance, if the sample space consists of a school's student body, a random variable could associate each student with their respective height. Random variables also enable us to redistribute outcomes of the sample space into new events. For example, if two dice are rolled, a random variable could represent the sum of the dice's outcomes.

It is crucial to remember that we possess the liberty to assign a random variable to any sample space, even when the assignment might not seem intuitively meaningful. As an illustration, one could assign a numerical value to each possible color in a deck of cards, draw two cards randomly, and sum the assigned numbers of these two cards. Although such a setup may not yield a significant interpretation, it is nonetheless possible to calculate probabilities based on this setup.

**Definition 12.3.2** Let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable, and let  $C \subset \mathbb{R}$  be a subset such that the set  $\{\omega \in \Omega : X(\omega) \in C\}$  constitutes an event. The probability of  $X$  belonging to  $C$ , expressed as  $P(X \in C)$ , is given by  $P(X \in C) = P(\{\omega \in \Omega : X(\omega) \in C\})$ .

The probability of a random variable  $X$  essentially configures a probability space over the line of real numbers, specifically over the range of  $X$ .

■ **Example 12.7** Let  $\Omega = \{1, 2, 3, 4, 5, 6\}$  be the sample space of tossing a die, and  $P$  a probability that assigns  $1/6$  to each single outcome in  $\Omega$ . Let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable defined as:

$$X(\omega) = \begin{cases} 0 & \text{if } \omega \text{ is even (2, 4, 6),} \\ 1 & \text{if } \omega \text{ is odd (1, 3, 5).} \end{cases}$$

This discrete random variable maps the outcomes of the die toss to either 0 (if the outcome is even) or 1 (if the outcome is odd). For  $C = \{0\}$ , the probability  $P(X \in C) = P(X = 0) = P(\{2, 4, 6\}) = 1/2$ . For  $C = \{1\}$  the probability  $P(X \in C) = P(X = 1) = P(\{1, 3, 5\}) = 1/2$ . Through this transformation, the original discrete probability space has been mapped to the real numbers using the discrete random variable  $X$ , establishing a new probability over  $X$ 's range. ■

Unless we say the contrary, we will assume that  $\{\omega \in \Omega : X(\omega) = x_i\}$  is an event for all the points that compose the range of  $X$ .

■ **Example 12.8**  $(\Omega, \mathcal{A}, P)$  be a discrete probability space, where  $\Omega = \{1, 2\}$ ,  $\mathcal{A} = \{\emptyset, \Omega\}$ , and  $P(\emptyset) = 0$  and  $P(\Omega) = 1$ . And let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable defined as  $X(1) = 1$  and  $X(2) = 2$ . We are interested in the probability  $P(X = 1)$ , but since  $\{\omega \in \Omega : X(\omega) = 1\} = \{1\}$  is not an event, such probability cannot be computed. ■

Definition 12.1.3 introduced the concept of probability mass function for probability spaces based on the probabilities of the events. Next definition extends the concept of probability mass function to random variables.

**Definition 12.3.3** Let  $X$  be a discrete random variable over a discrete probability space, and let  $\{x_1, x_2, \dots\}$  be the range of  $X$ . The *probability mass function* of the discrete random variable  $X$ , abbreviated as p.f., is defined as the function  $f : \text{range}(X) \rightarrow [0, 1]$  such that  $f(x_i) = P(X = x_i)$ .

The set of points for which the probability mass function is greater than zero, that is  $\{x : f(x) > 0\}$ , is called the *support* of the distribution of  $X$ .

It is possible for two random variables to have identical probability mass functions but to differ in significant ways.

■ **Example 12.9** Let  $\Omega = \{H, T\}$  be the sample space of tossing a coin, and  $P$  a probability that assigns  $1/2$  to each outcome in  $\Omega$ . Let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable defined as:  $X = 1$  if the coin shows Head and  $X = 0$  if coin shows Tail. The distribution of  $X$  is  $P(X = 1) = P(Y = 1) = 0.5$ . The discrete random variables of this example and Example 12.7 have same probability distribution even if they are different discrete random variables. ■

Given the probability mass function of a random variable, we can derive the probability of any subset of the real line.

**Proposition 12.3.1** Let  $X$  be a discrete random variable with probability mass function  $f$ . The probability of each subset  $C$  of the real line can be determined from the relation  $P(X \in C) = \sum_{x_i \in C} f(x_i)$

*Proof.* Considering that each outcome in the sample space is associated with exactly one value in the range  $\{x_1, x_2, \dots, x_i, \dots\}$  of  $X$ , we have that:

$$P(X \in C) = P(\{\omega \in \Omega : X(\omega) \in C\}) = \sum_{x_i \in C} P(X = x_i) = \sum_{x_i \in C} f(x_i)$$

■

Next proposition outlines a fundamental property, that the total sum of probabilities across all possible outcomes of a random variable is 1.

**Proposition 12.3.2** Let  $X$  be a discrete random variable with probability mass function  $f$ . If  $\{x_1, x_2, \dots\}$  is the range of  $X$ , then  $\sum_{i=1}^{\infty} f(x_i) = 1$ .

*Proof.* Considering that  $X$  is a total function, that each outcome in the sample space is associated with exactly one value in the range  $\{x_1, x_2, \dots\}$ , and given the axiomatic definition of probability we have that:

$$\begin{aligned}\sum_{i=1}^{\infty} f(x_i) &= f(x_1) + f(x_2) + \dots = P(X = x_1) + P(X = x_2) + \dots = \\ &= P(\{\omega \in \Omega : X(\omega) = x_1\}) + P(\{\omega \in \Omega : X(\omega) = x_2\}) + \dots = 1\end{aligned}$$

■

**Proposition 12.3.3** A function  $f(x)$  is a probability mass function of a random variable  $X$  if, and only if, i)  $f(x) \geq 0$  for all  $x$ , ii)  $\sum_{i=1}^{\infty} f(x_i) = 1$ .

*Proof.* TODO

■

The cumulative distribution function represents the probability that a random variable takes on a value less than or equal to a specific point.

**Definition 12.3.4** The *cumulative distribution function* (abbreviated c.d.f.)  $F$  of a discrete random variable  $X$  is the function  $F(x) = Pr(X \leq x)$  for all  $-\infty < x < \infty$

If  $X$  follows a distribution characterized by the probability mass function  $f(x)$ , its cumulative distribution function  $F(x)$  will exhibit the following behavior: at each distinct value  $x_i$  of  $X$ ,  $F(x)$  will display a jump equal to  $f(x_i)$ ; between these distinct values,  $F(x)$  remains unchanged.

The cumulative distribution function allows us to see how probabilities accumulate over the range of a random variable, offering insights into the overall distribution of the data.

■ **Example 12.10** Let's  $X$  be a discrete random variable that represents the grades of students in a class. Each grade is between 0 and 10. The probability that a student receives a grade of  $x$  is given by the probability mass function  $p(x)$ . The cumulative distribution function represents the probability that a randomly selected student scores  $x$  or less. For example, a value of  $F(7) = 0.6$  would mean that there's a 60% chance a student picked at random scored 7 or below.

■

The cumulative distribution function of a random variable is non-decreasing.

**Proposition 12.3.4** Let  $F(X)$  be the cumulative distribution function of a discrete random variable  $X$ . Then, if  $x_1 < x_2$  we have that  $F(x_1) \leq F(x_2)$ .

*Proof.* Given two values  $x_1$  and  $x_2$  where  $x_1 < x_2$ , the set of outcomes where  $X \leq x_1$  is a subset of the outcomes where  $X \leq x_2$ . Therefore, the probability of  $X$  taking a value less than or equal to  $x_1$  will be less than or equal to the probability of  $X$  taking a value less than or equal to  $x_2$ . Then  $F(x_1) \leq F(x_2)$ . ■

Next proposition delineates the asymptotic properties of the cumulative distribution function of a random variable, showcasing its bounds as we approach negative and positive infinity.

**Proposition 12.3.5** Let  $F(X)$  be the cumulative distribution function of a discrete random variable  $X$ . Then, we have that  $\lim_{x \rightarrow -\infty} F(x) = 0$  and that  $\lim_{x \rightarrow \infty} F(x) = 1$ .

*Proof.* As  $x$  tends to negative infinity, the probability that the discrete random variable  $X$  takes on a value less than or equal to this increasingly smaller  $x$  tends to zero. This is because there are fewer and fewer values (or none, depending on the specifics of the distribution) that  $X$  can assume which are less than this increasingly negative  $x$ . Therefore:

$$\lim_{x \rightarrow -\infty} F(x) = \lim_{x \rightarrow -\infty} P(X \leq x) = 0$$

As  $x$  tends to positive infinity, the probability that the discrete random variable  $X$  takes on a value less than or equal to this increasingly larger  $x$  approaches 1. This is because, given the unbounded increase of  $x$ , it encapsulates all possible values that  $X$  can take on. Therefore:

$$\lim_{x \rightarrow \infty} F(x) = \lim_{x \rightarrow \infty} P(X \leq x) = 1$$

■

The probability of  $X$  exceeding  $x$  is given by the complement of the cumulative distribution function at that point.

**Proposition 12.3.6** Let  $F(X)$  be the cumulative distribution function of a discrete random variable  $X$ . Then, for every  $x \in X$  we have that  $P(X > x) = 1 - F(x)$ .

*Proof.* The probability that  $X$  takes on a value greater than  $x$  plus the probability that  $X$  takes on a value less than or equal to  $x$  should sum up to 1. Given this, the probability that  $X$  takes a value greater than  $x$  is:

$$P(X > x) = 1 - P(X \leq x)$$

Using the definition of the cumulative distribution function, we get:

$$P(X > x) = 1 - F(x)$$

■

The following proposition establishes a relationship between the probabilities of a random variable  $X$  falling between two specific values and the corresponding differences in its cumulative distribution function values at those points.

**Proposition 12.3.7** Let  $F(X)$  be the cumulative distribution function of a discrete random variable  $X$ . Then, for all values  $x_1, x_2 \in X$  such that  $x_1 < x_2$  we have that  $P(x_1 < X \leq x_2) = F(x_2) - F(x_1)$

*Proof.* The probability that  $X$  is less than or equal to  $x_2$  is  $F(x_2)$ . From this, if we subtract the probability that  $X$  is less than or equal to  $x_1$ , which is  $F(x_1)$ , we'll get the probability that  $X$  falls strictly between  $x_1$  and  $x_2$ :

$$P(x_1 < X \leq x_2) = F(x_2) - F(x_1)$$

■

### 12.3.1 Multivariate Distributions

A multivariate probability distribution extends the concept of a probability distribution across multiple random variables, each with its own set of possible outcomes. Unlike univariate distributions that describe phenomena with a single random variable, multivariate distributions capture the relationships and dependencies between two or more variables. This allows for the exploration of complex phenomena where the outcome of interest is influenced by multiple factors simultaneously, providing insights into how these variables interact and impact the probability of various outcomes.

**Definition 12.3.5** Let  $X_1, X_2, \dots, X_n$  be  $n$  discrete random variables, where  $X_i : \Omega_i \rightarrow \mathbb{R}$  for  $i = 1, \dots, n$ . The *joint probability distribution* of  $X_1, X_2, \dots, X_n$  is defined as the collection of all probabilities of the form  $P((X_1, X_2, \dots, X_n) \in C)$  for all sets  $C \subset \mathbb{R}^n$  of real numbers such that  $\{(\omega_1, \omega_2, \dots, \omega_n) \in \Omega_1 \times \dots \times \Omega_n : (X_1(\omega_1), X_2(\omega_2), \dots, X_n(\omega_n)) \in C\}$  is an event.

The joint probability distribution of the random variables  $X_1, X_2, \dots, X_n$  defines a probability space in  $\mathbb{R}^n$ . If the random variables  $X_1, X_2, \dots, X_n$  each have a discrete distribution, then the joint distribution is also a discrete distribution.

**Definition 12.3.6** Let  $X_1, X_2, \dots, X_n$  be  $n$  discrete random variables over discrete probability spaces. The *joint probability mass function* of the discrete random variables  $X_1, X_2, \dots, X_n$  is defined as the function  $f : \text{range}(X_1) \times \dots \times \text{range}(X_n) \rightarrow [0, 1]$  such that  $f(x_1, \dots, x_n) = P(X_1 = x_1, \dots, X_n = x_n)$ .

■ **Example 12.11** A classic example of a bivariate discrete joint distribution involves rolling two six-sided dice. Let's define two discrete random variables:  $X_1$  is the outcome of the first die, and  $X_2$  is the outcome of the second die. Both  $X_1$  and  $X_2$  have a discrete uniform distribution over the set  $\{1, 2, 3, 4, 5, 6\}$ . The joint distribution of  $X_1$  and  $X_2$  describes the probability of each possible pair of outcomes when the two dice are rolled. The joint probability mass function  $f(x_1, x_2)$  for  $X_1$  and  $X_2$  can be expressed as:

$$f(x_1, x_2) = P(X_1 = x_1, X_2 = x_2) = \frac{1}{36}, \quad \text{for } x_1, x_2 \in \{1, 2, 3, 4, 5, 6\}$$

The joint distribution allows us to analyze the relationship between  $X_1$  and  $X_2$ . For instance, we can compute the probability that the sum of the two dice is equal to a certain number, or that one die shows a higher number than the other. ■

Before exploring the properties of multivariate random variables, we will introduce the concept of random vector. This concept simplifies notation and enhances clarity by grouping multiple random variables into a single entity.

**Definition 12.3.7** A *discrete random vector  $\mathbf{X}$*  is an ordered collection of  $n$  discrete random variables  $X_1, X_2, \dots, X_n$ , where  $X_i : \Omega_i \rightarrow \mathbb{R}$  for all  $i = 1, \dots, n$ .

Given the joint probability mass function of a vector of random variables, we can derive the probability of any subset within the multidimensional real space.

**Proposition 12.3.8** Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a discrete random vector with a joint probability mass function  $f$ . The probability of each subset  $C$  of the  $n$ -dimensional real space can be determined from the relation  $P(\mathbf{X} \in C) = \sum_{\mathbf{x} \in C} f(\mathbf{x})$ , where  $\mathbf{x} = (x_1, \dots, x_n)$ .

*Proof.* Considering that each outcome in the sample space is associated with exactly one value in the range of  $\mathbf{X}$ , we have that:

$$P(\mathbf{X} \in C) = P(\{\omega \in \Omega : \mathbf{X}(\omega) \in C\}) = \sum_{\mathbf{x} \in C} P(\mathbf{X} = \mathbf{x}) = \sum_{\mathbf{x} \in C} f(\mathbf{x})$$



The next proposition outlines a fundamental property, that the total sum of probabilities across all possible outcomes of a vector of random variables is 1.

**Proposition 12.3.9** Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a discrete random vector with a joint probability mass function  $f$ . If the range of  $\mathbf{X}$  is represented as a set of vectors  $\mathbf{x} = (x_1, \dots, x_n)$ , then  $\sum_{\mathbf{x}} f(\mathbf{x}) = 1$ .

*Proof.* Considering that  $\mathbf{X}$  is a total function, that each outcome in the sample space is associated with exactly one value in the range of  $\mathbf{X}$ , and given the axiomatic definition of probability we have that:

$$\sum_{\mathbf{x}} f(\mathbf{x}) = \sum_{\mathbf{x}} P(\mathbf{X} = \mathbf{x}) = P(\{\omega \in \Omega_1 : X_1(\omega) = x_1\}) + P(\{\omega \in \Omega_2 : X_2(\omega) = x_2\}) + \dots$$

■

A particular interesting case of multivariate distribution is given by the sum of  $n$  random variables. This is a little bit confusing scenario, since we are not adding  $n$  probability distributions, as the notation  $X_1 + \dots + X_n$  might suggest. Instead, we are defining a new random variable over the cartesian product of the original sample spaces.

**Definition 12.3.8** Let  $X_1, X_2, \dots, X_n$  be  $n$  discrete random variables, where  $X_i : \Omega_i \rightarrow \mathbb{R}$  for all  $i = 1, \dots, n$ . The sum distribution of  $X_1, X_2, \dots, X_n$ , denoted by  $X_1 + \dots + X_n$ , is defined as the discrete random variable  $X + \dots + X_n : \Omega_1 \times \dots \times \Omega_n \rightarrow \mathbb{R}$  that assigns to each  $(\omega_1, \omega_2, \dots, \omega_n) \in \Omega_1 \times \dots \times \Omega_n$  the number  $X_1(\omega_1) + X_2(\omega_2) + \dots + X_n(\omega_n)$ .

The concept of sum of  $n$  random variables will be applied in the law of large numbers (see Theorem 12.6.4), one of the most important theorems in probability theory.

■ **Example 12.12** Let  $X : \Omega \rightarrow \mathbb{R}$  be the discrete random variable representing the outcome of rolling a six-sided dice. The sum distribution corresponding to rolling three times a dice, denoted by  $S = X + X + X$ , is defined as the discrete random variable  $S : \Omega \times \Omega \times \Omega \rightarrow \mathbb{R}$  that assigns to each  $(\omega_1, \omega_2, \omega_3) \in \Omega \times \Omega \times \Omega$  the number  $X(\omega_1) + X(\omega_2) + X(\omega_3)$ . ■

### 12.3.2 Marginal Probability Mass Function

Given a multivariate discrete probability mass function, the marginal probability mass function of a subset of variables is derived by summing the joint probability mass function over all possible values of the remaining variables. This process essentially "marginalizes" out the variables not of interest, allowing focus on the probability mass function of a single variable or a subset of variables within the multivariate context.

**Definition 12.3.9** Let  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  be an  $n$ -dimensional random vector with joint probability mass function  $f_{\mathbf{X}}(x_1, x_2, \dots, x_n)$ , partition  $\mathbf{X}$  into two subvectors:  $\mathbf{Y} = (Y_1, Y_2, \dots, Y_k)$ , a  $k$ -dimensional random vector consisting of  $k$  discrete random variables selected from  $\mathbf{X}$ , and  $\mathbf{Z} = (Z_1, Z_2, \dots, Z_{n-k})$ , the remaining  $(n - k)$  discrete random variables of  $\mathbf{X}$ . The *marginal probability mass function*  $f_{\mathbf{Y}}$  of the random vector  $\mathbf{Y}$  is obtained by summing  $f_{\mathbf{X}}$  over all possible values of the variables in  $\mathbf{Z}$ . That is, for any specific values  $(y_1, y_2, \dots, y_k)$  of  $\mathbf{Y}$ ,

$$f_{\mathbf{Y}}(y_1, y_2, \dots, y_k) = \sum_{z_1} \sum_{z_2} \dots \sum_{z_{n-k}} f_{\mathbf{X}}(x_1, x_2, \dots, x_n),$$

where in each term of the sum,  $x_i = y_i$  for  $i$  corresponding to variables in  $\mathbf{Y}$ , and  $x_j = z_j$  for  $j$  corresponding to variables in  $\mathbf{Z}$ .

This definition underscores the process of marginalization in a discrete setting, which is key to understanding and analyzing the behavior of specific variables within a larger multivariate framework.

■ **Example 12.13** Consider two discrete random variables  $X$  and  $Y$ , each taking values in  $\{0, 1\}$ . Suppose their joint probability mass function is given by:

| $X \setminus Y$ | 0   | 1   |
|-----------------|-----|-----|
| 0               | 0.1 | 0.3 |
| 1               | 0.2 | 0.4 |

To find the marginal probability mass function of  $X$ , we sum over all possible values of  $Y$ :

$$f_X(0) = f_{X,Y}(0,0) + f_{X,Y}(0,1) = 0.1 + 0.3 = 0.4,$$

$$f_X(1) = f_{X,Y}(1,0) + f_{X,Y}(1,1) = 0.2 + 0.4 = 0.6.$$

■

While the marginal probability mass functions of the discrete random variables  $X_1, \dots, X_n$  can be obtained from their joint probability mass function by summing over the range of the other variables, the reverse process is not straightforward. Specifically, reconstructing the joint probability mass function of  $X_1, \dots, X_n$  from their marginal probability mass functions alone is not feasible without extra information about the dependence between  $X_1, \dots, X_n$ . This limitation arises because marginal probability mass functions encapsulate only the individual behavior of each variable, omitting details about how the variables interact or are related.

■ **Example 12.14** Let  $X$  and  $Y$  be discrete random variables, each taking values in  $\{0, 1\}$ , with marginal probability mass functions:

$$\begin{aligned} f_X(0) &= 0.5, & f_X(1) &= 0.5, \\ f_Y(0) &= 0.5, & f_Y(1) &= 0.5. \end{aligned}$$

Without additional information about the dependence structure between  $X$  and  $Y$ , the joint probability mass function cannot be reconstructed from the marginals alone. ■

A random vector  $\mathbf{X}$  is considered independent if the occurrence of an event associated with any of the random variables of  $\mathbf{X}$  does not influence the probability of an event associated with any other variable in  $\mathbf{X}$ . Independence among these variables indicates that there is no association or correlation among them, implying that knowing the outcome of one provides no information about the outcomes of the others.

**Definition 12.3.10** An  $n$ -dimensional random vector  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  is said to consist of *independent random variables* if for every choice of subsets  $A_1, A_2, \dots, A_n$  of  $\mathbb{R}$  such that  $\{X_i \in A_i\}$  is an event for  $i = 1, 2, \dots, n$ , the joint probability of these events can be expressed as the product of their individual probabilities:

$$P(X_1 \in A_1, X_2 \in A_2, \dots, X_n \in A_n) = P(X_1 \in A_1) P(X_2 \in A_2) \dots P(X_n \in A_n).$$

The concept of independence for a random vector  $\mathbf{X}$  simplifies the computation and understanding of joint probability distributions, particularly in complex problems involving multiple variables. It allows the joint probability distribution of the vector  $\mathbf{X}$  to be expressed as the product of the individual marginal distributions of  $X_1, X_2, \dots, X_n$ .

**Proposition 12.3.10** Let  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  be an  $n$ -dimensional random vector with joint probability mass function  $f_{\mathbf{X}}$  and marginal probability mass functions  $f_{X_1}, f_{X_2}, \dots, f_{X_n}$ . The random variables  $X_1, X_2, \dots, X_n$  are independent if and only if for every  $(x_1, x_2, \dots, x_n)$  in the support of  $\mathbf{X}$ , we have:

$$f_{\mathbf{X}}(x_1, x_2, \dots, x_n) = f_{X_1}(x_1) f_{X_2}(x_2) \dots f_{X_n}(x_n).$$

*Proof.* Assume that  $X_1, X_2, \dots, X_n$  are independent. Then, for any values  $x_1, x_2, \dots, x_n$ , we have:

$$\begin{aligned} f_{\mathbf{X}}(x_1, x_2, \dots, x_n) &= P(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) \\ &= P(X_1 = x_1) P(X_2 = x_2) \dots P(X_n = x_n) \\ &= f_{X_1}(x_1) f_{X_2}(x_2) \dots f_{X_n}(x_n). \end{aligned}$$

Conversely, assume that for all  $x_1, x_2, \dots, x_n$ ,

$$f_{\mathbf{X}}(x_1, x_2, \dots, x_n) = f_{X_1}(x_1)f_{X_2}(x_2)\dots f_{X_n}(x_n).$$

Then, for any subsets  $A_1, A_2, \dots, A_n$  of  $\mathbb{R}$ , we have:

$$\begin{aligned} P(X_1 \in A_1, X_2 \in A_2, \dots, X_n \in A_n) &= \sum_{(x_1, x_2, \dots, x_n) \in A_1 \times A_2 \times \dots \times A_n} f_{\mathbf{X}}(x_1, x_2, \dots, x_n) \\ &= \sum_{x_1 \in A_1} \sum_{x_2 \in A_2} \dots \sum_{x_n \in A_n} f_{X_1}(x_1)f_{X_2}(x_2)\dots f_{X_n}(x_n) \\ &= \left( \sum_{x_1 \in A_1} f_{X_1}(x_1) \right) \left( \sum_{x_2 \in A_2} f_{X_2}(x_2) \right) \dots \left( \sum_{x_n \in A_n} f_{X_n}(x_n) \right) \\ &= P(X_1 \in A_1)P(X_2 \in A_2)\dots P(X_n \in A_n). \end{aligned}$$

This equality holds for all subsets  $A_1, A_2, \dots, A_n$ , which implies that  $X_1, X_2, \dots, X_n$  are independent. ■

### 12.3.3 Conditional Probability Mass Function

The concept of conditional probability mass function offers a way to understand the probability of an event given that another event has occurred. In the particular case of discrete random variables, the conditional probability mass function of  $Y$  given  $X = x$  describes the probability mass function of  $Y$  under the condition that  $X$  takes a specific value  $x$ . This concept is pivotal for dissecting the interdependencies between discrete random variables, allowing us to refine our probability assessments based on new information.

**Definition 12.3.11** Let  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  be an  $n$ -dimensional random vector with joint probability mass function  $f_{\mathbf{X}}(x_1, x_2, \dots, x_n)$ . Partition  $\mathbf{X}$  into two subvectors:  $\mathbf{Y} = (Y_1, Y_2, \dots, Y_k)$ , a  $k$ -dimensional random vector consisting of  $k$  discrete random variables selected from  $\mathbf{X}$ , and  $\mathbf{Z} = (Z_1, Z_2, \dots, Z_{n-k})$ , the remaining  $n - k$  discrete random variables of  $\mathbf{X}$ . Let  $f_{\mathbf{Z}}$  represent the marginal probability mass function of  $\mathbf{Z}$  across its  $n - k$  dimensions. Provided that for any vector  $\mathbf{z} \in \mathbb{R}^{n-k}$  the condition  $f_{\mathbf{Z}}(\mathbf{z}) > 0$  holds, the *conditional probability mass function*  $f_{\mathbf{Y}|\mathbf{Z}}$  for  $\mathbf{Y}$  given  $\mathbf{Z} = \mathbf{z}$  is defined as:

$$f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} | \mathbf{z}) = \frac{f_{\mathbf{X}}(\mathbf{y}, \mathbf{z})}{f_{\mathbf{Z}}(\mathbf{z})}$$

Next example demonstrates how to compute the conditional probability mass function of  $Y$  given  $X$ .

■ **Example 12.15** Consider two discrete random variables  $X$  and  $Y$ , each taking values in  $\{0, 1\}$ , with the following joint probability mass function:

| $X \setminus Y$ | 0   | 1   |
|-----------------|-----|-----|
| 0               | 0.3 | 0.2 |
| 1               | 0.1 | 0.4 |

We can compute the marginal probability mass function of  $X$ :

$$f_X(0) = f_{X,Y}(0,0) + f_{X,Y}(0,1) = 0.3 + 0.2 = 0.5,$$

$$f_X(1) = f_{X,Y}(1,0) + f_{X,Y}(1,1) = 0.1 + 0.4 = 0.5.$$

Suppose we want to find the conditional probability mass function of  $Y$  given  $X = 0$ , denoted  $f_{Y|X}(y | x)$ . For  $x = 0$  we have:

$$f_{Y|X}(0 | 0) = \frac{f_{X,Y}(0,0)}{f_X(0)} = \frac{0.3}{0.5} = 0.6,$$

$$f_{Y|X}(1 | 0) = \frac{f_{X,Y}(0,1)}{f_X(0)} = \frac{0.2}{0.5} = 0.4.$$

For  $x = 1$  we have:

$$f_{Y|X}(0 | 1) = \frac{f_{X,Y}(1,0)}{f_X(1)} = \frac{0.1}{0.5} = 0.2,$$

$$f_{Y|X}(1 | 1) = \frac{f_{X,Y}(1,1)}{f_X(1)} = \frac{0.4}{0.5} = 0.8.$$

■

Next proposition generalizes the multiplication rule (see Equation 12.1) by combining marginal and conditional probability mass functions to derive the joint probability mass function for any configuration of discrete random variables within a random vector, accounting for the complex dependencies and interactions among multiple variables.

**Proposition 12.3.11** Let  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ ,  $f_{\mathbf{X}}(\mathbf{x})$ ,  $f_{\mathbf{Z}}(\mathbf{z})$ , and  $f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} | \mathbf{z})$  be defined as in Definition 12.3.11. Then, for each  $\mathbf{z}$  such that  $f_{\mathbf{Z}}(\mathbf{z}) > 0$  and each possible value of  $\mathbf{y}$ , the joint probability mass function is given by:

$$f_{\mathbf{X}}(\mathbf{x}) = f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} | \mathbf{z})f_{\mathbf{Z}}(\mathbf{z})$$

where  $\mathbf{x} = (\mathbf{y}, \mathbf{z})$  represents a specific instantiation of the random vector  $\mathbf{X}$ .

*Proof.* By the definition of the conditional probability mass function:

$$f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} | \mathbf{z}) = \frac{f_{\mathbf{X}}(\mathbf{y}, \mathbf{z})}{f_{\mathbf{Z}}(\mathbf{z})}, \quad \text{for } f_{\mathbf{Z}}(\mathbf{z}) > 0.$$

Rearranging this equation gives:

$$f_{\mathbf{X}}(\mathbf{y}, \mathbf{z}) = f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} \mid \mathbf{z}) f_{\mathbf{Z}}(\mathbf{z}).$$

Since  $\mathbf{x} = (\mathbf{y}, \mathbf{z})$ , we have:

$$f_{\mathbf{X}}(\mathbf{x}) = f_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y} \mid \mathbf{z}) f_{\mathbf{Z}}(\mathbf{z}).$$

■

Similarly, the conditional probability mass function of  $\mathbf{Z}$  given  $\mathbf{Y} = \mathbf{y}$ , denoted as  $f_{\mathbf{Z}|\mathbf{Y}}(\mathbf{z} \mid \mathbf{y})$ , can be combined with the marginal probability mass function of  $\mathbf{Y}$ ,  $f_{\mathbf{Y}}(\mathbf{y})$ , to yield the same joint probability mass function of  $\mathbf{X}$ :

$$f_{\mathbf{X}}(\mathbf{x}) = f_{\mathbf{Z}|\mathbf{Y}}(\mathbf{z} \mid \mathbf{y}) f_{\mathbf{Y}}(\mathbf{y}).$$

Bayes' theorem (see Theorem 12.2.2) provides a way to update our probability estimates for a hypothesis given new evidence. For random vectors, the theorem can be generalized to accommodate the multi-dimensional nature of the variables involved.

**Theorem 12.3.12 — Bayes' Theorem for Random Vectors.** Let  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  and  $\mathbf{Y} = (Y_1, Y_2, \dots, Y_m)$  be two random vectors representing different sets of discrete random variables. Suppose we are interested in the conditional probability distribution of  $\mathbf{X}$  given observed values of  $\mathbf{Y}$ , denoted as  $\mathbf{y} = (y_1, y_2, \dots, y_m)$ . The generalized Bayes' theorem for random vectors can be stated as:

$$P(\mathbf{X} = \mathbf{x} \mid \mathbf{Y} = \mathbf{y}) = \frac{P(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x}) P(\mathbf{X} = \mathbf{x})}{P(\mathbf{Y} = \mathbf{y})}$$

where  $P(\mathbf{Y} = \mathbf{y})$  is the marginal probability of  $\mathbf{Y}$ , which can also be expressed using the law of total probability as:

$$P(\mathbf{Y} = \mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x}) P(\mathbf{X} = \mathbf{x})$$

for discrete random vectors.

*Proof.* By the definition of conditional probability:

$$P(\mathbf{X} = \mathbf{x} \mid \mathbf{Y} = \mathbf{y}) = \frac{P(\mathbf{X} = \mathbf{x}, \mathbf{Y} = \mathbf{y})}{P(\mathbf{Y} = \mathbf{y})}.$$

Also, we can express the joint probability as:

$$P(\mathbf{X} = \mathbf{x}, \mathbf{Y} = \mathbf{y}) = P(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x}) P(\mathbf{X} = \mathbf{x}).$$

Substituting back into the first equation:

$$P(\mathbf{X} = \mathbf{x} \mid \mathbf{Y} = \mathbf{y}) = \frac{P(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x})P(\mathbf{X} = \mathbf{x})}{P(\mathbf{Y} = \mathbf{y})}.$$

■

This generalized form of Bayes' theorem allows us to update our belief about the probability distribution of a set of discrete random variables  $\mathbf{X}$  based on new information encapsulated in another set of discrete random variables  $\mathbf{Y}$ . It emphasizes the interplay between the prior information we have about  $\mathbf{X}$ , the likelihood of observing  $\mathbf{Y} = \mathbf{y}$  given  $\mathbf{X} = \mathbf{x}$ , and the evidence provided by the actual observation of  $\mathbf{Y} = \mathbf{y}$ .

Building on the familiar concept of independence between discrete random variables (see Definition 12.3.10), an important extension is the idea of conditional independence. This concept comes into play when the independence of a set of discrete random variables is considered in the context of being conditioned on another set of variables.

**Definition 12.3.12** Let  $\mathbf{Z}$  be a random vector with joint probability mass function  $f_{\mathbf{Z}}(\mathbf{z})$ . The variables of the random vector  $\mathbf{X} = (X_1, \dots, X_n)$  are *conditionally independent* given  $\mathbf{Z}$  if, for all  $\mathbf{z}$  such that  $f_{\mathbf{Z}}(\mathbf{z}) > 0$ , we have:

$$f_{\mathbf{X}|\mathbf{Z}}(\mathbf{x} \mid \mathbf{z}) = \prod_{i=1}^n f_{X_i|\mathbf{Z}}(x_i \mid \mathbf{z}),$$

where  $f_{\mathbf{X}|\mathbf{Z}}(\mathbf{x} \mid \mathbf{z})$  is the conditional probability mass function of  $\mathbf{X}$  given  $\mathbf{Z} = \mathbf{z}$ , and  $f_{X_i|\mathbf{Z}}(x_i \mid \mathbf{z})$  is the conditional probability mass function of  $X_i$  given  $\mathbf{Z} = \mathbf{z}$ .

## 12.4 Characterizing Distributions

A *measure of central tendency* is a number derived from a probability distribution, intended as a summary of that distribution. The most common measures of central tendency in use are the expected value and the median. Each of these measures provides a different approach to characterize distributions. It is also common to use *metrics of dispersion* to describe the variability of a distribution around the measures of centrality. We will review two metrics of dispersion, the variance and the standard deviation. The metrics of dispersion can also be used in case of bivariate distributions, under the names of covariance and correlation, to measure the *statistical relationship* between two discrete random variables. All these measures allow us to summarize and compare distributions.

**12.4.1****Measures of Central Tendency**

The most common measures of central tendency in use to characterize probability distributions are the expected value and the median.

**Expected Value**

The expected value of a discrete random variable is computed as the weighted average of all possible values of the variable, where weights are given by the probabilities of the outcomes.

**Definition 12.4.1** Let  $X$  be a discrete random variable whose probability mass function is  $f$ . The *expected value* of  $X$ , denoted by  $E(X)$ , is defined as:

$$E(X) = \sum_x xf(x)$$

The definition of expected value considers only the distribution of the discrete random variable, not the original outcomes. Thus, two different discrete random variables with the same distribution will have equal expected values, even if the underlying discrete probability spaces are different.

The term "expected value" can be somewhat misleading because, for many discrete distributions, the expected value is not necessarily one of the possible values. For instance, when rolling a six-sided die, the expected value is 3.5, which is not an actual outcome of the roll. This counterintuitive aspect of the concept of expected value has led to considerable confusion in scientific research.

A drawback of the expected value is that it can be greatly affected by a small change in the probability assigned to a large value of  $X$ .

**Example 12.16** Consider a company with a population of 100 employees, and we define a discrete random variable based on their salaries. Let's  $X = \{300, 6000\}$  with probabilities of 99/100 and 1/100 respectively. The expected salary is calculated as:

$$E(X) = (300 \times 99/100) + (6000 \times 1/100) = 357$$

Now, suppose that one of the base employees is promoted to the executive level with the same salary of \$6000. The recalculated expected salary would be:

$$E(X) = (300 \times 98/100) + (6000 \times 2/100) = 414$$

This example shows how just changing the salary of one single person can increase the expected salary of the company by more than 13%. ■

The expected value of the linear combination of  $n$  discrete random variables is the linear combination of their respective expected values.

**Proposition 12.4.1** Let  $X_1, \dots, X_n$  be  $n$  independent discrete random variables with expectations  $E(X_i)$ , and let  $a_1, \dots, a_n$  and  $b$  constants, then

$$E(a_1X_1 + \dots + a_nX_n + b) = a_1E(X_1) + \dots + a_nE(X_n) + b$$

*Proof.* We have that

$$\begin{aligned} E(a_1X_1 + \dots + a_nX_n + b) &= \sum_{x_1} \dots \sum_{x_n} (a_1x_1 + \dots + a_nx_n + b) f(x_1, \dots, x_n) = \\ &= \sum_{x_1} \dots \sum_{x_n} a_1x_1 f(x_1, \dots, x_n) + \dots + \sum_{x_1} \dots \sum_{x_n} a_nx_n f(x_1, \dots, x_n) + \sum_{x_1} \dots \sum_{x_n} b f(x_1, \dots, x_n) = \\ &= \sum_{x_1} a_1x_1 f(x_1) + \dots + \sum_{x_n} a_nx_n f(x_n) + b = a_1 \sum_{x_1} x_1 f(x_1) + \dots + a_n \sum_{x_n} x_n f(x_n) + b = \\ &= a_1E(X_1) + \dots + a_nE(X_n) + b \quad (12.2) \end{aligned}$$

■

The expected value of the product of  $n$  independent discrete random variables is equal to the product of the individual expected values.

**Proposition 12.4.2** Let  $X_1, \dots, X_n$  be  $n$  independent discrete random variables with expectations  $E(X_i)$ , then:

$$E\left(\prod_{i=1}^n X_i\right) = \prod_{i=1}^n E(X_i)$$

*Proof.* We have that

$$\begin{aligned} E(X_1 \cdot \dots \cdot X_n) &= \sum_{x_1} \dots \sum_{x_n} (x_1 \cdot \dots \cdot x_n) f(x_1, \dots, x_n) = \\ &= \sum_{x_1} \dots \sum_{x_n} x_1 f(x_1, \dots, x_n) \cdot \dots \cdot \sum_{x_1} \dots \sum_{x_n} x_n f(x_1, \dots, x_n) = \\ &= \sum_{x_1} x_1 f(x_1) \cdot \dots \cdot \sum_{x_n} x_n f(x_n) = E(X_1) \cdot \dots \cdot E(X_n) \quad (12.3) \end{aligned}$$

■

The expected value of the product of non-independent discrete random variables is not necessarily equal to the product of their individual expected values.

### The Median

The median of a discrete random variable is a measure of central tendency that represents the point that separates the higher half from the lower half of a probability distribution.

**Definition 12.4.2** Let  $X$  be a discrete random variable. The *median* of  $X$ , denoted by  $m$ , is the value that satisfies:

$$\Pr(X \leq m) \geq \frac{1}{2} \quad \text{and} \quad \Pr(X \geq m) \geq \frac{1}{2}$$

The definition of the median considers the distribution of the discrete random variable, ensuring that at least half of the probability mass lies on either side of the median.

The "median" is often more intuitive than "expected value" because it is always one of the possible values, although in the case of an even number of outcomes it is customary to use the average of the two middle values.

The median is a robust measure of central tendency, especially useful when the data has outliers or is skewed.

■ **Example 12.17** Consider the company from Example 12.16. The median of the salaries is calculated as the value  $m$  for which  $\Pr(X \leq m) \geq \frac{1}{2}$ . In this case,  $m = 300$ . Now, suppose that one of the base employees is promoted to the executive level with the same salary of \$6000. The recalculated median would still be  $m = 300$ . This example shows that the median is a more robust measure than the expected value in the presence of outliers. ■

## 12.4.2 Measures of Dispersion

The most common measures of dispersion in use to characterize probability distributions are the variance and its squared root, called standard deviation.

### The Variance

The variance of a discrete random variable is a measure of the spread or dispersion of the possible values of the variable around the expected value.

**Definition 12.4.3** Let  $X$  be a discrete random variable with expected value  $E(X)$  and probability mass function  $f$ . The *variance* of  $X$ , denoted by  $\text{Var}(X)$ , is defined as:

$$\text{Var}(X) = E[(X - E(X))^2] = \sum_x (x - E(X))^2 f(x)$$

Variance considers the distribution of the discrete random variable and provides a measure of how much the values differ from the expected value. For instance, if all possible values of a discrete random variable are the same,

the variance is zero.

As it was the case of the expected value, a drawback of the variance is that it can be influenced significantly by outliers because it involves squaring the deviations from the mean.

■ **Example 12.18** Consider the company from Example 12.16, with an expected salary of  $E(X) = 357$ : We calculate the variance as:

$$\text{Var}(X) = (300 - 357)^2 \times 99/100 + (6000 - 357)^2 \times 1/100 = 3218250$$

■

Next proposition states that for a linear combination of independent discrete random variables, the variance of the combination is the weighted sum of the variances of the individual variables, where the weights are the squares of the coefficients in the linear combination.

**Proposition 12.4.3** Let  $X_1, \dots, X_n$  be  $n$  independent discrete random variables with finite expected values, and  $a_1, \dots, a_n$  and  $b$  be arbitrary constants, then:

$$\text{Var}(a_1X_1 + \dots + a_nX_n + b) = a_1^2\text{Var}(X_1) + \dots + a_n^2\text{Var}(X_n)$$

*Proof.* Let  $Y = a_1X_1 + \dots + a_nX_n + b$ . The variance of  $Y$  is given by:

$$\text{Var}(Y) = \text{Var}(a_1X_1 + \dots + a_nX_n + b)$$

Since variance is unaffected by the addition of a constant, we can ignore  $b$ :

$$\text{Var}(Y) = \text{Var}(a_1X_1 + \dots + a_nX_n)$$

Now, using the linearity of variance for independent discrete random variables, we have:

$$\text{Var}(a_1X_1 + \dots + a_nX_n) = \text{Var}(a_1X_1) + \dots + \text{Var}(a_nX_n)$$

Next, we use the property that for any discrete random variable  $X_i$  and constant  $a_i$ ,  $\text{Var}(a_iX_i) = a_i^2\text{Var}(X_i)$ :

$$\text{Var}(a_1X_1) + \dots + \text{Var}(a_nX_n) = a_1^2\text{Var}(X_1) + \dots + a_n^2\text{Var}(X_n)$$

Therefore, we have:

$$\text{Var}(a_1X_1 + \dots + a_nX_n + b) = a_1^2\text{Var}(X_1) + \dots + a_n^2\text{Var}(X_n)$$

■

A key relationship in probability theory is the formula that expresses variance in terms of expected values. This relationship not only simplifies calculations but also provides deeper insight into the nature of variance as a measure of risk and variability.

**Proposition 12.4.4** Let  $X$  be a discrete random variable with expected value  $E(X)$ . Then, the variance of  $X$  is given by:

$$\text{Var}(X) = E(X^2) - [E(X)]^2$$

*Proof.* To prove this proposition, we start from the definition of variance:

$$\text{Var}(X) = E[(X - E(X))^2]$$

Expanding the square inside the expectation yields:

$$\text{Var}(X) = E[X^2 - 2X \cdot E(X) + (E(X))^2]$$

Applying the linearity of expectation, this becomes:

$$\text{Var}(X) = E(X^2) - 2E(X)E(X) + E((E(X))^2)$$

Since  $E(X)$  is a constant, the expectation of a constant is the constant itself, so:

$$E((E(X))^2) = (E(X))^2$$

Thus, the expression simplifies to:

$$\text{Var}(X) = E(X^2) - 2(E(X))^2 + (E(X))^2$$

Simplifying further, we cancel out terms:

$$\text{Var}(X) = E(X^2) - (E(X))^2$$



### Standard Deviation

The standard deviation is also a statistical measure that quantifies the dispersion or variability in a set of data values. However, unlike variance, which squares the differences from the mean, resulting in units that are the square of the original data units, the standard deviation is the square root of the variance. This adjustment is crucial because it brings the units back to the original units of the data, making the measure more intuitive and directly interpretable.

**Definition 12.4.4** Let  $X$  be a discrete random variable with expected value  $E(X)$ , variance  $Var(X)$  and probability mass function  $f$ . The *standard deviation* of  $X$ , denoted by  $\sigma$ , is defined as:

$$\sigma = \sqrt{Var(X)} = \sqrt{\sum_x (x - E(X))^2 f(x)}$$

The standard deviation provides a numerical summary of how scattered the values of  $X$  are around the mean. A smaller standard deviation indicates that the values tend to be closer to the mean (less spread out), whereas a larger standard deviation indicates that the values are more spread out from the mean.

**Proposition 12.4.5** Let  $X$  be a discrete random variable with standard deviation  $\sigma_X$ , and  $a$  and  $b$  be arbitrary constants, then the standard deviation  $\sigma_Y$  of the discrete random variable  $Y = aX + b$  is  $|a|$  times the standard deviation of  $X$ , i.e.,  $\sigma_Y = |a|\sigma_X$ .

*Proof.* The variance of  $Y$  can be calculated as:

$$\sigma_Y^2 = \sum_x (ax + b - (aE(X) + b))^2 f(x) = a^2 \sum_x (x - E(X))^2 f(x) = a^2 \sigma_X^2$$

Taking the square root of both sides, we get:

$$\sigma_Y = \sqrt{a^2 \sigma_X^2} = |a|\sigma_X.$$

■

### 12.4.3 Measures of Statistical Relationship

Covariance and correlation are measures of the relationship between two variables. Covariance indicates the direction of the linear relationship, showing whether the variables tend to increase or decrease together, but it doesn't standardize the scale of the data. Correlation, on the other hand, standardizes this relationship, giving a value between -1 and 1, where -1 means a perfect negative linear relationship, 1 means a perfect positive linear relationship, and 0 indicates no linear relationship.

#### Covariance

Covariance is a measure used to determine the degree to which two discrete random variables  $X$  and  $Y$  vary together. It gives an indication of the direction of the linear relationship between the variables.

**Definition 12.4.5** Let  $X$  and  $Y$  be two discrete random variables with finite expected values  $E(X)$  and  $E(Y)$  respectively. The *covariance* of  $X$  and  $Y$ , denoted by  $\text{Cov}(X, Y)$  is defined as

$$\text{Cov}(X, Y) = E[(X - E(X))(Y - E(Y))]$$

The sign of the covariance reveals the direction of the relationship between two variables. A positive covariance means that as  $X$  increases,  $Y$  also tends to increase, while a negative covariance indicates that as  $X$  increases,  $Y$  tends to decrease. A covariance of zero suggests no linear relationship between the variables, but it's important to note that this does not rule out the possibility of a non-linear relationship. Variables may still be related in complex, non-linear ways that covariance cannot capture.

Next proposition presents a fundamental way to calculate covariance, showing how it measures the degree to which two variables vary together in relation to their individual expected values.

**Proposition 12.4.6** Let  $X$  and  $Y$  be two discrete random variables with finite expected values  $E(X)$  and  $E(Y)$  respectively. We have that

$$\text{Cov}(X, Y) = E(XY) - E(X)E(Y)$$

*Proof.* The covariance of  $X$  and  $Y$  is defined as the expected value of the product of their deviations from their respective means. Expanding the product within the expectation gives:

$$\text{Cov}(X, Y) = E[(X - E(X))(Y - E(Y))] = E[XY - XE(Y) - YE(X) + E(X)E(Y)]$$

Using the linearity of expectation, this expression can be simplified as:

$$E[XY - XE(Y) - YE(X) + E(X)E(Y)] = E(XY) - E(XE(Y)) - E(YE(X)) + E(E(X)E(Y))$$

And since  $E(Y)$  and  $E(X)$  are constants:

$$E(XE(Y)) = E(X)E(Y) \quad \text{and} \quad E(YE(X)) = E(Y)E(X)$$

Hence, the expression further simplifies to:

$$E(XY) - E(X)E(Y) - E(Y)E(X) + E(X)E(Y) = E(XY) - E(X)E(Y)$$

Thus, the covariance of  $X$  and  $Y$  is proved to be:

$$\text{Cov}(X, Y) = E(XY) - E(X)E(Y)$$



Next proposition states an important result that highlights the relationship between independence and covariance for discrete random variables.

**Proposition 12.4.7** If  $X$  and  $Y$  are independent discrete random variables with  $0 < \text{Var}(X) < \infty$  and  $0 < \text{Var}(Y) < \infty$  then  $\text{Cov}(X, Y) = 0$

*Proof.* Let  $X$  and  $Y$  be independent discrete random variables. By the definition of covariance, we have:

$$\text{Cov}(X, Y) = E[(X - E[X])(Y - E[Y])].$$

Expanding this, we get:

$$\text{Cov}(X, Y) = E[XY] - E[X]E[Y].$$

Since  $X$  and  $Y$  are independent, the expectation of the product of independent random variables is the product of their expectations:

$$E[XY] = E[X]E[Y].$$

Substituting this into the expression for covariance, we get:

$$\text{Cov}(X, Y) = E[X]E[Y] - E[X]E[Y] = 0.$$

Thus,  $\text{Cov}(X, Y) = 0$ . ■

The converse is not true as a general rule. Two dependent discrete random variables can be uncorrelated.

### Correlation

The magnitude of the covariance is not standardized, meaning that it can be difficult to interpret the strength of the relationship without context. This is why correlation, which normalizes covariance, is often preferred to measure the strength of the linear relationship between two variables.

**Definition 12.4.6** Let  $X$  and  $Y$  be two discrete random variables with finite variances  $\text{Var}(X)$  and  $\text{Var}(Y)$  respectively. Then the *correlation* of  $X$  and  $Y$ , denoted by  $\text{Cor}(X, Y)$ , is defined as

$$\text{Cor}(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}}$$

Correlation is a statistical measure that quantifies the strength and direction of a linear relationship between two discrete random variables. Unlike covariance, correlation is dimensionless and standardized, providing a value between -1 and 1.

**Proposition 12.4.8** The correlation coefficient  $\rho_{X,Y}$  lies in the range  $[-1, 1]$ .

*Proof.* Correlation normalizes covariance by the product of the standard deviations of  $X$  and  $Y$ , thus it is dimensionless:

$$\rho_{X,Y} = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

where  $\sigma_X = \sqrt{\text{Var}(X)}$  and  $\sigma_Y = \sqrt{\text{Var}(Y)}$ . By the Cauchy-Schwarz inequality, we have:

$$(\mathbb{E}[XY])^2 \leq \mathbb{E}[X^2]\mathbb{E}[Y^2]$$

Substituting for expectations from the definitions of variance and covariance, and rearranging terms, the absolute value of the covariance does not exceed the product of the standard deviations of  $X$  and  $Y$ :

$$|\text{Cov}(X, Y)| \leq \sigma_X \sigma_Y$$

This implies:

$$-1 \leq \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y} \leq 1$$

Thus:

$$-1 \leq \rho_{X,Y} \leq 1$$

■

Correlation is a vital tool in statistics for assessing how strong a linear relationship exists between two variables. A correlation close to  $-1$  or  $+1$  indicates a strong linear relationship, whereas a correlation close to  $0$  suggests a weak linear relationship. This metric is particularly useful in fields such as finance, economics, and the natural sciences where understanding the relationship between variables is crucial for modeling and prediction.

■ **Example 12.19** Let's take a practical example of two random variables: "hours of exercise per week" ( $X$ ) and "weight loss in kilograms over a month" ( $Y$ ). Suppose we gather the following data from 5 individuals:

| Person | Hours of Exercise (X) | Weight Loss (Y) |
|--------|-----------------------|-----------------|
| 1      | 5                     | 1               |
| 2      | 10                    | 2               |
| 3      | 15                    | 3               |
| 4      | 20                    | 3.5             |
| 5      | 25                    | 4               |

Now, let's calculate the correlation between the hours of exercise and the weight loss to see how strongly these two variables are related. Using the formula for the Pearson correlation coefficient:

$$r = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

we calculate  $r \approx 0.98$ , indicating a very strong positive correlation. This means that as the hours of exercise increase, weight loss tends to increase as well. ■

The next proposition establishes that if two discrete random variables are independent and have finite variances, their correlation must be zero.

**Proposition 12.4.9** Let  $X$  and  $Y$  be two independent discrete random variables with finite variances  $\text{Var}(X)$  and  $\text{Var}(Y)$ , then we have that  $\text{Cor}(X, Y) = 0$

*Proof.* Let  $X$  and  $Y$  be two independent discrete random variables with finite variances. The correlation between  $X$  and  $Y$  is given by:

$$\text{Cor}(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)} \sqrt{\text{Var}(Y)}}.$$

By the definition of covariance, we have:

$$\text{Cov}(X, Y) = E[XY] - E[X]E[Y].$$

Since  $X$  and  $Y$  are independent, we know that:

$$E[XY] = E[X]E[Y].$$

Substituting this into the expression for covariance, we get:

$$\text{Cov}(X, Y) = E[X]E[Y] - E[X]E[Y] = 0.$$

Thus, the covariance is zero. Therefore, the correlation becomes:

$$\text{Cor}(X, Y) = \frac{0}{\sqrt{\text{Var}(X)} \sqrt{\text{Var}(Y)}} = 0.$$



## 12.5 Common Distributions

In this section, we will define and discuss several important families of distributions that are widely used in applications of probability theory. Specifically, we will introduce the families of discrete distributions: uniform, Bernoulli, binomial, and discrete normal. We will briefly explain how each of these distribution families arises in practical problems and why they may serve as appropriate probability models for certain experiments. For each family, we will present the form of the probability mass function and discuss some of the fundamental properties of the distributions within the family.

### 12.5.1 Uniform Distribution

The uniform distribution is the simplest probability distribution in probability theory. It models scenarios where all outcomes are equally likely.

**Definition 12.5.1** A discrete random variable  $X$  is said to follow an *uniform distribution* if the probability of each value is the same. Specifically, if  $X$  takes on values in the set  $\{x_1, x_2, \dots, x_n\}$ , the probability mass function is given by:

$$P(X = x_i) = \frac{1}{n} \quad \text{for } i = 1, 2, \dots, n.$$

An example of the discrete uniform distribution would be throwing a fair die. In contrast to other distributions studied in this section, the uniform distribution is non-parametric, meaning it does not depend on a parameter to be fully specified.

The expected value of a discrete uniform random variable  $X$  that takes on values in  $\{x_1, x_2, \dots, x_n\}$  is  $E(X) = \frac{x_1 + x_2 + \dots + x_n}{n}$ . The variance of a discrete uniform random variable  $X$  that takes on values in  $\{x_1, x_2, \dots, x_n\}$  is  $Var(X) = \frac{1}{n} \sum_{i=1}^n (x_i - E(X))^2$ .

### 12.5.2 Bernoulli Distribution

The Bernoulli distribution is one of the simplest and most fundamental probability distributions in probability theory. It models scenarios where there are only two possible outcomes, often termed "success" and "failure."

**Definition 12.5.2** A discrete random variable  $X$  is said to follow a *Bernoulli distribution* with parameter  $p$ , with  $0 \leq p \leq 1$ , if it takes the value 1 (representing "success") with probability  $p$  and the value 0 (representing "failure") with probability  $1 - p$ . The probability mass function is

given by:

$$P(X = x) = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0. \end{cases}$$

The Bernoulli distribution arises naturally in many applied problems, especially those involving binary outcomes, in which the parameter  $p$  represents the probability of success. For example, in medical studies, the outcome of a treatment can be a success (cure) or a failure (no cure), which can be modeled using a Bernoulli distribution.

The expected value of a Bernoulli random variable  $X$  with parameter  $p$  is  $E(X) = p$ . This makes intuitive sense, as the expected value represents the average outcome of many trials, which would tend toward the probability of success. The variance of a Bernoulli random variable is given by  $\text{Var}(X) = p(1 - p)$ . The variance measures the spread or dispersion of the distribution around the expected value. The product  $p(1 - p)$  achieves its maximum value when  $p = 0.5$ , indicating maximum uncertainty when the probabilities of success and failure are equal.

**Definition 12.5.3** Let  $X_1, X_2, \dots$  be a sequence of discrete random variables that follows a Bernoulli distribution with parameter  $p$ , then it is said that  $X_1, X_2, \dots$  are *Bernoulli trials* with parameter  $p$ . An infinite sequence of independent Bernoulli trials is called a *Bernoulli process*.

The Bernoulli process serves as a foundation for more complex stochastic processes and is used in various applications, such as modeling binary events over time.

### 12.5.3 Binomial Distribution

The binomial distribution is a discrete probability distribution that models the number of successes in a fixed number of independent Bernoulli trials, each with the same probability of success. This distribution is widely used in probability theory for various applications involving binary outcomes.

**Definition 12.5.4** A discrete random variable  $X$  is said to follow a *binomial distribution* with parameters  $n$  and  $p$  if it represents the number of successes in  $n$  independent Bernoulli trials, each with success probability  $p$ . The probability mass function is given by:

$$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k} \quad \text{for } k = 0, 1, 2, \dots, n.$$

The parameter  $n$  is the number of trials, and  $p$  is the probability of success on each trial. The term  $\binom{n}{k}$  is the binomial coefficient, representing

the number of ways to choose  $k$  successes out of  $n$  trials.

The binomial distribution arises naturally in many applied problems where the outcomes are binary (success/failure).

**■ Example 12.20** Suppose a company produces light bulbs, and historically, they know that 95% of the bulbs they produce are functional (success) and 5% are defective (failure). The company wants to check the quality of a batch of 20 light bulbs by randomly selecting and testing them. This problem can be modeled using the binomial distribution, where the number of trials is  $n = 20$  and the probability of success is  $p = 0.95$ . For example, the probability of exactly 18 out of 20 light bulbs being functional is:

$$P(X = 18) = \binom{20}{18} (0.95)^{18} (0.05)^2.$$

■

The expected value of a binomial random variable  $X$  with parameters  $n$  and  $p$  is  $E(X) = np$ . This represents the average number of successes in  $n$  trials. The variance of a binomial random variable is  $Var(X) = np(1 - p)$ . The variance measures the spread or dispersion of the distribution around the expected value.

#### 12.5.4 Discrete Normal Distribution

The normal distribution is a continuous probability distribution that frequently appears in natural and social sciences, primarily due to the Central Limit Theorem (see Theorem 12.6.5). However, many real-world situations involve discrete outcomes, such as counts of events or integer-valued random variables. In these cases, a discrete analog to the normal distribution is required. This need does not represent a limitation, as continuous distributions often emerge as limiting abstractions of inherently discrete processes.

In this section, we introduce the discrete normal distribution, which serves as a discrete counterpart to the normal distribution for integer-valued random variables. We derive its key properties through formal definitions and propositions.

To define the discrete normal distribution, we first introduce the concept of a standardized discrete random variable.

**Definition 12.5.5** Let  $X$  be a discrete random variable with expected value  $E(X)$  and variance  $Var(X)$ . The *standardized discrete random variable* of  $X$ , denoted by  $Z$ , is defined as:

$$Z = \frac{X - E(X)}{\sqrt{Var(X)}}$$

This normalization centers the distribution around 0 (the standardized expected value) and scales it by the standard deviation.

■ **Example 12.21** If the discrete random variable  $X$  follows a binomial distribution with parameters  $n$  and  $p$ , its standardized version  $Z$  is given by:

$$Z = \frac{X - np}{\sqrt{np(1-p)}}$$

The probability mass function of  $Z$  can be written as:

$$P(Z = k) = P\left(Z = \frac{k - np}{\sqrt{np(1-p)}}\right) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, \dots, n.$$

■

The normal distribution arises as the limit of the standardized binomial distribution as  $n \rightarrow \infty$ . However, the normal distribution is continuous, while the binomial distribution is discrete. To define a discrete analog to the normal distribution, we consider a discrete distribution whose probabilities are proportional to the normal probability density function evaluated at integer points.

**Definition 12.5.6** A discrete random variable  $X$  is said to follow a *discrete normal distribution* with parameters  $\mu$  and  $\sigma^2$  if it has the following probability mass function:

$$P(X = k) = \frac{1}{Z} \exp\left(-\frac{(k - \mu)^2}{2\sigma^2}\right), \quad k \in \mathbb{Z},$$

where  $Z$  is the normalization constant given by:

$$Z = \sum_{k=-\infty}^{\infty} \exp\left(-\frac{(k - \mu)^2}{2\sigma^2}\right).$$

This distribution can be regarded as a fundamental model for systems where outcomes are inherently discrete, such as the distribution of counts or events.

In practice, this distribution retains the discrete nature of integer-valued random variables while adopting the bell-shaped curve characteristic of the normal distribution. The discrete normal distribution thus encapsulates the essence of the normal distribution while remaining defined on the set of integers.

**Proposition 12.5.1** The normalization constant  $Z$  in the discrete normal distribution satisfies:

$$Z \approx \sqrt{2\pi\sigma^2},$$

for large  $\sigma$ , and thus the probability mass function can be approximated by:

$$P(X = k) \approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(k-\mu)^2}{2\sigma^2}\right).$$

*Proof.* For large  $\sigma$ , the sum in  $Z$  can be approximated by an integral:

$$Z = \sum_{k=-\infty}^{\infty} \exp\left(-\frac{(k-\mu)^2}{2\sigma^2}\right) \approx \int_{-\infty}^{\infty} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx = \sqrt{2\pi\sigma^2}.$$

Therefore, for large  $\sigma$ , the discrete normal PMF approaches the continuous normal PDF evaluated at integer points. ■

■ **Example 12.22** Suppose we want to model the number of times a fair six-sided die shows a "6" in 100 rolls. The number of successes  $X$  follows a binomial distribution with parameters  $n = 100$  and  $p = \frac{1}{6}$ . The expected value and variance are:

$$E(X) = np = \frac{100}{6} \approx 16.67, \quad \text{Var}(X) = np(1-p) = 100 \times \frac{1}{6} \times \frac{5}{6} \approx 13.89.$$

For large  $n$ , the binomial distribution can be approximated by the discrete normal distribution with  $\mu = 16.67$  and  $\sigma^2 = 13.89$ . The PMF is:

$$P(X = k) \approx \frac{1}{Z} \exp\left(-\frac{(k-16.67)^2}{2 \times 13.89}\right),$$

where  $Z$  is the normalization constant.

We can use this approximation to compute probabilities, such as  $P(X = 20)$  or  $P(X = 15)$ . For instance:

$$P(X = 20) \approx \frac{1}{Z} \exp\left(-\frac{(20-16.67)^2}{2 \times 13.89}\right) \approx \frac{1}{Z} \exp\left(-\frac{11.11}{27.78}\right).$$

■

The following proposition highlights an important property of the discrete normal distribution, specifically its symmetry around the mean.

**Proposition 12.5.2** The discrete normal distribution is symmetric about its mean  $\mu$ . Formally, for any integer  $k$ :

$$P(X = \mu + k) = P(X = \mu - k).$$

*Proof.* The symmetry follows directly from the functional form of the PMF:

$$P(X = \mu + k) = \frac{1}{Z} \exp\left(-\frac{(\mu+k-\mu)^2}{2\sigma^2}\right) = \frac{1}{Z} \exp\left(-\frac{k^2}{2\sigma^2}\right).$$

Similarly,

$$P(X = \mu - k) = \frac{1}{Z} \exp\left(-\frac{(\mu - k - \mu)^2}{2\sigma^2}\right) = \frac{1}{Z} \exp\left(-\frac{(-k)^2}{2\sigma^2}\right) = \frac{1}{Z} \exp\left(-\frac{k^2}{2\sigma^2}\right).$$

Since both expressions are equal, the distribution is symmetric about  $\mu$ . ■

The discrete normal distribution serves as a bridge between the discrete and continuous realms of probability distributions. It retains many key properties of the continuous normal distribution, such as symmetry, centrality, and the bell-shaped curve, while accommodating the discrete nature of certain discrete random variables. By formally defining the distribution and studying its properties, we gain a powerful tool for modeling real-world phenomena where outcomes are discrete but exhibit normal-like behavior.

This distribution is especially useful in areas such as manufacturing, education, and epidemiology, where event counts or integer-valued outcomes naturally arise, and where the underlying variability can be captured using a normal-like distribution. The discrete normal distribution allows us to model these scenarios more precisely, ensuring that our probabilistic models align with the real-world structure of the data.

## 12.6 Large Random Samples

A random sample is a collection of independent and identically distributed random variables. Random samples are fundamental in probability theory, providing a basis for making inferences about an unknown probability distribution. Two key results that arise from random samples are the law of large numbers, which ensures that the sample mean converges to the expected value of the distribution as the sample size increases, and the central limit theorem, which states that the distribution of the sample mean approaches a normal distribution as the sample size becomes large, regardless of the original population's distribution.

[...] statistical methods do not directly describe reality but provide a conceptual framework (i.e., a model) for understanding how data is generated and how we can make inferences about the population from which the data comes. [...] Random sampling assumes that every member of the population has an equal probability of being selected, and selections are independent of one another. This means the sample is a simplified, idealized representation of how real-world data might be generated. [...] The model assumes perfect randomness, which rarely occurs in practice. True random sampling is often an unattainable ideal, but it provides a baseline against which deviations can be measured. [...] Random sampling allows for the

development of probability distributions that describe the expected behavior of sample statistics (e.g., sample mean, variance). This makes it possible to quantify uncertainty and draw conclusions about the population. [...] The random sampling model is a representation of how data might be collected, but it is not the same as reality [...] models are tools that work well in specific contexts, but their assumptions often do not hold universally. The practical use of random sampling depends on how well the model fits the real-world context.

### 12.6.1 Random Samples

The concept of a random sample is pivotal in the field of statistical learning, as assuming a set of discrete random variables constitutes a random sample greatly simplifies the mathematical underpinnings of inferential methods. Nevertheless, the criteria for a collection of variables to be considered a random sample are not always met in real-world scenarios.

**Definition 12.6.1** Let  $f$  be a probability mass function, and let  $X_1, X_2, \dots, X_n$  be a collection of discrete random variables. The variables  $X_1, X_2, \dots, X_n$  form a *random sample* from the distribution  $f$  if each discrete random variable  $X_i$  follows the probability mass function  $f$ , and the variables  $X_1, X_2, \dots, X_n$  are independent of each other. These discrete random variables are also referred to as *independent and identically distributed (i.i.d.)*. The number  $n$  of discrete random variables is referred to as the *sample size*.

The joint probability mass function  $g$  of a random sample  $X_1, X_2, \dots, X_n$  is given by:

$$g(x_1, x_2, \dots, x_n) = f(x_1) f(x_2) \cdots f(x_n)$$

■ **Example 12.23** A factory produces light bulbs, and the quality control department is interested in determining the proportion of defective bulbs. The number of defective bulbs in a batch of 100 bulbs is modeled using a binomial distribution, where each bulb has a probability  $p$  of being defective. The department randomly selects 5 batches, each containing 100 bulbs. Let the discrete random variables  $X_1, X_2, X_3, X_4, X_5$  represent the number of defective bulbs in each of these 5 batches. Each  $X_i$  follows the binomial distribution  $\text{Binomial}(100, p)$ . The discrete random variables  $X_1, X_2, X_3, X_4, X_5$  are independent and identically distributed, and thus, they form a random sample. ■

In the area of statistical inference, it is also highly convenient to compute the sample mean, as the average of  $n$  discrete random variables.

**Definition 12.6.2** Let  $X_1, X_2, \dots, X_n$  be  $n$  discrete random variables. The *sample mean* of  $X_1, X_2, \dots, X_n$ , denoted by  $\bar{X}_n$ , is defined as:

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$$

Do not confuse  $\frac{1}{n}(X_1 + X_2 + \dots + X_n)$ , which is a discrete random variable<sup>2</sup>, with the expected value  $E(X_1 + X_2 + \dots + X_n)$ , which is a real number.

As we will show in the next proposition, the concept of sample mean is particularly relevant for the case of  $X_1, X_2, \dots, X_n$  being a random sample.

**Proposition 12.6.1** Let  $X_1, \dots, X_n$  be a random sample with finite mean  $E(X_i) = \mu$  and finite variance  $\text{Var}(X_i) = \sigma^2$ , and let  $\bar{X}_n = \frac{1}{n}(X_1 + \dots + X_n)$  be the sample mean. Then we have that  $E(\bar{X}_n) = \mu$  and  $\text{Var}(\bar{X}_n) = \sigma^2/n$ .

*Proof.* Using the linearity of expectation, we have:

$$E(\bar{X}_n) = E\left(\frac{1}{n} \sum_{i=1}^n X_i\right) = \frac{1}{n} \sum_{i=1}^n E(X_i) = \frac{1}{n} \cdot n \cdot \mu = \mu$$

Using the properties of variance and the fact that the  $X_i$  are independent, we have:

$$\text{Var}(\bar{X}_n) = \text{Var}\left(\frac{1}{n} \sum_{i=1}^n X_i\right)$$

Since variance is homogeneous of degree 2, we can factor out the constant  $\frac{1}{n}$ :

$$\text{Var}(\bar{X}_n) = \left(\frac{1}{n}\right)^2 \text{Var}\left(\sum_{i=1}^n X_i\right)$$

For independent discrete random variables, the variance of the sum is the sum of the variances:

$$\text{Var}\left(\sum_{i=1}^n X_i\right) = \sum_{i=1}^n \text{Var}(X_i) = n \cdot \sigma^2$$

Thus:

$$\text{Var}(\bar{X}_n) = \left(\frac{1}{n}\right)^2 \cdot n \cdot \sigma^2 = \frac{\sigma^2}{n}$$

■

---

<sup>2</sup>Recall that with  $X_1 + \dots + X_n$  we are defining a new discrete random variable over the Cartesian product of the original sample spaces, not adding  $n$  probability distributions.

The probability distribution of  $\bar{X}_n$  will be more concentrated around the mean value  $\mu$  that was the original distribution.

In the area of statistical inference, it is also highly convenient to compute the sample variance as a measure of the dispersion of  $n$  discrete random variables. In particular, we will compute the sample variance of random samples.

**Definition 12.6.3** Let  $X_1, X_2, \dots, X_n$  be a set of discrete random variables with sample mean  $\bar{X}_n$ . The *sample variance* of  $X_1, X_2, \dots, X_n$ , denoted by  $S^2$ , is defined as:

$$S^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X}_n)^2$$

Do not confuse the sample variance, which is a statistic based on observed values, with the theoretical variance of the distribution. We will study some important properties of the variance of large random samples in Section 12.6.

■ **Example 12.24** Continuing from the previous example, suppose we have the observed values of  $X_1, X_2, \dots, X_5$  as follows:

$$X_1 = 6, \quad X_2 = 4, \quad X_3 = 5, \quad X_4 = 7, \quad X_5 = 3$$

The sample mean is:

$$\bar{X}_5 = \frac{1}{5} (6 + 4 + 5 + 7 + 3) = \frac{25}{5} = 5$$

The sample variance is:

$$S^2 = \frac{1}{5-1} \sum_{i=1}^5 (X_i - 5)^2 = \frac{1}{4} [(6-5)^2 + (4-5)^2 + (5-5)^2 + (7-5)^2 + (3-5)^2]$$

$$S^2 = \frac{1}{4} (1 + 1 + 0 + 4 + 4) = \frac{10}{4} = 2.5$$

Thus, the sample variance is 2.5. ■

The sample standard deviation is used to estimate the standard deviation of a population based on a sample taken from it. Unlike the population standard deviation, which uses the true mean of the entire population, the sample standard deviation uses the mean of the sample as an estimate of the true mean.

**Definition 12.6.4** Let  $X_1, X_2, \dots, X_n$  be  $n$  discrete random variables with sample mean  $\bar{X}_n$  and sample variance  $S^2$ . The *sample standard deviation*

of  $X_1, X_2, \dots, X_n$ , denoted by  $S$ , is defined as:

$$S = \sqrt{S^2} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X}_n)^2}$$

■ **Example 12.25** Using the sample variance calculated in the previous example, the sample standard deviation is:

$$S = \sqrt{S^2} = \sqrt{2.5} \approx 1.5811$$

■

The sample covariance is used to estimate the covariance between two populations based on a sample taken from them.

**Definition 12.6.5** Let  $X_1, X_2, \dots, X_n$  and  $Y_1, Y_2, \dots, Y_n$  be two sets of discrete random variables with sample means  $\bar{X}$  and  $\bar{Y}$  respectively. The *sample covariance* between  $X$  and  $Y$  is defined as:

$$s_{XY} = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})$$

■ **Example 12.26** Suppose we have paired observations of two discrete random variables  $X$  and  $Y$ :

| $i$   | 1 | 2 | 3 | 4 | 5  |
|-------|---|---|---|---|----|
| $X_i$ | 2 | 4 | 6 | 8 | 10 |
| $Y_i$ | 1 | 3 | 5 | 7 | 9  |

Compute the sample means:

$$\bar{X} = \frac{1}{5} \sum_{i=1}^5 X_i = \frac{30}{5} = 6, \quad \bar{Y} = \frac{1}{5} \sum_{i=1}^5 Y_i = \frac{25}{5} = 5$$

Compute the sample covariance:

$$s_{XY} = \frac{1}{5-1} \sum_{i=1}^5 (X_i - 6)(Y_i - 5)$$

$$s_{XY} = \frac{1}{4} [(2-6)(1-5) + (4-6)(3-5) + (6-6)(5-5) + (8-6)(7-5) + (10-6)(9-5)]$$

$$s_{XY} = \frac{1}{4} [(-4)(-4) + (-2)(-2) + (0)(0) + (2)(2) + (4)(4)] = \frac{1}{4} [16 + 4 + 0 + 4 + 16]$$

Thus, the sample covariance between  $X$  and  $Y$  is 10. ■

## 12.6.2 Law of Large Numbers

The law of large numbers is a theorem that states that the sample mean of a large random sample, i.e. a large number of independent and identically distributed discrete random variables, should be close to the expected value of the discrete random variables, and that the more variables we add, the closer will be to that value. Before to prove this important theorem we have to prove two other related propositions: Markov's inequality and Chebyshev's inequality.

Markov's inequality provides a simple way to measure how unlikely it is for a discrete random variable to take on large values, given its expected value. It leverages the fact that if the average value of discrete random variable  $X$  is small, then it is improbable for  $X$  to take on a large value (recall Example 12.16). Markov's inequality is particularly useful because it makes no assumptions about the distribution of the discrete random variable, apart from it being non-negative and having a finite expected value.

**Proposition 12.6.2 — Markov's Inequality.** Let  $X$  be a nonnegative discrete random variable, i.e.  $P(X \geq 0) = 1$ , with expected value  $E(X)$ . Then for every real number  $t > 0$  we have that

$$P(X \geq t) \leq \frac{E(X)}{t}$$

*Proof.* Let  $f$  be the probability mass function of  $X$ . Since  $X$  is non-negative we have that  $E(X) = \sum_{x>0} xf(x)$ . Then

$$E(X) = \sum_{x>0} xf(x) = \sum_{x=0}^t xf(x) + \sum_{x=t}^{\infty} xf(x) \geq \sum_{x=t}^{\infty} xf(x) \geq \sum_{x=t}^{\infty} tf(x) = t \sum_{x=t}^{\infty} f(x) = tP(X \geq t)$$

that is,  $E(X) \geq tP(X \geq t)$ . ■

For example, if the expected value of a discrete random variable  $X$  is 10, and we are interested in the probability that  $X$  is at least 50, Markov's inequality tells us that  $P(X \geq 50) \leq 10/50 = 0.2$ . While Markov's inequality is very general and easy to use, it is often not tight (i.e., the bound is not very close to the true probability). This is because it does not use any other information about the distribution of  $X$  other than its expectation.

Chebyshev's inequality provides a bound on the probability that a discrete random variable deviates from its expected value by more than a specified number of standard deviations. Unlike Markov's inequality, which only requires non-negativity, Chebyshev's inequality does not require the discrete random variable to be nonnegative, but it requires knowledge of both the

mean and variance of the discrete random variable. The inequality is applicable to any discrete random variable with a finite mean and variance, regardless of the specific distribution.

**Corollary 12.6.3 — Chebyshev's inequality.** Let  $X$  be a discrete random variable with expected value  $E(X)$  and variance  $\text{Var}(X)$ . Then for every real number  $t > 0$  we have that

$$P(|X - E(X)| \geq t) \leq \frac{\text{var}(X)}{t^2}$$

*Proof.* Applying Markov's inequality we have that

$$P(|X - E(X)| \geq t) = P\left((X - E(X))^2 \geq t^2\right) \leq \frac{E\left((X - E(X))^2\right)}{t^2} = \frac{\text{var}(X)}{t^2}$$

■

Chebyshev's inequality provides insight into the concentration of a discrete random variable around its mean. It quantifies the idea that most of the probability mass of a discrete random variable lies within a certain range of its mean, with fewer values appearing further away. This concept is particularly useful in understanding the spread of a distribution and the likelihood of large deviations.

The last element we need to formally prove the law of large numbers is to introduce the concept of convergence in probability for discrete random variables. Intuitively, a sequence of discrete random variables converges in probability to a value  $b$  if  $X_n$  lies around  $b$ .

**Definition 12.6.6** Let  $X_1, X_2, \dots$  be a sequence of discrete random variables. It is said that the sequence  $X_1, X_2, \dots$  converges in probability to  $b$ , denoted by  $X_n \xrightarrow{P} b$ , if for every positive real number  $\varepsilon > 0$  we have that

$$\lim_{n \rightarrow \infty} P(|X_n - b| < \varepsilon) = 1$$

Convergence in probability essentially means that as the sequence progresses (i.e., as  $n$  increases), the discrete random variables  $X_n$  become arbitrarily close to the number  $b$  with high probability. The distance between  $X_n$  and  $b$  can be made as small as desired, and the likelihood of  $X_n$  being far from  $b$  diminishes as  $n$  increases.

Given the above definitions and partial results, we can now formally introduce and prove the law of large numbers.

**Theorem 12.6.4 — Law of Large Numbers.** Let  $X_1, \dots, X_n$  be a random sample with finite expected value and finite variance, and let  $\bar{X}_n = \frac{1}{n}(X_1 + \dots + X_n)$  be the sample mean. Then we have that

$$\bar{X}_n \xrightarrow{P} \mu$$

*Proof.* Since the variables  $X_1, \dots, X_n$  are independent and identically distributed, we have that the variance of the sample mean is  $\text{Var}(\bar{X}_n) = \frac{\sigma^2}{n}$  and that the mean is  $E(\bar{X}_n) = \mu$  (see Proposition 12.6.1). Applying the Chebyshev's inequality to the discrete random variable  $\bar{X}_n$  we have that

$$P(|\bar{X}_n - \mu| \geq \varepsilon) \leq \frac{\sigma^2}{n\varepsilon^2}.$$

From there we can obtain

$$P(|\bar{X}_n - \mu| < \varepsilon) = 1 - P(|\bar{X}_n - \mu| \geq \varepsilon) \geq 1 - \frac{\sigma^2}{n\varepsilon^2}.$$

As  $n$  approaches infinity, the above expression approaches 1. Applying the definition of convergence in probability, we have that

$$\bar{X}_n \xrightarrow{P} \mu$$

■

It is very important to note that the law of large numbers only works in case that the discrete random variables are independent and identically distributed. Also, that the law is true only in the limit. If we have a finite number of discrete random variables, the sample mean will be close to the distribution mean, but not necessarily equal.

Let's see an example of the use of the law of large numbers in practice.

■ **Example 12.27** Consider the experiment of rolling a fair six-sided die  $\Omega = \{1, 2, 3, 4, 5, 6\}$ , being each side equally probable with  $P(\omega) = 1/6$  for  $\omega \in \Omega$ . Let  $X : \Omega \rightarrow \mathbb{R}$  be a discrete random variable that maps each side of the die with the number depicted on it. The probability mass function  $f : \text{range}(X) \rightarrow [0, 1]$  of  $X$  assigns  $1/6$  to each value of  $\text{range}(X)$ . The expected value of  $X$  is  $E(X) = 3.5$ .

Let's consider a random sample of size two, that is, throwing two dice (one after the other, so they are distinguishable), and call the corresponding discrete random variables  $X_1$  and  $X_2$ . The sample mean would be the discrete random variable  $\frac{X_1+X_2}{2} : \Omega \times \Omega \rightarrow \mathbb{R}$  that maps each pair  $(\omega_1, \omega_2) \in \Omega \times \Omega$  to the real value  $X_1(\omega_1) + X_2(\omega_2)/2$ .

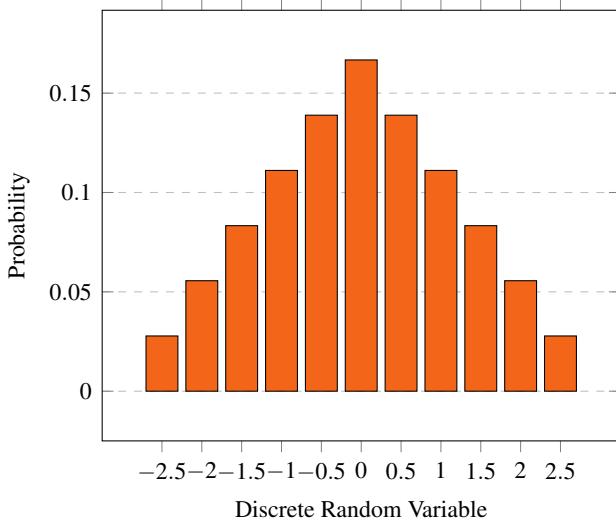


Figure 12.1: Probability mass function of the discrete random variable  $\frac{X_1+X_2}{2} - \mathbb{E}(X)$ .

The law of large numbers proposes to study the discrete random variable  $\frac{X_1+X_2}{2} - \mathbb{E}(X) : \Omega \times \Omega \rightarrow \mathbb{R}$ , whose probability mass function is depicted in Figure 12.1. And lets, for example, compute the probability  $P(|\frac{X_1+X_2}{2} - \mathbb{E}(X)| < 1)$  which is 0.444.

Now let's consider a random sample of size ten. In Figure 12.2 is depicted the probability mass function of the discrete random variable  $\frac{X_1+X_2+\dots+X_{10}}{10} - \mathbb{E}(X) : \Omega \times \dots \times \Omega \rightarrow \mathbb{R}$ . The probability  $P(|\frac{X_1+X_2+\dots+X_{10}}{10} - \mathbb{E}(X)| < 1)$  is 0.973. ■

It is important to clarify that the law of large numbers refers to the sample mean, that is  $\frac{1}{n} \sum_{i=1}^n X_i$ , and it is not necesarily true for other formulas, like for example, the deviation from the theoretical expected value  $\sum_{i=1}^n X_i - n \times \mathbb{E}(X)$  which not only it does not converge, but inscreases in absolute value as  $n$  increases (see Example 12.28).

■ **Example 12.28** If we toss a fair coin, the probability that the outcome will be head is equal to 1/2. According to the law of large numbers, the proportion of heads in a large number of coin tosses will be close to 1/2. However, the difference between the number of heads and tails will not be close to zero. In fact, the larger the number of coin tosses, the larger will be this difference. This is a highly conterintuitive fact, since most of the people think that the more we toss the coin, the closer will be the number of heads

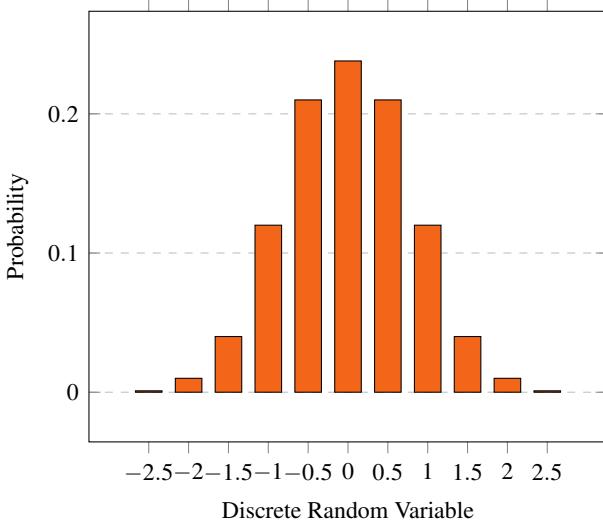


Figure 12.2: Probability mass function of the discrete random variable  $\frac{X_1+X_2+\dots+X_{10}}{10} - E(X)$ .

to the number of tails, which is not true. ■

### 12.6.3 Central Limit Theorem

Let  $X_1, \dots, X_n$  be a sample of  $n$  independent and identically distributed discrete random variables with mean  $\mu$  and variance  $\sigma^2$ . As we saw in the previous section, the law of large numbers states that the sample average  $\bar{X}_n$  converges in probability to  $\mu$  as  $n$  increases. The central limit theorem states that the distribution of the difference between the sample average  $\bar{X}_n$  and the population mean  $\mu$ , when multiplied by the factor  $\sqrt{n}$  approximates to the normal distribution with mean 0 and variance  $\sigma^2/n$ . The theorem is true regardless of the shape of the original random variables.

**Theorem 12.6.5 — Central Limit Theorem.** Let  $X_1, \dots, X_n$  be a random sample of size  $n$  from a distribution with mean  $\mu$  and finite variance  $\sigma^2$ . Define the standardized sum as

$$Z_n = \frac{\bar{X}_n - \mu}{\sigma/\sqrt{n}},$$

where  $\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$  is the sample mean. Then, as  $n \rightarrow \infty$ , the probability density function of  $Z_n$  converges to the probability density function of the

standard normal distribution  $\phi(x)$ , that is,

$$\lim_{n \rightarrow \infty} f_{Z_n}(x) = \phi(x),$$

where  $f_{Z_n}(x)$  is the probability density function of  $Z_n$ , and  $\phi(x)$  is the probability density function of the standard normal distribution, given by

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right).$$

*Proof.* Let  $X_1, \dots, X_n$  be independent and identically distributed (i.i.d.) discrete random variables with mean  $\mu$  and finite variance  $\sigma^2$ . Define the sample mean as

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i.$$

We are interested in the distribution of the standardized variable

$$Z_n = \frac{\bar{X}_n - \mu}{\sigma/\sqrt{n}}.$$

First, consider the sum of the  $X_i$ 's, which we write as

$$S_n = \sum_{i=1}^n X_i.$$

The sample mean can then be written as

$$\bar{X}_n = \frac{S_n}{n}.$$

Thus, the standardized variable  $Z_n$  becomes

$$Z_n = \frac{S_n - n\mu}{\sigma\sqrt{n}}.$$

To analyze the behavior of  $Z_n$  as  $n \rightarrow \infty$ , we consider the sum  $S_n = \sum_{i=1}^n X_i$ . According to the Law of Large Numbers,  $S_n/n$  converges to  $\mu$ , and hence  $Z_n$  should converge to a normal distribution due to the nature of sums of independent discrete random variables.

For large  $n$ ,  $Z_n$  can be approximated by considering the Taylor expansion of the exponential function. Since the  $X_i$ 's are i.i.d., the distribution of  $S_n$  can be approximated by a normal distribution with mean  $n\mu$  and variance  $n\sigma^2$ . The key idea is that as  $n$  increases, the distribution of  $Z_n$  approaches a

normal distribution because the sum of i.i.d. discrete random variables tends to be normally distributed by the Central Limit Theorem.

Let us consider the moment generating function (MGF) of  $Z_n$ . The MGF of  $Z_n$  is given by:

$$M_{Z_n}(t) = \mathbb{E}[\exp(tZ_n)] = \mathbb{E}\left[\exp\left(t\frac{S_n - n\mu}{\sigma\sqrt{n}}\right)\right].$$

For large  $n$ , by the Central Limit Theorem, the MGF of  $Z_n$  approaches that of a standard normal variable  $N(0, 1)$ :

$$M_{Z_n}(t) \approx \exp\left(\frac{t^2}{2}\right),$$

which implies that  $Z_n$  converges in distribution to  $N(0, 1)$  as  $n \rightarrow \infty$ .

Since  $Z_n$  converges in distribution to  $N(0, 1)$ , the probability density function of  $Z_n$ , denoted by  $f_{Z_n}(x)$ , must converge to the probability density function of a standard normal distribution  $\phi(x)$ , given by:

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right).$$

Thus, we have

$$\lim_{n \rightarrow \infty} f_{Z_n}(x) = \phi(x).$$

Therefore, as  $n \rightarrow \infty$ , the distribution of the standardized sum  $Z_n$  approaches the standard normal distribution, completing the proof. ■

The Central Limit Theorem is a fundamental concept in probability theory used in statistical analysis and inference. It allows us to compute the probability that the sample average is close to the distribution mean. It is important to recall the conditions for the central limit theorem to be true: the samples must be independent and identically distributed, the original distribution has to have a finite variance, and the sample size must be sufficiently large.

**Example 12.29** Suppose a factory produces light bulbs, and the lifespan of each light bulb is a discrete random variable with a mean of  $\mu = 1000$  hours and a standard deviation of  $\sigma = 50$  hours. The factory tests a random sample of  $n = 36$  light bulbs to estimate the average lifespan of the bulbs produced in a particular batch. What is the probability that the sample mean lifespan of these 36 bulbs is between 990 and 1010 hours?

We can apply the Central Limit Theorem to solve this problem because we are dealing with the sample mean of a large number of independent,

identically distributed discrete random variables (lifespans of light bulbs). First, define the sample mean  $\bar{X}_n$  as:

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i,$$

where  $X_i$  represents the lifespan of the  $i$ -th light bulb in the sample, and  $n = 36$  is the sample size.

By the Central Limit Theorem, for sufficiently large  $n$ , the distribution of the sample mean  $\bar{X}_n$  approaches a normal distribution with mean  $\mu$  and standard deviation  $\sigma/\sqrt{n}$ :

$$\bar{X}_n \sim N\left(\mu, \frac{\sigma}{\sqrt{n}}\right) = N\left(1000, \frac{50}{\sqrt{36}}\right) = N\left(1000, \frac{50}{6}\right) = N(1000, 8.33).$$

Next, we calculate the z-scores corresponding to 990 hours and 1010 hours using the formula:

$$z = \frac{X - \mu}{\sigma/\sqrt{n}},$$

where  $X$  is the value for which we want to find the z-score.

For  $X = 990$ :

$$z_{990} = \frac{990 - 1000}{8.33} \approx \frac{-10}{8.33} \approx -1.20.$$

For  $X = 1010$ :

$$z_{1010} = \frac{1010 - 1000}{8.33} \approx \frac{10}{8.33} \approx 1.20.$$

Now, we use the standard normal distribution table (or a calculator) to find the probabilities corresponding to these z-scores:

$$P(z_{990} \leq Z \leq z_{1010}) = P(-1.20 \leq Z \leq 1.20).$$

From the standard normal distribution table:

$$P(Z \leq 1.20) \approx 0.8849,$$

$$P(Z \leq -1.20) \approx 0.1151.$$

Thus, the probability that the sample mean lifespan of the 36 light bulbs is between 990 and 1010 hours is:

$$P(990 \leq \bar{X}_n \leq 1010) = P(-1.20 \leq Z \leq 1.20) = 0.8849 - 0.1151 = 0.7698.$$

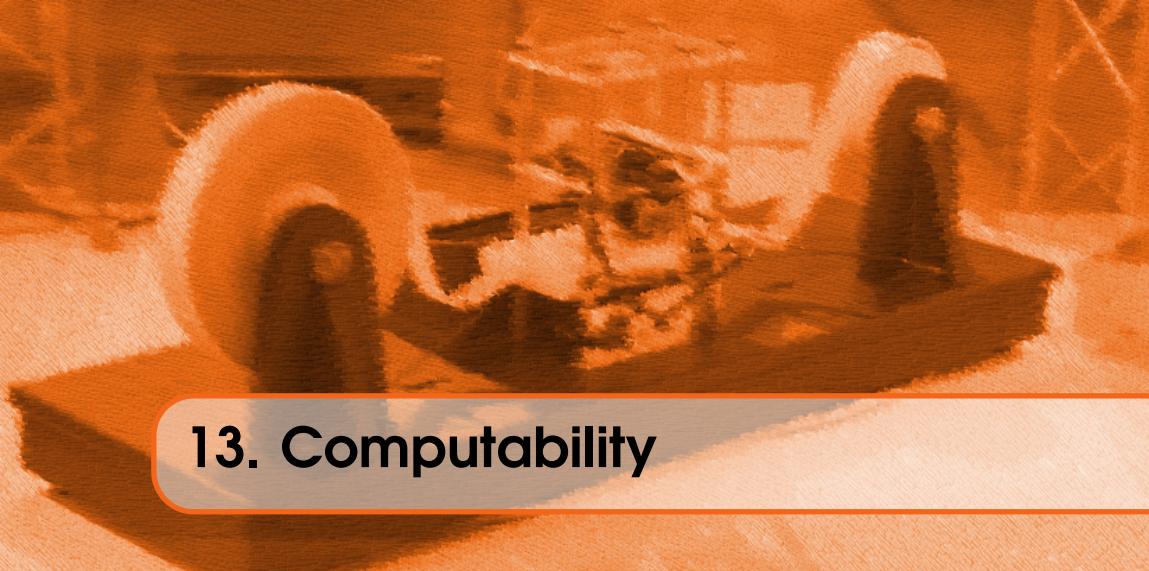
The probability that the sample mean lifespan of the 36 light bulbs falls between 990 and 1010 hours is approximately 76.98%. This result demonstrates how the Central Limit Theorem allows us to use the normal distribution to approximate the sampling distribution of the sample mean, even when the original data is not normally distributed, as long as the sample size is sufficiently large. ■

## References

[DeG+86] is a widely respected textbook in the fields of statistics and probability theory. First published in 1975, this book is known for its clear exposition of the fundamental concepts of probability and statistics, making it suitable for both beginners and those with some background in the subject. The book's approach balances theory and application, making it useful both for learning theoretical underpinnings and for applying probability and statistics to real-world problems.

[Chi13] offers a comprehensive introduction to the foundational aspects of probability, with a focus on the philosophical questions it raises. Childers explores various interpretations of probability, including frequentist, propensity, classical, Bayesian, and objective Bayesian, and presents these complex ideas in a way that is accessible even to those without a strong background in probability or mathematics.

An example of the problems associated with the misinterpretation of expected value is the St. Petersburg Paradox. Introduced by Nicholas Bernoulli in 1713, this paradox involves a gambling game with an infinite expected payoff, yet no reasonable person would pay more than \$25 to play it. Despite being three centuries old, the paradox continues to inspire new arguments and solutions in recent years (see [Hua13] for a historical review of the main proposed solutions).



## 13. Computability

*Caminante, no hay camino,  
se hace camino al andar.<sup>1</sup>*

Antonio Machado

We begin our review of the background required to understand the theory of nescience by providing a mathematical formalization of the concept of a *computable procedure*. Intuitively, a computable procedure is a method consisting of a finite number of instructions that, when applied to a problem, produce the correct answer after a finite number of steps. The key point is that the instructions must be clear and precise enough for any human to follow without aid. We can even go a step further and require that the instructions must be so straightforward that a machine could execute them. In 1936, British mathematician Alan Turing introduced a formal model for a family of hypothetical machines and posited that for every computable procedure (in its intuitive sense), there exists a *Turing machine* capable of computing it. The model was not only simple enough for precise mathematical analysis but also versatile and powerful.

Over the years, many alternative proposals have aimed to formalize the concept of computable procedure. Some have been very complicated, but

---

<sup>1</sup>Wanderer, there is no road, the road is made by walking.

all have proven equivalent to the concept of the Turing machine; that is, they solve the same set of problems. Two notable examples of alternative definitions are the *lambda calculus* by Alonzo Church and the *theory of recursive functions* by Kurt Gödel and Stephen Kleene. The *Church-Turing thesis* asserts that any formalization of the concept of a computable procedure, meeting some minimum requirements (such as performing a finite amount of work in a single step), is equivalent to a Turing machine. This thesis suggests an objective notion of a computable procedure that is independent of any specific formalization.

The concept of the Turing machine, initially referring to mechanical devices designed to solve individual problems, has been extended and universalized. A *universal Turing machine* exists that can resolve all computable problems by simulating the behavior of other specific machines, akin to how modern computers run algorithms written in various programming languages. This concept raises a significant question: Are there problems that are not computable? We will see that the answer is affirmative, certain well-defined problems exceed the computational capabilities of computers, and such problems are more common than initially anticipated. The notion of uncomputable functions will be pivotal in our theory of science.

Given the abstract nature of most entities studied in science, we employ the concept of the *oracle Turing machine* to aptly formalize our theory. An oracle Turing machine resembles a regular Turing machine but is augmented with the capability to query an external oracle, whose workings are not fully understood, to aid in its computations. This oracle can address problems that are unresolvable by standard Turing machines—essentially, it can solve uncomputable problems. The oracle is a theoretical construct that represents a source of solutions or information that is not bound by the limitations of computability. It acts as a ‘black box’ that instantly provides answers to specific questions or problems, enabling the oracle Turing machine to transcend its computational boundaries. The oracle is an abstract and non-mechanical entity, a theoretical tool used to explore the bounds of computation, rather than a physical or concrete machine that can be built or observed.

Turing machines illuminate the inherent limitations of our computational capabilities. This exploration into the abstract and theoretical realms of computation is not just a philosophical endeavor; it also possesses practical applications in the field of *computational complexity*. Located at the intersection of computer science and mathematics, computational complexity evaluates the challenges associated with solving problems, measured against the required resources, notably time and space. Problems are classified based on their intrinsic complexity and the computational effort required for their resolution. One of the key questions in this field is the elusive and yet un-

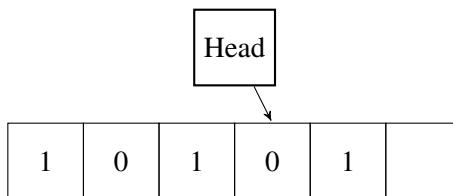


Figure 13.1: Turing Machine

solved  $P \stackrel{?}{=} NP$  question, which seeks to determine if the class  $P$  of problems, those that are easy to solve, coincides with the class  $NP$  of problems, whose solutions are easy to verify. In this book, our focus is not solely on the epistemological question of identifying which problems can be effectively solved given ample time and space, but also on those that can be resolved efficiently in time.

## 13.1 Turing Machines

A Turing machine is an extremely simplified model of a general-purpose computer, yet it is capable of solving any problem that real computers can address. Intuitively, one can envision the machine as consisting of a head that operates on a two-way infinite tape striped with symbols. At each time step, the machine reads the symbol under the head and decides to either write a new symbol on the tape, move the head one square to the left or right, or execute both actions. Algorithms are implemented using an internal table of rules housed within the control head, and the actual input to the algorithm is encoded on the tape. Once the machine reaches its final state, the algorithm's output can be read from the tape. Figure 13.1 depicts an example of a machine in its initial state, with the head located at the beginning of the input string.

The following definition formally introduces the concept of a Turing machine.

**Definition 13.1.1 — Turing Machine.** A *Turing machine* is a 7-tuple

$(Q, \Gamma, \sqcup, \Sigma, q_i, q_f, \tau)$  where:

- $Q$  is a finite, non-empty, set of *states*,
- $\Gamma$  is a finite, non-empty, set of *tape symbols*,
- $\sqcup \in \Gamma$  is the *blank symbol*,
- $\Sigma \subseteq \Gamma \setminus \sqcup$  is the set of *input symbols*,
- $q_o \in Q$  is the *initial state*,
- $q_f \in Q \setminus \{q_o\}$  is the *final state*,
- $\tau : (Q \setminus \{q_f\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$  is a partial *transition function*.

The algorithm executed by the machine is defined by the transition function  $\tau$ . This function dictates the machine's actions based on its current state and the tape symbol currently under the head. According to  $\tau$ , the machine transitions to a new state, writes a new symbol on the tape (or retains the existing one), and moves the head left, right, or keeps it stationary ( $L$ ,  $R$ , or  $S$  respectively). The machine follows a finite, uniquely determined sequence of steps until it reaches the final state  $q_f$  and *halts*, making no subsequent moves. The algorithm's output is the string of symbols  $s \in \Sigma^*$  remaining on the tape after halting. Some machines, however, may enter an infinite loop, never reaching a halting state. If a machine encounters an undefined transition, it will enter in an infinite loop, never halting.

The machine's input consists of a string of symbols, with the assumption that the machine's head is initially positioned at the first symbol of the input string. To address problems involving an object  $O$  that isn't a string, we must first develop a method to encode that object as a string, denoted as  $\langle O \rangle$ .

■ **Example 13.1** The following Turing machine is designed to solve the problem of adding two natural numbers. It consists of the set of states  $Q = \{q_o, q_1, q_f\}$ , the set of tape symbols  $\Gamma = \{0, 1, \sqcup\}$ , and the set of input symbols  $\mathcal{B} = \{0, 1\}$ . The transition function is defined in the table below, where rows are indexed by machine states, and columns by tape symbols:

|       | 0                  | 1                  | $\sqcup$   |
|-------|--------------------|--------------------|------------|
| $q_o$ | $(q_f, \sqcup, S)$ | $(q_1, \sqcup, R)$ | $\uparrow$ |
| $q_1$ | $(q_f, 1, S)$      | $(q_f, 1, R)$      | $\uparrow$ |

Table 13.1: Transition Rules

For natural numbers  $n$  and  $m$ , the input string is composed of  $n$  occurrences of the symbol '1', followed by a '0', and then followed by  $m$  occurrences of '1'. The machine's output will be a string of  $n+m$  consecu-

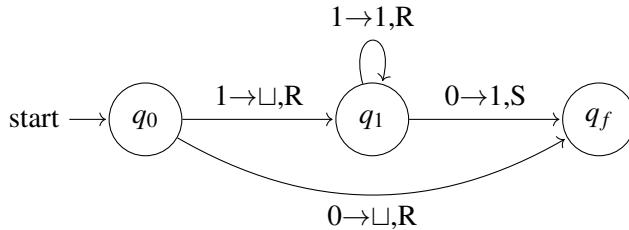


Figure 13.2: Example of Turing Machine

tive ‘1’s. For instance, to add the numbers 2 and 3, the input string should be  $\square 110111\square$ , resulting in the output string  $\square 11111\square$ . ■

A Turing machine can also be represented by a *state diagram*. A state diagram is similar to a labeled directed graph<sup>2</sup> where the vertices represent the states of the machine. The edges signify transitions from one state to another, and the edge labels indicate the symbol under the head that leads to the new state, the symbol that gets written on the tape, and the direction in which the head moves. Following these conventions, the state diagram for the Turing machine in Example 13.1 is depicted in Figure 13.2.

It is a remarkable fact that minor alterations to the definition of a Turing machine do not change its computational power. In other words, the definition is highly robust. In Example 13.2, it’s demonstrated that adding more tapes to the machine doesn’t expand the range of problems it can solve. Similar arguments can be made when adding finite storage to the control tape, allowing for parallel processing with multiple control heads, and so on.

■ **Example 13.2** A *multitape Turing machine* is a Turing machine equipped with multiple heads and their respective tapes. In the initial configuration, the input string resides in tape 1, while the other tapes are blank. The transition function for a multitape Turing machine is:

$$\tau : (Q \setminus q_f) \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k,$$

where  $k$  denotes the number of tapes. Multitape Turing machines are equivalent in power to standard Turing machines. We can validate this claim by devising a method for a standard Turing machine to mimic a multitape machine’s behavior. This requires encoding the content of multiple tapes onto a single tape, introducing a new symbol as a tape separator, and encoding the positions of the heads across the tapes with a distinct head location symbol. If we designate the tape separation symbol as  $|$  and the head location

<sup>2</sup>In this particular case, we allow loops and multiple edges originating from vertices.

symbol as  $h$ , a simulation tape for a machine with 3 tapes might appear as  $\sqcup 01h00|000h1|h0101\sqcup$ . The standard machine's operation would involve scanning the subtapes one by one, pinpointing the head's location, and executing the necessary transition. If the computation on one subtape necessitates writing a new symbol beyond its boundary, we'd need to shift subsequent symbols to accommodate the new one. While the simulation might operate at a slower pace than the original multtape machine, both machine types can solve an identical set of problems. ■

For the remainder of this book, without any loss of generality, we'll assume that the set of input symbols is  $\Sigma = \mathcal{B}$  and the set of tape symbols is  $\Gamma = \{0, 1, \sqcup\}$ .

In addition to providing a formal definition of a Turing machine, it's essential to formally outline its computational process. This entails detailing how the machine reads the input string, produces the output string, and transitions between states during computation. We will start by defining the concept of the machine's internal configuration. This configuration captures the machine's current state and position, as well as the present state of the tape.

**Definition 13.1.2** A *configuration* of a Turing machine  $T$  is the 3-tuple  $(q, s, i)$ , where  $q \in Q$  represents a state of the machine,  $s \in \Gamma^+$  denotes a string containing the tape's content (excluding the blank symbols), and  $1 \leq i \leq n$  is the index of the symbol  $s_i$  beneath the head. Here,  $s_1$  is the first non-blank symbol on the tape, and  $n = l(s)$ .

Configurations enable us to describe the current state of a Turing machine without any loss of information. At any stage of computation, one could halt the machine, record its configuration, and later resume the computation from the exact point of interruption using this configuration.

The following definition explains how we transition from one configuration to the next during computation.

**Definition 13.1.3** A configuration  $C = (q, s, i)$  yields another configuration  $C' = (r, s', j)$  if there exists a transition  $\tau : (q, s_i) = (r, s'_i, a)$ , where  $s = s_1 \dots s_{i-1} s_i s_{i+1} \dots s_n$ ,  $s' = s_1 \dots s_{i-1} s'_i s_{i+1} \dots s_n$ , and

$$j = \begin{cases} i+1 & \text{if } a = R \\ i-1 & \text{if } a = L \\ i & \text{if } a = S \end{cases} \quad (13.1)$$

Building on the concepts of configuration and one configuration yielding another, we can now formally articulate the notion of computation.

**Definition 13.1.4 — Computation.** Let  $T$  be a Turing machine,  $C_0$  its initial configuration, and  $C_n$  a configuration encompassing the final state  $q_f$ . A *computation* under machine  $T$  refers to a finite sequence of  $n + 1$  configurations  $(C_0, C_1, \dots, C_n)$  wherein each configuration  $C_k$  yields the subsequent configuration  $C_{k+1}$ , for all  $0 \leq k < n$ .

Computations are deterministic; meaning, for a given Turing machine  $T$  and an input string  $s$ , the configuration sequence is preordained. If machine  $T$  neither halts nor progresses with input  $s$ , we deduce the absence of computation.

■ **Example 13.3** The computation of the Turing machine described in Example 13.1 using the input string 110111 results in the following sequence of configurations:

- 1  $(q_0, 110111, 1)$
- 2  $(q_1, 10111, 1)$
- 3  $(q_1, 10111, 2)$
- 4  $(q_f, 11111, 2)$

■

Intuitively, a procedure is deemed computable by a human if it can be delineated through specific steps, executed systematically, without relying on intuition or ingenuity. This intuitive grasp aligns with the formalized concept of a Turing machine, bridging informal comprehension and the machine's rigorous definition—a cornerstone in the theory of computation. However, this alignment presents an intriguing challenge. Affirming that our grasp of computability mirrors a Turing machine's capabilities cannot be proven traditionally, as 'computability' lacks a well-defined interpretation. Consequently, some researchers categorize this as a *thesis*, avoiding the formal 'theorem' label. Turing himself opted to term it a *definition*, steering clear of denoting it as a theorem.

**Theorem 13.1.1 — Turing's Thesis.** A procedure is computable if, and only if, it can be executed by a Turing machine.

To further underscore the significance and robustness of the Turing machine as a model of computation, it's worth noting, as mentioned earlier in this chapter, that all alternative formalizations of computability proposed to date align in terms of their computational capabilities with that of the Turing machine. This universality underscores the Turing machine's central position in the realm of theoretical computer science.

## 13.2 Universal Turing Machines

In Section 13.1, we explored storing the current state of a Turing machine, its configuration, to pause and later resume computation. In Example 13.4, we will delve into a similar procedure, not for storing the machine's current state, but for saving a comprehensive description of the machine itself. This methodology facilitates the enumeration, or listing, of all possible Turing machines. Such enumeration is instrumental in demonstrating the existence of problems that cannot be solved by any Turing machine (refer to Section 13.3) and unveiling the pivotal concept of the *Universal Turing Machine*.

■ **Example 13.4** To describe a Turing machine concisely, we need to encode the transition function  $\tau : (Q \setminus \{q_f\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$ . This function can be represented as a collection of quintuples  $(q, s, r, t, a)$ , where  $q \in (Q \setminus \{q_f\})$ ,  $r \in Q$ ,  $s, t \in \Gamma$ , and  $a \in \{L, R, S\}$ . In this manner, any Turing machine  $T$  is fully described by a collection of quintuples:

$$(q_1, s_1, r_1, t_1, a_1), (q_2, s_2, r_2, t_2, a_2), \dots, (q_m, s_m, r_m, t_m, a_m)$$

where  $m \leq d(Q \setminus \{q_f\}) \times d(\Gamma)$ , with the stipulation that the first quintuple refers to the initial state and the second one to the final state; i.e.,  $q_1 = q_o$  and  $r_2 = q_f$ . A possible approach to describe these quintuples is to encode the elements of the set  $Q \cup \Gamma \cup \{L, R, S\}$  using a fixed-length binary code (refer to Definition 14.1.6 for more details), encoding the quintuple  $(q, s, r, t, a)$  as  $\langle q, s, r, t, a \rangle$ . The length of an encoded quintuple is  $5l$ , where  $l = \lceil \log(d(Q \cup \Gamma \cup \{L, R, S\})) \rceil$ . Following this convention, machine  $T$  is encoded as the binary string:

$$\langle T \rangle = \langle \bar{l}, \langle q_1, r_1, s_1, t_1, a_1 \rangle, \dots, \langle q_r, r_r, s_r, t_r, a_r \rangle \rangle$$

The length of the encoded machine, following this schema, would be  $l(\langle T \rangle) \leq 5lm + \log l + 1$ . ■

Since each Turing machine is composed by a finite set of quintuples, we can encode and list all the machines using a shortlex ordering. We associate each machine  $T$  with the index  $i$  corresponding to its position in this list, and we denote by  $T_i$  the  $i$ -th Turing machine. Each positive integer  $i$  encodes one, and only one, Turing machine. However, as Proposition 13.2.1 shows, all Turing machines have an infinite number of indexes. We associate each Turing machine with its smallest index.

**Proposition 13.2.1 — Padding Lemma.** Each Turing machine has infinitely many indexes.

*Proof.* Consider a Turing machine  $T_i$  encoded by the string  $\langle T_i \rangle$ . We can create a new encoding  $\langle T_j \rangle$  by appending a finite number of 0's to  $\langle T_i \rangle$ ,

such that  $\langle T_j \rangle = \langle T_i \rangle 0^n$  for some positive integer  $n$ . Since  $n$  can take on any positive integer value, there are infinitely many possible encodings  $\langle T_j \rangle$  for the same Turing machine  $T_i$ . ■

A universal Turing machine is a machine that can simulate the behavior of any other Turing machine on arbitrary input. The universal machine achieves this by reading both the description of the machine to be simulated (for instance, using the coding schema described in Example 13.4) and the input string for the computation from its own tape.

**Definition 13.2.1 — Universal Turing Machine.** A *Universal Turing Machine* is a Turing machine  $U$  such that  $U(\langle\langle T_i \rangle, s\rangle) = T_i(s)$  for all Turing machines  $T_i$  and all input strings  $s \in \mathcal{B}$ .

Naturally, we must prove that such a machine exists before we can utilize it. One could argue that a human being could decode the machine  $T_i$  and simulate its behavior with the input string  $s$ , and then refer to Theorem 13.1.1. A more rigorous approach would be to explicitly construct a universal Turing machine. However, providing a detailed description of one of these machines is beyond the scope of this book. Instead, we direct the reader to the references included at the end of the chapter for further exploration.

### 13.3 Non-Computable Problems

Turing machines enable us to delineate the set of problems that can be resolved through effective procedures or, in other words, by computers. It may be surprising to learn that numerous problems cannot be addressed using algorithms; such challenges lie beyond the computational capabilities of machines. We are not alluding to speculative queries like whether a computer can be intelligent or self-aware but to concrete, well-defined mathematical problems. We are also not referring to complex problems that demand a substantial amount of time to solve, as those, irrespective of their time consumption, remain computable.

One classic exemplar of non-computability is the *halting problem*. As illustrated in Algorithm 13.1, it involves a program or algorithm tasked with determining whether any given program (including itself) and input will eventually halt or continue to run indefinitely. Alan Turing proved that no algorithm can exist to solve this problem for all possible program-input pairs. This revelation wasn't a reflection on the limitations of technology or processing power but highlighted a profound theoretical limit intrinsic to computation.

The proposition below proves that the halting problem is non-computable.

**Algorithm 13.1** HALT function

---

```

procedure HALT( $A, I$ )
  if  $A(I)$  halts then
    return 1
  else
    return 0
  end if
end procedure

```

---

**Proposition 13.3.1 — Halting Problem.** Define HALT as in Algorithm 13.1. There does not exist a Turing machine that computes the HALT function for all possible pairs  $(A, I)$ , where  $A$  is a Turing machine and  $I$  is the input string to that machine.

*Proof.* The proof is by contradiction. Assume that the machine  $\text{HALT}$  exists, and define a new Turing machine  $\text{TC}$  such that  $\text{TC}(A) = 1$  if  $\text{HALT}(A, A) = 0$ , and  $\text{TC}(A)$  will never stop if  $\text{HALT}(A, A) = 1$ . Then the contradiction arises when we ask about the result of  $\text{TC}(\text{TC})$ : if  $\text{TC}(\text{TC})$  stops we have that  $\text{HALT}(\text{TC}, \text{TC}) = 0$  and that  $\text{TC}(\text{TC})$  should not stop, and if  $\text{TC}(\text{TC})$  does not stop then we have that  $\text{H}(\text{TC}, \text{TC}) = 1$  and thus  $\text{TC}(\text{TC})$  should stop. ■

The existence of such non-computable problems underscores the boundaries of mechanical computation. It illustrates that while Turing machines, and by extension, computers are profoundly powerful tools capable of solving an extensive array of problems, they are not omnipotent. A frontier of unsolvable problems exists, necessitating deeper exploration into the realms of mathematics, logic, and perhaps even philosophy to understand the inherent limits of computation.

The Halting Problem also has significant practical consequences in computer programming. For instance, it is impossible to write a program that can guarantee any other arbitrary program is bug-free or that all infinite loops with conditional exits will eventually halt for all possible inputs.

The next example introduces a well-defined, practical problem involving simple string manipulation that cannot be solved using computers.

■ **Example 13.5** Given two finite lists  $(\alpha_1, \dots, \alpha_n)$  and  $(\beta_1, \dots, \beta_n)$  of strings over some alphabet  $\Sigma$ , where  $d(\Sigma) \geq 2$ , the *Post Correspondence Problem* (PCP) asks to determine if there exists a sequence of  $K \geq 1$  indices  $(i_k)$ , with  $1 \leq i_k \leq n$  for all  $1 \leq k \leq K$ , such that  $\alpha_{i_1} \dots \alpha_{i_K} = \beta_{i_1} \dots \beta_{i_K}$ . For instance, given the sequences  $(a, ab, bba)$  and  $(baa, aa, bb)$ , a solution would be  $\alpha_3 \alpha_2 \alpha_3 \alpha_1 = \beta_3 \beta_2 \beta_3 \beta_1$ . No algorithm exists to solve PCP. Like many proofs of incomputability, the proof proceeds by showing that HALT can

be reduced to PCP, meaning if PCP is decidable, then the Halting Problem should be decidable as well. We will not detail the proof in this section; for interested readers, we refer to the references at the end of this chapter. ■

Non-computable problems are generally not derived directly from natural phenomena but from logical and mathematical constructs. To date, there are no known examples of non-computable problems manifesting plainly in natural phenomena. It's essential to distinguish between non-computability and unpredictability. Non-computable problems are those for which no algorithm can ever be created to solve them. In contrast, unpredictable systems (such as chaotic or complex systems) are theoretically computable but are unpredictable in practice due to factors like sensitivity to initial conditions or measurement precision.

## 13.4 Computable Functions and Sets

Each Turing machine  $T$  defines a function  $f_T : \mathcal{B}^* \rightarrow \mathcal{B}^*$  that assigns to each input string  $s \in \mathcal{B}^*$  an output string  $T(s) \in \mathcal{B}^*$ . The relationship between Turing machines and functions forms the basis for introducing the concept of a *computable function*.

**Definition 13.4.1** A function  $f : \mathcal{B}^* \rightarrow \mathcal{B}^*$  is *computable* if there exists a Turing machine  $T$  that defines the function  $f$  and halts for all the values of  $\mathcal{B}^*$ .

The terminology in computational theory can vary. While computable functions are occasionally referred to as *recursive functions*, this book opts for the term computable functions for consistency.

■ **Example 13.6** The function that assigns to each pair of natural numbers  $x$  and  $y$  their sum  $x + y$  is computable, as demonstrated in Example 13.1. ■

In real-world scenarios, certain functions don't provide a defined output for all possible inputs. Partial computable functions, characterized by Turing machines that don't halt for specific inputs, model these cases.

**Definition 13.4.2** A partial function  $f : \mathcal{B}^* \rightarrow \mathcal{B}^*$  is *partial computable* if there exists a Turing machine  $T$  that defines  $f$  for defined values and does not halt for undefined values.

The distinction between total computable functions and partial computable functions is significant in computability theory because it reflects the difference between problems that are always solvable by an algorithm (total) and those that are only solvable in some cases (partial).

■ **Example 13.7** The function  $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$  that assigns to each pair of

natural numbers  $x$  and  $y$  the number  $x - y$  is a partial computable function, since it is not defined in the case that  $x < y$ . Recall that according to our definition of Turing machine (see Definition 13.1.1), when the machine reaches an undefined configuration enters an infinite loop without ever halting.

■

We can expand the application of the principles of computability and partial computability to the domain of sets. We characterize sets through the lens of their characteristic functions that discern the membership of elements within the sets.

**Definition 13.4.3** A set  $A \in \mathcal{B}^*$  is *computable* if its characteristic function  $\chi_A$  is a total computable function. A set  $A \in \mathcal{B}^*$  is *computably enumerable* if its characteristic function  $\chi_A$  is a partial computable function, that is,  $\chi_A(a) = 1$  if  $a \in A$ , but  $\chi_A(a)$  is undefined if  $a \notin A$ .

The application of these concepts is illustrated through the example of the set of all Turing machines that halt for all inputs.

■ **Example 13.8** The set of all Turing machines that halt on all inputs, as demonstrated in 13.3.1, is not computable but is computably enumerable. ■

Next proposition provides an alternative characterization of computable sets.

**Proposition 13.4.1** A set  $A \in \mathcal{B}^*$  is computable if and only if  $A$  and its complement  $A^c$  are computably enumerable.

*Proof.* If  $A$  is computable, by definition, there exists a Turing machine that decides for any input  $x \in \mathcal{B}^*$  whether  $x \in A$  or  $x \notin A$ , halting in both cases. This implies that both  $A$  and its complement  $A^c$  can be enumerated by Turing machines. Thus, both  $A$  and  $A^c$  are computably enumerable.

Conversely, suppose  $A$  and  $A^c$  are computably enumerable. This means there exist two Turing machines,  $T_A$  and  $T_{A^c}$ , that enumerate the elements of  $A$  and  $A^c$ , respectively. To show that  $A$  is computable, construct a Turing machine  $T$  that, given an input  $x \in \mathcal{B}^*$ , simulates  $T_A$  and  $T_{A^c}$  in parallel to search for  $x$ . If  $x$  appears in the enumeration produced by  $T_A$ ,  $T$  halts and accepts  $x$  as an element of  $A$ . If  $x$  appears in the enumeration produced by  $T_{A^c}$ ,  $T$  halts and accepts  $x$ , indicating  $x \notin A$ . Since every element of  $\mathcal{B}^*$  must be in either  $A$  or  $A^c$  and both sets are computably enumerable,  $T$  will eventually halt for every input  $x$ , proving that  $A$  is computable. ■

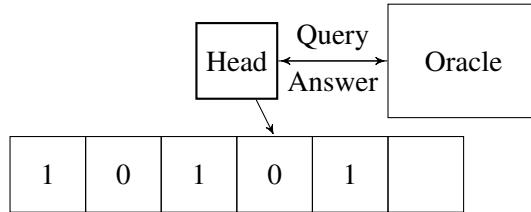


Figure 13.3: Oracle Turing Machine

## 13.5 Oracle Turing Machine

An oracle Turing machine (see Figure 13.3) is a theoretical model of computation that extends the capabilities of a standard Turing machine by providing it with an oracle. The oracle is a black box that can instantly compute certain answers, even for problems that are unsolvable or would take an impractical amount of time for a standard Turing machine to process. This model helps computer scientists and mathematicians explore the implications and boundaries of computational theory, including questions about complexity classes and the limits of what is computationally possible. The oracle Turing machine isn't a physical or implementable machine but rather a conceptual tool used in theoretical studies.

**Definition 13.5.1 — Oracle Turing Machine.** An *oracle Turing machine* with oracle set  $\mathcal{O}$  is a 8-tuple  $(Q, \Gamma, \sqcup, \Sigma, q_i, q_f, \tau, \mathcal{O})$  where:

- $Q$  is a finite, non-empty, set of *states*,
- $\Gamma$  is a finite, non-empty, set of *tape symbols*,
- $\sqcup \in \Gamma$  is the *blank symbol*,
- $\Sigma \subseteq \Gamma \setminus \sqcup$  is the set of *input symbols*,
- $q_o \in Q$  is the *initial state*,
- $q_f \in Q \setminus \{q_o\}$  is the *final state*,
- $\tau : (Q \setminus \{q_f\}) \times \Gamma \times \{0, 1\} \rightarrow Q \times \Gamma \times \{L, R, S\}$  is the *transition function*,
- $\mathcal{O} \subseteq \Sigma^*$  is the *oracle set*.

Building on the concept of a regular Turing machine (refer to Definition 13.1.1), an oracle Turing machine introduces the unique feature of an oracle set  $\mathcal{O}$ . This set comprises a subset of strings for which the oracle can instantly provide answers. The true strength of an oracle machine emerges when the set  $\mathcal{O}$  is non-computable; in cases where  $\mathcal{O}$  is computable, a regular Turing machine would be sufficient.

The transition function  $\tau$  for the oracle Turing machine is nuanced. At

each step the machine will query the oracle. Specifically, it sends the current string  $w$ —starting under the head and extending to the rightmost non-empty cell of the input tape—to the oracle. The oracle then responds with a binary answer: '1' if  $w$  is in  $\mathcal{O}$  or '0' if it isn't. The machine doesn't always utilize this response. There are steps where, despite receiving an answer, the computation proceeds unaffected by the oracle's response—essentially ignoring it. However, when the oracle's answer is pivotal, the machine makes a decision based on it. This decision could affect the next state, the next symbol to be written, or the next move (left, right, or stay). In essence, while the machine has the capability to continuously consult the oracle, it strategically chooses when to act on the information received. We could have modified the  $\tau$  function in such a way that the oracle is queried only when the answer is relevant for the computation, but that would require making important changes to the behaviour of the function, such as the introduction of new control states.

■ **Example 13.9** In the realm of theoretical computer science, an oracle can be invoked to "solve" the halting problem (refer to Theorem 13.3.1). The oracle is a hypothetical device or black box that, as if by magic, provides an instantaneous answer to a specific problem instance. The oracle set  $\mathcal{O}$  would consist of a collection of strings in the form  $\langle P, I \rangle$ , where  $P$  represents a program encoded as a string and  $I$  denotes its input. Given a program and its input, this oracle would instantly inform us whether the program halts on that input. Naturally, this concept is purely theoretical. No such oracle exists in reality, and the halting problem remains unsolvable in practical terms. ■

Turing machines are a subset of oracle Turing machines

**Proposition 13.5.1**

The machine is independent of the oracle set

**Proposition 13.5.2**

An oracle machine is characterized by two independent components: a particular Turing machine and a particular oracle set.

How to encode oracle Turing machines

■ **Example 13.10** ■

What means a function or set to be oracle computable

**Definition 13.5.2****13.6 Computational Complexity**

*Computational complexity* refers to the study of the efficiency of algorithms in terms of the resources they consume, such as time and space, to solve a given problem. By categorizing problems based on their inherent difficulty, computational complexity provides insights into the feasibility of solving problems within practical limits. It involves classifying problems into *complexity classes* offering a framework to analyze and compare the performance of algorithms. In this book we are interested mostly in the time required to solve a problem. We compute the running time of an algorithm as a function of the length of the string representing the input.

**Definition 13.6.1** Let  $M$  a Turing machine. The running time or *time complexity* of  $M$  is the function  $T_M : \mathbb{N} \rightarrow \mathbb{N}$ , where  $T_M(n)$  is the maximum number of steps that machine  $M$  takes for any input of length  $n$ .

*Big-O notation* is a mathematical notation that describes the upper bound of an algorithm's running time in the worst-case scenario, serving as a measure of its efficiency. Big-O notation provides a high-level analysis of an algorithm's performance, offering insights into how the running time requirements grow as the size of the input increases. It abstracts the details of the machine model and focuses on the most significant factors that contribute to the growth rate of the algorithm's complexity.

**Definition 13.6.2** Let  $f : \mathbb{N} \rightarrow \mathbb{N}$  and  $g : \mathbb{N} \rightarrow \mathbb{N}$  be two functions. We say that the function  $f$  is of order  $g$ , denoted  $f(n) = O(g(n))$ , if there exist constants  $c > 0$  and  $n_0 \geq 0$  such that for all  $n \geq n_0$ ,  $f(n) \leq c \cdot g(n)$ .

Big-O notation is instrumental in comparing different algorithms and selecting the most appropriate one for a given problem and dataset size.

■ **Example 13.11** Consider a polynomial function that represents the time complexity of an algorithm:

$$f(n) = 3n^3 + 2n^2 + 5n + 7$$

In the context of big-O notation, we are interested in the term with the highest growth rate as  $n$  approaches infinity, because it will eventually dominate the function. In this case, the term  $3n^3$  has the highest growth rate. Thus, we have

$$f(n) = O(n^3)$$

A complexity class consists of a collection of problems that are classified together due to their shared level of computational complexity. Each problem within a class shares similar characteristics of time complexity, ensuring a consistent measure of computational effort needed for resolution. These classes offer valuable insights and are integral in assessing and distinguishing the practical solvability and resource requirements of various computational challenges.

**Definition 13.6.3** A complexity class  $C$  is a collection of problems for which there exists a time bound function  $f(n)$  such that any problem  $P$  in  $C$  can be solved by a Turing machine  $M$  with time complexity  $T_M = f(n)$ , where  $n$  is the size of the input.

The class  $P$  consists of problems that can be solved in polynomial time by a Turing machine. In other words, for every problem in  $P$ , there exists an algorithm that can determine the solution in a number of steps that is a polynomial function of the size of the input. This class is foundational in computational complexity, serving as a baseline for measuring the efficiency of algorithms.

**Definition 13.6.4** The class problems that can be solved in polynomial time, denoted by  $P$ , is defined as the collection of problems for which there exists a Turing machine  $M$  that solve them and a polynomial  $p(n)$  such that the time complexity of  $M$  is  $p(n)$ .

Problems within  $P$  are considered tractable, meaning they can be practically solved even for large inputs. The concept of polynomial time solvability is crucial in distinguishing between problems that have efficient solutions in practice and those that do not.

■ **Example 13.12** A classic example of a problem in class  $P$  is the *shortest path problem*. In this problem, you're given a weighted graph and two vertices, and the objective is to find the path of minimum total weight between these two vertices. One common algorithm to solve this problem is Dijkstra's algorithm. It works by iteratively selecting the vertex with the smallest known distance from the start vertex, and updating the estimated distances to its neighbors. This algorithm runs in polynomial time, specifically  $O(V^2)$  for a graph with  $V$  vertices. Since there exists a polynomial-time solution to this problem it is in class  $P$ . ■

A *verifier* is a Turing machine that, given an input string and an additional string known as a *certificate* or witness, determines whether the input satisfies a particular property, resulting in a binary "yes" or "no" answer. The core idea behind a verifier is not necessarily to find a solution to a problem, but

rather to efficiently check or "verify" the validity of a proposed solution.

**Definition 13.6.5** A verifier is a Turing machine  $M$  that, given an input string  $x$  and an additional certificate string  $y$ , determines if  $x$  belongs to a certain subset  $S$  of strings. The machine satisfies two conditions:

- 1 If  $x$  belongs to subset  $S$ , there exists a certificate  $y$  such that  $V(x, y) = 1$ .
- 2 If  $x$  does not belong to subset  $S$ , then for every possible certificate  $y$ ,  $V(x, y) = 0$ .

Being the verifier a Turing machine, we can measure its time complexity, in terms of the length of the certificate. When a problem has solutions that can be verified in polynomial time using a verifier, it sheds light on the inherent complexity of that problem, situating it within specific computational classes.

**Definition 13.6.6** A problem is in the class  $NP$  if there exists a polynomial  $p(n)$  and a Turing verifier  $V$  such that for every instance  $x$ :

- 1 If  $x$  has a solution in subset  $S$ , then there exists a certificate  $y$  with length at most  $p(|x|)$  such that  $V$  confirms  $x$  as a valid instance of  $S$  using  $y$  in polynomial time.
- 2 If  $x$  does not have a solution in subset  $S$ , then for every possible certificate  $y$  with length at most  $p(|x|)$ ,  $V$  does not confirm  $x$  as a valid instance of  $S$ .

■ **Example 13.13** *TODO: Rewrite* A classic example of a problem in class  $NP$  is the *Traveling Salesman Problem* or TSP: given a list of cities and the distances between each pair of cities, the problem is to find the shortest possible route that visits each city exactly once and returns to the original city. To show that TSP is in  $NP$ , we don't need to demonstrate how to efficiently find the shortest route; we only need to show that, given a proposed route (or tour), we can efficiently verify whether that route satisfies the conditions of the problem and is shorter than or equal to a certain length. The verification process would be:

- Check if the proposed route visits each city exactly once and returns to the starting city. This can be done by traversing the proposed route and marking visited cities.
- Compute the total distance of the proposed route by summing up the distances between consecutive cities in the route.
- Compare the computed distance to the given threshold or limit.

Given a proposed solution (the route) and the distances between cities, a verifier can perform the above steps in polynomial time with respect to the number of cities, thereby verifying the solution's validity. In the context of the  $NP$  definition, the "certificate" for TSP would be the proposed route. If

TSP has a route shorter than or equal to a specific distance, then there exists a certificate (the route itself) that can be verified in polynomial time. If not, no such certificate would make the verifier confirm the solution. ■

The  $P = NP$  problem is one of the most fundamental questions in computer science. It asks whether every problem for which a potential solution can be verified quickly can also have its solution found quickly. In essence, if  $P$  equals  $NP$ , it would mean that finding solutions is as "easy" as checking them. Despite extensive study, no one has been able to prove that  $P = NP$  or  $P \neq NP$ , making it a longstanding open challenge in theoretical computer science.

**Theorem 13.6.1 — P=NP Problem.** The class of problems  $P$  is equal to the class of problems  $NP$ .

If  $P$  were to equal  $NP$ , it would signify a monumental shift in our understanding of computational tasks. Practically, many problems considered intractable today, especially those in cryptography, would become tractable. Secure communication over the internet relies on certain problems being hard to solve (like factoring large numbers). If  $P = NP$ , then encryption methods underpinning online security could be broken, potentially leading to a crisis in digital security. Conversely, numerous challenges in fields like optimization, drug discovery, and logistics could see groundbreaking solutions, revolutionizing industries. While it's tempting to view  $P = NP$  solely in the context of its challenges, especially to security, it would also herald unprecedented opportunities and advancements in multiple domains.

*Turing reducibility* is a fundamental concept in the realm of theoretical computer science and computability theory. It provides a means to compare the "computational difficulty" of problems. Specifically, a problem  $A$  is Turing reducible to a problem  $B$  if there exists an algorithm for solving  $A$  that may make use of a subroutine solving  $B$  an arbitrary number of times.

**Definition 13.6.7** Let  $A$  and  $B$  be two problems. We say that  $A$  is **Turing reducible** to  $B$ , denoted by  $A \leq_T B$ , if there exists a Turing machine that solves  $A$  by making use of one or more Turing machines that solve  $B$ .

If  $A$  is Turing reducible to  $B$  and vice versa, then the two problems are said to be Turing equivalent, indicating they have the same computational complexity.

■ **Example 13.14 TODO: rewrite.** Consider the problems of GRAPH ISOMORPHISM and SUBGRAPH ISOMORPHISM.

GRAPH ISOMORPHISM (GI): Given two graphs  $G_1$  and  $G_2$ , is there a one-to-one correspondence (isomorphism) between their vertices such that

the edges are preserved?

**SUBGRAPH ISOMORPHISM (SI):** Given two graphs  $G_1$  and  $G_2$ , does  $G_1$  contain a subgraph that is isomorphic to  $G_2$ ?

The GRAPH ISOMORPHISM problem is Turing reducible to the SUBGRAPH ISOMORPHISM problem. Here's the reasoning:

If we have an efficient solver for SUBGRAPH ISOMORPHISM, then we can use it to solve GRAPH ISOMORPHISM. For two graphs  $G_1$  and  $G_2$  of the same size, we ask the SI solver if  $G_2$  is a subgraph of  $G_1$  and vice versa. If both answers are affirmative, then  $G_1$  and  $G_2$  are isomorphic. Thus, an efficient solution for SI can be used to determine a solution for GI.

This example demonstrates the essence of Turing reducibility: by solving the more general problem (SUBGRAPH ISOMORPHISM), we inherently find a solution for the more specific problem (GRAPH ISOMORPHISM). ■

NP-hard

NP-Complete

## References

**TODO:** Briefly mention the historical emergence of the concept, including the works of pioneers like Alan Turing, Alonzo Church, and others.

The original paper from Alan Turing where the concepts of Turing machine, universal Turing machine, and non-computable problems were introduced is [Tur36], however it is a difficult to read paper for the contemporary reader. An easier to read introduction to computability theory, from the point of view of languages, can be found in [Sip12], and a more advanced introductions in [Coo03] and [Soa16]. In [Fer09] we can find a description of the most important computability models proposed so far. The Post Correspondence Problem was introduced by Emil Post in [Pos46]; for the details of the proof sketched in Example 13.5 please refer to [Sip12].

**TODO:** Here are a few seminal academic works on non-computability:

Turing, A. M. (1936). On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, s2-42(1), 230-265. Content: This foundational paper by Alan Turing introduces the concept of Turing machines, laying the groundwork for the theory of computation and establishing the halting problem's non-computability.

Church, A. (1936). An Unsolvable Problem of Elementary Number Theory. American Journal of Mathematics, 58(2), 345-363. Content: Alonzo Church presents the  $\lambda$ -calculus and establishes Church's Thesis, claiming that his formalism captures the intuitive notion of "computable," and proving the unsolvability of the Entscheidungsproblem.

Gödel, K. (1931). Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. Monatshefte für Mathematik, 38(1), 173-198. Content: Kurt Gödel's landmark paper where he introduces his incompleteness theorems, showing that within any sufficiently powerful mathematical system, there are statements that cannot be proven or disproven.

Post, E. (1946). A Variant of a Recursively Unsolvable Problem. Bulletin of the American Mathematical Society, 52(4), 264-268. Content: Emil Post presents a simpler, more accessible proof of unsolvability (non-computability) related to the halting problem and delves into the Post Correspondence Problem, a classic example of a non-computable problem.

Davis, M. (1958). Computability and Unsolvability. McGraw-Hill. Content: In this book, Martin Davis offers a comprehensive overview of the theory of computability and unsolvability, addressing topics like recursive functions, Turing machines, and Gödel's incompleteness theorems in an accessible manner.

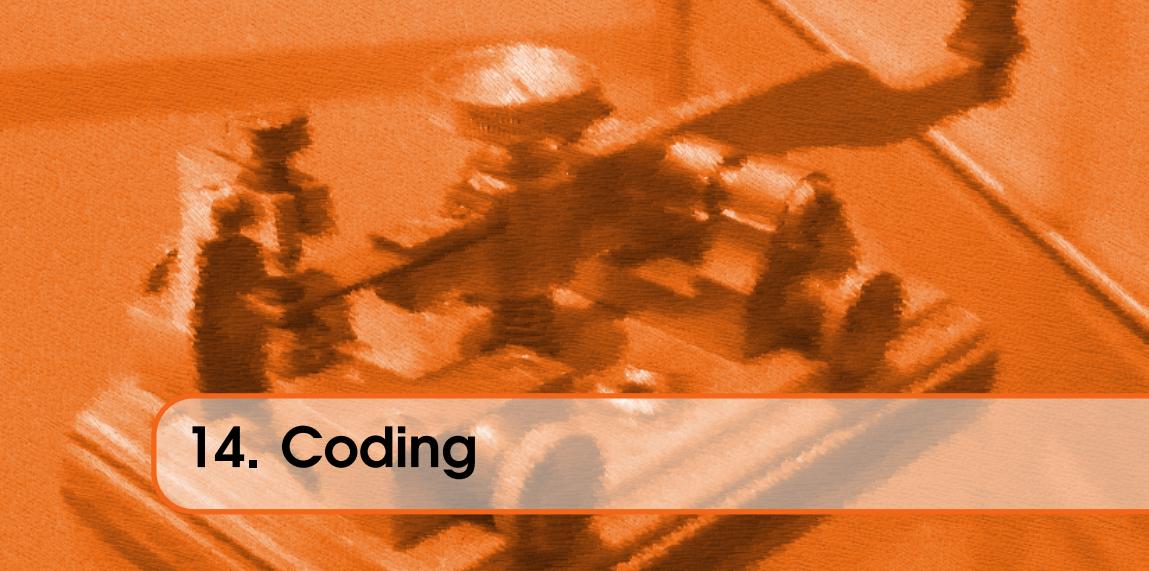
These references should provide a solid academic foundation for exploring the multifaceted and intriguing world of non-computability. Each offers unique insights and perspectives that collectively illuminate the complexity and depth of this area of study.

TODO: Add a reference to how to build a universal Turing machine.

TODO: Introduce other well-known non-computable problems besides the halting problem, like Turing's “Entscheidungsproblem” or the Busy Beaver function.

TODO: Relate non-computability to Gödel's incompleteness theorems to illustrate the inherent limitations in formal mathematical systems.

TODO: Explore philosophical discussions on the implications of non-computability for artificial intelligence and human cognition.



## 14. Coding

*Information is the resolution of uncertainty.*

Claude Shannon

In this section, we are going to review the conceptual ideas and main results behind coding theory and the related area of information theory.

Coding is the process of describing a sequence of symbols from some alphabet by a sequence of symbols from another alphabet. Coding has many practical applications, such as error detection, cryptography, and telecommunications. Here, our interest is in data compression, that is, encoding a message using fewer symbols than its original representation, without losing any information. Compression algorithms reduce the size of messages by identifying unnecessary elements and removing them, usually by means of computing and eliminating statistical redundancies. For example, data compression can be achieved by assigning shorter descriptions to the most frequent symbols from the source, and longer descriptions to the less frequent symbols. A particular type of codes, the prefix-free codes, will be discussed as playing a central role in this book. Prefix-free codes allow us to link coding theory with probability theory, a link that will be very useful in the context of the theory of nescience.

Information theory proposes that the amount of information we receive when some event happens is the negative logarithm of the probability of that event. In this sense, the theory assumes that information is equivalent to surprise: the more unlikely an event is, the more information we receive when the event occurs. We are not going to use that interpretation of information in our theory of nescience, but we will extensively use another concept from information theory: entropy. Entropy quantifies the amount of uncertainty involved in the value of a random variable or the outcome of a random process. Entropy is important to us because it establishes a limit to the compression of texts: it is not possible to find a code with an average word length smaller than the entropy of the source alphabet.

There exist many interesting concepts derived from entropy, like joint entropy, conditional entropy, or mutual information. However, these concepts are more relevant in the context of communication because they allow us to solve the problem of how to transmit information in a reliable manner over a noisy channel. Here, they are introduced for completeness purposes, and to compare them with our own definitions of joint nescience and conditional nescience.

## 14.1 Coding

Intuitively, coding refers to the process of losslessly describing a sequence of symbols (a message) coming from some alphabet by other sequences of symbols coming from a (potentially) different alphabet. There is no general agreement about what exactly a code is, as different authors propose different definitions in the literature. Fortunately, the definition of a prefix-free code, the kind of codes used in the theory of nescience, is a standard one.

Let  $\mathcal{S} = \{s_1, s_2, \dots, s_q\}$  be a finite set called *source alphabet*, and  $\mathcal{X} = \{x_1, x_2, \dots, x_r\}$  a finite set called *code alphabet*.

**Definition 14.1.1 — Code.** A *code* for  $\mathcal{S}$  is a total function  $C : \mathcal{S} \rightarrow \mathcal{X}^+$ .

If  $(s, x) \in C$  we say that  $s$  is the *source symbol* and  $x$  is the *code word*. If  $C$  is an injective function we say that the code is *nonsingular*.

Nonsingularity allows us to unambiguously describe the individual symbols of the source alphabet. For the rest of this book, whenever we talk about a code we mean a nonsingular code. Moreover, without any loss of generality, we will restrict ourselves to *binary codes*, that is,  $\mathcal{X} = \mathcal{B} = \{0, 1\}$ .

The property of nonsingularity can also be applied to strings of symbols. In order to do that, we have to extend the concept of code from symbols to strings.

**Definition 14.1.2** The *extension of order n* of a code  $C$  is a function  $C^n : \mathcal{S}^n \rightarrow \mathcal{B}^+$  defined as  $C^n(s_{i_1} \dots s_{i_n}) = C(s_{i_1}) \dots C(s_{i_n})$ , where  $C(s_{i_1}) \dots C(s_{i_n})$  is the concatenation of the code words corresponding to the symbols of the string  $s_{i_1} \dots s_{i_n} \in \mathcal{S}^n$ . An extension of order  $n$  of a code  $C$  is *nonsingular* if the function  $C^n$  is injective.

If it is clear from the context, we will also use the word *code* to refer to a nonsingular extension of order  $n$  of a code, and the elements of  $\mathcal{S}^n$  will be called *source words*.

■ **Example 14.1** The code  $C(a) = 0$ ,  $C(b) = 00$ ,  $C(c) = 01$ , and  $C(d) = 11$  is a nonsingular code, but its extension of order 2 is singular, since, for example,  $C^2(ab) = C^2(ba) = 000$ . ■

As we have seen in Example 14.1 not all nonsingular codes have non-singular extensions, that is, it might happen that we are not able to decode the original messages given their encoded versions. Unique decodability is a highly desirable property of codes.

**Definition 14.1.3** A code  $C$  is called *uniquely decodable* if its order  $n$  extension  $C^n$  is nonsingular for all  $n$ .

Next proposition provides a characterization of the unique decodability of codes.

**Proposition 14.1.1** A code  $C$  is uniquely decodable if, and only if, the function  $C^+ : \mathcal{S}^+ \rightarrow \mathcal{B}^+$  is injective.

*Proof.* If the function  $C^+$  is injective, then the restriction to  $C^n$  must be injective for all  $n$ . Now, let us assume that  $C^n$  is nonsingular for all  $n$  and let's prove that  $C^+$  must be nonsingular by contradiction: select two source words  $s_1 \in \mathcal{S}^n$  and  $s_2 \in \mathcal{S}^m$ ,  $n \neq m$ , and assume that they have the same code word  $C^+(s_1) = C^+(s_2)$ . Then construct the words  $s_3 = s_1 s_2$  and  $s_4 = s_2 s_1$ , both  $s_3$  and  $s_4$  have the same length, and  $C^+(s_3) = C^+(s_4)$ , which is a contradiction with the fact that  $C^{n+m}$  must be nonsingular. ■

■ **Example 14.2** The code  $C(a) = 0$ ,  $C(b) = 01$ ,  $C(c) = 011$ , and  $C(d) = 0111$  is a uniquely decodable code. For example, the code word 0010011 uniquely corresponds to the source word *abac*. Unique decodability is achieved because the 0 symbol plays the role of a delimiter, separating code words. ■

The Sardinas-Patterson theorem provides a necessary and sufficient condition for a code to be uniquely decodable. The theorem is based on an algorithmic approach, enabling the examination of a code's unique decodability through a sequence of iterative steps. Let  $C_0$  denote the set of code

words of a code  $C$ . We define the set  $C_n$  as

$$C_n = C^{-1}C_{n-1} \cup C_{n-1}^{-1}C$$

for all  $n \in \mathbb{N}$  and where  $C^{-1}C_{n-1}$  represents the left quotient of  $C$  and  $C_{n-1}$ . And let  $C_\infty$  be defined as

$$C_\infty = \bigcup_{n=1}^{\infty} C_n$$

**Proposition 14.1.2 — Sardinas-Patterson.** A code  $C$  is uniquely decodable if and only if the sets  $C_0$  and  $C_\infty$  are disjoint.

*Proof.* TODO: Pending ■

■ **Example 14.3** Given the code of Example 14.1 we have that

$$\begin{aligned} C_0 &= \{0, 00, 01, 11\} \\ C_1 &= \{0, 1\} \end{aligned}$$

And since  $C_0 \cap C_1 \neq \emptyset$  we can conclude that the code is not uniquely decodable.

Given the code of Example 14.2 we have that

$$\begin{aligned} C_0 &= \{0, 01, 011, 0111\} \\ C_1 &= \{1, 11, 111\} \\ C_2 &= \emptyset \end{aligned}$$

And since  $C_0 \cap C_\infty = C_0 \cap \bigcup_{n=1}^{\infty} C_n = C_0 \cap C_1 = \emptyset$  we can conclude that the code is uniquely decodable. ■

Next definition introduces the concept of prefix-free codes. Prefix-free codes will play a critical role in the computation of the amount of algorithmic information of an arbitrary string (described in Chapter 15), and in our own theory of nescience. Prefix-free codes also allow us to link coding theory and probability theory through the Kraft inequality (Theorem 14.2.1). Note that we prefer the name *prefix-free code* over the more standard *prefix code*, since the former more accurately describes the concept.

**Definition 14.1.4 — Prefix-free Code.** A code  $C$  is *prefix-free* if for all  $i, j$  where  $1 \leq i, j \leq q$  and  $i \neq j$ ,  $C(s_i)$  is not a prefix of  $C(s_j)$ .

Note that the fact that a string is a prefix of itself does not violate the prefix-free property because the condition specifically excludes considering a string as a prefix of itself ( $i \neq j$ ) for the purpose of determining whether a set of code words is prefix-free.

■ **Example 14.4** The code  $C(a) = 0$ ,  $C(b) = 10$ ,  $C(c) = 110$  and  $C(d) = 1110$  is a prefix-free code. Here, the 0 symbol plays the role of a comma as it was the case of Example 14.2, but its new position at the end of the code words is what makes the code prefix-free. ■

Prefix-free codes are uniquely decodable, as the next proposition proves.

**Proposition 14.1.3** Let  $C$  be a prefix-free code, then  $C$  is uniquely decodable.

*Proof.* Let  $C$  be a prefix-free code, and assume that  $C$  is not uniquely decodable. This implies there exist two different sequences of source symbols, say  $r$  and  $s$ , such that they encode to the same sequence of code words:  $C(r) = C(s)$ . Since  $C$  is prefix-free, the start of any code word sequence uniquely determines the first code word, as no other code word can be a prefix. Therefore, the first symbols in  $r$  and  $s$  must be the same, as they are encoded into the same first code word. However, this leads to the conclusion that  $r = s$ , contradicting our initial assumption that  $r$  and  $s$  are different. Therefore, our assumption must be false, and the code  $C$  must be uniquely decodable. ■

From an engineering point of view it is highly convenient to have codes whose source symbols can be decoded as soon as the corresponding code words are received, that is, it is not necessary to wait for the next code word in order to decode the current symbol, this is why some authors refer to these codes as *instantaneous codes*.

**Definition 14.1.5** A code  $C$  is *instantaneous* if, for any order  $n$  and for any sequence of code words  $C(s_{i_1}), C(s_{i_2}), \dots, C(s_{i_n})$ , each sequence of code words  $\mathbf{t} = C(s_{i_1})C(s_{i_2})\dots C(s_{i_m})\dots$  can be uniquely decoded as  $\mathbf{s} = s_{i_1}s_{i_2}\dots s_{i_m}\dots$  without ambiguity, regardless of the continuation of  $\mathbf{t}$ .

■ **Example 14.5** The code described in Example 14.2 is not instantaneous. For example, after receiving the sequence 011 the source symbol could be a  $c$  if the next symbol is a 0 or a  $d$  if it is a 1. ■

Prefix-free codes and instantaneous codes are essentially two terms for the same concept. A prefix-free code is one in which no code word is a prefix of any other code word. This property ensures that there is a clear demarcation between code words when they are concatenated in a sequence, allowing each code word to be decoded immediately upon receipt without the need to look ahead to subsequent code words for context.

**Proposition 14.1.4** A code  $C$  is instantaneous if, and only if, it is a prefix code.

*Proof.* Assume  $C$  is an instantaneous code, and suppose that  $C$  is not a prefix-free code. This means there exists at least a pair of code words, say  $C(s_i)$  and  $C(s_j)$ , such that  $C(s_i)$  is a prefix of  $C(s_j)$ . Consider a sequence where  $C(s_j)$  is transmitted. Since  $C(s_i)$  is a prefix of  $C(s_j)$ , the decoder would decode  $C(s_i)$  from the initial part of  $C(s_j)$ , leading to ambiguity, as the rest of  $C(s_j)$  could be seen as another code or part of the next code. This contradicts the assumption that  $C$  is instantaneous.

Assume  $C$  is a prefix code. This property ensures that once a code word is identified in a sequence, there can be no confusion as to where it ends, and the next code word begins. There is no need to look ahead to determine the boundary between code words, as the end of one code word cannot be mistaken for the beginning of another. Hence, a prefix code allows for instantaneous decoding. ■

The last type of codes we are going to review are fixed length codes. We will use fixed length codes to compute the length of a text when there are no regularities we can exploit to compress it.

**Definition 14.1.6** If all the code words of a code have the same length we say that the code is a *fixed length code*.

Fixed length codes have the property of being prefix-free (or instantaneous).

**Proposition 14.1.5** Let  $C$  be a fixed-length code, then  $C$  is prefix-free.

*Proof.* Let  $C$  be a fixed length code, and  $C(s_i)$  and  $C(s_j)$  the code words of two arbitrary source words  $s_i$  and  $s_j$ . Assume that  $C(s_i) <_p C(s_j)$ , given the fact that  $l(C(s_i)) = l(C(s_j))$  we have that  $C(s_i) = C(s_j)$  and so, the code  $C$  is prefix-free. ■

Of course, the converse of the previous proposition does not hold; not all prefix-free codes are of fixed length. The code described in Example 14.4 is prefix-free but it is not fixed-length.

Figure 14.1 provides a graphical representation of the relation among the different types of codes that have been introduced in this section.

## 14.2 Kraft Inequality

The Kraft inequality provides a condition for the existence of a prefix-free code given a set of code word lengths. Kraft's inequality states that for a given set of code word lengths in a binary code, the sum of the reciprocals of the powers of two corresponding to the code word lengths must be less than or equal to one. This condition is both necessary and sufficient; not only

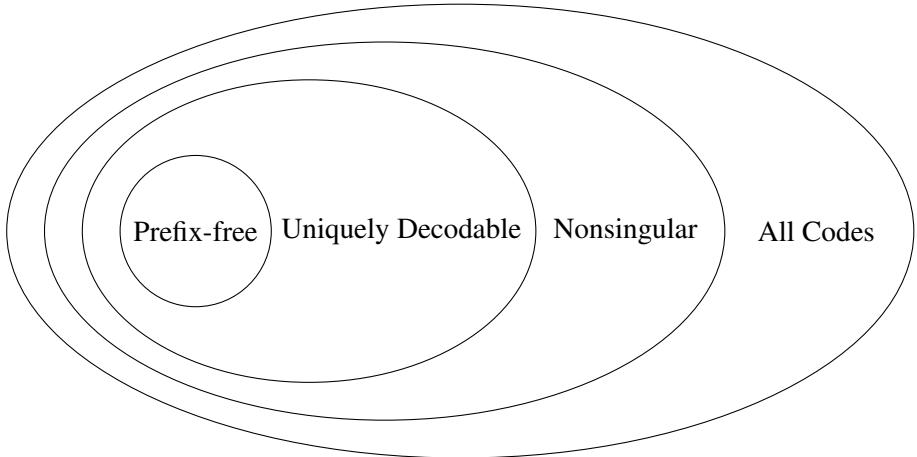


Figure 14.1: Classification of Codes

must any prefix-free code satisfy this inequality, but given a set of lengths that meets this condition, it is always possible to construct a corresponding prefix-free code. The elegance and utility of Kraft's inequality lie in its ability to link the lengths of code words with the mathematical properties of probability.

**Theorem 14.2.1 — Kraft Inequality.** Let  $\mathcal{L} = \{l_1, l_2, \dots, l_q\}$  be a set of lengths, where  $l_i \in \mathbb{N}$ , then there exists a binary prefix-free code  $C$  whose code words have the lengths of  $\mathcal{L}$  if, and only if,

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} \leq 1$$

*Proof.* Consider a binary tree whose branches are labeled with the symbols of the code alphabet, in such a way that the path from the root to the leaves traces out the symbols of a code word. The prefix-free condition implies that nodes representing complete code words cannot have descendants. An example of such a tree, for the code described in Example 14.4, is shown in Figure 14.2.

Let  $l_{max} = \max \{l_1, l_2, \dots, l_q\}$ , that is, the length of the longest code word from the set of lengths. There will be at most  $2^{l_{max}}$  leaf nodes in the tree, but at level  $l_i$  we have to prune  $2^{l_{max}-l_i}$  leaves, since the code is prefix-free. Summing over all the code words' lengths, we have that the total number of pruned leaves must be less than or equal to the maximum number of leaves,

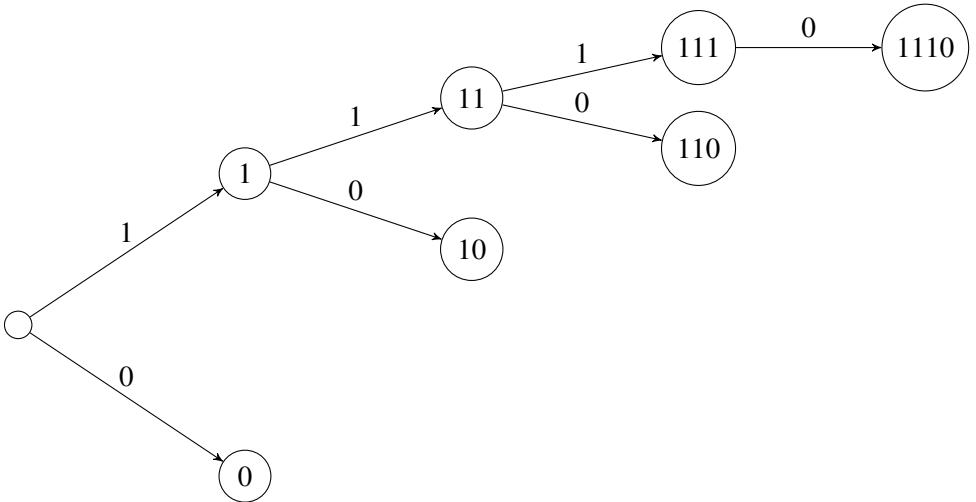


Figure 14.2: Prefix-free Tree

that is

$$\sum_{l_i \in \mathcal{L}} 2^{l_{max} - l_i} \leq 2^{l_{max}}$$

or, equivalently,

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} \leq 1$$

which is exactly the inequality we are trying to prove.

Conversely, given any set of code words' lengths  $\mathcal{L} = \{l_1, l_2, \dots, l_q\}$  that satisfy the Kraft inequality, we can always construct a binary tree, like the one in the Figure 14.2. Label the first node (lexicographically) of depth  $l_1$  as code word 1, and remove its descendants from the tree. Then label the first remaining node of depth  $l_2$  as code word 2, and so on. Proceeding this way, we construct a prefix code with the specified lengths. ■

Given a code  $C$  whose code word lengths  $\mathcal{L}$  satisfy the Kraft inequality does not necessarily mean that the code is prefix-free, since what the inequality states is that there exists a prefix-free code with those word lengths, not that all codes with those word lengths are prefix-free.

■ **Example 14.6** The code  $C(a) = 0$ ,  $C(b) = 111$ ,  $C(c) = 110$  and  $C(d) = 100$  satisfies the Kraft inequality, but it is not prefix-free. ■

McMillan's Inequality enriches our understanding of coding theory by establishing a crucial link between the lengths of code words in uniquely

decodable codes. This inequality mirrors Kraft's Inequality, yet it broadens the scope to encompass all uniquely decodable codes, not just those that are prefix-free. By demonstrating that the sum of the reciprocals of the powers of two for code word lengths must be less than or equal to one for a code to be uniquely decodable, McMillan's Inequality ensures that such codes can be constructed with a given set of lengths.

**Theorem 14.2.2 — McMillan's Inequality.** Let  $\mathcal{L} = \{l_1, l_2, \dots, l_q\}$  be a set of lengths, where  $l_i \in \mathbb{N}$ , then there exists a uniquely decodable code  $C$  whose code words have the lengths of  $\mathcal{L}$  if, and only if,

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} \leq 1$$

*Proof.* TODO: review

We must show that if there exists a uniquely decodable code with code word lengths  $l_1, l_2, \dots, l_q$ , then the inequality holds.

Suppose  $C$  is a uniquely decodable code with code word lengths  $l_1, l_2, \dots, l_q$ . Consider a sequence of code words produced by  $C$  that is long enough to contain any possible concatenation of  $n$  code words (for some  $n$  large enough). Let's denote the set of all such sequences as  $S_n$ .

Because  $C$  is uniquely decodable, each sequence in  $S_n$  corresponds to a unique sequence of source symbols. The number of different sequences in  $S_n$  is at least  $2^{nl}$ , where  $l$  is the smallest length among all  $l_i$ . This is because, for the shortest code word, we can consider a sequence of length  $nl$  and all possible fillings with the shortest code word. Since the code is uniquely decodable, none of these sequences can be the same as any sequence containing longer code words.

The total number of bits required to represent all sequences in  $S_n$  can indeed be calculated as  $(2^{-l_1} + 2^{-l_2} + \dots + 2^{-l_q})^n$ , considering the combinatorial possibilities of all code word concatenations.

Since each sequence in  $S_n$  must be different (uniquely decodable) and can be represented in a binary tree (where each leaf corresponds to a code word and the length of the path to the leaf corresponds to the length of the code word), the number of sequences in  $S_n$  cannot exceed  $2^{nl_{max}}$ , where  $l_{max}$  is the length of the longest code word.

Therefore, we have:

$$(2^{-l_1} + 2^{-l_2} + \dots + 2^{-l_q})^n \leq 2^{nl_{max}}$$

Taking the  $n$ th root of both sides, we have:

$$2^{-l_1} + 2^{-l_2} + \dots + 2^{-l_q} \leq 2^{l_{\max}}$$

Since  $2^{l_{\max}}$  is always less than or equal to 1 (because there must be at least one code word of the maximum length and thus occupying the "full" length of the binary tree), we obtain:

$$2^{-l_1} + 2^{-l_2} + \dots + 2^{-l_q} \leq 1$$

To prove sufficiency, we must show that if the inequality holds, then there exists a uniquely decodable code with those lengths.

If the inequality holds, we can use the Kraft construction to build a prefix-free code, which is always uniquely decodable, with code word lengths  $l_1, l_2, \dots, l_q$ . The construction starts by assigning the shortest code word first and ensuring no other code word is a prefix of any other. Since prefix-free codes are a subset of uniquely decodable codes, we have constructed the required code.

Therefore, by satisfying the inequality, we can construct a uniquely decodable code, proving that the condition is sufficient. ■

In the theory of nescience, we are primarily interested in prefix-free codes. This preference may seem to impose a limitation, as it appears more rational to consider the broader category of uniquely decodable codes. However, this perceived limitation does not actually exist. As the following theorem demonstrates, uniquely decodable codes also satisfy Kraft's inequality. This means that for any uniquely decodable code, there exists a prefix-free code with exactly the same code word lengths, ensuring no compromise in the generality of our analysis when focusing on prefix-free codes.

**Corollary 14.2.3** There is an instantaneous (prefix-free) code with word lengths  $l_1, \dots, l_q$  if and only if there is a uniquely decodable code with these word lengths.

*Proof.* **TODO: Review** This corollary is a direct consequence of McMillan's Inequality, which is a counterpart to Kraft's Inequality for uniquely decodable codes. McMillan's Inequality states that for a set of code word lengths  $\{l_1, l_2, \dots, l_q\}$ , a uniquely decodable code exists if and only if the sum of  $2^{-l_i}$  for all  $i$  is less than or equal to one. Given that prefix-free codes are a subset of uniquely decodable codes and also satisfy this condition, the existence of a uniquely decodable code with given word lengths implies the

possibility of constructing a prefix-free code with the same lengths. The reverse is inherently true by the definition of prefix-free codes, which are always uniquely decodable. ■

In the context of nescience, our interest lies not in the specific codes themselves but in the lengths of the code words. This emphasis allows us to abstract away from the details of code construction to concentrate on the mathematical properties and implications of these lengths, which are central to understanding and applying the principles of nescience.

### 14.3 Optimal Codes

In coding theory, a compact code is an optimal encoding strategy that minimizes the expected length of code words, a concept central to evaluating a code's efficiency. The expected length of a code is determined by the weighted average of the lengths of its code words, with weights corresponding to the probability distribution of the source symbols. By designing code words so that more frequent symbols are assigned shorter lengths and less frequent ones longer lengths, compact codes effectively reduce the average size needed to encode information. This principle is pivotal in data compression, as it allows for a significant reduction in the space required for storage or the bandwidth needed for transmission.

Let's establish  $\mathcal{S} = \{s_1, s_2, \dots, s_q\}$  as a finite source alphabet, with  $P$  denoting a probability distribution defined over the elements of  $\mathcal{S}$ .

**Definition 14.3.1** The *expected length* of a code  $C$ , denoted by  $L_C$ , is the weighted sum of the lengths of the code words, calculated as

$$L_C = \sum_{i=1}^q P(s_i)l_i,$$

where  $\mathcal{L} = \{l_1, l_2, \dots, l_q\}$  represents the lengths of the code words of  $C$ . We may simply use  $L$  to denote  $L_C$  when the code  $C$  is understood from the context.

Our goal is to identify a code  $C$  that minimizes the expected length  $L$  of the codewords  $\mathcal{L}$  under the given probability distribution  $P$ . Such an optimal code  $C$  would enable us to compress messages composed of the source alphabet  $\mathcal{S}$  effectively, reducing the number of symbols required to encode these messages.

Next, the following definition formalizes what it means for a code to be compact, which is a desirable attribute for efficient data encoding.

**Definition 14.3.2** A code  $C$  is *compact* if its average length  $L$  is less than or equal to the average length of all the other codes for the same source alphabet and code alphabet.

**Definition 14.3.3** A code  $C$  is *compact* if, for a given source alphabet  $\mathcal{S}$  and a corresponding probability distribution, its expected length  $L_C$  is minimized among all possible codes that can be constructed for  $\mathcal{S}$ .

This means  $C$  achieves the lowest possible weighted average codeword length, where each codeword length is weighted by the probability of its corresponding source symbol.

The existence of compact codes for all possible source alphabets is a foundational aspect of coding theory, suggesting that for every finite source alphabet, an optimally efficient encoding scheme can be devised. It is not entirely clear that compact codes exist for all possible source alphabets, so the following proposition must be proven.

**Proposition 14.3.1** For every finite source alphabet  $\mathcal{S} = \{s_1, s_2, \dots, s_q\}$  with a defined probability distribution over its symbols, there exists at least one code  $C$  that is compact.

*Proof.* The proof of this proposition involves constructing or identifying a code  $C$  for the source alphabet  $\mathcal{S}$  that achieves the minimum possible expected length  $L_C$ . This can be demonstrated through the use of Huffman's algorithm, which is designed to produce an optimal prefix code based on the probabilities of the source symbols. By definition, the Huffman code for a given probability distribution over  $\mathcal{S}$  minimizes the expected length of the codewords, thereby proving the existence of a compact code for any finite source alphabet with a given probability distribution. ■

This clarification not only completes the explanation but also aligns the definition of compactness with established principles in coding theory. It provides a more rigorous basis for discussing the efficiency of encoding schemes and underscores the role of probability distributions in determining the compactness of a code.

**TODO: Introduce this concept**

**Definition 14.3.4** The redundancy of a code is defined as

$$\eta = \frac{H((S))}{L}$$

Of course, our goal is to minimize the redundancy of codes.

Next theorem states that the entropy of the probability distribution  $P$  poses a limit to the average length of prefix-free codes.

**Theorem 14.3.2** The expected length  $L_C$  of any prefix-free  $r$ -ary code, given the probability distribution  $P$ , is greater than or equal to the entropy of  $P$ , that is

$$H_r(P) \leq L_C$$

with equality if, and only if,  $r^{-l_i} = P_i$  for all  $0 \leq i \leq q$ .

*Proof.*



**Definition 14.3.5** A probability distribution is called *D-adic* if each of the probabilities is equal to  $D^n$  for some  $n$ .

**Corollary 14.3.3** We have the equality in the theorem if, and only if, the distribution of  $X$  is D-adic.

*Proof.* TODO



Mention that in practice we will non-integer code word lengths. Show that, on average, the length of the encoded string will be less than 1 bit than using a code with code words with integer lengths

Kraft's inequality allows us to compare how efficient different codes are for the same source alphabet.

**Definition 14.3.6** Let  $C_1 : \mathcal{S} \rightarrow \mathcal{X}^+$  and  $C_2 : \mathcal{S} \rightarrow \mathcal{X}^+$  be two different codes. We say that code  $C_1$  is *more efficient* than code  $C_2$  if for all  $s \in \mathcal{S}$  we have that  $l(C_1(s)) \leq l(C_2(s))$ , and there exists at least one  $s' \in \mathcal{S}$  such that  $l(C_1(s')) < l(C_2(s'))$ .

■ **Example 14.7** The code described in Example 14.6 is more efficient than the code of Example 14.4. Of course, the problem with the code described in Example 14.6 is that it is not prefix-free, but since it satisfies Kraft's inequality, we know that there must exist another code with the same code word lengths that is prefix-free. For example,  $C(a) = 0$ ,  $C(b) = 10$ ,  $C(c) = 110$  and  $C(d) = 111$ . ■

We are interested in the most efficient possible codes.

TODO: This definition is wrong

**Definition 14.3.7** A code  $C$  is *complete* if there does not exist a code  $C'$  that is more efficient than  $C$ .

It turns out that Kraft's inequality provides a very useful characterization of complete codes.

**Proposition 14.3.4** A code  $C$  is complete if, and only if, its code word lengths  $\mathcal{L} = \{l_1, l_2, \dots, l_q\}$  satisfy the property:

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} = 1$$

*Proof.* Suppose the sum of the exponentiated inverses of the code word lengths equals 1:

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} = 1$$

This implies that the code utilizes the full capacity of the coding space. In other words, there is no redundancy or additional space in the code that could be used to encode a symbol with a shorter length. This is because each term in the sum  $2^{-l_i}$  represents the proportion of the coding space taken up by the code word of length  $l_i$ . If these proportions sum up to 1, then all possible code words of the given lengths are used, and there's no space left to introduce any new code word without increasing the length of at least one existing code word. Therefore, the code is as efficient as possible, hence complete.

Now suppose that the code  $C$  is complete. This means that there is no other code  $C'$  with the same source alphabet and the same or shorter code word lengths that is more efficient than  $C$ .

Assume for contradiction that the sum of the exponentiated inverses is less than 1:

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} < 1$$

This would imply that there is still unused coding space, meaning that it would be possible to construct a more efficient code by using this unused space to represent at least one symbol with a shorter code word, contradicting the assumption that  $C$  is complete. Therefore, the sum cannot be less than 1.

Since  $C$  is complete and cannot be less efficient, the only alternative left is that the sum is exactly 1:

$$\sum_{l_i \in \mathcal{L}} 2^{-l_i} = 1$$

Combining both directions, we have shown that a code  $C$  is complete if and only if the sum of the exponentiated inverses of its code word lengths equals 1, as stated in the proposition. ■

## 14.4 Entropy

In this section we are going to introduce the concept of *entropy*, as a measure of the uncertainty of a random variable. Entropy is a very difficult to grasp concept that can be applied in many different contexts, such as communications, statistics, finance, etc. Here we are interested in entropy because it will allow us to identify codes with the shortest possible average length.

Let  $A = \{a_1, a_2, \dots, a_n\}$  a finite set, and  $X$  a random variable defined over the set  $A$  with probability mass function  $p(a)$ .

**Definition 14.4.1 — Entropy.** The *entropy* of the random variable  $X$ , denoted by  $H(X)$  and measured in *bits*, is defined as:

$$H(X) = \sum_{a \in A} p(a) \log \frac{1}{p(a)}$$

Note that the entropy of  $X$  does not depend on the individual elements of  $A$ , but on their probabilities. It is easy to show that  $H(X) \geq 0$  since  $0 \leq p(a) \leq 1$  implies that  $-\log p(a) \geq 0$ . In case of  $p(a_i) = 0$  for some  $i$ , the value of the corresponding summand  $0 \log 0$  is taken to be 0, which is consistent with the limit  $\lim_{p \rightarrow 0^+} p \log p = 0$ . If we change the base of the logarithm to  $u$ , entropy will be scaled by a factor of  $\log_u 2$  (see Equation B.1).

■ **Example 14.8** Let  $X$  a random variable defined over the set  $A = \{a_1, a_2\}$ , with values  $p(a_1) = q$  and  $p(a_2) = 1 - q$ . Then, the entropy of  $X$  is given by:

$$H(X) = q \log \frac{1}{q} + (1 - q) \log \frac{1}{1 - q}$$

Figure 14.3 shows the entropy of  $X$  for different values of  $q$ . If  $q = 0$  or  $q = 1$  the entropy is 0, that is, there is no uncertainty about which value of  $A$  we will get. The maximum value of  $H$  is 1, and it is reached when  $q = 1/2$ ; that is, we could say that 1 bit is the uncertainty associated to two equally probable symbols. ■

Next proposition shows that the maximum value for entropy is the logarithm of the number of symbols of  $A$ , and that this value is reached when all the symbols have the same probability.

**Proposition 14.4.1** Given the random variable  $X$  we have that  $H(X) \leq \log n$ , and  $H(X) = \log n$  if, and only if,  $p(a_1) = p(a_2) = \dots = p(a_n)$ .

*Proof.* Consider the expression:

$$\log n - H(X) = \sum_{i=1}^n p(a_i) \log n - \sum_{i=1}^n p(a_i) \log \frac{1}{p(a_i)} = \sum_{i=1}^n p(a_i) \log np(a_i)$$

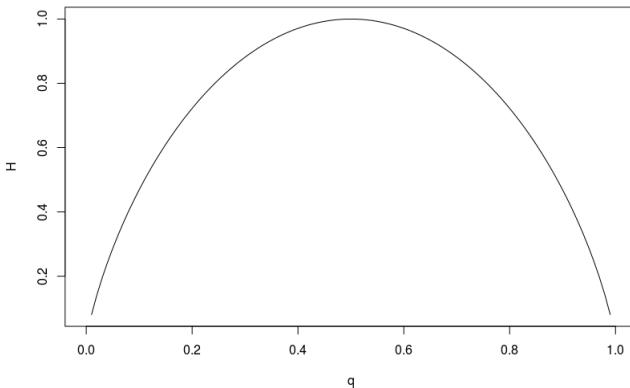


Figure 14.3: Binary Entropy Function

Applying property B.1 we have that:

$$\log n - H(X) = \log e \sum_{i=1}^n p(a_i) \ln np(a_i)$$

And applying property B.2 (equalling  $x = 1/n p(a_i)$ ):

$$\log n - H(X) \geq \log e \sum_{i=1}^n p(a_i) \left(1 - \frac{1}{np(a_i)}\right) \geq \log e \left(\sum_{i=1}^n p(a_i) - \frac{1}{n} \sum_{i=1}^n \frac{p(a_i)}{p(a_i)}\right) \geq 0$$

Which proves that  $H(X) \leq \log n$ .

The inequality becomes an equality if, and only if,  $p(a_i) = 1/n$  (given that the inequality B.2 becomes an equality if, and only if,  $x = 1$ ). ■

■ **Example 14.9** If we choose a random symbol from  $A$  according to the probability mass function  $p$ , entropy would be the minimum expected number of binary questions (Yes/No questions) required to identify the selected symbol. If the symbols of  $A$  are equiprobable, the expected number of questions is maximal and equal to  $\log d(A)$ . ■

We can extend the concept of entropy to a pair of random variables by means of using the joint probability mass function. In this way, the joint entropy will be a measure of the uncertainty associated to both variables. Let  $A = \{a_1, a_2, \dots, a_n\}$  and  $B = \{b_1, b_2, \dots, b_m\}$  two finite sets, and  $X$  and  $Y$  two random variables defined over the sets  $A$  and  $B$  respectively, with probability mass function  $p(a)$  and  $p(b)$ , and joint probability mass function  $p(a, b)$ .

**Definition 14.4.2** The *joint entropy* of the random variables  $A$  and  $B$ , denoted by  $H(A, B)$ , is defined as:

$$H(A, B) = \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a, b)}$$

Since  $p(a, b) = p(b, a)$  we have that the joint entropy does not depend of the order in which the random variables are selected, that is  $H(A, B) = H(B, A)$ . We can provide a similar definition for the joint entropy of a set of  $n$  random variables  $A_1, A_2, \dots, A_n$  using the joint probability mass function  $p(a_1, a_2, \dots, a_n)$ .

Adding a second random variable whose outcome is not known might increase the entropy, as following proposition proves.

**Proposition 14.4.2** We have that

$$H(A, B) \geq \max(H(A), H(B))$$

*Proof.*

$$H(A, B) = \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a, b)} \geq \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a)} = \sum_{a \in A} p(a) \log \frac{1}{p(a)} = H(A)$$

In the same way we can prove that  $H(A, B) \geq H(B)$ . Combining both inequalities we get the desired result. ■

The joint entropy of two random variables cannot be greater than the sum of their individual entropies.

**Proposition 14.4.3** We have that  $H(A, B) \leq H(A) + H(B)$  and  $H(A, B) = H(A) + H(B)$  if, and only if,  $p(a)$  and  $p(b)$  are statistically independent.

*Proof.* TODO: Prove without using the concept of conditional entropy nor mutual information. ■

The next derived concept from entropy that we are going to introduce is conditional entropy. Conditional entropy measures the uncertainty of a random variable given that the value of another random variable is known.

**Definition 14.4.3** The *conditional entropy* of the random variable  $B$  given the random variable  $A$ , denoted by  $H(B | A)$ , is defined as:

$$H(B | A) = \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(b | a)}$$

Since  $p(b | a) \neq p(a | b)$  we have that  $H(B | A) \neq H(A | B)$ . If  $H(B | A) = 0$  we have that the value of  $B$  is completely determined by the value of  $A$ .

Next proposition proves that knowing the value of a second random variable can never increase the uncertainty of a random variable.

**Proposition 14.4.4** Given the random variables  $X$  and  $Y$ , we have that  $H(Y | X) \leq H(Y)$ , and  $H(Y | X) = H(Y)$  if, and only if,  $p(a)$  and  $p(b)$  are independent.

*Proof.*

$$\begin{aligned} H(Y | X) &= \sum_{a \in A} \sum_{y \in B} p(a, b) \log \frac{1}{p(b | a)} = \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{p(a)}{p(a, b)} \\ &= \sum_{a \in A} \sum_{b \in B} p(a, b) \log p(a) - \sum_{a \in A} \sum_{b \in B} p(a, b) \log p(a, b) = -H(X) + H(X, Y) \end{aligned}$$

Applying Proposition 14.4.3 we have that  $H(Y | X) = H(X, Y) - H(X) \leq H(X) + H(Y) - H(X) = H(Y)$ . The iff equality is also proved by applying Proposition 14.4.3. ■

From an intuitive point of view we could expect that the uncertainty associated to a pair of random variables must be equal to the uncertainty of one of them plus the uncertainty of the second given that we know the outcome of the first one.

**Proposition 14.4.5 — Chain rule.** Given the random variables  $X$  and  $Y$  we have that  $H(X, Y) = H(X) + H(Y | X)$ .

*Proof.*

$$\begin{aligned} H(Y, X) &= \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a, b)} = \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a)p(a | b)} \\ &= \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a)} + \sum_{a \in A} \sum_{b \in B} p(a, b) \log \frac{1}{p(a | b)} = H(X) + H(Y | X) \end{aligned}$$
■

The last derived concept of entropy we are going to see is mutual information. Intuitively, the mutual information of two random variables  $X$  and  $Y$  measures the information that  $X$  and  $Y$  share, that is, how much knowing one of these variables reduces the uncertainty about the other.

**Definition 14.4.4** The *mutual information* of the random variable  $X$  and  $Y$ , denoted by  $I(X;Y)$ , is defined as:

$$I(X;Y) = \sum_{a \in A} \sum_{b \in B} p(a,b) \log \frac{p(a,b)}{p(a)p(b)}$$

Since  $p(a,b) = p(b,a)$  we have that  $I(X;Y) = I(Y;X)$ , that is, the order of the random variables does not affect the concept of mutual information.

Next proposition shows that mutual information is a positive quantity, and it is equal to 0 if, and only if, the random variables are independent.

**Proposition 14.4.6** Given the random variables  $X$  and  $Y$  we have that  $I(X;Y) \geq 0$ , and  $I(X;Y) = 0$  if, and only if, the variables  $X$  and  $Y$  are independent.

*Proof.* TODO: to be done ■

Introduce this proposition

**Proposition 14.4.7** Given the random variables  $X$  and  $Y$ , we have that:

$$I(X;Y) = H(X) - H(X | Y) = H(Y) - H(Y | X)$$

*Proof.* TODO: to be done ■

Introduce this proposition

**Proposition 14.4.8** Given the random variables  $X$  and  $Y$ , we have that:

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

*Proof.* TODO: to be done ■

Prove that  $I(\mathcal{S};\mathcal{S}) = H(\mathcal{S})$

Use the Venn diagrams in this example

■ **Example 14.10** ■

## 14.5 Huffman Algorithm

This section should be about compression algorithms. Not sure if only about algorithms based on information theory, or generic compression algorithms. Depends of what we need in practice

Mention that Huffman is not necessarily the optimal compression algorithm

From a practical point of view, there exists an algorithm, called *Huffman algorithm*, that provides a method to build compact prefix-free codes given

a probability distribution. For simplicity, we will study first the particular case of constructing binary prefix-free codes, and later I will provide its generalization to the case of D-ary prefix-free codes.

---

**Algorithm 14.1** Huffman Algorithm
 

---

```

procedure HUFFMAN( $Q$ )
   $T \leftarrow$  empty tree
  for  $i \leftarrow 1, d(Q) - 1$  do
    allocate a new node  $z$ 
     $z.\text{left} = x = \text{EXTRACT-MIN}(Q)$ 
     $z.\text{right} = y = \text{EXTRACT-MIN}(Q)$ 
     $z.\text{freq} = x.\text{freq} + y.\text{freq}$ 
     $\text{INSERT}(Q, z)$ 
  end for
  return  $T$ 
end procedure
  
```

---

The algorithm (see Algorithm 14.1) expects as input a source alphabet  $S = \{s_1, s_2, \dots, s_q\}$  and their corresponding probabilities  $P = \{p_1, p_2, \dots, p_q\}$ . For simplicity, we will merge both sets into a single one  $Q = \{(s_1, p_1), (s_2, p_2), \dots, (s_q, p_q)\}$ . The algorithm works by constructing a binary tree  $T$ , similar to the one used in the proof of Theorem 14.2.1. The algorithm requires  $d(Q) - 1$  iterations to finish. During each iteration, the two elements with the lowest probability are selected and removed from set  $Q$ , and a new tree node  $z$  is created, with the addition of the removed values, and added to the set  $Q$ . Once the tree has been constructed, we have to perform a tree transversal assigning a 0 to each left branch, and a 1 to each right branch, until we reach a leaf.

■ **Example 14.11** Assume we have the source alphabet  $S = \{a, b, c, d, e, f\}$  with the associated probabilities  $P = \{0.35, 0.16, 0.08, 0.12, 0.06, 0.23\}$ . In Figure are depicted the contents of the set  $Q$  and the tree  $T$  for each iteration of the algorithm. At the end of the algorithm, if we perform a traversal of the  $T$  tree, we will get the following prefix-free compact code for the source alphabet  $S$ :

| Source Word | Code Word |
|-------------|-----------|
| a           | 11        |
| b           | 00        |
| c           | 1011      |
| d           | 100       |
| e           | 1010      |
| f           | 01        |

The expected length of the code is  $L = 2.4$ , and its entropy is  $\mathcal{H} \approx 2.34$ .  
 Since the set of probabilities is not D-adic ... ■

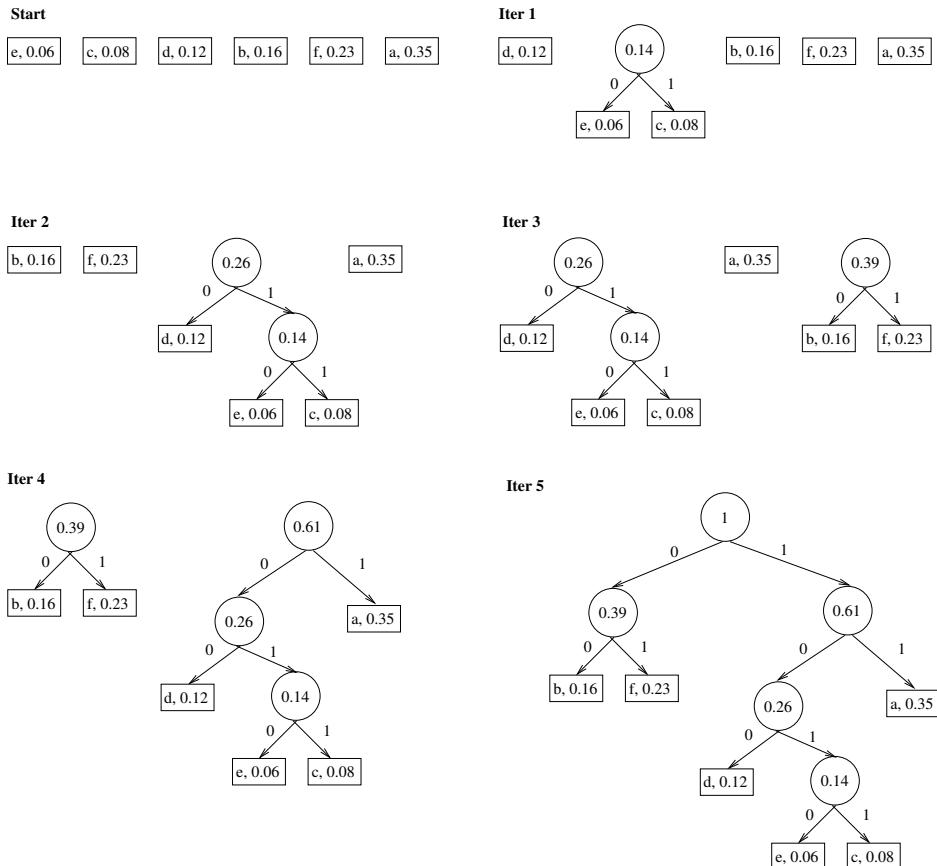


Figure 14.4: Huffman Algorithm

*This order is arbitrary; switching the left and right child of any node yields a different code of the same cost*

**Proposition 14.5.1** Given the probability is ...

*Proof.* TODO

The next theorem shows the optimality of the Huffman coding.

**Theorem 14.5.2** If  $C$  is a Huffman code then  $C$  is compact.

*Proof.* TODO

**TODO:** Rewrite the following paragraphs

*So not only is this code optimal in the sense that no other feasible code performs better, but it is very close to the theoretical limit established by the entropy*

*Although we have proved the theorem for a binary alphabet, the proof can be extended to establishing optimality of the Huffam coding algorithm for a D-ary alphabet as well.*

**TODO:** show how to extend the algorithm to D-ary codes

*Then -ary Huffman algorithm uses the 0, 1, ..., n-1 alphabet to encode message and build an n-ary tree [...] the same algorithm applies as for binary codes, except that the n least probable symbols are taken together , instead of just the 2 least probable. Note that for n greater than 2, not all sets of source words can properly form an n-ary tree for Huffman coding. In this case, additional 0-probability place holders must be added. This is because the tree must form and n to 1 contractor; for binary coding, this is a 2 to 1 contractor, and any sized set can form such a contractor. If the number fo source words is congruent to 1 module n-1, then the set of source words will form a proper Huffman tree.*

**Mention arithmetic coding**

## References

**TODO:** write this section

In 1948, Claude E. Shannon published a paper entitled "A Mathematical Theory of Communication", where he established the foundations of a new discipline, later called *information theory*.

Paper of Shannon ... Harley

Huffman -> D. A. Huffman. A method for the construction of minimum redundancy codes. Proc. IRE, 40:1098-1101, 1952.

The algorithm of huffman has been adapted from Cover. Here also you can find a proof that the algorithm is of order XX.

Kraft's inequality was published by Leon G. Kraft in 1949 as part of his Master Thesis [Kra49]. The inequality was independently rediscovered

and proved for the general case of uniquely decodable codes by Brockway McMillan in 1956 [McM56]. The proofs contained in this book have been adapted from [CT12].

The proof of proposition 14.4.1 has been adapted from [Abr63]. The proof of proposition 4.3.1 has been adapted from Abramson.

In the area of *digital signal processing* [GG12] it is common to apply a pre-processing step called *quantization*, in which a large number of samples from a continuous signal are mapped into a finite number of representative values. It can be a *scalar quantization* when the signal is one dimensional, or *vector quantization* in case of a multidimensional signal. The optimization goal is to identify a (pre-defined) number of quantized values such that the mean squared error between the selected values and the original signal is minimized. The problem is solved using the Lloyd-max algorithm [Llo82] (closely related to the kmeans clustering algorithm [**<empty citation>**] used in machine learning) in which the search space is partitioned in a collection of convex regions and their centroids used as quant, and then, are continuously adapted until some stopping criteria is reached. Although it can be shown that the algorithm converges to the optimal solution that minimizes the mean squared error, the selected intervals cannot be used as estimation of the original probability distribution.





## 15. Complexity

*Everything should be made as simple as possible,  
but not simpler.*  
Albert Einstein

In Chapter 14, the concept of the complexity of a string based on the lengths of the codewords of a prefix-free code was introduced. This definition is limited by two main factors: firstly, it necessitates prior knowledge of the set of possible strings, and secondly, it requires the definition of a probability distribution over this set a priori. It would be highly beneficial if we could expand the set of strings to encompass all strings (that is,  $\mathcal{B}^*$ ) without necessitating a probability distribution, thereby providing an absolute notion of string complexity. Unfortunately, even if these issues are addressed, a more significant limitation exists in studying the complexity of strings using codes: certain strings, expected to be classified as simple, cannot be compressed. For instance, the binary expansion of the constant  $\pi$  follows a uniform distribution over the set  $\{0, 1\}$  and, as such, cannot be compressed. However, it can be fully and effectively described by a very short mathematical formula. This necessitates an alternative definition of string complexity.

*Kolmogorov Complexity*, also known as *Algorithmic Information Theory*, offers a definition of the complexity of a string that explicitly addresses these

issues. Intuitively, the amount of information in a finite string is measured by the length of the shortest computer program capable of producing the string. This approach does not require prior knowledge of the set of valid strings or their probability distribution. Furthermore, objects like  $\pi$  are appropriately classified as having low complexity. We could argue that Kolmogorov complexity provides a universal definition of the amount of information that closely aligns with our intuitive understanding. To compute the Kolmogorov complexity of a string, it is necessary to agree upon a universal description method or computer language, along with a universal computer. One might question whether, by doing so, the complexity of a string becomes dependent on the chosen computer language. Fortunately, it has been demonstrated that this is not the case, as all reasonable (and sufficiently powerful) computer languages yield the same description length, up to a fixed constant that depends on the languages chosen, but not on the string itself. Unfortunately, Kolmogorov complexity introduces a significant issue: it is a non-computable quantity, and as such, must be approximated in practice.

At this point, one might wonder if it is possible to compute the complexity of any object in general, not just strings. The answer is yes, at least theoretically. Given an object  $x$ , the task is to provide an encoding method that represents the object as a string  $x$ . This encoding method would only be useful if we can losslessly and effectively reconstruct the original object from its encoded description. Unfortunately, providing such descriptions is not always feasible, either because the objects in question are abstract (as is often the case in mathematics) or because practical reconstruction of the object from its description is currently impossible (for example, with living organisms<sup>1</sup>).

## 15.1 Strings Complexity

In Section 13.1, the concept of the Turing machine, an idealized model of computers, was introduced. We saw that Turing machines can be represented as partial computable functions  $T : \mathcal{B}^* \rightarrow \mathcal{B}^*$ , which assign to each input string  $s \in \mathcal{B}^*$  an output string  $T(s) \in \mathcal{B}^*$  (Definition 13.4.1). We also introduced the concept of a universal Turing machine  $U : \mathcal{B}^* \times \mathcal{B}^* \rightarrow \mathcal{B}^*$  (Definition 13.2.1), a machine that can simulate the behavior of any other Turing machine; that is, for all  $(x, v) \in \mathcal{B}^* \times \mathcal{B}^*$ , we have that  $U(x, v) = T_x(v)$ . Later, in Section 14.1, the concept of a code, and in particular, the notion of a prefix-free code, was introduced (Definition 14.1.4). We saw that this kind of code presents important properties (Theorem 14.2.1). The next definition

---

<sup>1</sup>As of now, it is not possible to recreate an animal solely based on its DNA.

merges the best of both worlds, Turing machines and prefix-free codes, and introduces a new type of universal Turing machine.

**Definition 15.1.1** A *prefix-free universal Turing machine* is a universal Turing machine  $U : \mathcal{B}^* \times \mathcal{B}^* \rightarrow \mathcal{B}^*$  such that, for every  $v \in \mathcal{B}^*$ , the domain  $U_v$  is prefix-free, where  $U_v : \mathcal{B}^* \rightarrow \mathcal{B}^*$  and  $U_v(p) = U(p, v)$  for all  $p \in \mathcal{B}^*$ .

Using modern computer science terminology we could say that  $U$  is the computer,  $p$  is the program and  $v$  is the input to the program. Intuitively, the above definition requires that no computer program can be a prefix of any other program. This is not a limitation from the point of view of string lengths, since, by applying McMillan's theorem (Theorem 14.2.1), given a uniquely decodable program, we could always find a prefix-free one that computes exactly the same function and has the same length. Moreover, in practice, real computer programs are usually prefix-free; for example, the C programming language requires that all functions must be enclosed by braces {}.

**TODO:** Clarifying why we have to fix the input string  $v$  in  $U_v(p) = U(p, v)$ , since this is something that might confuse some readers.

The concept of a prefix-free universal Turing machine allows us to introduce a new definition of the complexity of a string that aligns more closely with our intuitive understanding of the amount of computational information contained in an object (encoded as a string).

**Definition 15.1.2 — Kolmogorov Complexity.** Fix a prefix-free universal Turing machine  $U : \mathcal{B}^* \times \mathcal{B}^* \rightarrow \mathcal{B}^*$ . The *Kolmogorov complexity* of a string  $s \in \mathcal{B}^*$ , denoted by  $K(s)$ , is defined as:

$$K(s) = \min_{p,v \in \mathcal{B}^*} \{l(p) + l(v) : U(p, v) = s\}$$

Intuitively, the shortest description of a string  $s$  is given by two elements: a program  $p$  (a self delimited Turing machine) that compress all the regular patterns of the string, and a new string  $v$  that comprises those parts of  $s$  that do not present any regularity. We have to find the optimum balance between increasing the complexity of the program, trying to grasp more regularities, or increase the size of the non-compressible part<sup>2</sup>.

**■ Example 15.1** Consider the string composed of one thousand times the substring "10", that is " $\underbrace{1010\dots1010}_{1.000 \text{ times}}$ ". We could write the following program:

<sup>2</sup>In the literature the Kolmogorov complexity of the string  $s$  is defined as  $K(s) = \min_{p \in \mathcal{B}^*} \{l(p) : U(p, \lambda) = s\}$ , that is, the length of the shortest computer program that without any additional input can print the string  $s$ . We prefer to use the two parts definition  $l(p) + l(v)$  because it is more in line with the requirements of the theory of nescience.

```
example(char *v) {
    for (int i=1; i<=1000; i++)
        printf("%s", v);
}
```

and then run it with:

```
example("10");
```

in order to print it. The length of the original string is 2.000 bits, but the length of the program is 480 bits (assuming that every symbol is encoded using an uniform code of 8 bits), and the length of the input is 2 bits, so we can conclude that the string has a low complexity. Of course, in order to compute the real Kolmogorov complexity of the string we should find the shortest Turing machine that prints that string.

On the contrary, a string composed of two thousands random 0's and 1's would have a high complexity, since it does not exists any program shorter than the program that prints the string itself. ■

As we mentioned in the preface of this chapter, Kolmogorov complexity would not be particularly useful if the complexity of strings depended on the choice of universal Turing machine. The following theorem demonstrates that this concern is unfounded, up to a constant that depends on the choice of machines, but not on the strings themselves. This establishes Kolmogorov complexity as an inherent property of the strings.

**Theorem 15.1.1 — Invariance theorem.** Let  $U$  and  $U'$  two universal Turing machines. Then, there exists a constant  $C_{U,U'}$ , that depends on  $U$  and  $U'$ , such that for each string  $s \in \Sigma^*$  we have that:

$$K_U(s) \leq K_{U'}(s) + C_{U,U'}$$

*Proof.* Let  $p, v$  be the shortest strings that  $U'(p, v) = s$ , then we have that  $U(\langle U', p, v \rangle, \lambda) = U'(p, v) = s$ , and that  $l(\langle U', p \rangle) = \log(l(\langle U' \rangle)) + 1 + l(\langle U' \rangle) + l(p) = K_{U'}(s) + C$ , where  $l(\langle U' \rangle)$  is the length of the machine  $U'$  using the encoding described in Section 13.2. ■

■ **Example 15.2** Consider a universal programming language, such as Java, and an alternative language, such as Python. We can write a Python interpreter in Java, that is, a Java program that takes a Python script as input and executes it. Then, to compute the complexity of a string  $s \in \mathcal{B}^*$  using Java,  $C_J(s)$ , it would be no greater than the complexity of the string using Python,  $C_P(s)$ , plus the length of the Python interpreter written in Java,  $C_{J,P}$ .

Importantly, the length of the interpreter,  $C_{J,P}$ , does not depend on the string  $s$ . ■

Although we have proved that Kolmogorov complexity does not depend on the selected universal Turing machine, the size of the constant  $C_{U,U'}$  could pose a limitation in practical applications, especially when computing the complexity of short strings where the constant might significantly exceed the complexity of the string itself. This challenge is addressed by the Minimum Description Length principle, as described in Section 16.3.

**Notation 15.1.** We denote by  $s^*$  the shortest program that outputs the string  $s$  on the universal Turing machine  $U$ , that is,  $s^* = \langle p, v \rangle$ ,  $U(s^*) = s$  and  $l(s^*) = K(s)$ . If more than one program satisfies these properties, we select the first one using a lexicographical order induced by  $0 < 1$ .

The size of the  $C_{U,U'}$  constant is not the only challenge presented by Kolmogorov complexity; another issue is its non-computability, that is, there is no algorithm capable of determining the shortest program that generates any given string. The following theorem on the uncomputability of Kolmogorov complexity marks a pivotal insight into the intrinsic limits of complexity theory. By exploring the theorem, we delve into the heart of computability theory, confronting the paradoxical reality that some of the most fundamental aspects of information are inherently beyond the reach of algorithmic computation.

**Theorem 15.1.2** The function  $K : \mathcal{B}^* \rightarrow \mathbb{N}$  that assigns to each string  $s$  its Kolmogorov complexity  $K(s)$  is not computable.

*Proof.* **TODO: Review** Assume for the sake of contradiction that  $K$  is computable. This means there exists a Turing machine  $T$  such that for any string  $s$ ,  $T$  halts with  $K(s)$  on its tape. Consider the following process, which uses  $T$  to construct a string  $s$  of arbitrary complexity: i) enumerate all binary strings using a shortlex ordering:  $s_1, s_2, \dots$ , ii) for each string  $s_i$ , compute  $K(s_i)$  using  $T$  and select  $s_i$  such that  $K(s_i) > l(s_i) + C$ , where  $C$  is a constant representing the length of this description plus the operation to add  $C$ . Such a process would effectively produce a string  $s$  whose Kolmogorov complexity is greater than its length plus a constant, which contradicts the definition of Kolmogorov complexity as the length of the shortest description. The contradiction arises because our assumption that  $K$  is computable allows us to construct a string that defies the bounds set by  $K$  itself. Therefore, the function  $K$  is not computable. ■

If  $K$  were computable, we could solve the Halting Problem by constructing a program that, for any input program and input, computes whether the

program halts by checking if its Kolmogorov complexity is finite. Since the Halting Problem is known to be undecidable, this provides a contradiction.

In practice, we approximate the Kolmogorov complexity using the compressed version of the string, employing standard compression algorithms, such as the Huffman algorithm described in Section 14.5.

## 15.2 Properties of Complexity

In this section, we delve into the properties of Kolmogorov complexity. We will explore the foundational principles that govern this complexity measure, including its invariance, symmetry, and non-computability. Through examining these properties, we gain deeper insights into the complex interplay between information, computation, and randomness.

Kolmogorov complexity is a finite positive natural number.

**Proposition 15.2.1** For all  $s \in \mathcal{B}^*$  we have that  $0 < K(s) < \infty$ .

*Proof.* Since  $K(s)$  is the length of a string, it must be greater than 0. The property  $K(s) < \infty$  is a consequence of Proposition 15.2.2 and the fact that we are only dealing with finite strings. ■

The Kolmogorov complexity of a string cannot surpass the sum of its own length and a constant.

**Proposition 15.2.2** There is a constant  $c$  such that for all  $s \in \mathcal{B}^*$  we have that  $K(s) \leq l(s) + c$ .

*Proof.* Let  $s \in \mathcal{B}^*$  be an arbitrary string, and consider the encoding of a Turing machine  $p$  such that for any input  $v = s$ , it halts and outputs  $s$ . The program  $p$  is designed to simply reproduce its input. Given this setup, when  $p$  is executed on a universal Turing machine  $U$  with  $s$  as input, it satisfies the condition  $U(p, s) = s$ . The length of  $p$  is a constant  $c$  across all strings  $s$ . This constancy arises because  $p$ 's mechanism (accepting an input and outputting it unchanged) does not vary with the size or content of  $s$ . By the definition of Kolmogorov Complexity  $K(s)$ , which seeks the minimum length of a program-input pair that generates  $s$ , the combination of  $p$  and  $s$  presents a feasible solution. Therefore, we have  $K(s) \leq l(s) + l(p) = l(s) + c$ . ■

The size of the constant  $c$  depends on the specific encoding schema used by the selected universal Turing machine  $U$ , but it is independent of the string  $s$ . In Section 15.6, we will explore the characteristics of random strings, which are defined as strings that cannot be compressed. Such strings exhibit a Kolmogorov complexity that exceeds their own length, that is,  $K(s) > l(s)$ .

The absolute difference in Kolmogorov complexity between any string  $x$  and its transformed counterpart  $f(x)$ , via a computable bijection, is bounded by a constant  $c$ . That is, not only does  $f$  not increase the complexity of  $x$  by more than a constant, but also  $f$  does not decrease the complexity by more than a constant.

**Proposition 15.2.3** Let  $f : \mathcal{B}^* \rightarrow \mathcal{B}^*$  is a computable bijection, then there exists a constant  $c$  such that  $|K(f(x)) - K(x)| < c$ .

*Proof.* Let  $P_f$  be the program that computes  $f$  and  $P_{f^{-1}}$  the program that computes the inverse of  $f$ . For any string  $x$ , let  $P_x$  be the shortest program that generates  $x$ . Then, a program  $P_{f(x)}$  that generates  $f(x)$  can be constructed by concatenating  $P_x$  with  $P_f$ . The length of this program is  $|P_{f(x)}| = |P_f| + |P_x|$ . Since  $|P_f|$  is a constant that does not depend on  $x$ , we can say that  $K(f(x)) \leq K(x) + |P_f|$ . Similarly, given  $f(x)$ , we can construct a program  $P'_x$  to generate  $x$  by applying  $P_{f^{-1}}$  to  $f(x)$ . The length of this program is  $|P'_x| = |P_f| + |P_{f^{-1}}|$ . Thus,  $K(x) \leq K(f(x)) + |P_{f^{-1}}|$ . The two inequalities combined implies that  $|K(f(x)) - K(x)| \leq \max(|P_f|, |P_{f^{-1}}|) = c$ , where  $c$  is a constant that represents the maximum of the lengths of the programs that compute  $f$  and  $f^{-1}$ . This constant  $c$  does not depend on  $x$ , but rather on the complexity of the functions  $f$  and  $f^{-1}$ . ■

This proposition shows a remarkable stability of informational content under computable bijections, underscoring the intrinsic robustness of Kolmogorov complexity in the face of such transformations.

■ **Example 15.3** TODO: Provide an example clarifying this concept. ■

TODO: Elaborate on the fact that some strings are very compressible, and that happens for all string sizes.

## 15.3 Joint Kolmogorov Complexity

The joint Kolmogorov complexity of two strings  $s$  and  $t$  is defined as the length of the shortest program  $p$  that, when executed on an universal Turing machine  $U$ , outputs the pair  $\langle s, t \rangle$ , that is, in such a way that allows both strings to be unambiguously retrieved.

**Definition 15.3.1 — Joint Kolmogorov Complexity.** The *Joint Kolmogorov complexity* of the strings  $s, t \in \mathcal{B}^*$ , denoted by  $K(s, t)$ , is defined as:

$$K(s, t) = \min_{p, v \in \mathcal{B}^*} \{l(p) + l(v) : U(p, v) = \langle s, t \rangle\}$$

The notation  $K(s,t)$  and  $K(st)$  represent two different concepts in the context of Kolmogorov complexity.  $K(s,t)$  refers to the joint Kolmogorov complexity of two strings  $s$  and  $t$  as per Definition 15.3.1, meanwhile  $K(st)$  represents the Kolmogorov complexity of the concatenation of  $s$  and  $t$ , without any additional structure to distinguish between them, and so, Definition 15.1.2 is applied. The choice between  $K(s,t)$  and  $K(st)$  depends on whether it's important to preserve and utilize the distinction and relationship between  $s$  and  $t$ . If analyzing the interplay or the shared characteristics of  $s$  and  $t$  is relevant,  $K(s,t)$  is more appropriate. If the focus is on the information content of the combined sequence without regard to its origin from two separate strings,  $K(st)$  is used.

■ **Example 15.4** TODO: Provide an example clarifying this concept. ■

Our first proposition highlights a fundamental symmetry in Kolmogorov complexity, illustrating that the complexity of describing a pair of strings in either order differs by at most a constant. This reflects the intrinsic property that the information content is independent of the specific arrangement of the strings being described. The constant  $c$  encapsulates the overhead associated with the operations needed to reverse the order of the strings.

**Proposition 15.3.1** There is a constant  $c$  such that for all  $x,y \in \mathcal{B}^*$  we have that  $K(x,y) \leq K(y,x) + c$ .

*Proof.* The key to this proof lies in the symmetry of information and the properties of a universal Turing machine. Let  $U$  be a universal Turing machine, and let  $p_{xy}$  and  $p_{yx}$  be the shortest programs that output  $x$  followed by  $y$ , and  $y$  followed by  $x$ , respectively, when executed on  $U$ . To convert  $p_{xy}$  into a program that outputs  $yx$ , we need to add or modify a finite set of instructions to swap the output order. This set of instructions has a fixed length, denoted by  $c$ . Therefore, the program that first computes  $x$  and  $y$  and then swaps their order can be at most  $c$  bits longer than the program that directly computes  $y$  and  $x$ . ■

Next proposition underscores the subadditive nature of Kolmogorov complexity, proving that the total complexity of describing two strings jointly cannot exceed the sum of their individual complexities by more than a fixed constant, irrespective of the strings' content.

**Proposition 15.3.2** There is a constant  $c$  such that for all  $s,t \in \mathcal{B}^*$  we have that  $K(s,t) \leq K(s) + K(t) + c$ .

*Proof.* Let  $s^*$  and  $t^*$  be the shortest computer programs that generate  $s$  and  $t$ , respectively. Given that  $s^*$  and  $t^*$  are designed to be prefix-free, a universal Turing machine  $U$  can unambiguously decode each program

without needing any additional information and subsequently generate the strings  $s$  and  $t$ . The constant  $c$  accounts for the requirement of  $U$  to prepend the concatenated program for  $st$  with the length  $l(s)$  of  $s$ . This step ensures that the concatenated sequence  $st$  can be decoded unambiguously. ■

The last proposition states the relationship between the complexity of individual strings and their joint complexity, by establishing a lower bound on the joint complexity of two strings relative to their individual complexities.

**Proposition 15.3.3** There is a constant  $c$  such that for all  $s, t \in \mathcal{B}^*$  we have that  $K(s, t) \geq K(s) + c$  and  $K(s, t) \geq K(t) + c$ .

*Proof.* Assume for the sake of contradiction that  $K(s, t) < K(s) + c$ . Let  $p$  be the shortest computer program that outputs the pair  $\langle s, t \rangle$ , meaning that  $K(s, t) = l(p)$ . Since program  $p$  unambiguously generates  $s$  as part of the output pair, it follows that the complexity of generating  $s$  alone,  $K(s)$ , should not exceed the length of  $p$ . which is a contradiction with respect the intial assumption of  $K(s, t) < K(s) + c$ . ■

## 15.4 Conditional Kolmogorov complexity

In this section, we explore the concept of *conditional Kolmogorov complexity*, which examines how the complexity of a string  $s$  can be significantly decreased by assuming prior knowledge of another string  $t$ . This notion highlights the impact of assuming some background information on the compressibility of a description.

**Definition 15.4.1 — Conditional Kolmogorov Complexity.** The *conditional Kolmogorov complexity* of a string  $s \in \mathcal{B}^*$  given the string  $t \in \mathcal{B}^*$  is defined as:

$$K(s|t) = \min_{p,v \in \mathcal{B}^*} \{l(p) + l(v) : U(p, \langle v, t \rangle) = s\}$$

Similar to the non-conditional Kolmogorov complexity, the conditional complexity of a string  $s$ , given a background string  $t$ , depends on the strings themselves rather than the specific universal Turing machine used for computation. That is, for any two universal Turing machines,  $U$  and  $U'$ , it is easy to show that there exists of a constant  $C_{U,U'}$  such that for all strings  $s, t \in \Sigma^*$ , the inequality  $K_U(s|t) \leq K_{U'}(s|t) + C_{U,U'}$  holds. This property underscores the inherent machine-independence of conditional complexity measures.

■ **Example 15.5 TODO:** Provide an example clarifying this concept. ■

As it was the case of the unconditional Kolmogorov complexity, conditional Kolmogorov complexity is a finite positive natural number.

**Proposition 15.4.1** For all  $s, t \in \mathcal{B}^*$  we have that  $0 < K(s|t) < \infty$ .

*Proof.* Since  $K(s)$  is the length of a string, it must be greater than 0. The property  $K(s|t) < \infty$  is a consequence of Proposition 15.2.2 and the fact that we are only dealing with finite strings. ■

Next proposition posits that when a string  $s$  is conditioned upon itself, its complexity stabilizes to a universal constant  $c$ .

**Proposition 15.4.2** There is a constant  $c$  such that for all  $s \in \mathcal{B}^*$  we have that  $K(s|s) = c$ .

*Proof.* When a string  $s$  is conditioned on itself, the information needed to generate  $s$  from  $s$  can be encapsulated in a Turing machine that simply copies its input to its output. This machine, being independent of the specific content of  $s$ , has a fixed length  $c$ . ■

This proposition explores the relationship between unconditional and conditional Kolmogorov complexities, establishing an upper bound for the latter. It asserts that for any strings  $s$  and  $t$ , the complexity of  $s$  given  $t$  is at most the complexity of  $s$  alone, plus a constant  $c$ . This highlights the intuitive notion that having additional information at most reduces the complexity of describing a string, or in the worst case, adds a constant overhead, but does not increase it beyond this bound.

**Proposition 15.4.3** There is a constant  $c$  such that for all  $s, t \in \mathcal{B}^*$  we have that  $K(s|t) \leq K(s) + c$ .

*Proof.* There exists a Turing machine  $T'$  that ignores its input and  $t$  directly generates  $s$ . The length of  $T'$  is  $K(s)$  by definition. ■

The relationship between conditional, unconditional, and joint Kolmogorov complexities, offers a comprehensive perspective on the informational interdependencies of binary strings. It posits that the complexity of a string  $s$  given another string  $t$  is at most the complexity of  $s$  alone, which in turn is no greater than the joint complexity of both  $s$  and  $t$ . The proposition encapsulates the intuitive understanding that knowledge of additional data (in this case,  $t$ ) cannot increase the complexity of describing  $s$ , and that the combined description of two strings is at least as complex as describing each independently.

**TODO:** Prove the asymmetry of conditional Kolmogorov complexity, that is,  $K(s|t) \neq K(t|s)$ .

**Proposition 15.4.4** For all  $s, t \in \mathcal{B}^*$  we have that  $K(s|t) \leq K(s) \leq K(s, t)$ .

*Proof.* Given Proposition 15.4.3 and Proposition 15.3.3. ■

The Kolmogorov complexity chain rule is a fundamental principle that connects the joint complexity of two strings with their individual and conditional complexities. It asserts that the total complexity of a pair of strings  $s$  and  $t$  can be decomposed into the complexity of  $s$  plus the complexity of  $t$  given  $s$ . This relationship mirrors the additive property of entropy in information theory (see Proposition XXX) and provides a powerful tool for understanding the interplay between information content and conditional information in the context of Kolmogorov complexity.

**Proposition 15.4.5** For all  $s, t \in \mathcal{B}^*$  we have that  $K(s, t) = K(s) + K(t | s)$

*Proof.* TODO ■

■ **Example 15.6** TODO: Provide an example clarifying this concept. ■

## 15.5 Information Distance

In this section, we aim to introduce a universal metric for quantifying the absolute information distance between two or more individual entities encoded as strings of symbols. Intuitively, the information distance between two strings  $s$  and  $t$  can be understood as the length of the shortest computer program for a universal computer that enables the generation of  $s$  given  $t$  and vice versa.

**Definition 15.5.1** The *information distance* between two strings  $s, t \in \mathcal{B}^*$  with respect to a universal Turing machine  $U$ , denoted by  $ID_U(s, t)$ , is defined as

$$ID_U(s, t) = \min\{l(p) : U(p, s) = t, U(p, t) = s\}$$

It can be shown that for any pair of universal Turing machines,  $U_1$  and  $U_2$ , the information distance between two strings  $s$  and  $t$  differs by no more than a constant  $c$ ; this constant  $c$  depends on the choice of  $U_1$  and  $U_2$  but is independent of the specific strings  $s$  and  $t$ . It's also important to note that despite its theoretical utility, information distance is non-computable, meaning it does not exist and there is no algorithm to calculate it in practice.

The idea of using the conditional Kolmogorov complexity of  $s$  given  $t$ , that is  $K(s | t)$ , as a metric for information distance may seem appealing. However, this approach is unsuitable for capturing the essence of information distance because of its inherent asymmetry (refer to Proposition XXX). For example, the complexity  $K(\lambda | t)$  remains minimal even when  $t$  significantly differs from the empty string. Alternatively, employing the sum  $K(s | t) + K(t | s)$

as a measure of distance is also inadequate, as it overlooks the redundancy in the information necessary for transforming  $s$  into  $t$  and vice versa.

Next proposition shows the proper way in which the information distance between two strings can be computed using their conditional Kolmogorov complexity.

**Proposition 15.5.1** Let  $s, t \in \mathcal{B}^*$  be two binary strings, then we have that:

$$ID_U(s, t) = \max\{K(s | t), K(t | s)\} + O(\log \max\{K(s | t), K(t | s)\})$$

*Proof.* TODO: based on the maximal overlap. ■

■ **Example 15.7** TODO: Give an example based on the bitwise exclusive or.

■

Introduce the concept of max distance as:

$$E(x, y) = \max\{K(x | y), K(y | x)\}$$

Introduce the following proposition.

**Proposition 15.5.2**  $E(x, y)$  is a metric.

*Proof.*

Introduce the following proposition.

**Proposition 15.5.3**

$$E(x, y) = \max\{K(x | y), K(y | x)\} = K(xy) - \min\{K(x), K(y)\} + O(\log K(xy))$$

*Proof.* TODO: Pending ■

TODO: Introduce the concept of admisible information distance. Prove that  $E(x, y)$  is an admisible information distance. Prove that it is minimal for every admisible information distance. Explain this notion of universality.

This concept of information distance is universal because it encompasses all other computable distance metrics as special cases.

## Normalized Information Distance

Information distance is an absolute measure; however, when assessing similarity, we are often more concerned with relative measures. For instance, consider two strings each of length 1,000,000 that differ by 1000 bits; we would perceive these strings as being relatively more similar compared to two strings of length 1000 bits that differ by the same amount of 1000 bits. This concept of normalization implies that the size of the description required for transformation should be evaluated in the context of the sizes of the objects involved.

**Definition 15.5.2** The *normalized information distance* between two binary strings  $s, t \in \mathcal{B}^*$ , denoted by  $NID(s, t)$ , is defined as:

$$NID(s, t) = \frac{\max\{K(s | t), K(t | s)\}}{\max\{K(s), K(t)\}}$$

As expected, the normalized information distance is a number between zero and one.

**Proposition 15.5.4** The normalized information distance  $NID(s, t)$  takes values in the range  $[0, 1]$ .

*Proof.* TODO: Pending ■

Introduce the following proposition.

**Proposition 15.5.5** The normalized information distance  $NID(x, y)$  is a metric, up to negligible errors.

*Proof.* TODO: Pending ■

Introduce the following proposition.

**Proposition 15.5.6**

$$NID(x, y) = \frac{K(xy) - \min\{K(x), K(y)\} + O(\log K(xy))}{\max\{K(x), K(y)\}}$$

*Proof.* TODO: Pending ■

## Normalized Compression Distance

Although the normalized information distance metric is not computable, it has a potential wide range of applications. By approximating  $K$  with practical compressors, where  $Z(s)$  represents the binary length of the string  $s$  compressed using a compressor  $Z$  (such as "gzip", "bzip2", "PPMZ"), we can approximate the concept of normalized information distance.

**Definition 15.5.3** The *normalized compression distance* between two strings  $s, t \in \mathcal{B}^*$ , given the compressor  $Z$ , and denoted by  $NCD_Z(s, t)$ , is defined as:

$$NCD_Z(s, t) = \frac{\max\{Z(s | t), Z(t | s)\}}{\max\{Z(s), Z(t)\}}$$

Unfortunately, for the majority of the compressors it is very difficult to compute the compressed conditional version of a string  $Z(s | t)$ . Fortunately, we can rewrite the NCD in a different way in which we do not need to use conditional compressions.

**Proposition 15.5.7** The normalized compression distance between two strings  $s, t \in \mathcal{B}^*$ , given the compressor  $Z$ , satisfies:

$$NCD_Z(s, t) = \frac{Z(st) - \min\{Z(s), Z(t)\}}{\max\{Z(s), Z(t)\}}$$

*Proof.* TODO: prove. ■

The normalized compression distance constitutes a spectrum of distances, each defined by the choice of compressor  $Z$ . The effectiveness of  $Z$  determines how closely the normalized compression distance mirrors the normalized information distance, ultimately influencing the quality of the outcomes.

## 15.6 Incompressibility and Randomness

A string is considered incompressible if its Kolmogorov complexity is approximately equal to its length; in other words, there is no significantly shorter description or program that can produce it.

**Definition 15.6.1** For each constant  $c$  we say that a string  $s \in \mathcal{B}^*$  is  $c$ -incompressible if  $K(s) \geq l(s) - c$ .

Next proposition shows that there exists incompressible strings for all strings lengths.

**Proposition 15.6.1** For every length  $n$ , there exists a string of length  $n$  that is incompressible.

*Proof.* By a counting argument, the number of possible programs of length less than  $n$  is less than  $2^n$ , which is the number of binary strings of length  $n$ . Therefore, some strings of length  $n$  cannot be generated by any shorter program, rendering them incompressible. ■

We will extend the term *incompressible string* to encompass all the strings that are  $c$ -incompressible with small  $c$ . In this sense, it turns out that most strings are incompressible.

**Proposition 15.6.2** For any  $n$ , at least  $1 - \frac{1}{c}$  fraction of the strings of length  $n$  are  $c$ -incompressible.

*Proof.* Consider the number of programs of length  $n - c$  or less. There are  $\sum_{i=0}^{n-c} 2^i = 2^{n-c+1} - 1$  such programs. Since there are  $2^n$  strings of length  $n$ , and fewer than  $2^{n-c+1}$  programs to generate them, at least  $2^n - 2^{n-c+1} + 1$  strings of length  $n$  must be  $c$ -incompressible. The fraction of incompressible

strings is therefore at least  $1 - \frac{2^{n-c+1}-1}{2^n}$ , which approximates  $1 - \frac{1}{c}$  for large  $n$ . ■

The notion of randomness, especially in the context of sequences or strings, is often intuitively associated with unpredictability, lack of pattern, or absence of structure. Kolmogorov complexity formalizes this intuition by linking randomness to incompressibility. Specifically, a string is considered to exhibit randomness if it cannot be generated by any program significantly shorter than the string itself.

**Definition 15.6.2** We say that a string  $s \in \mathcal{B}^*$  is *random* if it is  $c$ -incompressible with  $c$  being small.

Random strings are characterized by high Kolmogorov complexity, which means they are incompressible. Such strings contain the maximum amount of information possible, precluding the possibility of a simpler, algorithmic shorthand for their representation. This is because the shortest program that can generate a random string is essentially the string itself.

■ **Example 15.8** Consider a string generated by flipping a fair coin to decide each bit: 1011010110110101. Assuming the coin flips are truly random, this string's Kolmogorov complexity is high, as there's no shorter program or pattern that can generate it besides enumerating the bits one by one. Any attempt to compress it would result in a program whose length is comparable to the string itself. ■

The unpredictability of random strings stems from their incompressibility. Since predicting any subsequent bit of a random string essentially requires knowledge of the entire string (due to the lack of patterns), such strings are inherently unpredictable.

Random strings are typical in the space of all strings. Most strings are random in the sense of Kolmogorov complexity, implying that most strings do not admit a significantly shorter description. This aligns with the principle that, in a large enough set of strings, structured or simple patterns are the exception rather than the rule.

■ **Example 15.9** Consider the set of all conceivable high-resolution digital photographs, each represented as a binary string encoding pixel colors and intensities. Within this immense collection, only a small fraction might display identifiable patterns or motifs, such as a picture of a blue sky or a uniform monochromatic background. These images could be succinctly described or compressed into shorter binary sequences because their repetitive nature or constrained color palettes. However, the vast majority of these possible digital photographs bear resemblance to "random" strings, in the sense that

they exhibit incompressible patterns. ■

## References

Kolmogorov complexity, named after the Soviet mathematician Andrey Kolmogorov, is a measure of the complexity of a string of text or other data. It is defined as the length of the shortest possible description of the string in some fixed universal description language. This measure is inherently uncomputable in general, as proven by the halting problem's undecidability, but it provides a powerful theoretical tool for understanding data complexity. Beyond theoretical interest, Kolmogorov complexity has applications in pattern recognition, data compression, and the study of randomness. It offers a framework for understanding the limits of compressibility and the nature of information.

The concept of Kolmogorov complexity emerged independently in the works of several researchers in the early 1960s. Andrey Kolmogorov introduced it in 1965 [Kol65], motivated by trying to formalize the concept of randomness and complexity through the lens of information theory. Ray Solomonoff laid the groundwork for algorithmic information theory, introducing a related concept that would later be recognized as a form of Kolmogorov complexity [Sol64]. His work focused on the idea of describing data compactly using probabilistic models. Almost simultaneously with Kolmogorov, Gregory Chaitin developed similar ideas [Cha69]. Chaitin is known for introducing the concept of algorithmic randomness and for his work on the incompleteness theorem, which relates to the limits of formal systems in proving the complexity of sequences. The motivations behind the development of Kolmogorov Complexity were multifaceted, encompassing the desire to better understand the nature of information, randomness, and the limits of computation and prediction, thus laying the groundwork for numerous applications in theoretical computer science, mathematics, and beyond.

**TODO: Include at least one reference to a introductory book.**

For those readers interested in delving into the details of Kolmogorov complexity, a variety of foundational texts and advanced treatments are available. [LV13] provides a comprehensive coverage to the concepts and applications of Kolmogorov complexity, and it's widely regarded as the definitive textbook on the subject, although it is not recommended for beginners. [Cal02] delves deeply into the foundations of algorithmic information theory, focusing on the rigorous mathematical exploration of randomness and complexity through Kolmogorov complexity. [CT12] broader in scope, this book provides an excellent foundation in information theory, including

discussions relevant to Kolmogorov complexity. It's a great resource for understanding the context in which Kolmogorov complexity operates within information theory.





## 16. Learning

*Some mathematical statements are true for no reason,  
they're true by accident.*

Gregory Chaitin

**Warning: This section still requires a significant amount of work!**

Machine learning refers to a large collection of algorithms designed to automatically build mathematical models based on sample data sets, usually with the aim of making predictions, classifying objects, or simply to better understand the structure of the data. In the past decade, machine learning algorithms have been highly successful in areas like self-driving cars, practical speech recognition, effective web search, and purchase recommendations.

In this chapter we are going to see how the problem of learning from data is formally formulated in the area of machine learning. In this sense, the chapter is a continuation of the introduction to discrete probability included in Section ???. Also, we are going to study in detail two particular approaches to machine learning that are highly related to our theory of nescience: the Minimum Description Length principle and the Minimum Message Length principle.

Most of the learning algorithms used today in practice are known since forty years ago. The high success of current machine learning applications is

largely due to the availability of huge, high-quality, training datasets, and to the advance of computing power, and in particular, thanks to the powerful graphical processing units (GPU) used in video-games. In Chapter 8 we will introduce a collection of new machine learning algorithms based on the theory of nescience, and we will compare them with the current, state of the art, algorithms.

## 16.1 Statistical Inference

*Statistical inference* has traditionally been presented as the way to make sense of reality through data. It applies probabilistic models to relate finite observations to broader populations, aiming to offer predictions and structured conclusions. However, statistical inference is not a direct reflection of reality but rather a tool that depends heavily on assumptions and idealized models. These models simplify complex phenomena, and the validity of their conclusions relies on how well these assumptions align with the underlying data-generating processes.

The random sampling model serves as a stepwise framework in statistical inference, guiding how we connect finite samples to broader populations:

**Step 1.- Defining the Population and Collecting Data:** The first step is to define the population of interest and determine how data will be collected. This involves specifying the sampling process and ensuring that each data point is drawn independently and identically distributed (iid) from the same population, with every unit having a nonzero probability of selection. Once the sampling process is defined, data is collected as a sample from the population, and each observation is treated as a realization of a random variable following the specified distribution.

**Step 2.- Constructing and Checking the Random Sampling Model:** Based on the assumptions about how the data was generated, the random sampling model is mathematically expressed. For example, , where is the unknown population distribution with parameter . This step formalizes the relationship between the data and the population. Real-world data rarely fits these idealized assumptions perfectly, so it is essential to check for deviations such as selection biases or lack of independence and address them appropriately.

**Step 3.- Applying the Likelihood Function and Estimating Parameters:** The likelihood function plays a central role in parameter estimation by quantifying how plausible different parameter values are given the observed data . Methods such as maximum likelihood estimation (MLE) or Bayesian inference refine these estimates, translating raw data into structured knowledge about the population. For example, the sample mean is often used as an estimate of the population mean .

**Step 4.- Quantifying Uncertainty and Generalizing to the Population:** Recognizing that the sample is just one realization of a random process, uncertainty must be quantified to understand how sample statistics behave across repeated samples. Sampling distributions, confidence intervals, and hypothesis tests help describe this variability. The Central Limit Theorem ensures that for large samples, the sample mean approximates a normal distribution, making inference more reliable. Finally, results are generalized from the sample to the population using frequentist methods such as confidence intervals and p-values or Bayesian methods that update beliefs about by incorporating prior knowledge.

The random sampling model provides a structured, step-by-step approach for making inferences from data. However, each step depends on assumptions that rarely hold perfectly in practice. As such, conclusions drawn from statistical inference should be viewed as tentative and context-dependent, subject to revision when new data or insights become available.

**Definition 16.1.1** A *statistical model* is a random variable, together with a specification of its probability distribution, and the identification of the parameters, denoted by  $\theta$ , of that distribution.

A statistic is a function of the observable data. A statistic is used to summarize or describe some aspect of the sample. For example, the sample mean and the sample variance are statistics because they summarize data drawn from a sample.

**Definition 16.1.2** Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a random sample. A *statistic* is a random variable  $T = r(X_1, \dots, X_n)$ , where  $r()$  is an arbitrary real-valued function of  $n$  variables.

A *parametric random variable* is a random variable  $X$  that belongs to a family of functions parameterized by  $\theta$ . The parameter  $\theta$  can either be a single scalar or a vector of values, and it is treated itself as a random variable that follows a probability distribution. The set  $\Theta = \{\theta_1, \theta_2, \dots\}$ , consisting of all possible values of  $\theta$ , is called the *parameter space*, and its elements are referred to as *parameters*. When the parameter  $\theta$  is unknown, the distribution of the random variable  $X$  is said to be conditional on  $\theta$ , denoted by  $f(X | \theta)$ .

As was the case in Chapter 12, in this section, we are only considering parametric discrete random variables defined over discrete probability spaces.

■ **Example 16.1** We have seen in Definition 12.5.4 that a binomial distribution with parameters  $n$  and  $p$  is a model for a family of experiments in which we are interested in knowing the number of successes in a sequence of  $n$  independent binary trials, where the probability of success is  $p$ . If  $X$  is a random variable following a binomial distribution, the probability of getting

exactly  $k$  successes is given by:

$$\Pr(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

In statistical inference, we are usually interested in the inverse problem. That is, we have the actual result of an experiment composed of  $n$  trials, in which we know how many successes  $k$  we have obtained, and we would like to estimate the parameter  $p$ , that is, the probability of success. ■

We assume that the true value of the unknown parameter  $\theta$  can be inferred, typically by analyzing a collection of data samples. The observable data  $\mathbf{X} = (X_1, \dots, X_n)$  is modeled as a random sample from the distribution  $f(X | \theta)$  conditional on  $\theta$ .

An estimator of a parameter  $\theta$  is a function of the random sample  $\mathbf{X}$  that we hope provides a value close to the unknown parameter  $\theta$ .

**Definition 16.1.3** Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a random sample from a discrete random variable  $X$  with parameter  $\theta$ . An *estimator* of the parameter  $\theta$  is a real-valued function  $\delta(X_1, \dots, X_n)$ . If  $X_1 = x_1, \dots, X_n = x_n$  are observed, then  $\delta(x_1, \dots, x_n)$  is called the *estimate* of  $\theta$ .

The estimator itself is a random variable, and its probability distribution can be derived from the joint distribution of  $X_1, \dots, X_n$ . Every estimator is a statistic, but not every statistic is an estimator. The estimate is a real number.

Estimators help us estimate the quantities of interest in statistical inference and quantitatively assess the quality of the results. A good estimator is one where it is highly likely that the error  $\delta(\mathbf{X}) - \theta$  will be close to zero.

In practice, we begin our analysis by deriving a distribution for the parameter  $\theta$  based on theoretical assumptions and our current knowledge of the experiment.

When selecting an estimator, it is important to measure how "good" an estimate is. One common approach is to quantify the loss or cost associated with choosing an estimate that deviates from the true parameter value. This is done using a loss function, which assigns a numerical penalty to the difference between the estimate and the actual parameter.

**Definition 16.1.4** A loss function is a real-valued function of two variables,  $L(\theta, a)$ , where  $\theta \in \Theta$  and  $a$  is a real number.

The loss function represents the cost incurred when the true parameter value is  $\theta$  and the estimate is  $a$ . In other words,  $L(\theta, a)$  quantifies the loss when there is a discrepancy between the estimate and the true value.

Let  $\xi(\theta)$  denote the prior probability mass function (p.m.f.) of  $\theta$  on the set  $\Theta$ . Given a particular estimate  $a$ , the expected loss for discrete random

variables is computed as:

$$E [L(\theta, a)] = \sum_{\theta \in \Theta} L(\theta, a) \xi(\theta)$$

The expected loss is the weighted average of the losses, where the weights are the probabilities  $\xi(\theta)$  assigned to each possible value of  $\theta$ . The goal is to choose an estimate  $a$  that minimizes this expected loss.

The loss function  $L(\theta, a) = (\theta - a)^2$  is called squared error loss. Let  $\theta$  be a real-valued parameter. Suppose that the squared error loss function is used and that the posterior mean of  $\theta$ ,  $E(\theta | \mathbf{X})$ , is finite. Then, a Bayes estimator of  $\theta$  is  $\delta^*(\mathbf{X}) = E(\theta | \mathbf{X})$ .

The loss function  $L(\theta, a) = |\theta - a|$  is called absolute error loss. When the absolute error loss function is used, a Bayes estimator of a real-valued parameter  $\delta^*(\mathbf{X})$  equal to a median of the posterior distribution of  $\theta$ .

### 16.1.1 Maximum Likelihood Estimator

The Maximum Likelihood Estimator (abbreviated MLE) is a method of estimating the parameters of a probability model. It is one of the most commonly used techniques in statistics for fitting model parameters to observed data. The core idea behind MLE is to find the parameter values that maximize a likelihood function, which represents the probability of observing the given data under the model. MLE is widely used due to its desirable properties for large samples, but its effectiveness relies on correct model specification and may struggle with small sample sizes or computational difficulties.

The likelihood represents the probability (or likelihood) of observing the data as a function of the parameters of the model.

**Definition 16.1.5** Let  $X_1, X_2, \dots, X_n$  be  $n$  independent and identically distributed discrete random variables, and let  $P(X_i = x_i | \theta)$  be the probability mass function of  $X_i$ , where  $\theta \in \Theta$  is an unknown parameter, and  $\Theta$  is the parameter space. The *likelihood function* is then defined as:

$$L(\theta | X_1, X_2, \dots, X_n) = \prod_{i=1}^n P(X_i = x_i | \theta)$$

where  $x_1, x_2, \dots, x_n$  are the observed data points.

The likelihood function is a product of probabilities. Taking the logarithm of the likelihood, what it is called the *log-likelihood function*, simplifies this product into a sum:

$$\ell(\theta | X_1, X_2, \dots, X_n) = \log L(\theta | X_1, X_2, \dots, X_n) = \sum_{i=1}^n \log f(X_i | \theta)$$

Sums are generally much easier to work with than products, particularly when performing differentiation for optimization purposes. This transformation makes it more straightforward to compute derivatives and apply optimization algorithms.

Maximum likelihood estimation is a method that determines values for the parameters of a model. The parameter values are found such that they maximise the likelihood that the process described by the model produced the data that were actually observed.

**Definition 16.1.6** The *Maximum Likelihood Estimator*, abbreviated as MLE, for discrete random variables is defined as the parameter value that maximizes the likelihood function, i.e.,

$$\hat{\theta}_n = \arg \max_{\theta \in \Theta} L(\theta | X_1, X_2, \dots, X_n)$$

Equivalently, the MLE can also be found by maximizing the log-likelihood function:

$$\hat{\theta}_n = \arg \max_{\theta \in \Theta} \ell(\theta | X_1, X_2, \dots, X_n)$$

**■ Example 16.2** Consider  $X_1, X_2, \dots, X_n$  i.i.d. random variables from a Bernoulli distribution with parameter  $p$ . The probability mass function is:

$$P(X_i = x_i | p) = p^{x_i} (1 - p)^{1-x_i}$$

where  $X_i \in \{0, 1\}$ . The likelihood function for this sample is:

$$L(p | X_1, X_2, \dots, X_n) = \prod_{i=1}^n p^{X_i} (1 - p)^{1-X_i}$$

Which can be rewritten as:

$$L(p) = p^S (1 - p)^{n-S}$$

where  $S = \sum_{i=1}^n X_i$  is the total number of successes (number of times  $X_i = 1$ ). The function  $f(p) = p^a (1 - p)^b$  attains its maximum at  $p = \frac{a}{a+b}$ . In our case,  $a = S$  and  $b = n - S$ , so the likelihood function  $L(p)$  reaches its maximum at:

$$\hat{p} = \frac{S}{S + (n - S)} = \frac{S}{n}$$

Therefore, the maximum likelihood estimator for  $p$  is:

$$\hat{p} = \frac{1}{n} \sum_{i=1}^n X_i$$

which is simply the sample mean of the Bernoulli trials (the proportion of successes). ■

The maximum likelihood estimator is consistent.

**Proposition 16.1.1** Let  $\hat{\theta}_n$  be the MLE for  $\theta$  when the random variables are discrete. Under regularity conditions (such as identifiability of the model and the existence of a unique maximizer of the likelihood function), the MLE is consistent, i.e.,

$$\hat{\theta}_n \xrightarrow{P} \theta_0 \quad \text{as } n \rightarrow \infty$$

where  $\theta_0$  is the true value of the parameter.

*Proof.* **TODO: Finish** 1. Show that the expected log-likelihood is maximized at the true parameter  $\theta_0$ . 2. Prove that the empirical likelihood function converges uniformly to the expected log-likelihood as the sample size increases. 3. Establish that  $\hat{\theta}_n$  converges to  $\theta_0$  as  $n \rightarrow \infty$ . ■

The MLE has several notable disadvantages. Firstly, it is highly dependent on sample size; for small samples, MLE may exhibit bias or high variance, as its desirable asymptotic properties, such as consistency, are guaranteed only for large datasets. Additionally, MLE requires correct model specification. It is sensitive to assumptions about the underlying model, and if the model is misspecified, the resulting estimates can be misleading. From a computational perspective, maximizing the likelihood function can pose numerical challenges, often necessitating complex optimization techniques that are computationally demanding and may result in convergence to local rather than global maxima. Furthermore, MLE is sensitive to outliers; since it seeks to maximize the likelihood based on observed data, extreme values can disproportionately affect the estimates, leading to distortions.

■ **Example 16.3** Consider a situation where we are dealing with a uniform distribution  $U(0, \theta)$ , where  $\theta$  is the unknown upper bound of the distribution. We have a sample  $X_1, X_2, \dots, X_n$  of i.i.d. observations from this distribution, and we wish to estimate the parameter  $\theta$  using the maximum likelihood estimator. The log-likelihood function is:

$$\ell(\theta | X_1, X_2, \dots, X_n) = -n \log \theta, \quad \text{subject to } \theta \geq \max(X_1, X_2, \dots, X_n)$$

To maximize the log-likelihood, observe that  $\ell(\theta)$  is a decreasing function of  $\theta$ . Therefore, the maximum likelihood occurs when  $\theta$  is minimized, subject to the constraint  $\theta \geq \max(X_1, X_2, \dots, X_n)$ . Hence, the MLE for  $\theta$  is:

$$\hat{\theta} = \max(X_1, X_2, \dots, X_n)$$

The MLE  $\hat{\theta}$  tends to underestimate the true value of  $\theta$  when the sample size is small. This is because the maximum observation in a sample is unlikely to exactly reach the true upper bound, leading to a systematic bias. For example, if  $\theta = 10$  and we take a sample of size 5, the expected value of  $\hat{\theta}$  is  $\frac{5}{6} \cdot 10 = 8.33$ , meaning we systematically underestimate the true parameter.

Since the MLE depends only on the maximum observed value, the estimate  $\hat{\theta}$  is highly sensitive to outliers. If an unusually large value appears in the sample, it can drastically inflate the estimate of  $\theta$ , even if the rest of the data suggests a much smaller upper bound. This makes the MLE fragile in the presence of anomalous observations.

Another issue with the MLE in this case is that it ignores most of the data. The MLE is solely determined by the largest value in the sample, meaning the other  $n - 1$  data points do not contribute to the estimate. This seems inefficient because the estimator does not use all available information, which could be helpful in producing a more reliable estimate. ■

### 16.1.2 Bayesian Inference

Unlike the Maximum Likelihood Estimator (MLE), which selects the parameter values that maximize the probability of the observed data, *Bayesian inference* takes a fundamentally different approach by treating parameters as probability distributions rather than fixed values. Bayesian inference also incorporates prior knowledge, allowing for an informed estimation process even before observing data. This approach enables Bayesian methods to dynamically update beliefs as new data becomes available. Bayes' theorem (see Theorem 12.2.2) formalizes this process by combining a prior probability mass function with the likelihood of the observed data to produce a posterior probability mass function, which represents the updated belief about the parameter.

**Definition 16.1.7** Let  $f(X | \theta)$  be the probability mass function of a discrete random variable  $X$  with parameter  $\theta$ . The probability distribution of the parameter  $\theta$ , denoted by  $\xi(\theta)$ , is called the *prior distribution*.

The prior distribution must be defined over the parameter space  $\Theta$ . It is called the prior distribution because it represents our knowledge or belief about the parameter  $\theta$  before observing any data.

**Definition 16.1.8** Let  $f(X | \theta)$  be the probability mass function of a discrete random variable  $X$  with parameter  $\theta$ , and let  $\mathbf{X} = (X_1, \dots, X_n)$  be a random sample from  $X$ . The conditional probability mass function of the parameter  $\theta$  given the observed values  $X_1 = x_1, \dots, X_n = x_n$ , denoted  $\xi(\theta | x_1, \dots, x_n)$ , is called the *posterior distribution*.

The posterior distribution represents our updated knowledge of the parameter  $\theta$  after taking into account the observed data. A strongly informative prior will dominate the posterior, while a weak or uniform prior will let the data speak for itself.

The posterior pmf is computed using Bayes' theorem:

$$\xi(\theta | \mathbf{X}) = \frac{L(\mathbf{X} | \theta) \xi(\theta)}{P(\mathbf{X})}$$

where:

- $\xi(\theta)$  is the prior distribution, which encodes our belief about  $\theta$  before seeing the data.
- $L(\mathbf{X} | \theta)$  is the likelihood function, representing the probability of observing the data given a particular value of  $\theta$ .
- $P(\mathbf{X})$  is the marginal probability, ensuring the posterior pmf is a valid probability mass function, computed as:

$$P(\mathbf{X}) = \sum_{\theta \in \Theta} L(\mathbf{X} | \theta) \xi(\theta)$$

Thus, the posterior pmf  $\xi(\theta | \mathbf{X})$  represents an updated belief about  $\theta$  after incorporating the observed data.

**■ Example 16.4** Suppose we have a biased coin with an unknown probability  $\theta$  of landing heads. We want to estimate  $\theta$  using Bayesian inference after observing a few coin flips. Before flipping the coin, we assume a prior belief about  $\theta$ . Let's say we assume a uniform prior:

$$\xi(\theta) = \begin{cases} \frac{1}{3}, & \theta \in \{0.2, 0.5, 0.8\} \\ 0, & \text{otherwise} \end{cases}$$

This prior suggests we believe  $\theta$  is equally likely to be 0.2, 0.5, or 0.8 before any data is observed. Now, we flip the coin  $n = 3$  times and observe  $x = 2$  heads. Assuming independent flips, the likelihood function is given by:

$$p(X = x | \theta) = \binom{3}{2} \theta^2 (1 - \theta)^1$$

For the possible values of  $\theta$ :

$$p(2 | 0.2) = \binom{3}{2} (0.2)^2 (0.8)^1 = 3(0.04)(0.8) = 0.096$$

$$p(2 | 0.5) = \binom{3}{2} (0.5)^2 (0.5)^1 = 3(0.25)(0.5) = 0.375$$

$$p(2 | 0.8) = \binom{3}{2} (0.8)^2 (0.2)^1 = 3(0.64)(0.2) = 0.384$$

Using Bayes' rule:

$$\xi(\theta | X = 2) = \frac{p(X = 2 | \theta)\xi(\theta)}{P(X = 2)}$$

where the denominator is the marginal probability:

$$P(X = 2) = \sum_{\theta} p(X = 2 | \theta)\xi(\theta)$$

$$P(X = 2) = (0.096 \times \frac{1}{3}) + (0.375 \times \frac{1}{3}) + (0.384 \times \frac{1}{3})$$

$$P(X = 2) = 0.032 + 0.125 + 0.128 = 0.285$$

Now, calculating the posterior probabilities:

$$\xi(0.2 | X = 2) = \frac{0.096 \times \frac{1}{3}}{0.285} = \frac{0.032}{0.285} \approx 0.112$$

$$\xi(0.5 | X = 2) = \frac{0.375 \times \frac{1}{3}}{0.285} = \frac{0.125}{0.285} \approx 0.439$$

$$\xi(0.8 | X = 2) = \frac{0.384 \times \frac{1}{3}}{0.285} = \frac{0.128}{0.285} \approx 0.449$$

Before observing the data, we believed  $\theta$  was equally likely to be 0.2, 0.5, or 0.8. After observing 2 heads in 3 flips, our belief has shifted: the most probable value for  $\theta$  is now 0.8 (44.9% probability), followed by 0.5 (43.9% probability). The probability that  $\theta = 0.2$  has dropped significantly to 11.2%. ■

A Bayes estimator is a point estimate derived from the posterior distribution in Bayesian inference, similar to the Maximum Likelihood Estimator (MLE), which selects the parameter values that maximize the likelihood function. However, while MLE relies solely on observed data, the Bayes estimator incorporates both prior knowledge and observed data through the posterior distribution. It is a single value that represents the "best guess" for the unknown parameter by minimizing the expected loss under a given loss function.

Suppose we can observe the value  $\mathbf{x}$  of the random vector  $\mathbf{X}$  before estimating  $\theta$ , and let  $\xi(\theta | \mathbf{x})$  denote the posterior pmf of  $\theta$  on  $\Omega$ . For each estimate  $a$  the expected loss will be

$$E[L(\theta, a) | \mathbf{x}] = \sum_{\theta \in \Theta} L(\theta, a) \xi(\theta | \mathbf{x})$$

**Definition 16.1.9** Let  $L(\theta, a)$  be a loss function. For each possible value  $\mathbf{x}$  of  $\mathbf{X}$ , let  $\delta^*(\mathbf{x})$  be a value of  $a$  such that  $E[L(\theta, a) | \mathbf{x}]$  is minimized. Then  $\delta^*$  is called a *Bayes estimator* of  $\theta$ . Once  $\mathbf{X} = \mathbf{x}$  is observed,  $\delta^*(\mathbf{x})$  is called a *Bayes estimate* of  $\theta$ .

Another way to describe a Bayes estimator  $\delta^*$  is to note that, for each possible value  $\mathbf{x}$  of  $\mathbf{X}$ , the value  $\delta^*(\mathbf{x})$  is chosen so that

$$E[L(\theta, \delta^*(\mathbf{x})) | \mathbf{x}] = \min_{\forall a} E[L(\theta, a) | \mathbf{x}]$$

The theory of Bayes estimators provides a satisfactory and coherent theory for the estimation of parameters. To apply the theory, it is necessary to specify a particular loss function, and also a prior distribution for the parameter. Specifying a meaningful prior distribution in a multidimensional parameter space  $\Omega$  is particularly challenging because it requires incorporating dependencies between multiple parameters, which may not be well understood. Additionally, computational complexity increases significantly as the number of parameters grows, making posterior calculations more difficult.

## 16.2 Machine Learning

Machine learning is the study of algorithms that, given a finite sample of individuals from a population, learn a rule to make predictions about new, previously unseen individuals. Each individual is represented by a vector of attributes, one of which is designated as the target. Since the true relationship between the predictors and the target is unknown, the algorithm must approximate it. The primary objective is to construct a function, called a model, that uses the observed attributes to predict the target with minimal error, both on the training examples and on future individuals drawn from the same population.

Although statistical inference and machine learning share the goal of using data to draw conclusions about unseen cases, they tend to emphasise different aspects of this task. Classical statistical inference often begins with a probabilistic model and seeks estimators that maximise likelihood, yield interpretable parameters, and satisfy theoretical properties such as consistency, efficiency, or convergence. In contrast, machine learning typically starts with a flexible class of functions, often implemented as large-scale software artefacts such as neural networks, and focuses on minimising an empirical loss that reflects predictive accuracy, even in the absence of formal guarantees. In practice, however, the boundary is fluid: statisticians often minimise risk functions, and many machine learning methods are grounded in statistical theory.

In this book, we present machine learning as a distinct discipline, adopting a fully deterministic approach. We introduce models, residuals, and optimization criteria without relying on probability theory or its results. This choice is deliberate: by avoiding probabilistic assumptions, we align the theoretical foundations of machine learning with the framework of the theory of nescience, facilitating the integration and reuse of results across both domains.

We begin by defining the basic mathematical objects that will underpin the rest of the discussion: populations and their individuals.

**Definition 16.2.1** A *population* is a non-empty, well-defined set  $\mathcal{S}$ . The elements  $\mathbf{x} \in \mathcal{S}$  are called *individuals*.

Throughout this book, we restrict our attention to countable populations. Both finite and countably infinite cases are permitted.

Each individual is described by a fixed list of attributes (also called features), and each attribute has a domain that specifies its set of admissible values.

**Definition 16.2.2** Each individual  $\mathbf{x}$  in the population  $\mathcal{S}$  is characterized by  $p$  attributes ( $p \geq 1$ ), such that  $\mathbf{x} = (x_1, x_2, \dots, x_p) \in \mathcal{S}$ . For each  $i \in 1, \dots, p$ , the *domain* of the  $i$ -th attribute is defined as  $\mathcal{D}_i = \{x_i : \mathbf{x} \in \mathcal{S}\}$ .

The set  $\mathcal{D}_i$  may be any type of set—for example, the real numbers  $\mathcal{D}_i = \mathbb{R}$ , a set of integers such as  $0, 1, \dots, k$ , or a collection of categorical labels like red, green, blue. An attribute is called *quantitative* if the elements of  $\mathcal{D}_i$  can be measured numerically. Quantitative attributes can be further classified as *discrete* if they take a countable number of distinct values, or *continuous* if they can take any value within a given range or interval.

An attribute is called *qualitative* or *categorical* if it represents characteristics that cannot be measured numerically. A qualitative attribute is said to be *nominal* if there is no natural order among its categories, and *ordinal* if such an order exists.

■ **Example 16.5** Consider a population consisting of the inhabitants of a small village, where we are interested in studying three attributes: age, gender, and daily water consumption. The attribute representing age, mapping individuals to their age in years, is a quantitative discrete attribute, as it takes integer values. Gender, mapping individuals to categories like "Male" or "Female," is a qualitative nominal attribute, since the categories have no inherent order. Daily water consumption, mapping individuals to the number of liters they drink per day, is a quantitative continuous attribute, as it can take any real value within a measurable range. ■

Broadly speaking, machine learning algorithms can be classified into

two main categories: supervised and unsupervised. In *supervised* learning, we are given a collection of training samples (also called *predictors*) along with their corresponding observed target values (or *labels*), and our goal is to predict the target for new, previously unseen observations. In contrast, in *unsupervised* learning, no target values are provided; we are given only training samples, and the objective is to uncover the underlying structure of the data. We postpone the discussion of unsupervised learning to [Section XX](#).

**Definition 16.2.3** Let  $\mathcal{S}$  be a population whose individuals  $\mathbf{x}$  are characterized by  $p$  *attributes*, with  $\mathbf{x} = (x_1, x_2, \dots, x_p) \in \mathcal{S}$ . A *target variable*, denoted by  $\mathbf{y}$ , is a vector corresponding to another attribute of the individuals in  $\mathcal{S}$ , distinct from the  $p$  attributes used as predictors. The *domain* of the target variable is  $\mathcal{D}_y = y_i$ .

We typically choose statistical learning methods based on whether the target attribute is quantitative or qualitative. (Whether the predictors are qualitative or quantitative is generally considered less critical.) Supervised learning algorithms are applied to *regression problems* when the target attribute is quantitative, and to *classification problems* when the target attribute is qualitative.

We assume the existence of an underlying function, or model, that relates the predictors to the target.

**Definition 16.2.4** Let  $\mathcal{S}$  be a population and  $\mathbf{y}$  a target attribute. A *model* is a function  $f : \mathcal{S} \rightarrow \mathbf{y}$  such that, for every individual  $\mathbf{x} \in \mathcal{S}$ , we have  $y = f(\mathbf{x})$ .

In practice, and for most of the populations, the set of attributes measured is not sufficient to fully characterize the target variable, meaning that  $f(\mathbf{x})$  do not maps to  $y_k$ , but to something approximated. This fact is modeled in the discipline of machine learning using a random variable, and saying that  $y_k = f(\mathbf{x}) + \varepsilon_k$  where the errors  $\varepsilon_1, \dots, \varepsilon_n$  are independent,  $\mathbb{E}[\varepsilon_k] = 0$ , and  $\text{Var}[\varepsilon_k] < \infty$ .

In practice, for most populations, the measured attributes are not sufficient to fully determine the target variable. This means that  $f(\mathbf{x})$  does not exactly map to the observed value  $\mathbf{y}$ , but rather to an approximation. In the traditional formulation of machine learning, this uncertainty is modeled using a random variable, assuming that

$$y_k = f(\mathbf{x}) + \varepsilon_k,$$

where the errors  $\varepsilon_1, \dots, \varepsilon_n$  are independent random variables, with  $\mathbb{E}[\varepsilon_k] = 0$  and  $\text{Var}[\varepsilon_k] < \infty$ .

Learning algorithms operate on a finite sample of data, called the *training set*, from which they infer an approximation to the unknown function  $f$ .

**Definition 16.2.5** A *training data set*, is a finite subset of the population  $\mathcal{S}$ , consisting of a collection of predictor  $\mathbf{X}$  and the corresponding target  $\mathbf{y}$ .

Let  $x_{ij}$  represent the value of the  $j$ -th predictor for the  $i$ -th observation, where  $i = 1, 2, \dots, n$  and  $j = 1, 2, \dots, p$ . Correspondingly, let  $y_i$  represent the target value for the  $i$ -th observation. Then, the training data consist of the pairs  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , where each  $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ .

The objective of supervised learning is to construct a model  $\hat{f} : \mathcal{S} \rightarrow \mathbf{y}$  such that the predicted value  $\hat{y}$  is as close as possible to the true value  $y(\mathbf{x})$ .

**Definition 16.2.6** Given a training set  $\mathcal{T}$ , a learning procedure returns an *estimator*

$$\hat{f} : \prod_{i \neq j} \mathcal{D}_i \longrightarrow \mathcal{D}_j.$$

For a new individual  $\mathbf{x} \notin \mathbf{x}_1, \dots, \mathbf{x}_n$ , the predicted target value is

$$\hat{y} = \hat{f}(\mathbf{x}_{-j}),$$

where  $\mathbf{x}_{-j}$  denotes the vector of predictor values, excluding the target.

The challenge for the machine learning algorithms is to ensure that the estimator  $\hat{f}$  generalizes well to the entire population  $\mathcal{S}$ , even though it was constructed using only the training data  $\mathbf{X}$ .

## 16.2.1 Parametric vs. Non-parametric Models

Most of the statistical learning methods can be characterized as either parametric or non-parametric. With *parametric* methods we select a priori a functional form, and we fit the free parameters of this functional form. In contrast, with *non-parametric* models we do not make any assumption about the form of the models. Given their flexibility, non-parametric methods usually require more training data to train, and they are more difficult to interpret by humans.

Parametric methods involve a two-step model-based approach: 1) First, we make an assumption about the functional form, or shape, of  $f$ ; 2) After a model has been selected, we need a procedure that uses the training data to fit or train the model. Parametric methods reduce the problem of estimating  $f$  down to one of estimating a set of parameters. The potential disadvantage of a parametric approach is that the model we choose will usually not match

the true unknown form of  $f$ . More flexible models can fit many different possible functional forms for  $f$ . But fitting a more flexible model requires estimating a greater number of parameters. More complex models can lead to a phenomenon known as overfitting the data, which essentially means they follow the errors, or noise, too closely.

TODO: Provide an example, like linear regression.

Non-parametric methods do not make explicit assumption about the functional form of  $f$ . By avoiding the assumption of a particular functional form for  $f$ , they have the potential to accurately fit a wider range of possible shapes for  $f$ . However, with non-parametric methods, a very large number of observations is required in order to obtain an accurate estimate for  $f$ .

TODO: Provide an example.

Concept of regularization to control model complexity. Ridge and Lasso: definitions and intuition. Regularized objective: Effect on optimization and generalization. Practical considerations: choosing.

TODO: Talk about the overfitting problem

There exists a trade-off between flexibility and interpretability of the machine learning methods. If we are interested in inference, then restrictive models are much more interpretable. If we are only interested in prediction, the interpretability of the predictive model is simply not of interest.

## 16.2.2 Model Accuracy

The error term  $\varepsilon$  introduced in Equation ?? correspond to variables that have not been taken into account in our study, and other effects that can not be measured. This kind of error is called *irreducible*, since there is nothing we can do to reduce it. A second type of error, called *reducible*, refers to the fact that our estimate  $\hat{f}$  of the function  $f$  might not be perfect. It is called reducible because with better estimates the error will decrease.

Derive this property

We are interested in the average error made by our model. Assuming that both  $\hat{f}$  and  $X$  are fixed, it can be shown that

$$E(Y - \hat{Y})^2 = E[f(X) + \varepsilon - \hat{f}(X)]^2 = [f(X) - \hat{f}(X)]^2 + Var(\varepsilon)$$

The irreducible error is an upper bound to the accuracy of our predictions. Unfortunately, in practice, this bound is almost always unknown.

### Mean Squared Error

A common metric used to quantitatively evaluate and compare the performance of the different machine learning algorithms is to compute the mean

square error (MSE) of the predictions made,

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{f}(X_i))^2$$

where  $Y_i$  is the  $i$ th observed value, and the  $\hat{f}(X_i)$  is the prediction that  $\hat{f}$  gives for the  $i$ th vector of predictors.

### Explain how MSR relates to MLE

We are interested in the capability of the model  $\hat{f}$  to generalize to previously unseen data, that is, to correctly make predictions based on input vectors not included in the training dataset  $\mathcal{X}$ . In this sense, our goal should be to select that method with the lowest MSE over a test dataset, that is, over a collection of input vectors that have not been used for the training of the algorithm. When a model  $\hat{f}$  has a very low train MSE but very high test MSE we say that the model overfits the training data.

If the response variable is qualitative the quantity we seek to minimize is the average number of misclassification made by the model, that is,

$$\frac{1}{n} \sum_{i=1}^n I(Y_i \neq \hat{f}(X_i))$$

where  $\hat{f}(X_i)$  is the predicted class for the  $i$ th observation, and  $I$  is a function that equals 1 if  $Y_i \neq \hat{f}(X_i)$  and zero otherwise.

### 16.2.3 No free lunch theorem

**Inductive Bias and the Role of Assumptions.** Why learning is not possible without assumptions. Inductive bias: choosing  $H$ , priors, constraints. Examples: smoothness, sparsity, linearity. The No Free Lunch Theorem (informal statement)

There is no free lunch in statistics: no one method dominates all others over all possible data sets. On a particular data set, one specific method may work best, but some other method may work better on a similar but different data set

### 16.2.4 The bias-variance trade-off

It is possible to show that the expected test MSE, for a given value  $x_0$ , can be always decomposed into the sum of three fundamental quantities: the variance of  $\hat{f}(x_0)$ , the squared bias of  $\hat{f}(x_0)$  and the variance of the error term  $\varepsilon$ :

$$E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) [Bias(\hat{f}(x_0))]^2 + Var(\varepsilon)$$

where  $E(y_0 - \hat{f}(x_0))^2$  defines the expected test MSE, and refers to the average test MSE that would obtain if we repeatedly estimated  $f$  using a large number of training sets, and tested each at  $x_0$ .

In order to minimize the expected test error, we need to select a statistical learning method that simultaneously achieves low variance and low bias. The expected test MSE can never lie below  $Var(\varepsilon)$ , the irreducible error. Variance refers to the amount by which  $\hat{f}$  would change if we estimated it using a different training data set. In general, more flexible statistical methods have higher variance. Bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much more simpler model. Generally, more flexible methods result in less bias. As a general rule, as we use more flexible methods, the variance will increase and the bias will decrease. The relative rate of change of these two quantities determines whether the test MSE increases or decreases.

The relationship between bias, variance, and test set MSE is referred to as the bias-variance trade-off. It is easy to obtain a method with extremely low bias but high variance, or a method with very low variance but high bias. The challenge lies in finding a method from which both the variance and the squared bias are low. In a real-life situation in which  $f$  is unobserved, it is generally not possible to explicitly compute the test MSE, bias, or variance for a statistical learning methods. Alternative approaches, like for example cross-validation, are used to estimate the test MSE using the training data.

### 16.2.5 Generative vs. discriminative models

TODO: Pending

### 16.2.6 Discretization of Continuous Variables

When summarizing large masses of raw data, it is often useful to distribute the data into classes, or categories, and to determine the number of individuals belonging to each class, called absolute frequency. The following definition formally introduces this concept.

**Definition 16.2.7** Let  $\mathcal{S}$  be a population consisting of  $n$  individuals, and let a variable  $X : \mathcal{S} \rightarrow \mathcal{D}$  represent the mapping of individuals in  $\mathcal{S}$  to values in the set  $\mathcal{D}$ , where  $k$  is the cardinality of  $\mathcal{D}$ . The *absolute frequency*, also known simply as *frequency*, denoted by  $n_i$  for  $1 \leq i \leq k$ , quantifies the number of individuals in  $\mathcal{S}$  for which  $X$  assigns the value corresponding to the  $i$ -th category of  $\mathcal{D}$ .

The sum of the frequencies must be equal to population size, that is,  $\sum_{i=1}^k n_i = n$ .

If the variable  $X$  is continuous, the elements of  $\mathcal{D}$  are referred to as *class intervals*. The endpoints of these intervals are known as *class limits*; the smaller number is termed the *lower class limit*, and the larger number, the *upper class limit*. A *class interval* that lacks either an upper or a lower class limit is known as an *open class interval*. The *width* of a class interval is defined as the difference between the upper and lower class boundaries, denoted by  $a_i = e_i - e_{i-1}$ . The *class mark*, or the midpoint of a class interval, is calculated as  $a_i = \frac{e_i + e_{i-1}}{2}$ . It is assumed that all observations within a specific class interval are equivalent concerning their categorical assignment. Refer to Section ?? for more information about how to discretize a continuous variable into discrete intervals.

**■ Example 16.6** In a study measuring adult heights within a community, researchers record heights ranging from 150 cm to 200 cm and organize them into 10 cm class intervals: 150-159 cm, 160-169 cm, 170-179 cm, 180-189 cm, and 190-199 cm. Each interval's lower and upper class limits are respectively the start and end points, such as 150 cm and 159 cm for the first interval. If the last interval had no specified upper limit, it would be considered an open class interval. The class width, typically 10 cm, is the operational span between boundaries, and the class mark, calculated as the midpoint of each interval (e.g., 154.5 cm for the first interval), provides a central value for summarizing data within that range. ■

A tabular arrangement of data by classes together with the corresponding class frequencies is called a frequency distribution, or frequency table.

**Definition 16.2.8** Let  $\mathcal{S}$  be a population consisting of  $n$  individuals, and let  $X : \mathcal{S} \rightarrow \mathcal{D}$  be a variable that maps individuals in  $\mathcal{S}$  to  $k$  distinct values of the set  $\mathcal{D}$ . A *frequency distribution* is represented the set of pairs  $\{(d_i, n_i) : 1 \leq i \leq k\}$ , where  $d_i$  denotes the  $i$ -th interval and  $n_i$  represents the number of individuals from  $\mathcal{S}$  whose value under  $X$  falls within the interval  $d_i$ .

Frequency distributions are useful for statistical analysis and helps in visualizing data by grouping values, which simplifies the understanding of distribution and central tendencies within the data.

The relative frequency of a class is the frequency of the class divided by the total frequency of all classes

**Definition 16.2.9** Let  $\mathcal{S}$  be a population consisting of  $n$  individuals, and let a variable  $X : \mathcal{S} \rightarrow \mathcal{D}$  represent the mapping of individuals in  $\mathcal{S}$  to values in the set  $\mathcal{D}$ , where  $k$  is the cardinality of  $\mathcal{D}$ . The *relative frequency*, denoted by  $f_i$ , is the ratio  $f_i = \frac{n_i}{n}$  for  $1 \leq i \leq k$ .

The sum of the relative frequencies is equal to one, that is,  $\sum_{i=1}^k f_i = 1$ .

The total frequency of all values less than the upper class boundary of a given class interval is called the cumulative frequency up to and including that class interval.

**Definition 16.2.10** Let  $\mathcal{S}$  be a population consisting of  $n$  individuals, and let a variable  $X : \mathcal{S} \rightarrow \mathcal{D}$  represent the mapping of individuals in  $\mathcal{S}$  to values in the set  $\mathcal{D}$ , where  $k$  is the cardinality of  $\mathcal{D}$ . The *cumulative frequency*, denoted by  $N_i$  for  $1 \leq i \leq k$ , represents the total number of individuals in  $\mathcal{S}$  for which  $X$  assigns a value less than or equal to the upper limit of the  $i$ -th category of  $\mathcal{D}$ . This is mathematically expressed as  $N_i = \sum_{j=1}^i n_j$ , where  $n_j$  is the absolute frequency of the  $j$ -th category.

By accumulating the frequencies up to each category or interval, cumulative frequencies provides a running total that shows how many data points fall below a certain value.

## Discretization Algorithms

**TODO:** Rewrite this section.

Let  $\mathcal{X}$  a continuous random variable that follows a probability density function  $P_{\mathcal{X}}$ , and assume we have collected  $n$  independent and identically distributed samples  $\mathbf{x} = \{x_1, \dots, x_n\}$  from  $\mathcal{X}$ . We are interested in computing the length of a compressed version of  $\mathbf{x}$  using an optimal compressor. Unfortunately, and except for some degenerate distributions, there is no lossless compression algorithm that produces a string with fewer bits than encoding directly the elements  $\mathbf{x}$ . Compression algorithms for continuous data only work in case that the elements of  $\mathbf{x}$  are not independent, as it is the case with images or sound. But, if this is not the case, the only option available to compress  $\mathbf{x}$  is to use a lossy compression algorithm, where some information is lost.

We are looking for an algorithm to produce a finite non-overlapping partition of  $m$  discrete intervals  $D = \{[d_o, d_1], (d_1, d_2], \dots, (d_{m-1}, d_m]\}$ , where  $d_o = \min \mathbf{x}_j$ , and  $d_m = \max \mathbf{x}_j$ , and  $d_i < d_{i+1}$  for  $i = 0, 1, \dots, m - 1$ , assign a unique label to each interval, and encode the elements of  $\mathbf{x}$  using this labeling schema. As compression algorithm we will use an optimal length code given the relative frequencies of the labels in the encoded vector. In this sense, our goal is to have a collection of intervals with sufficiently number of samples (so they are statistically significant) and that the distribution of frequencies resembles the original probability distribution  $P_{\mathcal{X}}$ .

A discretization algorithm is a mapping between a (possibly huge) number of numeric values and a reduced set of discrete values, and so, it is a process in which some information is potentially lost. The choice of

discretization algorithm is something that could have a high impact in the practical computation of the nescience. We are interested in a discretization algorithm that produces a large number of intervals (low bias), with a large number of number of observations per interval (low variance). Common techniques include *equal width discretization*, *equal frequency discretization* and *fixed frequency discretization*. However, these techniques require the optimization of an hyperparameter, and so, they are not suitable for our purposes.

In a *proportional discretization approach* the number of intervals  $m$  and the number of observations per interval  $s$  are equally proportional to the number of observations  $n$ . The algorithm starts by sorting the values of  $\mathbf{x}_j$  in ascending order and then discretizing them into  $m$  intervals of approximately  $s$  (possibly identical) values each. In this way, as the number of training observations increases, both interval frequency and number of intervals increases, taking advantage of the larger number of observations. In the same way, when the number of observations decreases, we reduce both.

### 16.2.7 k-means Clustering

K-means clustering is an algorithms that partitions  $n$  observations into  $k$  clusters in such a way that each observation belongs to the cluster with the nearest mean. In this way, we can replace the observation that belong to a cluster by its means, as a discretization. The optimization criteria in k-means is to minimize the within-cluster variance. Given the fact that the problem is NP-Complete, some approximation algorithms are used instead.

TODO: Define the concept of Voronoi diagram / Voronoi cell

### 16.2.8 Decision Trees

A decision tree is a mathematical model  $f$  that predicts the value of a target variable  $\mathbf{y}$  by learning simple *if-else* decision rules inferred from the training set  $(\mathbf{X}, \mathbf{y})$  (see Example 16.7). Trees are simple and easy to interpret, but they do not give good accuracy.

The nodes of the tree contain pairs of values  $(j, w)$ , where  $1 \leq j \leq p$  is a feature index and  $w \in \mathbb{R}$  is a threshold, and the tree leafs contain labels of  $\mathcal{G}$  in case of a classification problem, or numbers in case of a regression problem. Given a vector  $\mathbf{x} \in \mathbb{R}^p$  we perform a tree traversal checking at each node if  $x_j \leq w$  to decide if we continue with the left or right branch of the node, until a leaf is reached. We associate the value  $\hat{y}$  of the reached leaf with the vector  $\mathbf{x}$ .

■ **Example 16.7** In Figure 16.1, left side, it is shown an example of a dataset composed by two classes, red dots and blue dots. We want to find a decision

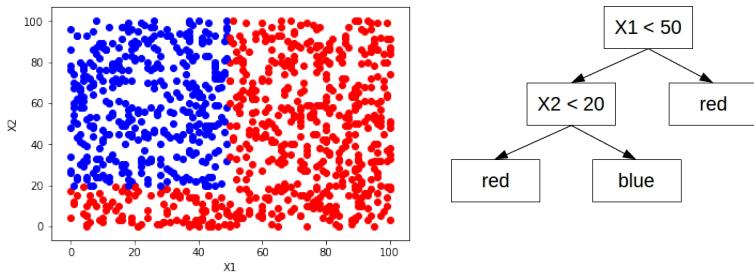


Table 16.1: Example of Decision Tree

tree such that given the features  $X_1$  and  $X_2$ , it returns if the corresponding dot is blue or red. A possible solution to this problem is depicted in the right side of the figure. This decision tree can be also encoded as a function in a programming language, for example in Python, as next code shows.

```

def tree(X1, X2):
    if X1 < 50:
        if X2 < 20:
            return "red"
        else:
            return "blue"
    else:
        return "red"
  
```

The algorithms for the construction of decision trees usually work by recursively partitioning the training set  $\mathbf{X}$  in such a way that the values of the target vector  $\mathbf{y}$  are grouped together, until all partitions are composed by a single label. The problem with these building methods is that they produce very complex trees that overfit the training data. Overfitted trees not only lead to poor predictive capabilities on non-training data, but also produce models that can be exceedingly difficult to interpret. A common approach to avoid overfitting in decision trees is to force an early stopping of the algorithm before the tree becomes too complex. Popular stopping criteria include limiting the maximum depth of the tree, requiring a minimum number of sample points at leaf nodes, or computing the accuracy gain yielded by adding new nodes. However, those heuristics demand the optimization of hyperparameters which makes the training process computationally expensive.

TODO: briefly describe bagging, random forests, and boosting [...] produce multiple trees which are then combined to yield a single consensus prediction [...] combining a large number of trees can often result in dramatic improvement in prediction accuracy, at the expense of some loss of

interpretability [...]

### 16.2.9 Time Series Analysis

A time series is a sequence of measurements taken at successively equally spaced points in time, so there exists a natural ordering of the observations. Examples of time series include the daily closing prices of the Standard and Poor's 500 index, the monthly number of passengers of an airline, or the yearly gross domestic product of a country.

**Definition 16.2.11** A *time series* of length  $n \in \mathbb{N}$ , denoted by  $\{x_t : t = 1, \dots, n\}$  or  $\{x_t\}$ , is a sequence  $\{x_1, x_2, \dots, x_n\}$  of *observed values*.

The elements  $x_i$  of the series correspond to values sampled at fixed time intervals  $1, 2, \dots, n$ . The sampling interval must be short enough to provide a very close approximation to the original continuous signal. Observed values could be continuous, discrete or even categorical.

In statistics, a time series is usually represented as a sequence of  $n$  random variables, and a particular time series is a realization of this representation. In this sense,  $\{x_t\}$  would denote a collection of random variables. In this book, we do not follow this approach for time series formalization.

Time series forecasting refers to the process of building a model to predict future values of the series based on the previously observed values. Time series forecasting is based on identifying the mean features in data and the random variation, and it is generally based on the assumption that present characteristics will continue in the near future, something that cannot be validated in practice.

**Notation 16.1.** Given the time time series  $\{x_t : t = 1, \dots, n\}$  we denote  $\hat{x}_{t+k|t}$  the forecast made at time  $t$  for a future value at time  $t + k$ , where  $k$  is the number of steps in the future.

#### Trends and Seasons

Many time series ... and a repeating seasonal component.

a systematic change in a time series that does not appear to be periodic is known as a trend. [...] A repeating pattern [...] within any fixed period [...] is known as seasonal variation [...] cycles [...] do not correspond to some fixed natural period. [...]

trend [...] change direction in unpredictable times [...] stochastic trend [...]

The main features of many time series are trends and seasonal variations that can be modelled deterministically with mathematical functions of time.

it is usually appropriate to remove trends and seasonal effects before comparing multiple series.

**Definition 16.2.12** Let  $\{x_t\}$  be a time series, a *simple additive model* is defined as

$$x_t = m_t + s_t + z_t$$

where  $m_t$  is called the *trend component*,  $s_t$  is the *seasonal component*, and  $z_t$  is the *error term*.

If the time series presents the property that the seasonal component increases as the trend increases, it might be better to use a multiplicative model.

**Definition 16.2.13** Let  $\{x_t\}$  be a time series, a *simple multiplicative model* is defined as

$$x_t = m_t s_t + z_t$$

where  $m_t$  is called the *trend component*,  $s_t$  is the *seasonal component*, and  $z_t$  is the *error term*.

In practice, a simple approach of estimating the trend of a time series is to compute a moving average.

**Definition 16.2.14** Let  $\{x_t\}$  be a time series, a *simple moving average* of length  $l$  is

The best results are achieved when the length  $l$  of the moving average is equal to the length of the seasonal component. The seasonal component can be estimated by

**Definition 16.2.15** Additive

$$\hat{s}_t = x_t - \hat{m}_t$$

Multiplicative

$$\hat{s}_t = \frac{x_t}{\hat{m}_t}$$

[...] many series are dominated by a trend and/or a seasonal effect [...] A simple additive decomposition model is given by

$$x_t = m_t + s_t + z_t$$

where, at time  $t$ ,  $x_t$  is the observed series,  $m_t$  is the trend,  $s_t$  is the seasonal effect, and  $z_t$  is an error term that is, in general, a sequence of correlated random variables with mean zero.

If the seasonal effect tends to increase as the trend increases, a multiplicative model may be more appropriate

$$x_t = m_t s_t + z_t$$

### ■ Definition 16.2.16

Once we have identified any trend and seasonal effects, we can deseasonalise the time series and remove the trend. If we use the additive decomposition method, we first calculate the seasonality adjusted time series and then remove the trend by subtraction. This leaves the random component, but the random component is not necessarily well modelled by independent random variables. In many cases, consecutive variables will be correlated. If we identify such correlations, we can improve our forecast, quite dramatically if the correlations are high.

### Second Order Properties

A possible approach to forecast future values of a time-series based variable is to extrapolate the current trend and to apply some adaptative estimations.

#### Exponential Smoothing

**Definition 16.2.17** Let's  $x$  and  $y$  two time series. The cross covariance function of  $x$  and  $y$  as a function of a lag  $k$ , denoted  $\gamma(x, y)$ , is defined as:

$$\gamma(x, y) = E$$

### ■ Definition 16.2.18

#### Autocorrelation, Cross-correlation and Partial Autocorrelation

Another important feature of most time series is that observations close together in time tend to be correlated (serially dependent)

two unrelated time series will be correlated if they both contain a trend

Autocorrelation measures the (Pearson) correlation of a time series with a delayed version of itself, and as a function of that delay. Autocorrelation is intended to estimate the degree of similarity of an observation with respect to previous observations.

**Definition 16.2.19** Let  $\{X_t\}$  be a time series with mean  $\mu$  and variance  $\sigma^2$ . The *autocorrelation* function, denoted by  $\rho$ , is defined as:

$$\rho_x(k) = \frac{E[(x_t - \mu)(x_{t+k} - \mu)]}{\sigma^2}$$

The value  $k$  is called *lag*.

The autocorrelation function is not defined for all time series, because the mean may not exist (time series with a trend), or the variance may be zero (constant time series). A time series for which the autocorrelation is defined is called *second order stationary*.

The *sample autocorrelation* is computed in practice by:

$$\hat{\rho}_x(k) = \frac{\frac{1}{n} \sum_{t=1}^{n-k} (x_t - \bar{x})(x_{t+k} - \bar{x})}{\left(\frac{1}{n} \sum_{t=1}^n (x_t - \bar{x})\right)^2}$$

On the contrary of what happened with autocorrelation, sample autocorrelation is defined in case of time series with a trend. However, we must be carefull about the interpretation of the results. In general, sample autocorrelation is applied over the residuals of a time series once the trend and the seasonal components have been removed.

A correlogram is a plot of the sample autocorrelations  $\hat{\rho}(k)$  versus time lags  $k$  (see Figure XXX). The dotted lines are drawn at  $-\frac{1}{n} \pm \frac{2}{\sqrt{n}}$ . If  $\hat{\rho}(k)$  is outside these lines for a value of  $k$  we have evidence against the null hypothesis that  $\rho(k) = 0$  at the 5% level (See Section XXX). It is expected that 5% of the estimates  $\hat{\rho}(k)$  fall outside these lines.

■ **Example 16.8** TODO: Insert Figure. If the example is drawn using `matplotlib.pyplot.acorr`, the above paragraph is not true. Investigate how the shared areas in the correlogram used by `matplotlib` are computed. ■

Crosscorrelation measures ...

**Definition 16.2.20** Let  $\{\mathbf{x}_t\}$  and  $\{\mathbf{y}_t\}$  be time series with means  $\mu_x$  and  $\mu_y$  and variances  $\sigma_x^2$  and  $\sigma_y^2$ . The *crosscorrelation* function, denoted by  $\rho$ , is defined as:

$$\rho_{x,y}(k) = \frac{E[(x_{t+k} - \mu_x)(y_t - \mu_y)]}{\sigma_x \sigma_y}$$

The value  $k$  is called *lag*.

If the crosscorrelation is defined it is said that the combined model is *second order stationary*.

The *sample crosscorrelation* is computed in practice by:

$$\hat{\rho}_{x,y}(k) = \frac{\frac{1}{n} \sum_{t=1}^{n-k} (x_t - \bar{x})(y_{t+k} - \bar{y})}{\frac{1}{n} \sum_{t=1}^n (x_t - \bar{x}) \frac{1}{n} \sum_{t=1}^n (y_t - \bar{y})}$$

In general, sample crosscorrelation is applied over the residuals of both time series once trends and the seasonal components have been removed.

In the same way we draw a correlogram we can plot a crosscorrelogram of the crosscorrelation between time series.

■ **Example 16.9** TODO: Insert Figure. If the example is drawn using `matplotlib.pyplot.acorr`, the above paragraph is not true. Investigate how the shared areas in the correlogram used by `matplotlib` are computed. ■

### Structural Time Series

A univariate structural time series model is one which is formulated in terms of components which, although unobservable, have a direct interpretation. It not only provides the basis for making predictions of future observations, but it also provides a description of the salient features of a time series.

Examples of structural components could be a trend, a seasonal effect, a cycle, an intervention, or the noise.

**Definition 16.2.21** Let  $\mathbf{x}_t$  be a time series. The structural decomposition of  $\mathbf{x}_t$  is given by

$$y_t = \mu_t + \psi_t + \gamma_t + \dots + \varepsilon_t$$

where  $\mu_t, \psi_t, \gamma_t, \dots$  is a finite collection of additive stochastic terms, called structural components, and  $\varepsilon_t$  is a random term composed by independent and identically distributed samples with zero mean.

■ **Example 16.10** dd ■

### State Space Model

**Definition 16.2.22** Let  $y_t$  be a time series. The state space decomposition of  $y_t$  is given by

$$\begin{aligned} y_t &= \mathbf{Z}_t \boldsymbol{\alpha}_t + \varepsilon_t \\ \boldsymbol{\alpha}_{t+1} &= \mathbf{T}_t \boldsymbol{\alpha}_t + \mathbf{R}_t \boldsymbol{\eta}_t \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{\alpha}_t &\text{ explain} \\ \mathbf{Z}_t &\text{ explain} \\ \mathbf{T}_t &\text{ explain} \end{aligned}$$

$$\mathbf{y} = f(\mathbf{X}) + \varepsilon \tag{16.1}$$

measurement equation and transition equation

### Multivariate Time Series

TODO: Introduce this section.

A time series could be multivariate, where a dependant variable  $\{x_{m+1}\}$  is sampled together with a collection of  $m$  independent variables  $\{\mathbf{x}_i\}$ .

**Definition 16.2.23** A *multivariate time series* of length  $n \in \mathbb{N}$  composed by  $m+1 \in \mathbb{N}$  variables, denoted by  $\{\mathbf{x}_t^i : i = 1, \dots, m+1 \text{ and } t = 1, \dots, n\}$  or  $\{\mathbf{x}_t\}$ , is a sequence  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  of *observed vector values*. The time series  $\{\mathbf{x}^{m+1}\}$  is called the *independent variable*, and the time series  $\{\mathbf{x}^i : i = 1, \dots, m\}$  the *dependant variables*.

TODO: cross-correlation and partial cross-correlation.

## 16.3 Minimum Message Length

The *Minimum Message Length* (MML) is based on the idea that a good theory, or explanation, for a dataset is a small collection of premises under which the data is not surprising. The best theories are those which are short, able to explain most of the data, and with a high accuracy. An *explanation message* is composed by two parts: the first part comprises all the premises induced from the data, including numerical values; the second part contains all the data that cannot be derived from the premises. The message also assumes the existence of some already known and accepted premises (prior knowledge). Given the prior premises and the message it should be possible to recover the original dataset. According to the MML, theories are not rejected due to contradictory measurements, they only make the second part of the message longer.

In the MML principle, a message is a lossless encoded version of the original data. The first part of the message contains a probabilistic model about the data, and the second part is the data encoded using this model. We are interested in finding the shortest possible explanation message. If the length of the explanation message is longer than the original data, the theory is considered unacceptable.

Bayes' theorem (see Theorem ??) states that the probability  $P(H | E)$  of a hypothesis  $H$  given an evidence  $E$  is:

$$P(H | E) = \frac{P(E | H)P(H)}{P(E)}$$

We are interested in finding the hypothesis  $H$  with the highest posterior probability  $P(H | E)$  of being true, assuming a fixed evidence  $E$ . That is, we are looking for maximize  $P(E | H)P(H)$  or, equivalently, maximize  $P(H \wedge E)$ .

The *Minimum Message Length* principle (MML for short) is based on the idea that the length of encoding  $H \wedge E$  as a binary string using an optimal

code is equal to  $-\log_2 P(H \wedge E)$  (see Theorem 14.3.2). That is:

$$l(H \wedge E) = -\log_2 P(H \wedge E) = -\log_2 P(E | H)P(H) = -\log_2 P(E | H) - \log_2 P(H)$$

The most probable model  $H$  would be that model that allows us to encode  $H \wedge E$  with the shortest possible string. The encoded string would be composed by two parts, a general assertion about the data, and a detailed description of the data assuming that the assertion is true.

Let  $\mathbb{X}$  be a discrete set composed by all possible datasets,  $\mathcal{X}$  a random variable taking values on  $\mathbb{X}$ , and  $f(X | \theta)$  a probability distribution for  $\mathcal{X}$  given the parameter  $\theta$ .

**Definition 16.3.1** Let  $\Theta$  be a discrete set of possible parameters for  $f$  with probability distribution  $h(\theta)$   $\theta \in \Theta$ , and let  $\hat{\theta} \in \Theta$  be an inferred parameter. An *assertion* is the encoded version of  $\hat{\theta}$  using an optimal code given the probability distribution  $h$ .

The length of the assertion given an optimal binary code is  $-\log_2 h(\hat{\theta})$  (see Section 14.3). Note that  $\theta$  could be a single scalar or a vector of values. Moreover,  $\theta$  could be related to more than one family of probability distributions  $f$ .

**Definition 16.3.2** Let  $X \in \mathbb{X}$  be a dataset, and  $\hat{\theta} \in \Theta$  an inferred value for the distribution  $f$ . A *detail* is the encoded version of  $X$  using an optimal code given the probability distribution  $f(X | \hat{\theta})$ .

The length of the detail given an optimal binary code is  $-\log_2 f(X | \hat{\theta})$ . That is, the length of the detail is the negative of the log-likelihood of  $X$  given  $\hat{\theta}$ .

**Definition 16.3.3** Let  $X \in \mathbb{X}$  be a dataset, and  $\hat{\theta} \in \Theta$  an inferred value for the distribution  $f$ . A *message* for the dataset  $X$  given an inference  $\hat{\theta} \in \Theta$  is the concatenation of the assertion for  $\hat{\theta}$  and the corresponding detail for  $X$  given  $\hat{\theta}$ .

The length of a message given an optimal binary code is  $-\log_2 h(\hat{\theta}) - \log_2 f(X | \hat{\theta})$ . The length of a message allow us, for example, to compare the posterior probabilities of two competing explanations or hypotheses  $\hat{\theta}_1$  and  $\hat{\theta}_2$ .

**Definition 16.3.4** Let  $X \in \mathbb{X}$  be a dataset. The *Minimum Message Length* of  $X$ , denoted by  $MML(X)$ , is given by:

$$MML(X) = \arg \min_{\hat{\theta} \in \Theta} (-\log_2 h(\hat{\theta}) - \log_2 f(X | \hat{\theta}))$$

In practice, the actual messages will not be constructed, since our interest is in the length of the messages, not in their content. It is assumed that the sets  $\mathbb{X}$  and  $\Theta$  and the functions  $f(X | \hat{\theta})$  and  $h(\theta)$  are known a priori, and so, it is not necessary to include them as part of our encoding message.

■ **Example 16.11** Consider an experiment in which we toss a weighted coin 100 times. Denote by 1 if we get a face and 0 a cross, so that each experiment is a binary string of length 100. Our collection of all possible datasets is  $\mathbb{X} = \mathcal{B}^{100}$ ,  $\theta$  is a number in the interval  $[0, 1]$ , the likelihood  $f(X | \theta)$  follows a binomial distribution (that is,  $f(X | \theta) = \theta^n(1 - \theta)^{100-n}$  where  $n$  is the number of faces in  $X$ ), and since we do not know anything about how the coin is weighted, we could assume that  $h(\theta)$  is the uniform distribution in the interval  $[0, 1]$  (that is,  $h(\theta) = 1$  for  $\theta \in \Theta$ ). Under these assumptions, the length of a message for  $X$  given an inferred parameter  $\hat{\theta}$  would be:

$$-\log_2 h(\hat{\theta}) - \log_2 f(X | \hat{\theta}) = -n \log_2 \hat{\theta} - (100 - n) \log_2 (1 - \hat{\theta})$$

We are interested in finding the value of  $\hat{\theta}$  that minimizes the length of the encoded version of  $X$ , that is, the minimum message length for  $X$ . ■

A Maximum A Posteriori analysis of the experiment in Example 16.11 would provide the same inference for  $\hat{\theta}$  than the Minimum Message Length approach. Moreover, given that  $h(\theta)$  follows an uniform distribution, a Maximum Likelihood approach would reach exactly the same value for  $\hat{\theta}$ .

## 16.4 Minimum Description Length

The *Minimum Description Length* (MDL) principle is a reformulation of the Kolmogorov complexity with the goal to make it applicable to solve practical problems. MDL explicitly address the two most important practical limitations of the Kolmogorov complexity: its uncomputability (in general, the Kolmogorov complexity cannot be computed), and the large constants involved in the invariance theorem (that makes it inapplicable to short strings). The approach of MDL to these problems is to scale down Kolmogorov complexity until it does become applicable: instead of using general-purpose computer languages, MDL is based on fixed languages and coding functions.

In contrast to Kolmogorov that states that the complexity of a string is equal to the length of the shortest program that prints that string, MDL proposes that our capacity to learn about a string is equivalent to our ability to compress that string. The idea behind MDL is to describe a dataset with the help of an hypothesis: the more the hypothesis fits the data (and here good fit equals learning), the more we can compress the data given that hypothesis.

In our particular case, we are interested in MDL because it will allow us to compute the nescience of a topic given a dataset, instead of requiring

a text describing the topic. Thus, given a topic  $t$  and a sample dataset  $D = \{x_1, x_2, \dots, x_n\}$ , the nescience of a particular hypothesis  $H$  will be related to the capacity of that hypothesis to compress the dataset.

MDL comes into two versions, *simplified (two-part code) MDL* and *refined MDL*. Simplified MDL is easier to understand, and it will allow us to introduce some important concepts and notation. The extension of the concept of nescience to datasets will be based on the refined version of MDL.

### TODO: Explain how this relates to cross entropy

In this section we are going to introduce the two-part code version of the minimum description length principle for probabilistic models.

*Given a set of candidate models (a set of probability distributions (for example first-order Markov chains) or functions of the same functional form (for example the  $k$ th degree polynomials))  $\mathcal{H}^{(1)}, \mathcal{H}^{(2)}, \dots$  the simplified, two-part version, of the Minimum Description Length Principle [Gr=0000FC05] states that the best point hypothesis (a single probability distribution (e.g. a Markov chain will all parameters values specified)  $H \in \mathcal{H}^{(1)} \cup \mathcal{H}^{(2)} \cup \dots$  to explain the data  $D = (x_1, \dots, x_n) \in \mathcal{X}^n$  is the one that minimizes the sum  $L(M) + L(D | M)$ , where  $L(M)$  is the length (in bits) of the model description, and  $L(D | M)$  is the length (in bits) of the data encoded with the help of the hypothesis ... there is only one reasonable choice for this code ... the so-called Shannon-Fano code ... Each hypothesis  $H$  may be viewed as a probability distribution over  $\mathcal{X}^n$ . For each such distribution there exists a code with length function  $L$  such that for all  $x^n \in \mathcal{X}^n$  we have that  $L(x^n) = -\log_2 P(x^n | H)$ . The quantity  $L(M)$  depends on each model.*

*A description of the data “with the help of” a hypothesis means that the better the hypothesis fits the data, the shorter the description will be. A hypothesis that fits the data well gives us a lot of information about the data. Such information can always be used to compress the data. This is because we only have to code the errors the hypothesis makes on the data rather than the full data.*

*The sum of the two description length will be minimized at a hypothesis that is quite (but not too) “simple”, with a good (but not perfect) fit.*

The length of the data given the model, that is,  $L(D | M)$ .

$$-\log P(D) = L_C(D)$$

This choice for C gives a short codelegth to sequences which have high probability according to (k, ttita) while it gives a high codelength to sequences with low probability. The codelength thus directly reflects the goodness-of-fit of the data with respect to (k,tita) measured in terms of the probability fo D according to (k,tita).

When we say we “code the data D with the help of probabilistic hypothesis P” we mean that we code D using the Shannon-Fano code corresponding to P.

$$L(D | P) := -\log P(D)$$

the code with these lengths is the only one that would be optimal if P where true. (mention we are only interested in code lengths, we are not interested in to find the code itself).

For  $L(M)$  we use the standard code for integers.

■ **Example 16.12** Markov Chain Hypothesis Selection: Suppose we are given data  $DX^n$  where  $X = \{0, 1\}$ . We seek a reasonable model for D that allows us to make good predictions of future data coming from the same source. We decide to model our data using the class B of all Markov chains [...] we face the problem of overfitting: for a given sequence  $D = (x_1, \dots, x_n)$ , there may be a Markov chain P of very high order that fits data D quite well but that will behave very badly when predicting future data from the same source. ■

### 16.4.1 Refined MDL

In refined MDL, we associate a code for encoding D not with a single  $H \in \mathcal{H}$  but with the full model  $\mathcal{H}$  ... we design a single one-part code with lengths  $\bar{L}(D | H)$  (called the stochastic complexity of the data given the model). This code is designed such that whenever there is a member of (parameter in)  $\mathcal{H}$  that fits the data well, in the sense the  $\bar{L}(D | H)$  is small, then the codelength  $\bar{L}(D | H)$  will also be small. Codes with this property are called universal codes.

There are at least four types of universal codes:

- 1 The normalized maximum likelihood (NML) code and its variations.
- 2 The Bayesian mixture code and its variations.
- 3 The prequential plug-in code.
- 4 The two-part code.

Refined MDL is a general theory of inductive inference based on universal codes that are designed to be minimax, or close to minimax optimal. It has mostly been developed for model selection, estimation and prediction.

## 16.5 Multiobjective Optimization

Multiobjective optimization is the area of mathematics that deals with the problem of simultaneously optimizing two or more conflicting functions.

Multiobjective optimization has been applied in many areas of science, including engineering, economics and logistics, where there is no single solution that simultaneously satisfies all objectives, so a decision must be made in the presence of trade-offs between the conflicting goals.

From a formal point of view, we are interested in solving the following *multiobjective optimization* problem:

$$\begin{aligned} \text{minimize} \quad & \{f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_k(\mathbf{x})\} \\ \text{subject to} \quad & \mathbf{x} \in \mathbf{S} \end{aligned}$$

where  $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $i = 1, \dots, k$ , are two or more objective *objective functions*, and the nonempty set  $\mathbf{S} \subset \mathbb{R}^n$  is the *feasible region*, whose elements  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  are *decision vectors*. The image of the feasible region  $f(\mathbf{S}) \subset \mathbb{R}^k$ , denoted by  $\mathbf{Z}$ , is called *objective region*, and its elements  $\mathbf{z} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_k(\mathbf{x}))$  *objective vectors*. In some applications, the feasible region is formed by a collection of inequality constraints  $\mathbf{S} = \{\mathbf{x} \in \mathbb{R}^n \mid g(\mathbf{x}) = (g_1(\mathbf{x}), \dots, g_m(\mathbf{x})) \leq 0\}$ .

In this book we will be dealing with nonlinear multiobjective minimization problems, where at least one of the objective functions, or the constraint functions, is not linear. Objective functions can be also incommensurable, that is, measured in different units or in different scales.

Since the objective functions are conflicting, it does not exist a single solution that is optimal with respect to every objective function (the objective region is partially ordered).

**Definition 16.5.1** A decision vector  $\mathbf{x} \in \mathbf{S}$  *dominates* another decision vector  $\mathbf{y} \in \mathbf{S}$  if  $f_i(\mathbf{y}) \leq f_i(\mathbf{x})$  for all  $i \in \{1, \dots, k\}$  and  $f_j(\mathbf{y}) < f_j(\mathbf{x})$  for at least one  $j \in \{1, \dots, k\}$ . An objective vector  $\mathbf{z} \in \mathbf{Z}$  *dominates* another objective vector  $\mathbf{w} \in \mathbf{Z}$  if  $w_i \leq z_i$  for all  $i \in \{1, \dots, k\}$  and  $w_j < z_j$  for at least one  $j \in \{1, \dots, k\}$ .

Dominance can be studied from the point of view of decision variable space or objective space. An objective vector dominates another objective vector if, and only if, its corresponding decision vector also dominates the other decision vector.

We are interested in those objective vectors for which none of its individual components can be improved without deteriorating at least one of the others.

**Definition 16.5.2** A decision vector  $\mathbf{x} \in \mathbf{S}$  is *Pareto optimal* if there does not exist another decision vector  $\mathbf{y} \in \mathbf{S}$  such that  $\mathbf{y}$  dominates  $\mathbf{x}$ . An objective vector  $\mathbf{z} \in \mathbf{Z}$  is Pareto optimal if there does not exist another objective vector  $\mathbf{w} \in \mathbf{Z}$  such that  $\mathbf{w}$  dominates  $\mathbf{z}$ .

Pareto optimality can be also studied from the point of view of decision variable space or objective space. An objective vector is Pareto optimal if, and only if, its corresponding decision vector is Pareto optimal.

**Definition 16.5.3** The set of Pareto optimal solutions, denoted by  $\mathbf{P}_D$ , is called the *Pareto optimal set*. The set of Pareto optimal solutions in the space of objectives, denoted by  $\mathbf{P}_O$ , is called the *Pareto frontier*.

Sometimes, in practice, it is convenient to use a more restrictive definition of the concept of optimality, in which we identify those vectors for which there does not exist any other vector that improves over all the components simultaneously.

**Definition 16.5.4** A decision vector  $\mathbf{x} \in \mathbf{S}$  is *weakly Pareto optimal* if there does not exist another decision vector  $\mathbf{y} \in \mathbf{S}$  such that  $f_i(\mathbf{y}) < f_i(\mathbf{x})$  for all  $i = 1, \dots, k$ . An objective vector  $\mathbf{z} \in \mathbf{Z}$  is weakly Pareto optimal if there does not exist another objective vector  $\mathbf{w} \in \mathbf{Z}$  such that  $w_i < z_i$  for all  $i = 1, \dots, k$ .

An objective vector is weakly Pareto optimal if its corresponding decision vector is weakly Pareto optimal. Obviously, the Pareto optimal set is a subset of the weakly Pareto optimal set.

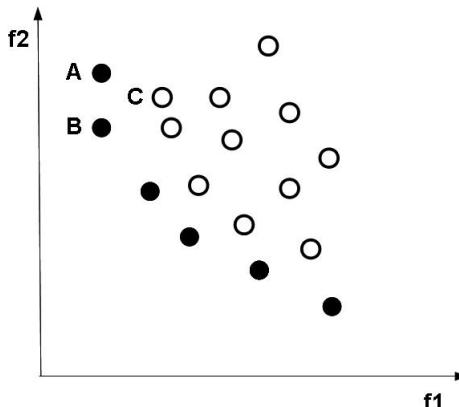


Figure 16.1: Pareto optimality.

■ **Example 16.13** **TODO:** Provide an example based on a multi-variable function. In figure 16.1 we have depicted a sample of the objective region of a multiobjective optimization problem composed by two real-valued objective functions  $f_1$  and  $f_2$  that we are interested in minimizing. White points are not weakly Pareto optimal since there exist points that improve both components at the same time (for example, point **B** improves point **C** in both functions).

Black points are weakly Pareto optimal since there is no point that improves both components at the same time. Point **A** is not Pareto optimal since point **B** improves one component without deteriorating the other. ■

Mathematically speaking all the solutions that compose the Pareto optimal set are equally good. However, for the majority of the practical applications, it is highly desirable to have a single solution. Finding this solution requires additional information not included in the definition of the optimization problem. The relation of preference between objective function values is expressed using a *decision maker*, that it is supposed to have additional insights about the problem to solve. In this sense, solving a multiobjective optimization problem would require to find those feasible decision vectors that are Pareto optimal and that satisfy the additional requirements imposed by the decision maker.

In practice, we assume that the preferences of the decision maker can be expressed using a value function.

**Definition 16.5.5** A *value function* is a function  $U : \mathbb{R}^k \rightarrow \mathbb{R}$  that assigns to each objective vector  $\mathbf{z} = (z_1, \dots, z_k)$  a single real value  $U(\mathbf{z})$ .

### value functions are maximized

Value functions allow us to order the vectors of the objective region  $\mathbf{Z}$ . We are interested in applying the value function to the Pareto optimal subset to find a unique solution to the multiobjective optimization problem.

#### 16.5.1 Range of the Solutions

We are interested in investigating the range of the solutions included in the Pareto optimal set. To do that, we have to find the lower and upper bounds of this set. In the rest of this section, we assume that the objective functions are bounded over the feasible region  $\mathbf{S}$ .

**TODO:** rewrite the concepts of nadir and ideal vector using the supremum and infimum.

An objective vector that minimizes all objective functions is called an ideal objective vector.

**Definition 16.5.6** A vector  $\mathbf{z}^* \in \mathbb{R}^k$  is called *ideal* if each of its components  $z_i$ , minimizes the objective function  $f_i(\mathbf{x})$  subject that  $\mathbf{x} \in \mathbf{S}$ .

If there exists an ideal objective vector that belongs to the feasible region, that is  $\mathbf{z}^* \in \mathbf{Z}$ , then that vector would be a solution of the optimization problem, and that solution would be unique. Ideal vectors are lower bounds to the Pareto set.

The upper bound of the Pareto optimal set is given by the nadir objec-

tive vector. The nadir vector can be estimated using the decision vectors calculated when obtaining the objective ideal vector.

**Definition 16.5.7** Let  $\mathbf{z}^* \in \mathbb{R}^k$  be an ideal objective vector. The set of decision vectors  $\{\mathbf{x}_1^*, \dots, \mathbf{x}_k^*\}$  used to compute  $\mathbf{z}^*$  is called the *payoff table* for  $\mathbf{z}^*$ . That is, if  $\mathbf{z}^* = \{z_1^*, \dots, z_k^*\}$ , we have that  $z_i^* = f_i(\mathbf{x}_i^*)$ .

It turns out that  $f_i(\mathbf{x}_i^*)$  ( $i = 1, \dots, k$ ) is minimal for all for all the elements of the payoff table.

Having the payoff table we can provide an estimation for the nadir vector. For simplicity, let's  $f_{ij}^*$  denote the value of the objective function  $f_i$  computed over the vector  $\mathbf{x}_j^*$ , where  $i, j = 1, \dots, k$ .

**Definition 16.5.8** Let  $\mathbf{z}^* \in \mathbb{R}^k$  be an ideal objective vector, and  $\{\mathbf{x}_1^*, \dots, \mathbf{x}_k^*\}$  its payoff table. The *nadir* objective vector  $\mathbf{z}^{nad} = \{z_1^{nad}, \dots, z_k^{nad}\}$  is given by  $z_i^{nad} = \max_j f_{ij}^*$ .

The ideal objective vector and the nadir objective vector may, or may not, be feasible. In figure 16.2 are depicted the ideal (vector **E**) and nadir (vector **F**) vectors of the multiobjective optimization problem of Example 16.13.

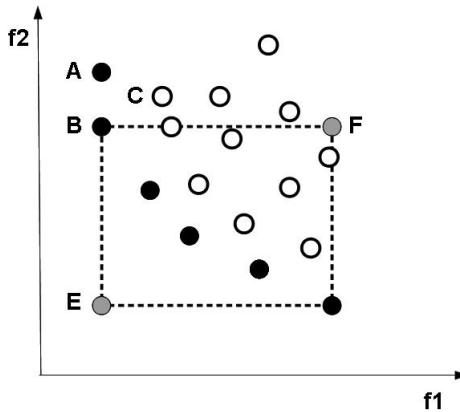


Figure 16.2: Ideal and Nadir vectors.

For some applications, the range of values of the objective functions can differ by orders of magnitude. In those situations, it is advisable to normalize them, so the values are in the same scale. We can use the ideal and nadir vector for this normalization process, by replacing each objective function  $f_i(\mathbf{x})$  ( $i = 1, \dots, k$ ) by the normalized function

$$\frac{f_i(\mathbf{x}) - z_i^*}{z_i^{nad} - z_i^*}$$

### 16.5.2 Trade-offs

Since the functions we want to minimize are conflicting, sometimes we have to assume that the only way to gain a benefit in one aspect of the problem is to lose something in another aspect. How much we have to give up in one objective to improve a certain quantity in the other is called a trade-off.

**Definition 16.5.9** Let  $\mathbf{x}^1, \mathbf{x}^2 \in \mathbf{S}$  be two decision vectors. The *ratio of change* between the functions  $f_i$  and  $f_j$  for the vectors  $\mathbf{x}^1, \mathbf{x}^2$ , denoted by  $\Delta_{ij}$ , is defined as:

$$\Delta_{ij}(\mathbf{x}^1, \mathbf{x}^2) = \frac{f_i(\mathbf{x}^1) - f_i(\mathbf{x}^2)}{f_j(\mathbf{x}^1) - f_j(\mathbf{x}^2)}$$

for all  $i, j = 1, \dots, k$  such that  $f_j(\mathbf{x}^1) - f_j(\mathbf{x}^2) \neq 0$ .

$\Delta_{ij}$  is called a *partial trade-off*, involving  $f_i$  and  $f_j$  between  $\mathbf{x}^1$  and  $\mathbf{x}^2$  if  $f_l(\mathbf{x}^1) = f_l(\mathbf{x}^2)$  for all  $l = 1, \dots, k$ ,  $l \neq i, j$ . If  $f_l(\mathbf{x}^1) \neq f_l(\mathbf{x}^2)$  for at least one  $l = 1, \dots, k$ , and  $l \neq i, j$  then  $\Delta_{ij}$  is called a *total trade-off*.

If the trade-off between two objective functions is very small or very large, that is, if a small change in one aspect of the optimization problem has a significant impact in another aspect, we have a case that is similar to having a weakly Pareto solution that it is not Pareto optimal. In some practical applications is convenient to filter out those solutions that present this undesirable behaviour.

**Definition 16.5.10** A decision vector  $\mathbf{x} \in \mathbf{S}$  is *properly Pareto optimal* if it is Pareto optimal and if there exists a real number  $M > 0$  such that for each  $f_i$  and  $\mathbf{y} \in \mathbf{S}$  satisfying  $f_i(\mathbf{y}) < f_i(\mathbf{x})$ , there exists at least one  $f_j$  such that  $f_j(\mathbf{x}) < f_j(\mathbf{y})$  and

$$\frac{f_i(\mathbf{x}) - f_i(\mathbf{y})}{f_j(\mathbf{y}) - f_j(\mathbf{x})} \leq M$$

An objective vector  $\mathbf{z} \in \mathbf{Z}$  is properly Pareto optimal if the decision vector corresponding to it is properly Pareto optimal.

A solution is properly Pareto optimal if there is at least one pair of objectives functions for which a small decrement in one objective is possible only at the expense of a large increment in the other objective.

Note that the properly Pareto optimal set is a subset of the Pareto optimal set, and the Pareto optimal set is a subset of the weakly Pareto optimal set.

### 16.5.3 Optimization Methods

Generating Pareto optimal solutions plays an important role in multiobjective optimization, and mathematically the problem is considered to be solved when the Pareto optimal set is found [...] However, this is not always enough. We want to obtain only one solution. This means that we must find a way to put the Pareto optimal solutions in a comple order. This is why we need a decision maker an his preference structure.

In general, multiobjective optimization problems are solved by scalarization [...] scalarization means converting the problem into a single or a family of single objective optimization problems with a real-valued objective function

three requirement are set for a scalarization function: 1) It can cover any Pareto optimal solution. 2) Every solution is Pareto optimal.

[...] classify the methods according to the participation of the decision maker in the solution process. The classes are: 1) methods where no articulation of preference information is used (no-preference methods) 2) methods where a posteriori articulation of preference is used (a posteriori methods) 3) methods whre a priori articulation of preference information is used (a priory methods), and 4) methods whre progressive articulation of preference information is used (interactive methods).

In no-preference methods, the knowledge of the decision maker is not taken into account, and the optimization problem is solved using a relatively simple method. In a posteriori methods, the set (or part of it) of Pareto optimal points is identified and then the decision maker select the preferred solution among the alternatives.

### Global Criterion

The global criterion is a non-preference method in which the distance between some reference point and the feasible objective region is minimized. In this method, all the objective functions are considered to be equally important. As reference point it is usually used the ideal vector, and as metric it is common to use a  $L_p$ -metric. Under these assumptions, the goblal criterion method becomes the following minimization problem:

$$\begin{aligned} \text{minimize} \quad & \left( \sum_{i=1}^k (f_i(\mathbf{x}) - z_i^*)^p \right)^{\frac{1}{p}} \\ \text{subject to} \quad & \mathbf{x} \in \mathbf{S} \end{aligned}$$

Different values for  $p$  result in different solutions to the minimization problem. Common values for  $p$  are 1, 2 or  $\infty$ .

**Proposition 16.5.1** The solution of the  $L_p$ -based global criterion is Pareto optimal.

*Proof.* Let  $\mathbf{x}$  be a solution of the  $L_p$ -based global criterion problem, with  $1 \leq p < \infty$ , and assume that  $\mathbf{x}$  is not Pareto optimal. Then, according to Definition 16.5.2 there must exist a point  $\mathbf{y} \in \mathbf{S}$  such that  $f_i(\mathbf{y}) \leq f_i(\mathbf{x})$  for all  $i = 1, \dots, k$ , and  $f_j(\mathbf{y}) < f_j(\mathbf{x})$  for at least one  $j$ . Then we have that  $(f_i(\mathbf{y}) - z_i^*)^p \leq (f_i(\mathbf{x}) - z_i^*)^p$  for all  $i \neq j$  and  $(f_j(\mathbf{y}) - z_j^*)^p < (f_j(\mathbf{x}) - z_j^*)^p$ . Adding all these terms and raising to the  $1/p$  power, we obtain

$$\left( \sum_{i=1}^k (f_i(\mathbf{y}) - z_i^*)^p \right)^{\frac{1}{p}} < \left( \sum_{i=1}^k (f_i(\mathbf{x}) - z_i^*)^p \right)^{\frac{1}{p}}$$

which is a contradiction with the fact that  $\mathbf{x}$  is a solution to the minimization problem. ■

Although all the solutions selected by the  $L_p$ -based global criterion are Pareto optimal, as we have proved in previous proposition, there are solutions of the optimal Pareto set that will never be selected by this method. In practice it is convenient to normalize the range of the objective values, so that those points closer to the ideal vector do not receive more importance. A common normalization term used in practice is  $z_i^{nad} - z_i^*$ .

### Weighting Method

The weighting method is a simple way to generate different Pareto optimal solutions.

In the weighting method [...] the idea is to associate each objective function with a weighting coefficient and minimize the weighted sum of the objectives. In this way, the multiple objective functions are transformed into a single objective function. We suppose that the weighting coefficients  $w_i$  are real numbers such that  $w_i \geq 0$  for all  $i = 1, \dots, k$ . It is also usually supposed that the weights are normalized, that is  $\sum_{i=1}^k w_i = 1$ .

The multiobjective optimization problem is modified into the following problem, to be called a weighting problem:

$$\text{minimize } \sum_{i=1}^k w_i f_i(\mathbf{x})$$

subject to  $\mathbf{x} \in \mathbf{S}$

where  $w_i \geq 0$  for all  $i = 1, \dots, k$  and  $\sum_{i=1}^k w_i = 1$ .

[...] weighting coefficient zero makes no sense. It means that we have included in the problem some objective function that has no significance at all.

The objective functions should be normalized or scaled so that their objective values are approximately of the same magnitude [...] Only in

this way can one control and manoeuvre the method to produce solutions of a desirable nature in proportion to the ranges of the objective functions. Otherwise the role of the weighting coefficients may be greatly misleading.

**Proposition 16.5.2** The solution of weighting problem is Pareto optimal if the weighting coefficients are positive, that is  $w_i > 0$  for all  $i = 1, \dots, k$ .

*Proof.* TODO ■

the solution of the weighting method is always Pareto optimal if the weighting coefficients are all positive or if the solution is unique [...] The weakness of the weighting method is that not all of the Pareto optimal solutions can be found.

the weighting method is used to generate Pareto optimal solutions

in practical calculations the condition  $w_i \geq \varepsilon$ , where  $\varepsilon > 0$ , must be used instead of the condition  $w_i > 0$  for all  $i = 1, \dots, k$ . This necessitates a correct choice as to the value of  $\varepsilon$ .

The weighting method can be used so that the decision maker specifies a weighting vector representing his preference information [...] In this case, the weighting problem can be considered (a negative of) a value function (remember that value functions are maximized).

If the weighting method is used as an a priori method one can ask what the weighting coefficients in fact represent. Often, they are said to reflect the relative importance of the objective functions. However, it is not at all clear what underlies this notion [...] instead of relative importance, the weighting coefficients should represent the rate at which the decision maker is willing to trade off values of the objective functions

if some of the objective functions correlate with each other, then seemingly "good" weighting vectors may produce poor results and seemingly "bad" weighting vectors may produce useful results

Weighting coefficients are not easy to interpret and understand for average decision makers.

Employing the weighting method as an a priori method presumes that the decision maker's underlying value function is or can be approximated by a linear function [...] it must be noted that altering the weighting vectors linearly does not have to mean that the values of the objective functions also change linearly. It is difficult to control the direction of the solutions by the weighting coefficients

Shall I explain the  $\varepsilon$ -Constraint method?

## References

**TODO:** Add paper of Turing about AI.

The minimum message length principle was developed by Chris Wallace, published for first time in 1968 in [WB68]. The book [Wal05], written by the same author, contains a detailed description of the principle.

A good introduction to the discipline of non-linear multiobjective optimization can be found in [Mie12]; Section 16.5 is largely based on this book.



## 17. Philosophy of Science

*To go where you don't know,  
you have to go the way you don't know.*

San Juan de la Cruz

The *philosophy of science* is the branch of philosophy that examines the foundations, methods, and implications of scientific inquiry. It explores how scientific knowledge is generated, the validity of scientific theories, the nature of scientific reasoning, and the role of values in science. By addressing fundamental questions about objectivity, reality, and the limits of scientific explanation, the philosophy of science helps us understand how science works and its impact on our understanding of the world.

The philosophy of science provides a theoretical framework for analyzing the core elements that constitute our theory of nescience. It enables us to critically examine the foundations and assumptions of our theory, guiding us toward the identification of the essential questions that we must address. Moreover, philosophical inquiry compels us to rigorously push the boundaries of our analysis, ensuring that our theory is explored to its ultimate consequences, both logically and conceptually. This approach strengthens the robustness of the theory, even if these results do not directly translate into practical applications.

A central theme in scientific methodology is the distinction between *discovery* and *justification*. Discovery refers to how new ideas or hypotheses emerge, often through creativity, intuition, or serendipity, while justification involves the use of evidence and logical reasoning to evaluate and validate these ideas. Philosophers of science have historically emphasized justification, focusing on methods to rigorously test and analyze scientific claims. However, both discovery and justification are critical to understanding scientific progress, and this chapter will address both aspects.

In this chapter, we provide a concise overview of key elements from the philosophy of science, as well as relevant concepts from other branches of philosophy, such as metaphysics, epistemology, and ontology, that are important to the theory of nescience. Certain other topics within the philosophy of science, such as the problem of objectivity (examining whether science can be truly objective or is influenced by social and personal values) or the role of values in science, including the relationship between ethical, social, and political values and scientific practices, are not included in this review, as they are not directly relevant to a mathematical theory of nescience.

The chapter begins with a brief introduction to the field and its importance in understanding scientific inquiry. We delve into the problem of which entities can be known, examining the scope of scientific knowledge and the nature of observable and unobservable phenomena. The chapter also addresses the concept of scientific representation, discussing how models, theories, and laws reflect aspects of reality. We examine how science discovers new knowledge, the principles behind the scientific method, and the various ways in which scientists formulate and test hypotheses. Finally, we explore the limits of science, considering the boundaries of what science can explain and where its explanatory power may fall short.

## 17.1 What is Science

*Science* is a systematic method of investigating the world around us, aimed at generating reliable knowledge through observation and reasoning. Unlike other forms of inquiry, science is rooted in the idea that knowledge must be based on evidence that can be tested and verified. By combining theoretical thinking with empirical data, science offers a powerful way to explain, predict, and understand natural phenomena. Sometimes science seeks explanations for practical purposes, but other times it is sought simply to satisfy our intellectual curiosity.

The *scientific method* is understood as the systematic process by which science acquires new knowledge. In schools, the scientific method is often taught as a series of steps: first observing and describing a phenomenon, then

coming up with a hypothesis to explain it, testing that hypothesis through experiments, analyzing the results, and finally making a conclusion. However, the idea of a single, universal scientific method that can be applied to all sciences isn't widely accepted anymore. Instead, scientists and philosophers recognize that different scientific fields require different methods because each field faces its own challenges and complexities.

The following is a list of the key features that make science distinct and valuable as a way of knowing [as opposed to other kinds of knowledge].

- *Empiricism:* (Do no use the word "empiricism" since this is a philosophical school and might confuse the reader.) At its core, science is an empirical endeavor, meaning it relies on observation, experimentation, and measurable evidence to understand the natural world. Scientific knowledge is grounded in data that can be gathered through direct or indirect observation, ensuring that claims can be tested and verified by others. [Science is derived from facts]
- *Testability:* A key characteristic of science is its focus on developing testable hypotheses, that is, statements or predictions that can be empirically investigated. This means that scientific claims must be falsifiable, open to potential disproof if the evidence does not support them. This distinguishes science from fields that rely on unfalsifiable or speculative claims.
- *Theoretical Frameworks:* Science is not merely about collecting facts; it seeks to develop broader explanations through models, laws, and theories. Models offer simplified representations of complex systems, while theories are well-supported frameworks that explain the underlying mechanisms of phenomena. These frameworks help interpret data and guide further investigation, offering coherence to the body of scientific knowledge.
- *Self-Correction:* A defining feature of science is its self-correcting nature. Scientific theories are not static; they evolve as new evidence comes to light. When new data contradicts a theory, science adapts, modifies, refines, or sometimes discards the theory in favor of one that better fits the evidence. This continuous process of revision ensures that scientific knowledge becomes more accurate over time.
- *Generalizability:* Science strives to uncover universal principles that apply across various contexts, not just to isolated cases. While science begins with specific observations, its goal is to identify general laws and patterns that explain a broad range of phenomena. This pursuit of generalizable knowledge allows science to predict future occurrences and provide deeper insights into the workings of the natural world.

The main task of the philosophy of science is to understand how tech-

niques such as experimentation, observation, and theory building enable scientists to discover the secrets of nature. The philosophy of science draws on other areas of philosophy, such as ontology, epistemology, metaphysics, and logic. *Ontology*, the branch of philosophy concerned with the nature of existence, addresses the question of which entities exist in the real world. It examines the types of entities that science can study, whether they are observable, abstract, or indeterminate, and engages with the philosophical debates surrounding their existence. Ontology provides the foundation for exploring the boundaries of scientific knowledge and the limits of inquiry. *Epistemology* is the branch of philosophy concerned with knowledge itself—how we acquire it, what justifies it, and what its limits are. While ontology deals with what exists, epistemology addresses how we come to know what exists. In the context of science, epistemology examines the methods, evidence, and reasoning that underpin scientific investigation, exploring how scientific knowledge is built, how reliable it is, and what counts as a justified belief in scientific practice. Both ontology and epistemology are part of the broader field of *metaphysics*, which deals with the fundamental nature of reality and existence. Metaphysics explores the most basic concepts and categories of being, such as time, space, causality, and possibility, as well as the relationship between mind and matter. *Logic*, the study of valid reasoning, underpins the entire scientific enterprise by providing the tools to analyze arguments, identify fallacies, and structure sound reasoning. In science, logic is used to ensure the coherence of theoretical frameworks, the validity of inferences drawn from data, and the consistency of explanations and predictions. Formal logic, including propositional and predicate logic, provides a systematic way to evaluate arguments and detect errors, while informal logic addresses reasoning in everyday scientific discourse. Together, ontology, epistemology, metaphysics, and logic provide a comprehensive philosophical foundation for understanding what science studies, how it builds knowledge, and the fundamental nature of the reality science seeks to explain.

This chapter on philosophy of science is structured around key components that illustrate how scientific knowledge is developed and justified (see Figure 17.1). It begins with a discussion on *Entities*, which can be either concrete or abstract, representing the objects of scientific study. From these entities, knowledge is acquired through *Observation*, which involves gathering empirical data and facts. These observations are then transformed into *Representations*, such as recorded data, or facts, that scientists use to analyze and interpret phenomena. Through the process of *Discovery*, these representations lead to the formulation of *Explanations*, which take the form of scientific *theories and laws* that describe underlying principles governing the natural world. The chapter also explores how scientific explanations

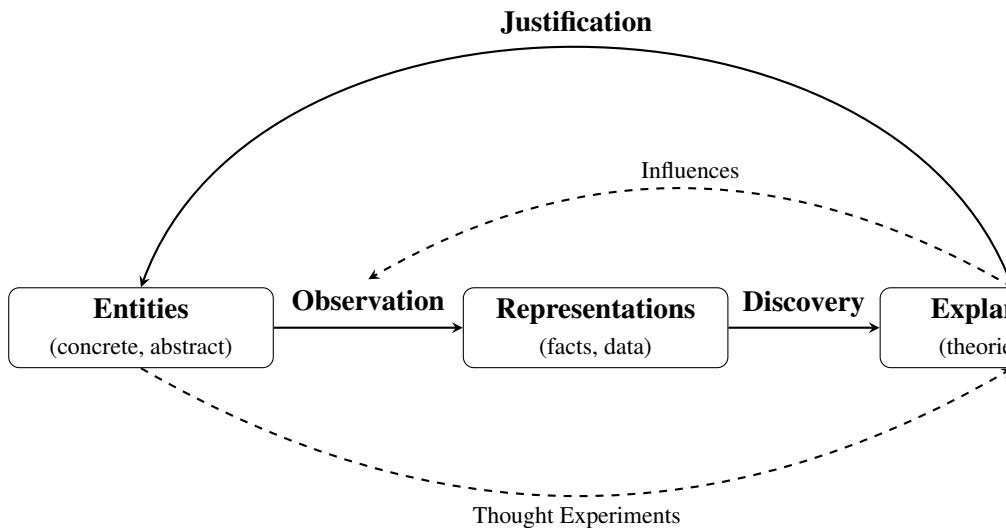


Figure 17.1: Chapter Organization

influence future observations, shaping what is investigated and how data is interpreted. Additionally, thought Experiments provide an alternative way of reasoning about scientific problems beyond direct empirical observation. Finally, the chapter addresses *Justification*, emphasizing the criteria by which scientific claims are validated, ensuring that discoveries and explanations are robust, reliable, and well-supported by evidence. This structure provides a comprehensive view of the scientific process, from the identification of entities to the formation and justification of scientific theories.

## 17.2 What is an Entity

Perhaps cover in this section. [...] science is derived from facts [...] facts are presumed to be claims about the world that can be directly established by a careful unprejudiced observation [...] via senses, which may not be reliable [...] how facts relate to theories, or require previous knowledge [...] (refer to epistemology?, the application of epistemology to science) Perhaps I should cover here two competing schools: empiricism and logical positivism.

In this section, we focus on the fundamental problem of determining which kinds of entities can be known or investigated by science.

A *knowable entity* is an object, phenomenon, or concept that can be investigated, understood, or described through scientific or intellectual inquiry. Knowable entities are those that, either directly or indirectly, can be observed,

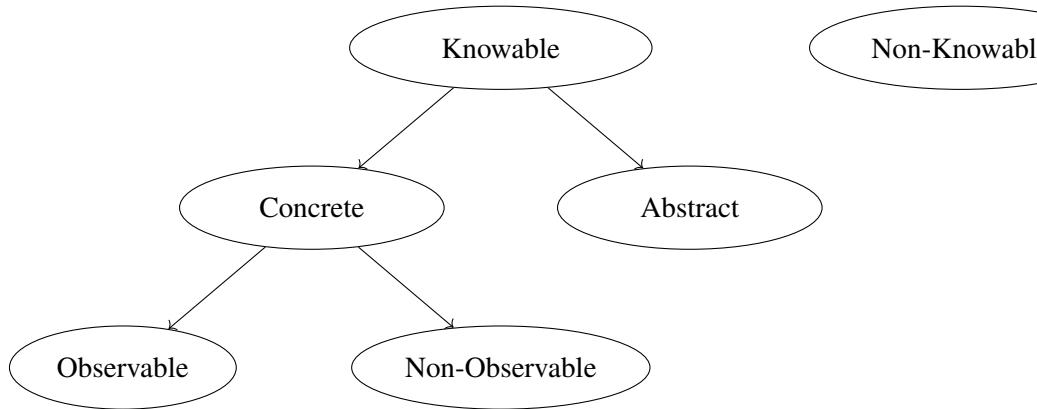


Figure 17.2: Classification of Research Entities

measured, inferred, or modeled, using available tools, methods, or theories. They are within the scope of human knowledge, and their properties can be analyzed or explained. Examples of knowable entities include the stars and planets, animals, or computer algorithms. A *non-knowable entity* refers to something that cannot be directly observed, measured, or understood using current scientific or intellectual methods. This could be due to limitations in technology, the abstract or metaphysical nature of the entity, or inherent epistemological boundaries. Non-knowable entities might remain beyond the reach of human understanding either temporarily (until methods evolve) or permanently (due to their nature). Examples of non-knowable entities include the nature of consciousness, the origin of the universe, or the existence of a deity. The distinction between knowable and non-knowable entities is not always fixed, but varies depending on the clarity and precision of the area of interest and the evolving nature of scientific understanding in each field.

Scientific research encompasses both *concrete entities* and *abstract entities*. Concrete entities refer to physical objects or phenomena that can be directly observed, measured, or interacted with, such as stars, cells, or chemical compounds. These entities form the basis of empirical research, where data is collected through direct observation or experimentation. In contrast, abstract entities are conceptual and do not have a physical presence, such as numbers, algorithms, or theoretical models. Abstract entities play a crucial role in scientific research, particularly in fields like mathematics and theoretical physics, where they provide the framework for understanding and modeling concrete phenomena. While abstract entities cannot be observed directly, they can be known through indirect methods. The inclusion of these abstract entities in scientific inquiry raises important philosophical questions

about their ontological status: Are these abstract entities real in the same way that physical objects are, or are they simply conceptual tools? This question remains an open and debated issue in the philosophy of science.

Finally, in scientific inquiry, there is also a distinction between *observable entities*, which can be directly perceived or measured, and *non-observable entities*, which cannot be directly observed but can still be detected or inferred from empirical evidence. Observable entities include things like trees, planets, and bacteria, objects that can be seen or measured using scientific instruments. Non-observable, but still detectable, entities include things like subatomic particles and gravitational forces. These entities are often crucial for explaining observable phenomena and are identified through the use of scientific instruments and theoretical frameworks. For example, we cannot observe a quark in the same way we observe a tree, but through scientific theory, experimentation, and the detection of indirect evidence, we infer its existence.

A central debate within the scope of science is the tension between *reductionism* and *holism*. Reductionism is the view that complex systems can be fully understood by breaking them down into their simplest, most fundamental parts. For example, a reductionist might argue that biological processes can be explained entirely by chemistry, and chemistry by physics. This approach assumes that understanding the smallest components of a system will provide a complete explanation of the whole. In contrast, holism argues that some phenomena cannot be fully understood by reducing them to their components. Instead, the whole system exhibits properties that cannot be predicted or explained by analyzing its parts in isolation. For example, in ecology, the interactions within an ecosystem can produce emergent properties that are not reducible to the behavior of individual species.

Finally, the scope of science is shaped by the debate between *scientific realism* and *anti-realism*, which addresses the question of whether the entities posited by scientific theories are real or merely useful constructs. The contrast between realism and anti-realism is most marked for sciences which make claims about non-observable entities. Scientific realism holds that the entities described by scientific theories exist independently of our knowledge of them. According to this view, successful scientific theories reveal truths about the world. In contrast, anti-realism argues that scientific theories are useful tools for predicting and organizing observations, but we should not necessarily believe that non-observable entities like electrons or gravitational waves are real. For anti-realists, the purpose of science is not to describe an independent reality, but rather to provide models that help us navigate and predict phenomena.

Scientists often seek to classify the objects they study into general kinds

or categories. A key goal of classification is to convey meaningful information, enabling us to better understand and navigate the complexities of the natural world. However, this process raises several intriguing philosophical questions. Do the categories used in science represent real, essential divisions in nature, or are they merely human-made constructs designed to impose order on a complex and often ambiguous reality (realism vs. anti-realism)? Since any set of objects can, in principle, be classified in various ways, how should we determine the most appropriate approach? Is there a 'correct' way to classify, or are all classification systems fundamentally arbitrary? Some philosophers argue that *natural kinds*, groups that correspond to divisions genuinely existing in the world, do exist, as opposed to merely reflecting human interests. Understanding whether and how such natural kinds exist can provide valuable insights into the structure of reality and inform scientific inquiry.

### 17.3 Observation

\* Scientific knowledge has a special status in part because it is founded on a secure basis, solid facts firmly established by observation.

\* Facts are claims about the world.

\* Facts are directly given to careful, unprejudiced observers via the senses. \* Facts are prior to and independent of theory \* Facts constitute a firm and reliable foundation for scientific knowledge.

\* There are reasons for doubting that facts acquired by observation and experiment are as straightforward and secure as has traditionally been assumed.

[...] what observers see, the subjective experience that they undergo, when viewing an object or scene is not determined solely by the images on their retinas, but depends also on the experience, knowledge and expectations of the observers [...] one has to learn to be a competent observer in science [...] The experienced and skilled observer does not have perceptual experiences identical to those of the untrained novice when the two confront the same situation. This clashes with a literal understanding of the claim that perception are given in a straightforward way via the senses. [...] observers viewing the same scene from the same place see the same thing but interpret what they see differently.

\* Before an observer can formulate and assent to an observation statement, he must be in possession of the appropriate conceptual framework and a knowledge of how to appropriately apply it. Facts, formulated as statements, presuppose quite a lot of knowledge.

Our search for relevant facts needs to be guided by our current state

of knowledge [...] the formulation of observation statements presupposes significant knowledge, and that the search for relevant observable facts in science is guided by that knowledge. [...] the fact that knowledge is necessary for the formulation of significant observation statements still leaves open the question of which of the statements so formulated are borne out by observation and which are not. The idea that knowledge should be based on facts that are confirmed by observation is not undermined by the recognition that the formulation of the statements describing those facts are knowledge-dependent.

The point is that if the knowledge that provides the categories we use to describe our observations is defective, the observation statements that presuppose those categories are similarly defective. [...] observation statements depend on the knowledge that forms the background against which the judgment is made [...] scientific revolution involved not just a progressive transformation of scientific theory, but also a transformation in what were considered to be the observable facts.

- \* Facts are fallible and subject to correction.
- \* Perceptions are influenced by the background and expectations of the observer, so that what appears to be an observable fact for one need not be for another.
- \* Judgments about the truth of observation statements depend on what is already known or assumed, thus rendering the observable facts as fallible as the presuppositions underlying them.
- \* Everyday observation is far from passive. There is a range of things that are done, many of them automatically and perhaps unconsciously, to establish the validity of a perception.
- \* Statements should be such that their validity can be tested in ways that involve routine, objective procedures that do not necessitate fine subjective judgement on the part of the observer.
- \* Observations suitable for constituting a basis for scientific knowledge are both objective and fallible. They are objective insofar as they can be publicly tested and straightforward procedures, and they are fallible insofar as they may be undermined by new kinds of test made possible by advances in science and technology.
- \* What is needed in science is not just facts but relevant facts. Which facts are relevant and which are not relevant to a science will be relative to the current state of development of that science.

Many kinds of processes are at work in the world around us, and they are all superimposed on, and interact with, each other in complicated ways. [...] It is not possible to arrive at an understanding of these various processes by careful observation of events as they typically occur. [...] To

acquire factors relevant for the identification and specification of the various processes at work in nature it is, in general, necessary to practically intervene to try to isolate the process under investigation and eliminate the effects of others. In short, it is necessary to do experiments. [...] If these are facts that constitute the basis for science, then those facts come in the form of experimental results rather than any old observable facts.

Experimental results are by no means straightforwardly given [...] the complexity of practical struggle involved in the production of an experimental result. [...] If experimental results constitute the facts on which science is based, then they are not straightforwardly given via the senses. They have to be worked for, and their establishment involves considerable know-how and practical trial and error as well as exploitation of the available technology. [...] Nor are judgments about the adequacy of experimental results straightforward. [...] Experimental results can be faulty if the knowledge informing them is deficient or faulty. [...] Experimental results are fallible, and can be updated or replaced for reasonably straightforward reasons. Experimental results can become outmoded because of advances in technology, they can be rejected because of some advance in understanding (in the light of which an experimental set-up comes to be seen as inadequate) and they can be ignored as irrelevant in the light of some shift in theoretical understanding.

\* Experimental results are required not only to be adequate, but also to be appropriate or significant. What is an adequate and significant experiment depends heavily on how the practical and theoretical situation is understood.

\* Experimental results are theory-dependent: All experiments will presume the truth of some theories to help judge that the set-up is adequate and the instruments are reading what they are meant to read.

\* Experimental results are fallible and revisable.

Two schools of thought that involve attempts to formalise [...] that scientific knowledge is derived from the fact, are empiricists and the positivism. Empiricists [...] held that all the knowledge should be derived from ideas implanted in the mind by way of sense perception. The positivist had a broader view of what fact amounts to. [...] The logical positivist [...] attempted to formalise it, paying close attention to the logical form of the relationship between scientific knowledge and the facts.

## 17.4 Scientific Representation

Science helps us understand the natural world by using different kinds of representations of research entities. These representations include measurements from scientific instruments, descriptions of observations, digital images like X-rays or MRI scans, and more. Scientific practice also often considers

mathematical equations, models, and theoretical constructs as valid forms of representation. The challenge of *scientific representation* is to identify the conditions that make a representation scientific and determine what makes an effective representation. The main issues discussed in this area of philosophy include:

*Scientific Representation Problem:* The scientific representation problem is about figuring out the necessary and sufficient conditions that make a representation valid in science. It explores whether these conditions are the same across all scientific fields or if they vary depending on the discipline or research context. For example, what qualifies as a valid representation in physics, which often uses mathematical models, might be different from what is used in biology, where visual and descriptive representations are common. This problem also questions whether scientific representations need to be adapted to specific research goals to be considered valid.

*Representational Demarcation Problem:* The representational demarcation problem looks at whether scientific representations are fundamentally different from other kinds of representations, like those found in art or everyday life. It examines what makes scientific representations unique, focusing on their purpose, accuracy, and the methods used to create them. Unlike artistic representations, which may emphasize subjective interpretation or aesthetic value, scientific representations are generally held to standards of precision, reliability, and empirical adequacy. Understanding these differences helps clarify the specific role that scientific representations play in knowledge production.

*Problem of Style:* The problem of style addresses the fact that the same entity can be represented in different ways, depending on the goals and methods of the research. Different styles of representation—such as diagrams, mathematical equations, physical models, or computer simulations—each have unique characteristics, including the intended audience, level of abstraction, and type of information conveyed. This issue also asks whether these styles are fixed or if new styles can be invented to meet emerging scientific needs. The flexibility of representation styles is crucial because it allows for new insights and different ways of understanding scientific phenomena.

*Standard of Accuracy:* The standard of accuracy problem is about determining what makes a scientific representation accurate. It involves figuring out how to distinguish between accurate and inaccurate representations by considering factors like how well the representation matches empirical data, captures important features of the phenomenon, and its ability to make predictions. This issue also explores whether accuracy should be seen as an objective standard or if it depends on the specific aims and context of the research. For example, a simplified model might still be considered accurate

if it effectively serves its purpose, such as making predictions or providing explanations.

*Problem of Ontology:* The problem of ontology in scientific representation deals with the nature of the entities that can serve as representations. It asks whether representations need to be concrete, like physical models or graphs, or if they can also be abstract, like mathematical equations or theoretical constructs. This issue also questions whether representations must be realistic or if more abstract, idealized forms can still be effective in scientific inquiry. Understanding these ontological aspects helps define the types of entities that are allowed in scientific discourse and how they relate to the real-world phenomena they represent.

There are also five conditions of adequacy that a scientific representation should satisfy to be considered effective and reliable:

*Requirement of Directionality:* The requirement of directionality examines the relationship between representations and the real world. Representations are meant to describe entities in the real world, but this condition raises the question of how, if at all, real-world entities might describe their representations. It challenges us to think about the direction of influence between the representation and the entity it aims to depict.

*Surrogate Reasoning:* Surrogate reasoning addresses how scientific representations allow researchers to generate hypotheses about the entities they represent. This condition explores how using a representation as a surrogate can lead to new insights or predictions about the target, effectively using the representation to stand in for the real-world entity during reasoning and analysis.

*Applicability of Mathematics:* The applicability of mathematics condition is concerned with how mathematical models can be used to represent the real world. It questions how abstract mathematical constructs can effectively describe complex physical systems and whether the success of mathematical representation depends on any special features of the target phenomena. This condition highlights the central role of mathematics in developing and understanding scientific theories.

*Possibility of Misrepresentation:* The possibility of misrepresentation addresses whether representations that are not fully accurate can still be considered valid scientific representations. It considers situations where simplifications or approximations are necessary and whether these less-than-perfect representations can still contribute valuable understanding of a phenomenon. This condition is important for understanding how idealizations and abstractions function in scientific practice.

*Targetless Models:* The targetless models condition explores whether we can allow representations that do not have a direct real-world counterpart.

It questions if a model that does not represent any existing entity can still be useful in scientific inquiry, perhaps as a way to explore theoretical possibilities or to understand potential scenarios. This condition emphasizes the creative and exploratory aspects of scientific modeling.

There have been multiple proposals to formally define the concept of scientific representation. Unfortunately, none of these proposals can provide a convincing answer to the questions and conditions of adequacy described above. In the rest of this section, we describe some of these proposals, identifying their advantages and drawbacks. To compare these proposals, we will present them as: "A scientific model  $M$  represents a target system  $T$  if, and only if ...".

*Stipulative Fiat:* The stipulative fiat proposal states that "a scientific model  $M$  represents a target system  $T$  if, and only if, a scientist stipulates that  $M$  represents  $T$ ." The main problem with this interpretation is that, since anything can be a representation if a scientist says so, it is difficult to guarantee the surrogate reasoning condition. If any model can be deemed a representation by simple stipulation, it becomes challenging to determine which representations are genuinely useful for making scientific inferences. Proponents of this theory acknowledge that while all representations may be stipulated, some are undeniably more useful than others.

*Similarity Conception:* The similarity conception proposes that "a scientific model  $M$  represents  $T$  if, and only if,  $M$  and  $T$  are similar." This conception addresses the surrogate reasoning condition since similarity between the model and the target allows us to derive similar properties. However, it introduces new challenges, particularly regarding the problem of style. The concept of similarity is often vague: in what sense are  $M$  and  $T$  similar? This vagueness can lead to issues with directionality and accuracy, as different aspects of similarity may not always align with what is relevant for scientific representation.

*Structuralist Conception:* The structuralist conception is based on the idea of isomorphism. According to this view, a scientific model  $M$  represents a target system  $T$  if the structure of  $M$  is isomorphic to the structure of  $T$ . In other words, there is a one-to-one correspondence between the elements and relationships in both  $M$  and  $T$ . This approach justifies surrogate reasoning because having the same structure implies that properties and relations in the model correspond to those in the target. Furthermore, since mathematics is fundamentally concerned with the study of structures, this conception also supports the applicability of mathematics in representing natural systems.

*Inferential Conception:* The inferential conception proposes that a model  $M$  is an epistemic representation of a target  $T$  if, and only if, the user adopts an interpretation of  $M$  in terms of  $T$ . This view emphasizes the role of

the user in giving meaning to the model, suggesting that representation is not an inherent property of the model itself but arises through its use in making inferences about the target system. This conception underscores the importance of context and interpretation in determining whether a model effectively represents its target.

*Fiction View of Models:* According to the fiction view of models,  $M$  represents  $T$  if and only if  $M$  functions as a prop in a game of make-believe that prescribes imagining certain things about  $T$ . This view draws an analogy between scientific modeling and storytelling, where models are treated as fictional constructs that facilitate imaginative engagement with the target system. Although this approach highlights the creative aspects of modeling, it raises questions about how such fictional constructs can be rigorously linked to real-world entities.

*Representation-As:* The representation-as approach suggests that a scientific model represents a target system as something, emphasizing that representation involves highlighting certain features of the target while downplaying others. This conception focuses on the interpretive aspect of modeling, where the modeler selects specific attributes of the target to represent, depending on the research goals. This approach allows for a flexible understanding of representation that can accommodate different styles and purposes, but it also implies that the usefulness of a representation is contingent on how well the modeler captures the relevant aspects of the target.

Each proposal has its strengths and weaknesses, highlighting the complexity of what it means for a model to effectively represent a target in scientific inquiry. Understanding these various perspectives is crucial for advancing our comprehension of the role of representation in science.

## 17.5 Scientific Discovery

Scientific discovery refers to the process through which new knowledge, ideas, or principles are uncovered within science. Unlike the systematic procedures associated with justification, discovery often involves creativity, intuition, and inspiration. While some discoveries arise from planned experiments or systematic observation, others occur unexpectedly, challenging existing paradigms or opening new fields of inquiry. The general agreement among philosophers is that the creative process of conceiving a new idea is a non-rational process that cannot be formalized as a set of steps. Understanding discovery is crucial for appreciating how science evolves and adapts, as it reveals the dynamic and often unpredictable nature of scientific progress.

The following two proposals assume that a domain-neutral logic of discovery can be formalized, offering attempts to develop such a framework.

- *Discovery as abduction:* Abductive reasoning is a mode of discovery that begins with surprising or anomalous phenomena and seeks to generate plausible hypotheses to explain them. This process is conceptualized as follows: (i) some unexpected data, such as  $p_1, p_2, \dots, p_n$ , is encountered; (ii) these data would be less surprising if a hypothesis of type  $H$  were true; and (iii) therefore, there is justification to develop a hypothesis of type  $H$ . Two types of abduction are distinguished: *selective abduction*, which involves choosing from known hypotheses, and *creative abduction*, which generates entirely new hypotheses. Abduction present some limitations. First, multiple hypotheses may explain the same phenomena, making additional criteria necessary to evaluate their merit. Second, the schema of abductive reasoning does not account for the act of conceiving a hypothesis itself.
- *Heuristic programming* is a computational approach designed to simulate and assist the creative aspects of human problem-solving. These programs operate as searches within a defined problem space, which includes all possible configurations for a given domain. Each configuration represents a specific state within the problem space, with two key states being the initial state, the starting point of the search, and the goal state, which represents the desired outcome. Operators define the moves that transition between states, while path constraints limit permissible moves within the problem space. Problem-solving in this context involves finding a sequence of operations that connects the initial state to the goal state. The core aim of this approach is to develop heuristics (practical rules or strategies) to efficiently navigate and solve complex problems. Heuristic programming has its limitations, scientific problem spaces are often ill-defined, and computer programs rely on experimental data, meaning that simulations frequently cover only specific aspects of scientific discovery.

Many philosophers argue that discovery is an important topic within the philosophy of science, even as they move away from the idea of a formal logic of discovery. A highly influential perspective is Thomas Kuhn's examination of how new facts and theories emerge in the so-called *paradigm shifts*. According to Kuhn, discovery is not a single event but rather a complex and prolonged process that often results in paradigm shifts. Paradigms consist of shared generalizations, theoretical commitments, values, and exemplars that unify a scientific community and shape its research practices. During periods of normal science, research focuses on expanding and refining the existing paradigm rather than pursuing novelty. Discovery typically begins with the recognition of anomalies—phenomena that defy the expectations established by the current paradigm. This process includes observing and

conceptualizing the anomaly, followed by revising the paradigm to accommodate it. During paradigm crises, theory-driven discoveries may occur as scientists propose speculative theories, develop new expectations, and conduct experiments or observations to test these ideas. Ultimately, a new paradigm emerges, transforming the once-anomalous phenomena into standard expectations.

Scientific discovery can be also viewed as inherently tied to creativity, with philosophers drawing from cognitive science, neuroscience, computational research, and environmental and social psychology to better understand how new ideas emerge. This perspective aims to demystify the mental processes behind creative thought, emphasizing that scientific creativity can be analyzed and understood philosophically. Central to this analysis are two pivotal cognitive mechanisms: analogies and mental models, which serve as fundamental tools in the generation of innovative ideas and the advancement of knowledge.

- *Analogy:* Analogies play a crucial role in scientific discovery by enabling the transfer of ideas from one domain to another. Philosophers distinguish between three types of analogies: positive, negative, and neutral. Positive analogies involve properties that are shared by both the model (the well-understood domain) and the target domain (the new domain). Negative analogies include properties that belong solely to the model and do not apply to the target domain. Neutral analogies are the most intriguing because they consist of properties whose relevance to the target domain remains uncertain. These neutral analogies are significant as they often lead to new insights and hypotheses about the less familiar domain. Additionally, there is a distinction between horizontal and vertical analogies. Horizontal analogies connect two domains at the same level of abstraction, whereas vertical analogies involve relationships between different levels of abstraction within the same domain.
- *Model-based reasoning:* The concept of model-based reasoning suggests that much of human thought, including probabilistic and causal reasoning as well as problem-solving, relies on mental models rather than formal logic or strict methodological rules. In this approach, the mind constructs structural representations of real-world or imagined situations and manipulates these models to explore possibilities and generate insights. Conceptual structures are viewed as models, and conceptual innovation involves creating new models using various operations. Analogical reasoning, or analogical modeling, is one of the primary forms of model-based reasoning, alongside visual modeling and simulative modeling, such as through experiments.

## 17.6 Scientific Explaination

If the reasoning that takes us from this factual basis to the laws and theories that constitute scientific knowledge is sound, then the resulting knowledge can itself be taken to be securely established and objective

In philosophy of science it is often made the assumption that there exists a single, distinct type of explanation that qualifies as "scientific." The concept of "scientific explanation" suggests at least two key elements: first, a contrast between explanations characteristic of science and those that are not, and second, a contrast between "explanations" and other forms of discourse, such as mere "descriptions." It is important to note that a set of claims can be true, accurate, and supported by evidence while still failing to qualify as explanatory.

Good scientific explanations are typically evaluated based on several key characteristics. While different philosophical frameworks might emphasize different criteria, the following characteristics are broadly agreed upon:

*Empirical Adequacy:* A good explanation must align with observed and experimental evidence. It should accurately describe the phenomena being explained, providing a detailed account of how the observed data supports the explanation. Additionally, it should integrate well with existing empirical findings, ensuring reliability and fostering broader scientific understanding.

*Logical Coherence:* The explanation should be logically consistent and free of contradictions. Its components should interconnect in a structured and harmonious way, enabling clear and valid reasoning. This coherence ensures that the explanation aligns with established logical principles and provides a solid foundation for understanding the phenomena under investigation.

*Causal Relevance:* Good explanations often identify the causal mechanisms responsible for the phenomenon. They should clarify how and why the phenomenon occurs, detailing the specific interactions and processes involved. By establishing a clear causal link, these explanations provide a framework for understanding not just what happens, but the underlying reasons and mechanisms driving the event. This depth of causality enables predictions, interventions, and broader scientific application.

*Generality:* Explanations that apply to a broader range of phenomena are considered more valuable. They should transcend individual instances to reveal patterns or principles that connect seemingly disparate phenomena. By addressing a wider scope, such explanations enhance our ability to generalize knowledge, foster predictive accuracy, and provide a unifying perspective across different domains of inquiry.

*Simplicity (Parsimony):* A good explanation should not be unnecessarily complex. Among competing explanations, the one that makes the fewest assumptions while still accounting for the phenomena is often preferred

(Occam's Razor). This principle emphasizes clarity and efficiency in reasoning, ensuring that explanations avoid superfluous details or unwarranted complexity. Simpler explanations are easier to test, communicate, and apply, fostering a more practical and streamlined understanding of the phenomena.

*Explanatory Depth:* Good explanations go beyond surface descriptions to provide a deeper understanding of the underlying mechanisms, principles, or causes. They delve into the fundamental elements that drive the phenomenon, offering insights into its origins and interconnections with related concepts. By unraveling these deeper layers, such explanations help uncover not just the "how," but the "why," enriching our comprehension and enabling a more profound application of knowledge.

*Testability:* The explanation must allow for predictions that can be tested and potentially falsified. This criterion ensures that the explanation is scientifically meaningful and subject to empirical scrutiny. By being testable, the explanation invites rigorous examination and challenges, which help to confirm its validity or expose its weaknesses. Testability also connects scientific explanations to the broader experimental process, enabling continuous refinement and adaptation in light of new evidence, thereby fostering scientific progress and reliability.

*Unification:* A good explanation often unifies disparate phenomena under a single framework or theory, showing how they are related. It highlights the connections and underlying principles that bring coherence to seemingly unrelated phenomena. By doing so, unification not only simplifies our understanding but also enhances the explanatory power of a theory, allowing for a more integrated and comprehensive perspective on the natural world.

*Use of Laws:* Good explanations often incorporate established scientific laws or theories to provide a robust foundation for their claims. By grounding explanations in well-established principles, they ensure credibility and consistency with the broader scientific framework. This approach not only strengthens the explanatory power of the claims but also facilitates their integration into existing knowledge systems, thereby fostering coherence, predictability, and utility in advancing scientific understanding.

*Practical Applicability:* In some cases, the usefulness of an explanation in solving problems, guiding further research, or applying knowledge in practical ways adds to its value. An explanation with practical applicability not only enhances theoretical understanding but also bridges the gap between abstract knowledge and real-world implementation. It can inform decision-making, inspire technological innovations, and address pressing societal challenges. By demonstrating utility in diverse contexts, such explanations underscore the relevance of scientific inquiry to everyday life and future advancements.

*Asymmetry:* The requirement of asymmetry in scientific explanation is a widely discussed topic in the philosophy of science. According to most philosophical accounts, explanation is indeed considered to be an asymmetric relation, meaning that if X explains Y, it should not also be true that Y explains X under the same laws and additional facts. However, there are debates and complications surrounding this idea. Let's examine the arguments for and against asymmetry in scientific explanations.

*Explanation as prediction:* When we provide a covering law explanation for a phenomenon, the laws and specific facts we reference could have allowed us to predict the phenomenon's occurrence. This highlights that a scientific explanation is inherently a potential prediction. Conversely, every reliable prediction is inherently a potential explanation, illustrating that explanation and prediction are structurally symmetric in their foundational principles.

The following paragraphs briefly introduce the most relevant proposals to characterize what is a scientific explanation.

The *Deductive-Nomological model*, or DN model, emphasizes the importance of deductive reasoning and general laws. According to this model, a phenomenon is explained by demonstrating how it logically follows from a general law combined with specific initial conditions. This explanation comprises two main components: the *explanans*, which includes the general laws and initial conditions, and the *explanandum*, which is the phenomenon to be explained. For the explanation to be valid, the explanans must be true and logically entail (see logical deduction at Section 17.7) the explanandum, meaning the phenomenon can be deduced from the laws and conditions provided. A DN model answers the question "Why did the explanandum occur?" by showing that the phenomenon resulted from specific circumstances  $C_1, C_2, \dots, C_i$ , in conjunction with laws  $L_1, L_2, \dots, L_j$ . For example, the motion of a particular pendulum can be explained by applying Newton's laws of motion (general laws) along with specific details such as the pendulum's length and initial displacement (specific initial conditions). Alternatively, the general behavior of pendulums can also be explained using the same laws and reasoning, since the DN model can be applied to both particular occurrences and general patterns.

The *Statistical-Relevance model*, or SR model focuses on explaining phenomena through statistical relationships rather than strict deductive reasoning. Unlike the Deductive-Nomological model, which requires logical entailment from general laws, the SR model emphasizes the identification of statistically relevant factors that significantly influence the likelihood of a phenomenon. In this model, given some class or population  $A$ , an attribute  $C$  is *statistically relevant* to another attribute  $B$  if and only if  $P(B|A, C) \neq P(B|A)$ . This means

that  $C$  affects the probability of  $B$  within the context of  $A$ . An explanation involves identifying such statistically relevant factors and evaluating their impact within a reference class (a group of events or entities sharing common characteristics). For example, in explaining the likelihood of developing a particular disease, an SR explanation might highlight factors such as age, genetic predisposition, or lifestyle choices, showing how these variables alter the probability of the disease occurring. By uncovering these statistical relationships, the SR model provides a method for explaining probabilistic phenomena that cannot be addressed deterministically.

The *Causal-Mechanical model*, or CM model, of scientific explanation emphasizes understanding phenomena by uncovering the underlying causal mechanisms that produce them (see causality at Section 17.7). This model asserts that explanations are not just about identifying laws or statistical relationships but about revealing the actual processes and interactions that link causes to effects. A causal-mechanical explanation requires tracing a continuous causal chain, often through detailed physical or biological processes, to show how an event is brought about. For example, explaining the boiling of water involves identifying the causal mechanism: heat energy transfers to the water molecules, increasing their kinetic energy until intermolecular bonds are overcome, leading to a phase change from liquid to gas. The CM model prioritizes clarity in how individual components interact and influence each other, providing a deeper understanding of the phenomenon by grounding it in observable and empirically testable mechanisms. This approach is particularly effective in fields like biology, physics, and engineering, where complex systems and their interactions play a central role in explanation.

The *unificationist account* of scientific explanation, emphasizes the power of explanation through the unification of diverse phenomena under a single, coherent framework of principles and patterns. According to this model, the primary aim of scientific explanation is to reduce the number of independent assumptions and derive a wide range of phenomena from a minimal, consistent set of explanatory patterns. An explanation is considered successful if it contributes to this unifying framework by connecting seemingly disparate observations through common principles. For instance, Newtonian mechanics unifies the motions of celestial bodies and terrestrial objects under the same laws of motion and gravitation. The unificationist approach highlights the importance of simplicity, generality, and coherence in scientific theories, proposing that the value of an explanation lies in its ability to integrate knowledge into an organized, explanatory schema. By offering a comprehensive understanding of diverse phenomena, this account showcases the interconnectedness and systematic nature of scientific inquiry.

The *pragmatic theories* of scientific explanation emphasize the context-

dependent and audience-specific nature of explanations, focusing on their purpose and practical utility rather than strict formal structures. These theories argue that explanations are answers to "why" questions posed within a specific context, and their adequacy depends on how well they address the interests and background knowledge of the audience. A scientific explanation, therefore, is not inherently tied to a universal standard but varies depending on the explanatory goals, such as prediction, understanding, or control. For instance, explaining why a bridge collapsed might involve detailed structural analysis for engineers, whereas a simplified account focusing on the immediate cause, like high winds, might suffice for the general public. Pragmatic approaches recognize that explanatory demands can differ across disciplines, situations, and audiences, making the effectiveness of an explanation contingent on its relevance, clarity, and alignment with the inquirer's needs. This perspective underscores the interplay between scientific knowledge and its communication within varied practical contexts.

## 17.7 Scientific Justification

scientific knowledge can neither be conclusively proved nor conclusively disproved

Logicians make an important distinction between deductive and inductive inference [...] The two statements above the line are called the premises of the inference, while the statements below the line is called the conclusion. This is a deductive inference because it has the following property: if the premises are true, then the conclusion must be true too [...] This is sometimes expressed by saying that the premises of the inference entail the conclusion [...] In a typical inductive inference, we move from premises about objects that we have examined to conclusions about objects of the same sort that we haven't examined.

When we reason deductively, we can be certain that if we start with true premises we will end up with a true conclusion. By contrast, inductive reasoning is quite capable of taking us from true premises to a false conclusion [...] Scientists reason inductively whenever they move from limited data to a more general conclusion, which they do all the time.

Popper claimed that scientist only need to use deductive inferences.

Although a scientific theory (or hypothesis) can never be proved true by a finite amount of data, it can be proved false, or refuted.

Hume argued [Hume's problem of induction] that the use of induction cannot be rationally justified at all [...] whenever we make inductive inferences, we presuppose the 'uniformity of nature' [...] our reasoning depend on the assumption that objects fwe haven't examined will be similar, in relevant

aspects, to object of the same sort we have examined [...] we cannot prove the uniformity assumption.

Inductive inferences have all the same structure, the premise has had the form 'all examined Fs have been G' and the conclusion the from 'other Fs are also G' [...] Reasoning of this sort is known as 'inference to the best explanation' (IBE) [...] The basic idea behind IBE – reasoning from one's data to a theory or hypothesis that explains the data – good explanation should be simple [...] Preferring a theory which explains the data in terms of the fewest number of causes seems sensible. But are there any objective grounds for thinking that such a theory is more likely to be true than a less simple rival?

Thomas Kuhn introduced the concept of scientific paradigms, which revolutionized the understanding of scientific progress. A paradigm encompasses the set of practices, theoretical frameworks, methodologies, and standards that define a scientific discipline during a specific period. According to Kuhn, normal science operates within the confines of a prevailing paradigm, solving puzzles and extending the framework. However, scientific revolutions occur when anomalies accumulate—phenomena that the existing paradigm cannot adequately explain—leading to a paradigm shift. This shift replaces the old framework with a new one that better accommodates the observed data, fundamentally altering the trajectory of scientific inquiry. Kuhn's insights emphasize the sociocultural and historical dimensions of science, challenging the notion of continuous, cumulative progress.

[...] scientific revolutions, periods of great upheaval when existing scientific ideas are replaced with radically new ones [...] these revolutions led to a fundamental change in the scientific worldview, the overthrow of an existing act of ideas by a completely different set [...] revolutions happen relatively infrequently [...] 'normal science' [...] the ordinary day-to-day activities that scientists engage in when their disciplines is not undergoing revolutionary change [...] A paradigm consists of two main components: first, a set of fundamental theoretical assumptions which all members of a scientific community accept; secondly, a set of 'exemplars' or particular scientific problems which have been solved by means of those theoretical assumptions [...] When scientists share a paradigm [...] they agree on how future research in the field should proceed, on which problems are the pertinent ones to tackle, on what the appropriate methods for solving those problems are, and on what an acceptable solution of the problems would look like [...] a paradigm is an entire scientific outlook, a constellation of shared assumptions, beliefs, and values which unite a scientific community and allow normal science to take place. [...] The job of the normal scientist is to try to eliminate these minor puzzles while making a few changes as possible to the paradigms [...] normal

scientists are not trying to test the paradigm. On the contrary, they accept the paradigm unquestioningly, and conduct their research within the limits it sets [...] as anomalies accumulate, a burgeoning sense of crisis envelops the scientific community. [...] A variety of alternatives to the old paradigm are proposed, and eventually a new paradigm becomes established [...] The essence of a scientific revolution is thus the shift from an old paradigm to a new one. [...] Kuhn argued that adopting a new paradigm involves a certain act of faith on the part of the scientists. [...] if paradigm shifts work the way Kuhn says, it is hard to see how science can be regarded as a rational activity at all. [...] the facts about the world are paradigm-relative, and thus change when paradigms change [...] Kuhn espouse a radical form of anti-realism about science.

[...] competing paradigms are typically 'incommensurable' with one another [...] Incommensurability is the idea that two paradigms may be so different as to render impossible any straightforward comparison of them with each other, there is no common language into which both can be translated [...] concepts cannot be explained independently of the theories in which they cannot be explained independently of the theories in which they are embedded. This idea, which is sometimes called 'holism' [...] replacement of 'wrong' ideas by 'right' ones [...] later paradigms are not better than earlier, just different [...] old and new paradigms to be incompatible [...] lack of a common language [...] incommensurability of standards.

[...] theory-ladenness of data [...] the ideal of theory-neutrality is an illusion, data are invariably contaminated by theoretical assumptions [...] a dispute between competing paradigms could not be resolved by simply appealing to 'the data' or 'the facts', for what a scientist counts as data, or facts, will depend on which paradigms they accept [...] To be objectively true, a theory must correspond to the facts, but the idea of such a correspondence makes little sense if the facts themselves are infected by our theories [...] perception is heavily conditioned by background beliefs [...] scientists' experimental and observational reports are often couched in highly theoretical language.

[...] there is 'no algorithm' for theory choice in science. [...] no one has ever succeeded in producing such an algorithm. [...] For one thing, there may be trade-offs: theory A maybe simpler than theory B, but B may fit the data more closely. So an element of subjective judgment, or scientific common sense, will often be needed to decide between competing theories. [...] is not that paradigm shifts are irrational, but rather that a more relaxed, pragmatic concept of rationality is required to make sense of them.

[...] as did the idea of a sharp dicotomy between the context of discovery and justification. [...] His doctrines of paradigm shifts, normal and

### revolutionary science, incommensurability, and theory landeness.

Paul Feyerabend, in his provocative work "Against Method," argued that there is no universal scientific method that guarantees the success of science. He criticized the rigid methodological rules proposed by philosophers like Popper and Kuhn, suggesting that science has advanced precisely because of its methodological anarchy. Feyerabend contended that "anything goes" in science, meaning that historical scientific breakthroughs often occurred by defying established rules and norms. For example, Galileo's use of persuasive rhetoric and creative reasoning, rather than strict adherence to empirical observation, was crucial in advancing heliocentrism. Feyerabend's ideas challenge the assumption that scientific progress is orderly and rational, emphasizing instead the role of historical, cultural, and personal factors in shaping scientific practice. While his views remain controversial, they highlight the complexity and diversity of scientific inquiry and question the feasibility of prescribing universal methodological principles.

The hypothetico-deductive method is a widely recognized approach in the philosophy of science, describing how scientific theories and knowledge are developed and tested. It begins with the formulation of a hypothesis, which serves as a tentative explanation for a phenomenon or a set of observations. From this hypothesis, scientists deduce specific, testable predictions or consequences, often in the form of "if-then" statements. These predictions are then subjected to empirical testing through experiments or observations. If the results align with the predictions, the hypothesis is supported but not conclusively proven; if the results contradict the predictions, the hypothesis may need to be revised or rejected. This iterative process underscores the provisional nature of scientific knowledge.

Falsificationism emphasizes that scientific theories can never be conclusively proven but can be rigorously tested through attempts to falsify them. An example of this is Einstein's theory of general relativity, which predicted that light would bend near massive objects like the sun. This prediction was empirically tested during the solar eclipse of 1919, when astronomers observed starlight bending as it passed near the sun, providing a crucial opportunity to potentially falsify the theory—but instead offering strong support for it. According to this perspective, a theory is scientific only if it is falsifiable—that is, if it makes predictions that could, in principle, be shown to be false by empirical evidence. In this view, the strength of a scientific theory lies in its ability to withstand attempts at falsification, and progress in science occurs when falsified theories are replaced by better, more robust ones. This perspective shifts the focus from verification to critical testing, highlighting the dynamic and self-correcting nature of scientific inquiry.

One challenge within the hypothetico-deductive method is addressing

the origin of hypotheses themselves. While the method provides a systematic way to test and refine hypotheses, it does not dictate where these initial ideas come from. In the past, the generation of hypotheses was often the direct result of careful observations of natural phenomena. However, as science has progressed, hypotheses have increasingly become products of creativity, intuition, or inspiration drawn from prior knowledge, analogies, or even serendipitous observations. This aspect highlights the interplay between the logical structure of scientific testing and the imaginative processes that fuel scientific discovery. Philosophers of science have long debated whether hypothesis generation is a purely rational process or one influenced by subjective and contextual factors, underscoring the complexity of scientific creativity.

*Statistical inference* has become a significant approach within the scientific method, providing a framework for deriving conclusions from data in the face of uncertainty. By employing probabilistic models and statistical techniques, scientists can estimate the likelihood that observed phenomena are consistent with a given hypothesis. Techniques such as hypothesis testing, point estimation, and confidence intervals allow researchers to quantify uncertainty and make data-driven decisions. This approach is particularly powerful in fields where direct experimentation is difficult or impossible, such as cosmology or epidemiology. However, reliance on statistical methods also raises important questions about the interpretation of probabilities (see Section 12.1) and the potential for misuse, such as overfitting models or neglecting prior assumptions. Despite these challenges, statistical inference remains an indispensable tool for connecting empirical data to theoretical models in modern science.

*The Bayesian approach* is an application of the statistical reasoning to the scientific method. Based on Bayes' theorem (see Theorem 12.2.2), the Bayesian approach provides a formal framework for updating the probability of a hypothesis as new evidence emerges. By iteratively adjusting probabilities, the Bayesian approach dynamically reflects how beliefs evolve as evidence accumulates. Each time an experiment is successfully performed the likelihood of the hypothesis to be true increases. This iterative process allows for a dynamic adjustment of beliefs, reflecting how evidence accumulates over time. Bayesian methods are particularly useful in contexts of uncertainty or incomplete data, such as medical diagnosis, or climate modeling.

Let  $P(h | e)$  denote the probability of a hypothesis  $h$  in the light of evidence  $e$ .  $P(e | h)$  denote the probability to be ascribed to the evidence  $e$  on the assumption that the hypothesis  $h$  is correct,  $P(h)$  the probability ascribed to  $h$  in the absence of knowledge of  $e$ , and  $P(e)$  the probability ascribed to  $e$  in the absence of any assumption about the truth of  $h$ . The Bayes' theorem (see

Theorem 12.2.2) can be written:

$$P(h | e) = P(h) \frac{P(e | h)}{P(e)}$$

$P(h)$  is referred to as the prior probability since it is the probability ascribed to the hypothesis prior to consideration of the evidence  $e$ , and  $P(h | e)$  is referred to as the posterior probability, the probability after the evidence  $e$  is taken into account. So the formula tell us how to change the probability of a hypothesis to some new, revised probability in the light of some specified evidence.

The factor  $P(e/h)$  is a measure of how likely  $e$  is given  $h$ . It will take a maximum value of 1 if  $e$  follows from  $h$  and a minimum value of zero if the negation of  $e$  follows from  $h$ .

With sufficient amount of evidence, bayes converges? (weigth of isotopes), making less relevant the problem of subjective priors (conjugate priors?)

■ **Example 17.1** Something established barely changes with new experiments, bold confirmed conjectures do ■

When an experiment fails to confirm a theory, Bayesian inference allows for a probabilistic adjustment of the theory's credibility by updating its posterior probability in light of the evidence, unlike falsificationism, which considers a theory definitively refuted when evidence contradicts its predictions.

Bayesian inference, while powerful, has notable limitations:

Objective prior probabilities of hypothesis are difficult to obtain (how do we list all possible hypothesis?). Subjective priors represent degrees of belieif. its reliance on subjective priors can introduce bias, requiring careful justification;

it is sometimes applied in contexts lacking a well-defined probability space, undermining its validity; and it cannot accommodate scientific inferences that leap to entirely new theories, as conditionalization is incapable of handling hypotheses or concepts absent from the prior probability space. Despite these limitations, Bayesian inference continues to shape modern scientific practices, highlighting the interplay between evidence, prior knowledge, and probabilistic reasoning.

*Causality* refers to the relationship between events where one event (the cause) brings about or influences another event (the effect), and it asserts that explaining a natural phenomenon involves identifying its causes. In the philosophy of science, there is ongoing debate about whether causal explanations differ fundamentally from those based on general laws, as

deducing an event from a general law often implicitly reveals its cause. A key feature of causality is its inherent asymmetry, if  $x$  is the cause of  $y$ , then  $y$  cannot be the cause of  $x$ ; this asymmetry is a crucial distinction that sets causal explanations apart from mere statistical correlations. Many challenges arise when relying on empirical observations to justify causal relationships. One perspective emphasizes the development of rigorous methodologies, generally by statisticians and computer scientists, to infer causality from observational data, addressing issues such as *confounding factors* (variables that obscure the true causal relationship) and the principle that correlation does not imply causation. *Randomized controlled trials* stand as a gold standard in causal inference, where confounding variables are systematically controlled by design to isolate the effect of the factor under investigation. In contrast, empiricists argue that causal relations cannot be directly observed and suggest that causality is merely a conceptual framework humans impose on the world to make sense of observed regularities.

## 17.8 The Limits of Science

[...] philosophical enquiry has its own proprietary methods, which can reveal truths of a sort that science cannot. [...] They include logical reasoning, the use of thought experiments, and what it is called 'conceptual analysis', which tries to delimit a particular concept by relying on our intuitions about whether a particular case fall under it. [...] It is often felt that natural sciences such as physics, chemistry, and biology are in a more advanced state than social sciences such as economics, sociology, and anthropology; the former can formulate precise laws with great predictive power, while the latter usually cannot. [...] It is never possible to prove that a scientific theory is true, in the strict sense of proof, for the inference from data to theory is invariably non-deductive.

### Non-detectable entities

[...] sience can in principle explain everything? Or are there some phenomena that must forever elude scientific explanation? [...] However much the science of the future can explain, the explanations it gives will have to make use of certain fundamental laws and principles. Since nothing can explain itself, it follows tat at least some of these laws and principles will themselves remain unexplained [...] some things will never be explained, but does not tell use what they are.

Science has long been a powerful tool for understanding and shaping the world. However, its scope is not unlimited. There are domains where science struggles to provide answers, such as consciousness or moral values. Metaphysical questions, such as the nature of existence or why there is

something rather than nothing, often fall outside the realm of empirical investigation. Similarly, science can describe the consequences of actions but cannot determine moral values or prescribe what ought to be done. This section contains a brief description of the main topics related to the limits of science.

### Reductionism vs. Holism

One key debate in understanding the limits of science is the tension between *reductionism* and *holism*. Reductionism asserts that complex phenomena can be fully understood by breaking them down into their most basic components and laws. For example, biology might be reduced to chemistry, and chemistry to physics. However, holism argues that some phenomena, especially in systems like ecosystems, societies, or even consciousness, cannot be fully understood without considering the interactions and emergent properties of the whole system. This debate underscores the challenge of addressing complexity within scientific frameworks.

The different scientific disciplines are designed for explaining different types of phenomena [...] there is a distinction of labour between the different sciences: each specializes in explaining its own particular set of phenomena [...] it is widely held that the different branches of science are not all on a par: some are more fundamental than others [...] physics can subsume all the higher-level sciences? [...] The objects studied by the higher-level sciences are multiply realized at the physical level.

Scientific inquiry excels in answering questions about the natural world, but it encounters boundaries when addressing issues that are metaphysical, moral, or inherently subjective. For example, while science can study the mechanics of evolution, it does not address whether life has an intrinsic purpose. Similarly, while it can analyze the factors influencing human behavior, it cannot resolve normative questions about justice or fairness. These limitations highlight the need for interdisciplinary approaches and alternative methods of inquiry.

*Realism vs. Anti-realism:* As it was already mentioned in Section 17.2, another critical debate in philosophy of science concerns the nature of scientific knowledge: does science uncover true aspects of reality (*scientific realism*), or does it merely provide useful models and predictions without necessarily representing reality (*scientific anti-realism*)? Anti-realism suggests that scientific theories are instruments for helping us predict observable phenomena, rather than as attempts to describe the underlying nature of reality. The scientific realism vs scientific anti-realism debate underscores a debate in philosophy between two opposing schools of thought called *realism* and *idealism*. Realism holds that the physical world exists independently of

human thought and perception. Idealism claims that the physical world is in some way dependent on the conscious activity of humans.

The underdetermination argument: Anti-realist emphasize that the empirical data to which scientific theories are responsible consist of fact about observable entities and processes. [...] fact about observable phenomena provide the ultimate data from theories that posit unobservable entities and processes. [...] Anti-realists argue that the empirical data 'underdetermine' the theories scientist put forward on their basis [...] the data can in principle be explained by many different, mutually incompatible, theories. [...] Undetermination leads naturally to the anti-realist conclusion that agnosticism is the rational attitude to take towards theories about the unobservable region of reality. [...] But, say the realists, it does not follow that all of these possible explanations are equally good. [...] why simple theories should be though more likely to be true.

### Objectivity

Science is often regarded as an objective endeavor, but this view is contested. Scientific practices are influenced by the cultural, social, and personal contexts of researchers. For example, funding priorities, societal values, and individual biases can shape research questions, methodologies, and interpretations. The ideal of objectivity remains central to scientific inquiry, but recognizing these influences is crucial for understanding the limitations and potential biases inherent in scientific work.

#### Address the self-correcting nature of science

### The Demarcation Problem

The *demarcation problem* addresses the challenge of distinguishing science from non-science or pseudoscience. This philosophical issue has significant implications for the credibility and authority of scientific knowledge. Karl Popper's criterion of falsifiability, a theory is scientific if it can be tested and potentially falsified, has been influential but not universally accepted. Some disciplines, like evolutionary psychology or string theory, straddle the line, sparking debate about their scientific status. The demarcation problem highlights the complexity of defining the scope and boundaries of science.

Science thrives on uncertainty, yet there are questions that may remain permanently beyond its reach. For example, the origins of the universe, the nature of dark matter, or the ultimate fate of existence may elude definitive answers. Furthermore, the tools and methodologies of science may be inherently limited in addressing phenomena that require new paradigms or approaches. This recognition does not diminish science but rather emphasizes its role as one part of a broader human quest for knowledge.

While science provides the means to achieve certain ends, it does not

dictate the ethical use of its discoveries. Ethical questions, such as the use of genetic engineering or artificial intelligence, require value-based judgments that go beyond empirical evidence. Integrating ethical considerations with scientific progress is essential to navigate the societal impact of science responsibly.

## **References**

Add references to the following entries from the Stanford Encyclopedia of Philosophy:

- Scientific Representation
- Scientific Explanation
- Scientific Method
- Scientific Discovery

?? contains an interesting review of the concept of science, the scientific method, and the role that technology plays in our society. The author proposes that the goal of science should be quality, although the concept of quality is left undefined, and how to reconcile the rational and romantic points of view in science. The book also contains some advice about which is the right state of mind to pursue a scientific problem, and how to deal with the inevitable failures.

# Appendix

|          |                                      |            |
|----------|--------------------------------------|------------|
| <b>A</b> | <b>Advanced Properties .....</b>     | <b>469</b> |
| A.1      | The Axioms of Science                |            |
| A.2      | Scientific Method                    |            |
| A.3      | The Inaccuracy - Surfeit Trade-off   |            |
| A.4      | Graspness                            |            |
| A.5      | Effort                               |            |
| A.6      | Human Understanding                  |            |
| A.7      | Areas in Decay                       |            |
| <b>B</b> | <b>Advanced Mathematics .....</b>    | <b>491</b> |
| B.1      | Distinctiveness of Metrics           |            |
| B.2      | Distinctiveness of Metrics           |            |
| B.3      | Statistical Significance of Analysis |            |
| <b>C</b> | <b>About Quotes and Photos .....</b> | <b>499</b> |
| <b>D</b> | <b>Notation .....</b>                | <b>507</b> |
|          | <b>Bibliography .....</b>            | <b>511</b> |
|          | Books                                |            |
|          | Articles                             |            |
|          | <b>Index .....</b>                   | <b>515</b> |





## A. Advanced Properties

*Invert, always invert.*

Carl Gustav Jacob Jacobi

This chapter covers more advanced mathematical properties of the concept of nescience. This chapter can be safely skipped by those readers interested only in the applications of nescience. However, it is highly recommended to read it, since it provides a deeper understanding of what nescience is exactly.

**TODO:** Explain the abstract nature of the axioms vs. the interpretation we have provided in the previous chapters. Mention that perhaps there exists other interpretations.

Since the time of David Hilbert, mathematics is about the study of the properties of abstract objects and their relations, without paying too much attention to what these objects are or represent. Mathematicians create (or discover) abstract frameworks of logic that can have multiple interpretations. It is up to applied scientists to provide those interpretations. In the same way, we can provide an abstract definition of the concept of nescience, and study its properties, without making any explicit reference to science nor to the scientific method. In doing so, we lose the interpretability of our theory, but, on the other side, we could apply the same results to other disciplines.

Extend this introduction with a very short review of the topics covered in the chapter.

## A.1 The Axioms of Science

**TODO:** Say something intelligent here.

In this section we are going to propose a collection of axioms that formalize what is this thing called science.

In the previous section we have provided a formalization of the theory of nescience based in set theory and first order logic. Set theory is not the only available framework in which mathematics can be formalized. We could have used type theory or category theory instead. Having meaningful axioms of our theory in multiple formalization frameworks, such as set theory, type theory, and category theory, will constitute strong evidence that nescience is a fundamental concept in mathematics, and not a mere artifact of a particular formalization.

### A.1.1 Model Theory

Our first approximation to the problem of how to formalize the theory of nescience is a collection of axioms based on first-order logic and the ZFC axioms of set theory with equality (see Appendix ??). In this section we will also prove some basic results and explain how these axioms relate to the new ideas we have introduced in this book. For this axiomatization we are not going to follow the approach described in the previous chapters, that is, we are not going to explicitly define the quantities of miscoding, surfeit and inaccuracy. Instead, we will introduce the concatenation and conditional operations, list their fundamental properties, and explain how nescience is affected by them. The remaining concepts follow naturally from these basic axioms.

**Definition A.1.1** Let  $\mathcal{L}$  be a non-empty set called *language* whose elements are called *strings*. We define *science* as the first-order logic structure  $(\mathcal{L} \mid \lambda, \mathcal{O}, <_N, \oplus, |)$  where:

- (i)  $\lambda \in \mathcal{L}$  is distinct element called *neutral*,
- (ii)  $\mathcal{O}$  is a relation called *oracle*,
- (iii)  $<_N$  is a relation called *nescience*,
- (iv)  $\oplus$  is a binary function called *concatenation*, and
- (v)  $|$  is a binary function called *conditional*

that satisfies the following axioms:

$$A1 \quad \forall t \in \mathcal{L} \, t\mathcal{O}t.$$

- A2  $\forall s, t \in \mathcal{L}$  if  $s\mathcal{O}t$  then  $t\mathcal{O}s$ .  
 A3  $\forall r, s, t \in \mathcal{L}$  if  $r\mathcal{O}s$  and  $s\mathcal{O}t$  then  $r\mathcal{O}t$ .  
 A4  $\forall t \in \mathcal{L} \neg t <_N t$ .  
 A5  $\forall s, t \in \mathcal{L}$  if  $s <_N t$  then  $\neg t <_N s$ .  
 A6  $\forall r, s, t \in \mathcal{L}$  if  $r <_N s$  and  $s <_N t$  then  $r <_N t$ .  
 A8  $\forall s, t \in \mathcal{L} \oplus(s, t) = \oplus(t, s)$ .  
 A9  $\forall s \in \mathcal{L} \oplus(s, \lambda) = \oplus(\lambda, s) = s$ .  
 A10  $\forall r, s, t \in \mathcal{L} \oplus(\oplus(r, s), t) = \oplus(r, \oplus(s, t))$ .  
 A11  $\forall s \in \mathcal{L} |(s, \lambda) = s$ .  
 A12  $\forall s \in \mathcal{L} |(s, s) = \lambda$ .

Axioms A1, A2 and A3 state that the oracle  $\mathcal{O}$  is an equivalence relation; the nescience relation  $<_N$  described by Axioms A4, A5 and A6 is a strict partial order; the concatenation operation  $\oplus$  defined by Axioms A8, A9 and A10 together with  $\mathcal{L}$  forms a free monoid; and Axioms A11 and A12 define the behaviour of the conditional operator  $|$ .

We still need two more axioms to fully characterize the behaviour of our logic structure. But first we have to introduce some and additional concept.

**Definition A.1.2** Let  $\mathcal{L}/\mathcal{O}$  be the quotation set defined by the oracle relation over the language set. We call *entity*, denoted by  $[e]$ , to every class of this quotation set, that is,  $[e] \in \mathcal{L}/\mathcal{O}$ .

With the next Axiom we require that every entity contains at least one minimal string with respect to the nescience ordering.

**Definition A.1.3 — Axiom A13.** For every entity  $[e] \in \mathcal{L}/\mathcal{O}$  there exists at least one  $r \in [e]$  such that it is minimal with respect to  $<_N$ , that is, it does not exist an  $s \in [e]$  such that  $s <_N r$ .

Our last axiom refers to how nescience decreases.

**Definition A.1.4 — Axiom A14.** Let  $t \in \mathcal{L}$  be a non-minimal element, then:

- (i) there exists  $s$  such that  $\oplus(t, s) <_N t$ , or
- (ii) there exists  $s$  such that  $|(t, s) <_N t$ .

### Interpretation of the Axioms

The first thing to note about our proposal is that the set  $\mathcal{E}$  of entities in which we are interested is not part of the axioms. Instead, the entities are studied indirectly through a language set  $\mathcal{L}$ . Furthermore, we have not provided any indication about the nature of the elements of  $\mathcal{L}$  besides that it must be a non-empty set. In this sense,  $\mathcal{L}$  could be anything that satisfy our axioms.

For example,  $\mathcal{L}$  could be the set  $\mathcal{B}^*$  of finite binary strings, in which case nescience will be based on string length; or it could be the set  $\mathbb{N}$  of natural numbers, and derive nescience from the ordering of these numbers<sup>1</sup>. This approach of studying the properties of a set without specifying the nature of its elements (the standard approach used in the mathematics of the last century) has some advantages and disadvantages. The main advantage is that perhaps we could apply our theory to other areas (yet to be discovered) for which the theory is not intended. The limitation is that this abstract nature of the axioms makes more difficult to apply the theory in practice, for example, to derive practical algorithms that compute the new proposed metrics.

The only tool we have at our disposal to match the strings of  $\mathcal{L}$  with the entities of  $\mathcal{E}$  is the oracle, and this matching has to be done in an indirect way. The oracle defines an equivalence relation that splits the set  $\mathcal{L}$  into equivalence classes. Each equivalence class corresponds to an entity of  $\mathcal{E}$ , and it might happen that not all entities of  $\mathcal{E}$  have an associated class (what we have called the unknowable unknown). All the strings are related to some entity, and some of them would be better than others (i.e. lower nescience). No string can be part of two different entities. For each collection of  $\mathcal{E}$  there could be more than one valid oracle. The details (inner workings) of how the oracles match strings to entities is not covered by the axioms.

We cannot provide a quantitative measure for the concept of nescience, since numbers are not part of our axioms. Even if we use as underline set  $\mathcal{L}$  the set of natural numbers  $\mathbb{N}$ , the symbol  $+$  and its properties are not part of our logic structure. Instead what we have provided is a relative ordering of the different elements of  $\mathcal{L}$ , intuitively, according to how much we do not know given those strings. The ordering has to be partial, i.e. not every pair of elements of  $\mathcal{L}$  can be compared. This partial order is applicable not only in case of strings that belong to different entities, but also it can happen with strings that belong to the same entity. The order has also to be strict, i.e. nescience equality is not defined. The problem of assuming a non-strict total order for nescience is that if  $s <_N t$  and  $t <_N s$  then it should be the case that " $s = t$ ", that is, if two strings have the same nescience, they should be the same string, and this is not necessarily the case. How nescience is assigned to the elements of  $\mathcal{L}$  is not covered by the axioms.

The problem of having multiple styles for the same entity is not explicitly addressed by the axioms, although it is implicit through the use of minimal elements (each minimal element could be related to each particular style). We do require that each equivalence class has at least one minimal element.

---

<sup>1</sup> TODO: Be a little bit more specific and mention in each case which one is the neutral element, and how concatenation and conditional could be defined. Ideally, provide a third, non-trivial, example.

An equivalence class could have more than one minimal element, and we do not require that classes have a minimum. Intuitively, each minima would correspond to each of the valid ways of representing the entity. We do not expect strings from different representation styles to be directly comparable. Finally, we do not require the existence of global minimum nor global minimas for the set  $\mathcal{L}$ .

If a string  $s$  is not a local minima for its class with respect to the nescience ordering, there must exists another string  $t$  with smaller nescience. There are two ways in which we can find this string, that is, to decrease our nescience about the string: either by concatenating the string with another one, or by conditioning the string to another one.

If the string  $s$  is missing some critical informaton required to encode the entity in which we are interested, we could extend  $s$  with additional symbols, by means concatenating  $s$  with other strings with the relevant symbols. If the string  $s$  contains non-relevant or even wrong information, we could get rid of that information by assuming that  $s$  is the concatenation of two or more strings, and removing the non-relevant substrings. Concatenation also allow us to discover new entities: concatenating two strings that belong to different entities might lead to a new, previously unknown, entity. In this sense, concatenation would implement the exploration part in this schema of exploration/exploitation in which science is characterized.

Conditioning is the tool that we have to our disposal to leverage on already existing knowledge. The difference between concatenation and conditioning is that concatenation add something to our string, meanwhile conditioning allow us to use something (if needed) without extending our current string. Conditional would implement the exploitation part in this science schema of exploration/exploitation.

## Basic Properties

First of all we will introduce some notational conventions to simplify the algebraic operations dealing with nescience.

**Notation A.1.** *We will use the infix notation for the concatenation function, that is, we will write  $s \oplus t$  instead of the  $\oplus(s, t)$ . Moreover, given Axiom A10, we will drop parenthesis in case of multiple concatenations.*

*We will use the infix notation for the conditional function, that is, we will write  $s | t$  instead of the  $|(s, t)$ .*

It is also convenient to introduce the symbol  $=_N$  to represent the case that our unknown given the strings  $s$  and  $t$  is the same. We will apply this symbol only if both strings belong to the same equivalence class.

**Notation A.2.** Let  $[e]$  be an entity, and  $s, t \in [e]$  two strings. We denote by  $s =_N t$  the case that  $(s, t) \notin <_N$ .

Conditional is non-decreasing wrt nescience

**Proposition A.1.1** Let  $s$  and  $t$  two arbitrary strings, then we have that  $s <_N (s \mid t)$ .

*Proof.* TO DO ■

TODO: Introduce the concepts of minimal and valid.

TODO: Prove that minimal implies valid.

TODO: Prove that joining two valid strings is a valid string.

TODO: Distributive Law -  $((r \mid s) \leftrightarrow t) <_n ((r \leftrightarrow t) \mid (s \leftrightarrow t))$ .

### Axiomatic Definition of the Elements of Nescience

As we have said at the introduction of this section, the collection of proposed axioms do not explicitly mention the concepts of miscoding, inaccuracy and surfeit. Moreover, in the set  $\mathcal{L}$  we do not distinguish between representations and models. All these elements are introduced as a particular interpretation of the axioms, the one we have been developing in Part II of this book. In this section we are going to prove that our interpretation of science satisfies the science axioms.

We start by assuming that the underline set  $\mathcal{L}$  is the set of all finite binary strings  $\mathcal{B}^*$ . We also assume that this set is composed by two kind of strings: representations, that are regular strings, and models, that are strings that encode a particular Turing machine assuming a fixed universal oracle machine. Each equivalence class will contain all possible representations and descriptions of that entity, including the wrong ones. We have also to assume that the oracle knows if a particular string is a representation or a description, and proceeds accordingly. For example, if the string is a representation, the oracle could do nothing, leaving the string as is, but if the string is a model, then the oracle could run the model (using our universal Turing machine) and use the output as a representation. Distinguishing between representations and descriptions also implies that the operation of concatenation has more sense in case of representations, and the operation of conditioning would be intented for descriptions. Concatenation would be used to include in our representations all the information needed to encode the entity (string concatenation), and conditional would be used to make descriptions shorter, by leveraging on other, already existing, models (conditional complexity).

TODO: Prove that our mododified universal oracle machine is an equivalence relation

TODO: Introduce the concepts os miscoding, surfeit and inaccuracy, and nescience.

TODO: Prove that nescience is a strinct partial order.

TODO: Prove that string concatenation is a free monoid. For the case of regular strings, for the case of models, and for mixed concatenations.

TODO: Prove that the axioms related to conditional satisfied the axioms. For the case of regular strings, for the case of models, and for mixed concatenations.

TODO: Prove that each equivalence class has at least one minimal element.

TODO: Prove that the nescience reduction axiom in fact is satisfied.

## A.1.2 Type Theory

In this section we are going to provide a formalization of the theory of nescience in the context of type theory (see Section ??). Type theory has some advantages over set theory. The most important is that it is a constructive theory, which means that we do not assume the existence of abstract mathematical entities that satisfy some properties, but we show how to construct those entities. For example, there is no need to resort to an abstract, uncomputable, oracle. In type theory, all propositions and proven theorems have an equivalent computable function or algorithm, and thus, we can provide computer programs to solve all problems addressed by our theory (see Section 8.1 for a practical implementation of some of these algorithms in the form of a software library).

A second advantage of type theory is that the lambda terms used in the theory are by definition computable functions, and that lambda calculus is a Turing machine capable of universal computation. In the theory of nesciece we require that our models be lambda terms, and that our universal Turing machine be the lambda calculus. In this sense, our models are native elements within the theory, not external artifacts based on an arbitrary, axiomatically defined, universal machine. The models that describe entities are lambda terms executed in the univeral Turing machine of lambda calculus.

The third advantage is that there are software implementations of type theory, such as the Coq proof assistant (see Section ??), which allow us to mechanically verify the correctness of our theory. In the rest of this section, we will use the coq syntax for definitions, propositions and theorems so that interested readers can use a computer and a Coq interpreter to verify by themselves that the proofs are correct. We have also included an appendix where we briefly describe the Coq language, so that those readers who prefer traditional mathematics can translate the Coq scripts into classical mathematical definitions and propositions.

## Foundamental Concepts

We start by introducing a new type, called `String`. The type `string` is defined recursively in the following way: the empty string, called `lambda`<sup>2</sup>, is a string, and the sucessor of a string, given the function `S`, is also a string. For us strings are just a list of lambdas, and our theory is based on the collection of all finite strings  $\{\lambda, \lambda\lambda, \lambda\lambda\lambda, \dots\}$ . Both descriptions and representations will be finite strings of lambdas.

```
Inductive String : Set :=
| lambda : String
| S      : String -> String.
```

Before to introduce the main concepts of the theory of nescience, we have to define a helper funcion called `Difference`. The `Difference` function computes the absolute difference between two strings, that is, the excess of lambdas of the longer string with respect to the shortest one. For example, if we have the strings  $\lambda\lambda$  and  $\lambda\lambda\lambda$ , the difference would be the string  $\lambda$ . Given two strings  $r$  and  $s$ , `Difference` is defined recursively as:

```
Fixpoint Difference (r s : String) : String :=
match r, s with
| lambda, lambda => lambda
| lambda, s        => s
| r     , lambda   => r
| S r' , S s'    => Difference r' s'
end.
```

We say that a string  $r$  has smaller *surfeit* than a string  $s$  if the string  $r$  is shorter, in terms of number of lambdas, than the string  $s$ . Given two strings  $r$  and  $s$ , the `Surfeit` function is defined recursively, and returns `true` if the string  $r$  has less lambdas than the string  $s$ . Applied to descriptionss, we prefer those with the smaller number of lamdas.

```
Fixpoint Surfeit (r s : String) : bool :=
match r, s with
| lambda, lambda => false
| lambda, _       => true
| _, lambda      => false
| S r' , S s'   => Surfeit r' s'
end.
```

The *inaccuracy* of a string  $r$  with respect to a target string  $t$  is based on the absolute difference between the two strings, that is, how close is the string to the target. Given two strings  $r$  and  $s$ , and a target string  $t$ , the `Inaccuracy` function returns `true` if the absolute difference between the strings  $r$  and  $t$  contains less lambdas than the absolute difference between the strings  $s$  and  $t$ . Inaccuracy will be applied to the ouput of the models.

---

<sup>2</sup>Do not confuse the  $\lambda$  symbol with a  $\lambda$ -term.

```
Definition Inaccuracy (r s : String) (t : String) : bool :=
  Surfeit (Difference r t) (Difference s t).
```

*Nescience* will be based on the concepts of surfeit and inaccuracy. Intuitively, we are looking for very short descriptions (low surfeit) and with very low error (low inaccuracy). The *Nescience* function gets as input five strings:  $mr$ ,  $ms$ ,  $r$ ,  $s$  and  $e$ , and returns *true* if  $mr$  is a shorter string than  $ms$ , and the inaccuracy of  $r$  is smaller than the inaccuracy of  $s$  with respect to  $e$ . Intuitively,  $mr$  and  $ms$  would be a string-based representations of our models (in our case, lambda-terms),  $r$  and  $s$  are the string-based outputs of these models, and  $e$  is a string-based representation of an entity.

```
Definition Nescience (mr ms : String) (r s : String) (e : String) : bool :=
  andb (Surfeit mr ms) (Inaccuracy r s e).
```

In practice, we generally do not know a string-based representation  $e$  of the entity we are interested in, so an indirect approach must be used to study that entity. *MisCoding* is this approach, and it will be introduced in a later section.

Instead of defining the new type *String* we could have reused the concept of natural number, that is, we could have introduced descriptions and representations as numbers. In that case, the *Difference* between two strings  $r$  and  $s$  would be the absolute difference  $|r - s|$  between the numbers  $r$  and  $s$ , and the *Surfeit* could be based on the strict order relation between numbers  $<$ . The concepts of *Inaccuracy* and *Nescience* would have been defined in the same way with numbers. However, natural numbers require additional properties that are not needed for the theory of nescience. For example, the multiplication of two natural numbers does not make any sense in our theory. Reusing natural numbers would have made our theory unnecessarily complex.

In the rest of this section, we will see how we can derive our theory of nescience using only these fundamental concepts. We will also prove the most significant results and derive algorithms for applying the theory in practice.

### Surfeit

*Surfeit* allow us to compare two strings. Recall that the *surfeit* of two strings  $r$  and  $s$  is true if, and only if,  $s$  is composed by more lambdas than  $r$ . *Surfeit* allow us to order the models (string based representations of the models) by its length. We are interested in those models with the lowest possible complexity. In the rest of this section we are going to review the properties of *surfeit*.

The *surfeit* of a string does not increase when we conditions that string

to another one. Intuitively, assuming some previous background knowledge already known as true allow us to reduce the complexity of our models.

```
Proposition SurfeitConditional : forall r s : String,
  Surfeit r (r |c s) = false.

Proof.
intros.
induction r as [| r' IHr'].
- simpl.
  reflexivity.
-
Admitted.
```

From a theoretical point of view, the limit to the process of conditioning would be to assume as true the current model.

```
Proposition SurfeitConditionalLambda : forall r s: String,
  Surfeit (lambda (r |c r)) = false.

Proof.
Admitted.
```

The surfeit of a string does not decrease when it is concatenated with another string. Intuitively, the concatenation of two models increases our unknown. This process of concatenating models can be used as an exploratory mechanism to discover new research topics (previously unknown entities).

```
Proposition SurfeitConcatenation : forall r s : String,
  Surfeit r (r <+> s) = false.

Proof.
intros.
Admitted.
```

From the point of view of surfeit, the operation of concatenation can be distributed over the operation of conditioning, as next proposition shows (**TODO: provide the intuition behind this distributive law**).

```
Proposition DistributiveLawSurfeit : forall r s t : String,
  Surfeit ((r |c s) <+> t) ((r <+> t) |c (s <+> t)) = true.

Proof.
Admitted.
```

## Inaccuracy

### Definition and results about inaccuracy

Inaccuracy does not increase with conditional

```
Proposition InaccuracyConditional : forall r s e : String,
  Inaccuracy r (r |c s) e = false.

Proof.
Admitted.
```

The limit would be conditioning a string to itself

```
Proposition InaccuracyConditionalLambda : forall r s: String,
  Inaccuracy lambda (r |c r) = false.

Proof.
Admitted.
```

### Distributive Law

```
Proposition DistributiveLawInaccuracy : forall r s t : String,
  Inaccuracy ((r |c s) <+> t) ((r <+> t) |c (s <+> t)) = true.

Proof.
Admitted.
```

## Entities, Representations and Descriptions

After we have described the fundamental concepts of the theory of nescience, in this section we are going to introduce the remaining elements of the theory, that is, the concepts of entity, representation and description.

We introduce the concept of *entity* recursively as a finite list of strings. Those strings correspond to the valid representations of the entity. We consider that the empty entity, denoted by *nil*, is also an entity.

```
Inductive Entity : Set :=
| nil : Entity
| add_repr : String -> Entity -> Entity.
```

A *universe* is a finite list of entities. A universe correspond to the particular research area in which we are interested. We consider that the empty universe, denoted by *empty*, is also a universe. The recursive definition of universe is given by:

```
Inductive Universe : Set :=
| empty : Universe
| add_entity : Entity -> Universe -> Universe.
```

A description is a pair composed by a recursive function (a  $\lambda$ -term) from strings to strings, what we call a *model*, and an input string to that model.

```
Definition Description (model : String -> String) (input : String) : String :=
  model input.
```

## Miscoding

### Definition and results about miscoding

## Properties of Strings

Before we move into the details of the theory of nescience, we are going to prove some basic properties that our concept of string satisfies. Also, besides to the already introduced concept of string difference, we are going to introduce two other strings operators: concatenation and conditional.

We say that two strings are *equal* if, and only if, they have the same number of sigmas. String equality is defined recursively as follows:

```
Fixpoint Equality (r s : String) : bool :=
  match r, s with
  | lambda, lambda => true
  | lambda, _          => false
  | _, lambda          => false
  | S r', S s'         => Equality r' s'
  end.
```

It is convenient to introduce a new infix operator to represent the string equality concept. That will help us to simplify new definitions and proofs.

```
Notation "x =? y" := (Equality x y)
```

In the same way, we introduce another infix operator to represent the concept of string difference.

```
Notation "x <d> y" := (Difference x y)
```

Strings with the difference operator form an abelian group. That is: it has a neutral element, it is commutative, associative and has an inverse element.

The neutral element for strings is the  $\lambda$  symbol. That  $\lambda$  is the neutral element with respect to the operation of difference is a direct consequence of the definitions of string and difference.

```
Proposition DifferenceNeutral : forall r : String,
  r <d> lambda = r /\ lambda <d> r = r.
Proof.
intros.
split.
- destruct r.
  * reflexivity.
  * reflexivity.
- destruct r.
  * reflexivity.
  * reflexivity.
Qed.
```

The property of being commutative of the difference operator is proved by ... **TODO**

```
Proposition DifferenceCommutative : forall r s : String,
  r <d> s = s <d> r.
Proof.
Admitted.
```

In the same way, the property of being commutative is shown by ... **TODO**

```
Proposition DifferenceAsociative : forall r s t : String,
  r <d> (s <d> t) = (r <d> s) <d> t.
Proof.
Admitted.
```

The commutative property can be generalized to finite collections of

strings, such that the use of paraenthesis is not needed ... **TODO**

Given an arbitrary string  $r$ , its inverse element is the string itself, as next proposition shows. The proposition is proved by induction over  $r$ .

```
Proposition DifferenceInverse : forall r : String,
  r <~> r = lambda.

Proof.
intros.
induction r as [| r' IHr'].
- reflexivity.
- simpl.
  apply IHr'.
Qed.
```

The operation of *concatenation* of two strings returns a new string composed by one of the strings appened at the end of the other. The operation of concatenation is defined recursively as:

```
Fixpoint Concatenate (r s : String) : String :=
  match r, s with
  | lambda, _      => s
  | S r' , _       => S (Concatenate r' s)
  end.
```

The infix notation for the concatenation operation is given by:

```
Notation "x <+> y" := (Concatenate x y)
```

The pair composed by strings and the concatenation operation form a commutative monoid. That is, it has a neutral element, and it satisfies the properties of being commutative and associative.

The neutral element of the operation of concatenation is  $\lambda$ . In order to prove that  $\lambda$  is indeed the neutral element we ... **TODO**

```
Lemma s_plus_lambda : forall s : String,
  s <+> lambda = s.

Proof.
intros.
induction s as [| s' IHs'].
- simpl.
  reflexivity.
- simpl.
  rewrite -> IHs'.
  reflexivity.
Qed.
```

Given the above lemma we can now prove that  $\lambda$  is the neutral element ... **TODO**

```
Proposition ConcatenationNeural : forall r : String,
  r <+> lambda = r /\ lambda <+> r = r.

Proof.
Admitted.
```

TODO: string concatenation is commutative, and its proof requires to additional lemmas.

```

Lemma plus_r_Ss : forall r s: String,
  S (r <+> s) = r <+> S s.
Proof.
intros.
induction r as [| r' IHr'].
- simpl.
  reflexivity.
- simpl.
  rewrite -> IHr'.
  reflexivity.
Qed.

```

```

Lemma S_equal : forall n m : String,
  n = m -> S n = S m.
Proof.
intros.
induction m as [| m' IHm'].
- rewrite -> H.
  reflexivity.
- rewrite -> H.
  reflexivity.
Qed.

```

```

Proposition ConcatenationCommutative : forall r s : String,
  r <+> s = s <+> r.
Proof.
intros.
induction s as [| s' IHs'].
- simpl.
  apply s_plus_lambda.
- simpl.
  rewrite <- IHs'.
  rewrite <- plus_r_Ss.
  reflexivity.
Qed.

```

### Associative property of concatenation ...

```

Proposition ConcatenationAsociative : forall r s t : String,
  r <+> (s <+> t) = (r <+> s) <+> t.
Proof.
intros.
induction r as [| r' IHr'].
- simpl.
  reflexivity.
- simpl.
  rewrite <- IHr'.
  reflexivity.
Qed.

```

Introduce the operator of string conditional, as a recursive definition.

```
Fixpoint Conditional (r s : String) : String :=
  match r, s with
  | lambda, _          => lambda
  | s      , lambda => s
  | S r' , S s'  => Conditional r' s
end.
```

### Infix notation.

```
Notation "x |c y" := (Conditional x y).
```

### Algebraic structure of the conditional operator.

#### Left Neutral Element

```
Proposition ConditionalLeftNeutral : forall r : String,
  r |c lambda = r.

Proof.
intros.
destruct r.
- reflexivity.
- reflexivity.
Qed.
```

#### Inverse Element

```
Proposition ConditionalInverse : forall r : String,
  r |c r = lambda.

Proof.
intros.
induction r as [| r' IHr'].
- simpl.
  reflexivity.
- simpl.
Admitted.
```

Having introduced the operations of difference, concatenation and conditional, we can study which of these operators can be distributed over the others, since not all possible combinations are valid. In fact, only three distributive laws are true in our theory of nescience.

The operation of concatenation can be distributed to the operation of conditional, as the next proposition shows. The proposition is proved by ... **TODO**.

```
Proposition DistributiveConditionalConcatenation : forall r s t : String,
  Surfeit ((r |c s) <+> t) ((r <+> t) |c (s <+> t)) = true.

Proof.
Admitted.
```

The concatenation operator can also be distributed to the difference operator. The proposition is proved by ... **TODO**.

```
Proposition DistributiveDifferenceConcatenation : forall r s t : String,
  Surfeit ((r <d> s) <+> t) ((r <+> t) <d> (s <+> t)) = true.
```

Proof.  
Admitted.

Finally, we have a third distributive law, that says that the difference operator can be distributed to the concatenation. Th proposition is proved by ... TODO.

```
Proposition DistributiveConcatenationDifference : forall r s t : String,
  Surfeit ((r <+> s) <d> t) ((r <d> t) <+> (s <d> t)) = true.
Proof.
Admitted.
```

## Equivalence of Axioms

In type theory the underline set  $\mathcal{S}$  is the collection of all finite unary strings.

Surfeit is based on string length.

Inaccuracy is based on the length of the absolute difference between the string an the target.

Nescience is based on lower surfeti and lower inaccuracy

\* strict \* antisymmetric \* transitive

When studying the properties of strings we have shown that the string concatenation operation is commutative, associative and has a neutral element, and that string conditional has a left neutral element and an inverse element.

### A.1.3 Category Theory

TODO: Interpretation of the theory of nescience in terms of category theory.

## A.2 Scientific Method

Next definition formally introduces the concept of scientific methodology in the context of the theory of nescience.

**Definition A.2.1** A *scientific methodology* is an effective procedure that produces a sequence of  $t_1, t_2, \dots, t_n, \dots$  where  $t_i \in \mathcal{D}$  such that  $N(t_i) < N(t_{i+1})$ .

Note that we do not require that in a scientific methodology the collection of  $t_i$  refer to the same entity.

Describe our proposal of scientific method. Short story: based on a exploration/exploitation approach, where the exploration is based in the concept of joint descriptions, and the exploitation in the concept of conditional description.

Compare against another proposals of scientific method (inductive-deductive, hypothetico-deductive, etc) and with other techniques for knowledge discovery and creativity (triz, etc.)

### A.2.1 Science as a Language

**TODO:** Provide an alternative definition of the set of descriptions, and prove that it is equivalent to our definition based on the description function

Since we require computable descriptions, we would like to know if the set of all possible descriptions of a topic is computable as well.

**Proposition A.2.1** Study if  $D_t$  is Turing-decidable, Turing-recognizable or none.

*Proof.* TODO ■

Although we know that it is not computable, we are interested in the set composed by the shortest possible description of each topic.

**Definition A.2.2** We define the set of *perfect descriptions*, denoted by  $\mathcal{D}^*$ , as:

$$\mathcal{D}^* = \{d_t^* : t \in \mathcal{T}\}$$

Since the set  $\mathcal{D}$  includes all the possible descriptions of all the possible (describable) topics, we can see this set as a kind of language for science. We do not call it universal language since it depends on the initial set of entities  $\mathcal{E}$  and the particular encoding used for these entities.

**Proposition A.2.2** The set  $\mathcal{D}$  is not Turing-decidable.

*Proof.* TODO: Because it depends on the set  $\mathcal{T}$  that it could be not computable ■

Say something here

**Proposition A.2.3** The set  $\mathcal{D}$  is Turing-recognizable.

*Proof.* TODO: The same argument as the previous proposition ■

A universal language is determined by a universal Turing machine. Given Two different universal Turing machines  $\delta_a$  and  $\delta_b$  defines two different universal languages  $\mathcal{L}_a$  and  $\mathcal{L}_b$ . Let  $\mathcal{L}_{a=b} = \{\langle d_a, d_b \rangle, d_a, d_b \in \mathcal{D} \mid \delta_a(d_a) = \delta_b(d_b)\}$ .

**Proposition A.2.4** Study if  $\mathcal{L}_{a=b}$  is Turing-decidable, Turing-recognizable or none.

*Proof.* TODO ■

Let  $\mathcal{L}_{\#t}$  the lagauge of valid descriptions (from a formal point of view) that do not describe any topic, that is,  $\mathcal{L}_{\#t} = \{d \in \mathcal{D} \mid \nexists t \in \mathcal{T}, \delta(d) = t\}$ .

**Proposition A.2.5** Study if  $\mathcal{L}_{\#t}$  is Turing-decidable, Turing-recognizable or none.

Let  $\mathcal{L}_{\#d}$  the lagauge topics that are not described by any description, that is,  $\mathcal{L}_{\#d} = \{t \in \mathcal{T} \mid \#d \in \mathcal{D}, \delta(d) = t\}$ .

**Proposition A.2.6** Study if  $\mathcal{L}_{\#d}$  is Turing-decidable, Turing-recognizable or none.

TODO: Extend the concept of conditional description to multiple topics.

TODO: Introduce the concept of "independent topics" based on the complexity of the conditional description. Study its properties.

TODO: Define a topology in  $\mathcal{T}$  and  $\mathcal{D}$ . Study the continuity of  $\delta$ . Study the invariants under  $\delta$ .

### A.3 The Inaccuracy - Surfeit Trade-off

TODO: Explain that given a miscoding, there is a trade-off between inaccuracy and surfeit, in the sense that there exists an optimal point beyond it the more we decrease the inaccuracy, the more we increase the surfeit, and so, the nescience keep constant. Explain how this trade-off imposes a limit to knowledge.

### A.4 Graspness

There are some topics whose nescience decrease with time much faster than others, even if the amount of research effort involved is similar. A possible explaination to this fact is that there are some topics that are inherently more difficult to understand.

We define the *graspness* of a topic as the entropy of the set of possible descriptions of that topic. A high graspness means a difficult to understand topic. Intuitively, a research topic is difficult if it has no descriptions much shorter than the others, that is, descriptions that significantly decrease the nescience. For example, in physics there have been a sucession of theories that produced huge advances in our understanding of how nature works (Aristotelian physics, Newton physics, Einstein physics), meanwhile in case of philosophy of science, new theories (deductivism, inductivism, empirism, falsation, ...) have a similar nescience than previous ones.

**Definition A.4.1 — Graspness.** Let  $D_t$  the set of descriptions of a topic  $t \in T$ . The *graspness* of topic  $t$ , denoted by  $G(t)$ , is defined by:

$$G(t) = \sum_{d \in D_t} 2^{-l(d)} l(d)$$

Grapsness is a positive quantity, whose maximum is reached when all the descriptions of the topic have the same length:

**Proposition A.4.1** Let  $D_t = \{d_1, d_2, \dots, d_q\}$  the set of descriptions of a topic  $t \in T$ , then we have that  $G(t) \leq \log q$ , and  $G(t) = \log q$  if, and only if,  $l(d_1) = l(d_2) = \dots = l(d_q)$ .

*Proof.* Replace  $P(s_i)$  by  $2^{-l(d_i)}$  in Proposition 14.4.1. ■

If  $G(t) = \log q$  then we have that  $N_t = 0$  for all  $\hat{d}_t$ . In that case, it does not make any sense to do research, since no new knowledge can be acquired. This is the case of pseudosciences, like astrology, where the nescience of descriptions does not decrease with time. If fact, graspsness can be used as a distinctive element of what constitutes *science* and what does not.

**Definition A.4.2** A topic  $t \in T$  is *scientable* if  $G(t) < \log d(D_t) - \varepsilon$ , where  $\varepsilon \in \mathbb{R}$  is a constant.

We can extend the concept of graspsness from topics to areas, for example, by means of computing the average graspsness of all the topics included in the area.

**Definition A.4.3** Let  $A \subset T$  a research area. The *graspsness* of area  $A$ , denoted by  $G(A)$ , is defined by:

$$G(A) = \frac{1}{d(A)} \sum_{t \in A} G(t)$$

## A.5 Effort

**TODO:** Provide a characterization of the effort, measured in terms of number of operations, or time, to reduce the nescience. Based in the concept of computational complexity. Provide a physical interpretation in terms of information.

## A.6 Human Understanding

In his landmark paper "*On Computable Numbers with an Application to the Entscheidungsproblem*", where the concept of computable function was proposed for the first time, when the author Alan M. Turing talked about computers, he was thinking about human computers, not machines. In this sense, according to Turing, everything that is computable, can be computed by a human, at least in theory.

The first limitation is about computation time. We expect that a human is able to find a solution to a particular instance of a problem in order of seconds,

maybe minutes or hours, but definitely, in less than a life time, otherwise another human have to start again from scratch to solve the problem. So, given the average computing speed of a human brain, we identify as human solvable problems only those problems with a complexity smaller than a fixed number of steps.

The second limitation is about program size. It might happen that the algorithms to solve a problem is simply too big to fit in our brains. Of course, we could argue that as we saw in Example XX, only X states and Y tape symbols are sufficient to implement a universal Turing machine capable of solving any computable problem, that is, any problem for which exist a Turing machine that can solve it. However, by just following a set of instruction we do not mean that we understand a problem. We understand a problem when we have made the problem for ourselves, in the sense, that the problem is somehow stored in our brain in our own language. So this limits the problem to those whose length, including the necessary background, can be stored in our brain.

We are looking for solutions that minimize at the same time the computational complexity and the Kolmogorov complexity.

**Definition A.6.1** We say that a problem  $P$  is *human solvable* if there exist an algorithm  $TM$  to solve  $P$  such that  $t(n)$  and  $K(P) < l$ .

Given the above definition, we could argue that most of the problems we know are not human solvable, but in fact, they have been solved by human. We use two strategies to deal with too complex problems for our individual brains. The fist strategy is that those time-consuming mechanical parts of the problems are left for computers, so we can reduce the computing time. The second one is that we split the problems into subproblems and each of use specializes in those subproblems.

In this section we are interested in to study the nature of these problems in which this strategy cannot be applied, and so, they are out of reach of individual, nor groups of, humans.

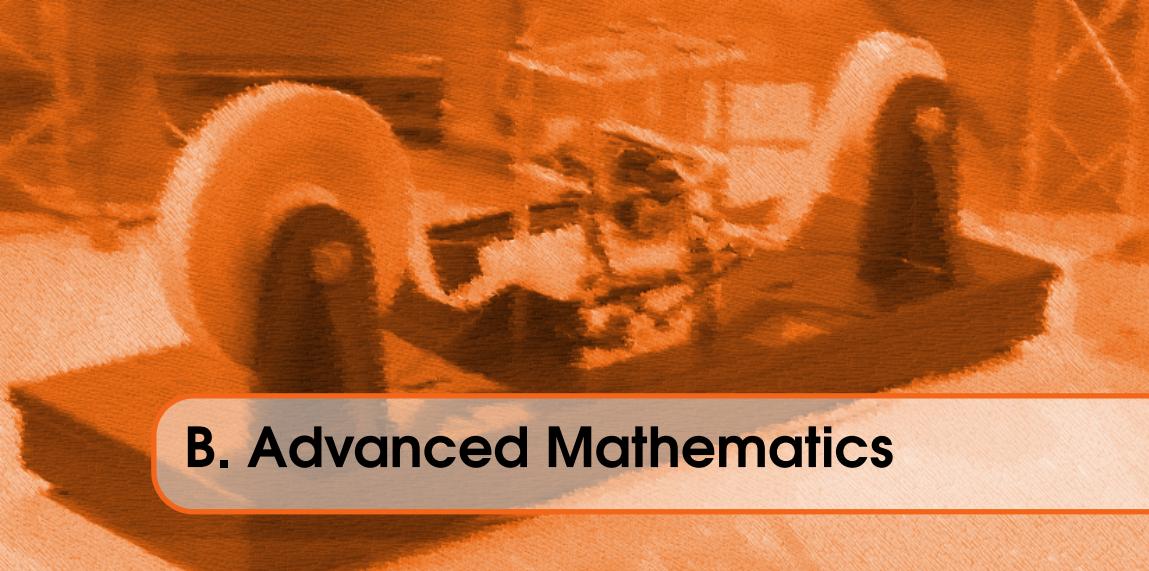
## A.7 Areas in Decay

Provide a model of the decay in the interest of research areas. For example, a good explanatory variable could be the number of interesting questions: the less interesting questions, the more the area is near to its end as a interesting research area, of course, as long as its topics are not used as tools.

## References

Mention the polemic between Hilbert and Fridge given a reference to the book of Mosterin.





## B. Advanced Mathematics

*We are to admit no more causes of natural things  
than such as are both true and sufficient  
to explain their appearances.*

Isaac Newton

TODO: Provide a definition and some basic properties.

$$\log_a x = \frac{1}{\log_b a} \log_b x \quad (\text{B.1})$$

TODO: And introduce the following inequation

$$\ln \frac{1}{x} \geq 1 - x \quad (\text{B.2})$$

with the equality if, and only if,  $x = 1$ .

TODO: Characterization of balanced trees with logs.

TODO: The length of the canonical encoding of integers using logs.

## B.1 Distinctiveness of Metrics

Let's  $\mathbf{X}$  be a dataset composed by  $m$  samples,  $\mathbf{y}$  a response variable that can take two possible values 1 or  $-1$ ,  $\hat{f}$  a model trained using a supervised machine learning algorithm, and  $\hat{\mathbf{y}}$  the vector of predicted values by  $\hat{f}$  when given as input  $\mathbf{X}$  (that is,  $\hat{\mathbf{y}} = \hat{f}(\mathbf{X})$ ). There exists multiple ways to measure the quality of the predicted values  $\hat{\mathbf{y}}$ , like for example accuracy, precision, recall, F1 score, etc. In this technical report we are interested in compare the *distinctiveness* [REF], that is, the capacity to discriminate between multiple candidate  $\hat{\mathbf{y}}$ , of the different error metrics. The closer to the theoretical limit of  $2^m$  possible values, the better is the metric for model training purposes, since the risk of getting stuck in a local minima is smaller [REF].

A *confusion matrix* [REF] is a table that allows to visualize the performance of a trained model. Each row of the matrix represents the number of instances in a predicted class of  $\hat{\mathbf{y}}$  while each column represents the number of instances in an actual class of  $\mathbf{y}$ .

[Fix Table]

|                 |          | Actual Class |          |
|-----------------|----------|--------------|----------|
|                 |          | Positive     | Negative |
| Predicted Class | Positive | $tp$         | $fp$     |
|                 | Negative | $fn$         | $tn$     |

We denote by  $p$  be the number of 1 in  $\mathbf{y}$ , and by  $n$  the number of  $-1$  (we have that  $m = p + n$ ). The following figure depicts a graphical representation of these concepts. The left area of the square correspond to the positive training values, and the right to the negative. The dark gray area represent the values predicted as positive, and the light gray area the values predicted as negative.

(insert figure)

For an easier comparison between metrics we will simplify the distinctiveness of each metric using *big-O* (asymptotic) notation. Recall that a function  $f(a)$  is of order  $\mathcal{O}(g(a))$  if there exists a positive constant  $c$  such that  $0 \leq f(a) \leq cg(a)$ , for sufficiently large  $a$ .

## B.2 Distinctiveness of Metrics

### B.2.1 Accuracy

Accuracy is the ratio between the number of correct predictions and the total number of testing values.

**Definition B.2.1** The accuracy of the predictions  $\hat{y}$  with respect to the training values  $y$  is given by:

$$acc = \frac{tp + tn}{tp + fp + tn + fn}$$

**Proposition B.2.1** The total number of possible values for the metric accuracy is given by:

$$\#acc = m + 1$$

*Proof.* The denominator  $tp + fp + tn + fn$  is fixed to the value  $m$ , meanwhile the numerator could go from no values correctly predicted to all values correctly predicted, and so, accuracy could take one of the following  $m + 1$  possible values:

$$\frac{0}{m}, \frac{1}{m}, \dots, \frac{m}{m}$$

■

**Corollary B.2.2** The order of total number of possible values for the metric accuracy is:

$$\mathcal{O}(m)$$

*Proof.* Given Proposition [prop:Accuracy].

■

## B.2.2 Precision

Precision is the ratio between the correctly predicted positive values and the total number of predicted positive values.

**Definition B.2.2** The precision of the predictions  $\hat{y}$  with respect to the testing values  $y$  is defined as:

$$pre = \frac{tp}{tp + fp}$$

**Proposition B.2.3** The total number of possible values for the metric precision is given by:

$$\#pre \leq (p + 1)(n + 1)$$

*Proof.*  $tp$  can take  $p + 1$  different values (from no true positives to  $p$  true positives), and  $fp$  can take  $n + 1$  different values (from no false positive to  $n$

false positives), being the two values independent of each other. Precision can take at most  $(p+1)(n+1)$  possible values, since some of these fractions can be simplified into a common distinctive value (how many fractions can be simplified depends on the function  $\pi(q)$ , that is, the number of primes less than  $q$ , which is unknown.) ■

**Corollary B.2.4** The order of total number of possible values for the metric precision is:

$$\mathcal{O}(m^2)$$

*Proof.* Given Proposition [prop:Precision]. ■

### B.2.3 Recall

Recall measures the ratio between the number of correctly predicted positive values and the total number of positives.

**Definition B.2.3** The recall of the predictions  $\hat{\mathbf{y}}$  with respect to the testing values  $\mathbf{y}$  is defined as:

$$rec = \frac{tp}{tp + fn}$$

**Proposition B.2.5** The total number of possible values for the metric recall is given by:

$$\#rec = (p+1)$$

*Proof.* Given that

$$rec = \frac{tp}{tp + fn} = \frac{tp}{p}$$

and that  $tp$  can take  $p+1$  different values (from no true positives to all true positives). ■

**Corollary B.2.6** The order of total number of possible values for the metric recall is:

$$\mathcal{O}(m)$$

*Proof.* Given Proposition [prop:Recall]. ■

**B.2.4 F1**

F1 is the harmonic mean between precision and recall.

**Definition B.2.4** The  $F_1$  score of the predictions  $\hat{y}$  with respect to the testing values  $y$  is defined as:

$$F_1 = \frac{2}{\frac{1}{pre} + \frac{1}{rec}}$$

**Proposition B.2.7** The total number of possible values for the metric F1 is given by:

$$\#F1 \leq (n+1) \frac{(p+2)(p+1)}{2}$$

*Proof.* It is a well known property that

$$F_1 = \frac{2}{\frac{1}{pre} + \frac{1}{rec}} = \frac{2 \times pre \times rec}{pre + rec} = \frac{2tp}{2tp + fp + fn}$$

$tp$  can take  $p+1$  different values (from no true positives to  $p$  true positives), and  $fp$  can take  $n+1$  different values (from no false positive to  $n$  false positives), being these two values independent. Fixed a  $tp$  the number of  $fn$  is  $p - tp + 1$ , so the total number of  $tp$  and  $fn$  is given by:

$$\sum_{tp=0}^p (p - tp + 1) = p(p+1) - \frac{p(p+1)}{2} + (p+1) = \frac{2p(p+1) - p(p+1) + 2(p+1)}{2}$$

Then,  $F_1$  can take at most  $(n+1) \frac{(p+2)(p+1)}{2}$  possible values, since some of these fractions can be simplified into a common distinctive value. ■

**Corollary B.2.8** Corollary 12. The order of total number of possible values for the metric F1 is:

$$\mathcal{O}(m^3)$$

*Proof.* Given Proposition [prop:F1]. ■

**B.2.5 Area under the ROC curve**

The area under the Receiver Operating Characteristic (AUC) measures how well our model differentiates between the distributions of the two classes.

**Definition B.2.5** The area under the Receiver Operating Characteristic (AUC) score of the predictions  $\hat{y}$  with respect to the testing values  $y$  is defined as:

$$F_1 = \frac{2}{\frac{1}{pre} + \frac{1}{rec}}$$

**Proposition B.2.9** The total number of possible values for the metric F1 is given by:

$$\#F1 \leq (n+1) \frac{(p+2)(p+1)}{2}$$

*Proof.* The area under the Receiver Operating Characteristic (AUC) for the binary case can be computed as:

$$AUC = \frac{\sum_{i=1}^p (r_i - 1)}{pn}$$

The different values for the numerator  $\sum_{i=1}^p (r_i - 1)$  correspond to the number of possible ways in which the p values could appear in the testing dataset, that is given by

$$\binom{m}{p} = \frac{m!}{n!p!} = \frac{m(m-1)\cdots(m-p+1)}{p(p-1)\cdots 1}.$$

The denominator is the constant  $m$ . ■

**Corollary B.2.10** Corollary 15. The order of total number of possible values for the metric AUC is:

$$\mathcal{O}(AUC) = m^{(m/2)}$$

*Proof.* Given Proposition [prop:AUC]. ■

### Kolmogorov Inaccuracy

Kolmogorov inaccuracy is a measure of the distance between the predicted values and the real values.

**Definition B.2.6** The Kolmogorov inaccuracy of the predictions  $\hat{y}$  with respect to the testing values  $y$  is defined as:

$$kol = NCD(y, \hat{y})$$

**Proposition B.2.11** The total number of possible values for the metric Kolmogorov inaccuracy is given by:

$$\#kol \leq \frac{(p+2)(p+1)(n+2)(n+1)}{2}$$

*Proof.* Proof. The normalized compression distance between two vectors  $\mathbf{y}$  and  $\hat{\mathbf{y}}$  is given by

$$NCD(\mathbf{y}, \hat{\mathbf{y}}) = \frac{\hat{K}_C(\hat{\mathbf{y}}, \mathbf{y}) - \min\{\hat{K}_C(\hat{\mathbf{y}}), \hat{K}_C(\mathbf{y})\}}{\max\{\hat{K}_C(\hat{\mathbf{y}}), \hat{K}_C(\mathbf{y})\}}$$

using as compressor a code with optimal length, the Kolmogorov inaccuracy is given by:

$$\frac{-tp \log_2 \frac{tp}{m} - fp \log_2 \frac{fp}{m} - tn \log_2 \frac{tn}{m} - fn \log_2 \frac{fn}{m}}{\max\{-(tp + fp) \log_2 \frac{(tp+fp)}{m} - (tn + fn) \log_2 \frac{(tn+fn)}{m}, -p \log_2 \frac{p}{m} - n \log_2 \frac{n}{m}\}} - \frac{\min\{ -$$

$tp$  can take  $p + 1$  different values (from no true positives to  $p$  true positives), and  $tn$  can take  $n + 1$  different values (from no true negative to  $n$  true negatives), being this two values independent. Fixed a  $tp$  the number of  $fn$  is  $p - tp + 1$ , so the total number of  $tp$  and  $fn$  is given by:

$$\sum_{tp=0}^p (p - tp + 1) = p(p + 1) - \frac{p(p + 1)}{2} + (p + 1) = \frac{2p(p + 1) - p(p + 1) + 2(p + 1)}{2}$$

and fixed a  $tn$  the number of  $fp$  is  $n - tn + 1$ , so the total number of  $tn$  and  $fp$  is given by:

$$\sum_{tn=0}^n (n - fp + 1) = n(n + 1) - \frac{n(n + 1)}{2} + (n + 1) = \frac{2n(n + 1) - n(n + 1) + 2(n + 1)}{2}$$

Combining both expression, and taking into account that some of these fractions can be simplified into a common distinctive value, we get the desired result. ■

**Corollary B.2.12** The order of total number of possible values for the metric Kolmogorov Accuracy is:

$$\mathcal{O}(m^4)$$

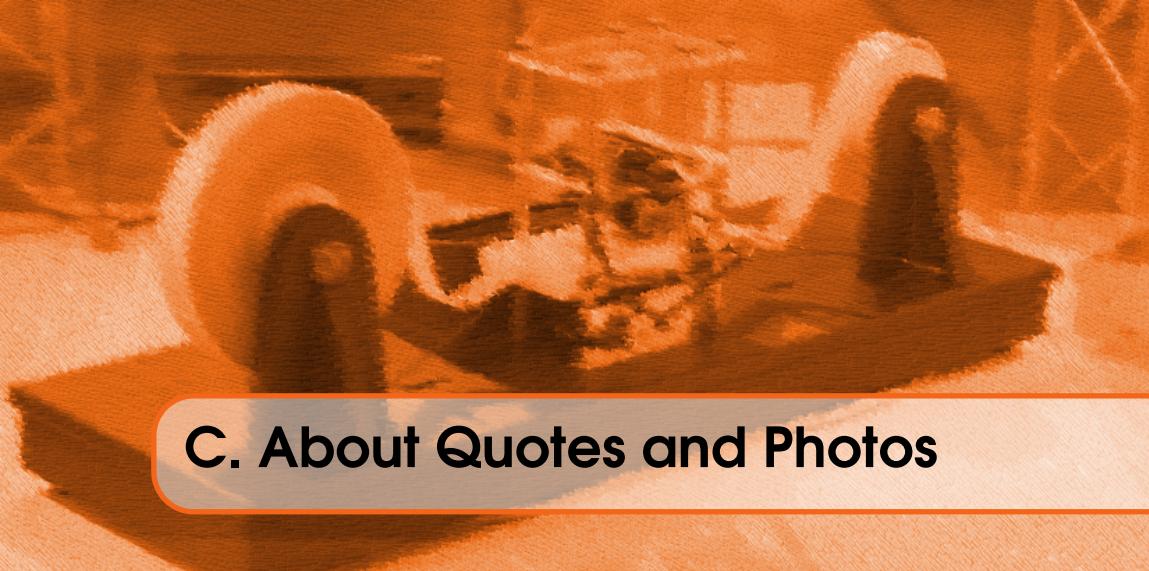
*Proof.* Given Proposition [prop:Accuracy]. ■

### Conclusion

In this technical report we have derived exact formulas for the distinctiveness of the most common metrics used in binary classification algorithms, and the novel metric of Kolmogorov inaccuracy. The highest value of distinctiveness is achieved with the Kolmogorov inaccuracy, followed by the F1 metric. The lowest values are for accuracy and recall.

Perhaps, an easier way to compare the different metrics in terms of distinctiveness would be to use asymptotic notation. In this sense, accuracy and recall will have a distinctiveness of  $\mathcal{O}(m)$ , precision of  $\mathcal{O}(m^2)$ , F1 of  $\mathcal{O}(m^3)$ , and Kolmogorov inaccuracy of  $\mathcal{O}(m^4)$ . This result is in line with the intuition, since accuracy and recall depend on only one parameter, precision on two, F1 on three, and Kolmogorov inaccuracy on four.

### **B.3 Statistical Significance of Analysis**



## C. About Quotes and Photos

*What we know is little,  
combined with tenacious concentration on a subject  
and what we are ignorant of is immense.*

Pierre-Simon Laplace

TODO: Explain why are important quotes and photos

### C.0.1 Quotes

Introduce this section.

*Perfection is achieved not when there is nothing more to add, but when there is nothing left to take away.* Antoine de Saint-Exupéry.

This is a critical idea in the theory of nescience, although there are some differences. I think Saint-Exupéry is talking more about what we have called redundancy, rather than the concept of surfeit. Moreover, Saint-Exupéry is true as long as the inaccuracy is zero.

*Computers are useless, they can only give you answers.* Pablo Picasso.  
As I have said in the Preface of the book, this quote triggered everything.

*If presented with a choice between indifferent alternatives, then one ought to select the simplest one.* Occam's razor principle.

I am sure that many people will claim that the theory of nescience is just Occam on steroids. By I think there are big differences, as I have already described in the book.

*Mathematics may be defined as the subject in which we never know what we are talking about, nor whether what we are saying is true.* Bertrand Russell.

Maybe I can talk here about Hilbert-Frege controversy, and what Russell means with this quote.

*Sometimes it's the people no one imagines anything of who do the things that no one can imagine.* Alan Turing.

Hopefully Turing is talking about me :)

*Information is the resolution of uncertainty.* Claude Shannon.

TODO: Explain

*Some mathematical statements are true for no reason, they're true by accident.* Gregory Chaitin.

TODO: Explain

*All great work is the fruit of patience and perseverance, combined with tenacious concentration on a subject over a period of months or years.* Santiago Ramón y Cajal.

Mention the book of Cajal.

*To go where you don't know, you have to go the way you don't know.* San Juan de la Cruz

TODO: Explain

*We are all agreed that your theory is crazy. The question which divides us is whether it is crazy enough.* Niels Bohr.

TODO: Explain

*Wanderer, there is no road, the road is made by walking.* Antonio Machado.

TODO: Explain

*A little inaccuracy sometimes saves tons of explanations.* Saki.

TODO: Explain

*Everything should be made as simple as possible, but not simpler.* Albert Einstein

**TODO:** Explain

*There are known knowns. These are things we know that we know. There are known unknowns. That is to say, there are things that we know we don't know. But there are also unknown unknowns. There are things we don't know we don't know.* Donald Rumsfeld.

**TODO:** Explain

*It is not the answer that enlightens, but the question.* Eugène Ionesco.

**TODO:** Explain

*Invert, always invert.* Carl Gustav Jacob Jacobi.

**TODO:** Explain

*Always look for tricks.* Antonio García.

**TODO:** Explain

*Anyone who regards games simply as games and takes work too seriously has grasped little of either.* Heinrich Heine.

**TODO:** Explain

*Science may be regarded as the art of data compression.* Li & Vitányi.

**TODO:** Explain

*To be surprised, to wonder, is to begin to understand.* José Ortega y Gasset.

**TODO:** Explain

*We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances.* Isaac Newton

**TODO:** Explain

*What we know is little, and what we are ignorant of is immense.* Pierre-Simon Laplace

**TODO:** Explain

*When academics encounter a new idea that doesn't conform to their preconceptions, there's often a sequence of three reactions: first dismiss, then reject, and finally declare it obvious.* S. Sloman and P. Fernbach.

**TODO:** Explain

## C.0.2 Photos

In this Appendix we explain the intended meaning of the photographs included at the beginning of each chapter. All the photographs are royalty-free (or at least this is what Google Images says). Photographs have been pre-processed with GIMP<sup>1</sup>, the GNU image manipulation program: we have applied a dotify filter (Artistic filter, GIMPressionist), and then altered the color map (Colors, Colorize) with a Hue/Saturation/Lightness levels of 24/84/10.

### The Torch Bearers



The Torch Bearers is an aluminum sculpture created by the American artist Anny Hyatt Huntington, and donated to the city of Madrid in 1955. The sculpture is currently located at Universidad Complutense campus. The sculpture represents an old dying man that before to die passes the torch (a symbol of knowledge) to a young riding man that will continue the quest of perfect knowledge. The artist created other copies in bronze of the same sculpture that are located in Valencia, La Habana, and several cultural organizations around the United States.

### Ancient Greek Philosophers



The carved busts of Greek philosophers Socrates, Antisthenes, Chrysippus, and Epicurus, located at the British Museum in London. The ideas and achievements of the ancient Greeks philosophers changed their world and had a huge influence in Western culture. Philosophers like Socrates, Plato and Aristotle formulated the first scientific explanations about how the world worked, and they pioneer a new way of thinking, based on reason and rational thought. The scientific explanation of how the Universe works formulated by Aristotle was accepted as true during more than two thousands years.

### Ars Magna

Ars Generalis Ultima or Ars Magna (The Ultimate General Art) was a book published by the Spanish philosopher Raimundo Lulio in 1305. The book contained the description of a mechanical device capable of answering any argument or question about Christian beliefs by means of using logic and reason. The machine operated by rotating a collection of concentrically arranged circles to combine a fixed set of fundamental concepts. In this way,

---

<sup>1</sup>[www.gimp.org](http://www.gimp.org)

the device could show all possible truths about the subject of inquiry. The method was an early attempt to use logical means to produce new knowledge, and it was the inspiration for the methodology of finding interesting questions described in this book.

### **Galileo's Telescope**



Galileo Galilei was an Italian astronomer, physicist, engineer, philosopher and mathematician. Galileo was the first scientist to clearly state the usefulness of mathematics to discover the laws of nature. Galileo also played a major role in the scientific revolution of the seventeenth century, introducing

important innovations in the scientific method. In 1610, Galileo built a telescope and looked up at the heavens. His discoveries revolutionized the field of astronomy and changed our understanding of the Universe.

### **Königsberg Bridges**



The old city of Königsberg in Prussia (now Kaliningrad, Russia) was laid on both sides of the Pregel river. The city had two large islands which were connected to each other and to mainland by seven bridges. The seven bridges problem is a classical problem in mathematics that asks to devise a walk

that would cross each of the seven bridges once and only once, starting and ending at any point, not necessarily the same. The Swiss mathematician Leonhard Euler proved in 1736 that the problem has no solution. The work of Euler laid the foundations of a new mathematical discipline: Graph Theory.

### **The Turing Machine**



In 1936, the British mathematician Alan Turing proposed a formal model for a hypothetical machine and claimed that his machine could compute anything that humans could compute following an algorithm. The model was a highly convincing one, and simple enough to allow precise mathematical analysis. In fact, the model had many of the ideas that ten years later electrical engineers used to build real computers. The Turing machine was one of this few moments in the history of science in which theory preceded practice.

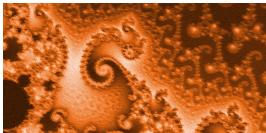
## Morse key



A switching device used to send messages along a wire. The system sends pulses of electric current using the Morse code. Morse code is named after Samuel F. B. Morse, the inventor of the telegraph. The code is composed by a standardized sequence of short and long signals called *dots* and *dashes*,

although it is not a binary code, since the code alphabet also includes symbols for the separation between letters and words. The average bit length per character for the English language is 2.53, a remarkable result, given the fact that the code was designed intuitively, without knowing any of the (later discovered) results of coding theory.

## Mandelbrot Set



The Mandelbrot set is created by sampling the complex numbers, and determining for each sample whether the result of iterating a function goes to infinity. Treating the real and imaginary parts of each number as image coordinates, pixels are colored according to how rapidly the sequence diverges, with black used for points where the sequence does not diverge. Images of the Mandelbrot set exhibit an elaborate boundary that reveals progressively ever-finer, self-similar, recursive detail at increasing magnifications. However, according to Kolmogorov complexity, the set presents a very low complexity.

## XXX: XXX

Xxx xxxx x xxx xx xxxxxx

## Athena's Owl



A silver coin depicting the owl that traditionally accompanies Athena. Athena is the virgin goddess of wisdom in Greek mythology. The owl has been used as a symbol of knowledge, wisdom, perspicacity and erudition throughout the Western world, perhaps because their ability to see in the dark. The

German philosopher Hegel famously noted that "the owl of [Athena] spreads its wings only with the falling of the dusk", meaning that philosophy comes to understand a historical condition just as it passes away. In this sense, Hegel asserts that Philosophy cannot be prescriptive because it understands only in hindsight.

## The Thinker



The Thinker is a bronze sculpture created by the French artist Auguste Rodin. The sculpture represents a nude male figure sitting on a rock with his chin resting on one hand. Originally created as part of a larger composition (The Gates of Hell), later the artist decided to treat the figure as an independent work, and at a larger size. There are about 28 full size castings located in museums and public places all around the world (Geneva, Brussels, San Francisco, New York, Buenos Aires, etc.), and many others at different scales.

A common interpretation of the sculpture is as an image of the deep thoughts required to find the right questions in philosophy.

## R.U.R.



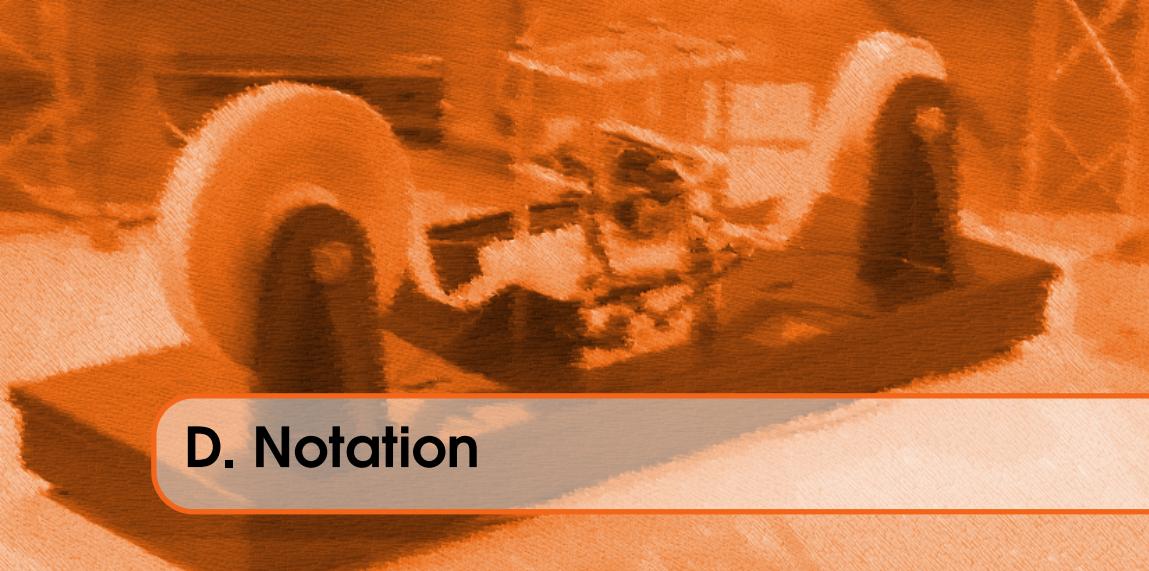
R.U.R. (Rossum Universal Robots) was a highly successful science fiction play written by the Czech Karel Čapek in 1920. The play introduced for the first time the word 'robot' as an alternative to other words used at that time like 'automaton' or 'android'. The word 'robot' derives from the Czech 'robita' meaning "forced labor of the kind that serfs had to perform on their masters' lands". The drama occurs in R.U.R., a factory that makes intelligent robots from artificial flesh and bones, so perfect that they can be mistaken for humans (the name Rossum is derived from the Czech word 'rozum' that means 'reason'). At the beginning robots were happy to work for humans, but then a rebellion starts and all the humans are murdered. At the end of the plot, robots realize that they do not have the knowledge to make new robots, and that by exterminating humans they have triggered their own extinction. The play is considered as a tragic satire about a naive humankind, the dangers of technology, and the obsolescence of God.

## Wikipedia Monument



The Wikipedia Monument is located in the city of Ślubice, Poland. The statue was designed by Armenian sculptor Mihran Hakobyan, and it was unveiled on October 2014, becoming the world's first monument to the online encyclopedia. The inscription reads: "With this monument the citizens of Ślubice would like to pay homage to thousands of anonymous editors all over the world, who have contributed voluntarily to the creation of Wikipedia,

the greatest project co-created by people regardless of political, religious or cultural borders. In the year this monument is unveiled Wikipedia is available in more than 280 languages and contains about 30 million articles. The benefactors behind this monument feel certain that with Wikipedia as one of its pillars the knowledge society will be able to contribute to the sustainable development of our civilization, social justice and peace among nations."



## D. Notation

*When academics encounter a new idea that doesn't conform to their preconceptions,*

*there's often a sequence of three reactions:  
first dismiss, then reject, and finally declare it obvious.*

S. Sloman and P. Fernbach

$\mathbb{N}$  set of natural numbers (including 0)

$\mathbb{Z}$  set or integers

$\mathbb{Z}^+$  set or positive integers

$\mathbb{Q}$  set of rational numbers

$\mathbb{R}$  set of real numbers

$\mathbb{R}^+$  set of positive real numbers

$x \in A$   $x$  is a member of  $A$

: set formation

$A \subset B$   $A$  is a subset of  $B$

$A \subseteq B$   $A$  is a subset or equal to  $B$

$\emptyset$  empty set

$d(A)$  cardinality of  $A$

$A \cup B$  union of  $A$  and  $B$

$A \cap B$  intersection of  $A$  and  $B$

|                     |  |
|---------------------|--|
| $A \setminus B$     | set difference                                       |
| $\bar{A}$           | complement of $A$                                    |
| $\mathcal{P}(A)$    | power set  |
| $(x, y)$            | ordered pair   |
| $A \times B$        | cartesian product                                    |
| $A^n$               | $n$ -fold cartesian product                          |
| $R$                 | binary relation                                      |
| $\leq$              | total order  |
| $\max(A)$           | maximum  |
| $\min(A)$           | minimum  |
| $f(x) = y$          | function   |
| $f(x) = \infty$     | undefined element                                    |
| $I_A$               | identity   |
| $f^{-1}$            | inverse function                                     |
| $f \circ g$         | composition  |
| $1_A$               | characteristic function                              |
| $\text{abs}(x)$     | absolute value                                       |
| $\lceil x \rceil$   | ceil   |
| $\lfloor x \rfloor$ | floor  |
| $l(s)$              | length of string                                     |
| $\lambda$           | empty string   |
| $s^R$               | reverse string                                       |
| $\mathcal{S}^n$     | set of strings of length n                           |
| $\mathcal{S}^+$     | set of all finite strings                            |
| $\mathcal{S}^*$     | set of all finite strings including the empty string |
| $<_p$               | prefix   |
| $\bar{s}$           | Self delimited string                                |
| $\langle O \rangle$ | Encoding as string of $O$                            |
| $G = (V, E)$        | graph  |
| $\deg(v)$           | degree of a vertex                                   |
| $N(v)$              | neighborhood of a vertex                             |
| $\text{indeg}(v)$   | in-degree of a vertex                                |
| $\text{outdeg}(v)$  | out-degree of a vertex                               |
| $\Omega$            | sample space   |
| $P(x)$              | probability of $x$                                   |
| $E(X)$              | expectation of $X$                                   |
| $T$                 | Turing machine                                       |
| $Q$                 | set of states  |
| $\Gamma$            | set of tape symbols                                  |
| $\sqcup$            | blank symbol   |

---

|                             |  |
|-----------------------------|--|
| $\Sigma$                    | input symbols  |
| $q_o$                       | initial state  |
| $q_f$                       | final state  |
| $\tau$                      | transition function  |
| $C$                         | configuration  |
| $C_o$                       | initial configuration  |
| $C_f$                       | final configuration  |
| $U$                         | universal Turing machine   |
| <br>                        |  |
| $\mathcal{E}$               | Set of entities  |
| $\mathcal{R}$               | Set of representations   |
| $\mathcal{R}_{\mathcal{E}}$ | Set of representations of $\mathcal{E}$                                |
| $t \in \mathcal{T}$         | Research topic   |
| $d \in \mathcal{D}_t$       | Description of a topic   |
| $\mathcal{D}$               | Set of descriptions  |
| $\mathcal{D}_{\mathcal{T}}$ | Set of valid descriptions of $\mathcal{T}$                             |
| $\mathcal{D}_t$             | Set of descriptions of $t \in \mathcal{T}$                             |
| $\delta$                    | Description function   |
| $d_t^*$                     | Perfect description of $t \in \mathcal{T}$                             |
| $d_{t,s}$                   | Joint description of $t, s \in \mathcal{T}$                            |
| $\mathcal{D}_{t,s}$         | Set of joint descriptions of $t, s \in \mathcal{T}$                    |
| $d_{t,s}^*$                 | Perfect joint description of $t, s \in \mathcal{T}$                    |
| $d_{t s^*}$                 | Conditional description of $t$ given $s, t, s \in \mathcal{T}$         |
| $\mathcal{D}_{t s^*}$       | Set of conditional descriptions of $t$ given $s, t, s \in \mathcal{T}$ |
| $d_{t s^*}^*$               | Perfect conditional description of $t$ given $s, t, s \in \mathcal{T}$ |
| $A \subset \mathcal{T}$     | Research area  |
| $\hat{A}$                   | Know subset of the area $A \subset \mathcal{T}$                        |
| $\mathcal{D}_{\hat{A}}$     | Description of the area $A$ given the known subset $\hat{A}$           |
| $\mathcal{D}_{\hat{A}}$     | Set of descriptions of the area $A$ given the known subset $\hat{A}$   |
| $d_{\hat{A}}^*$             | Perfect description of the area $A$ given the known subset $\hat{A}$   |
| <br>                        |  |
| $RG$                        | relevance graph  |
| $R_t$                       | relevance of topic $t$   |
| $IP_t$                      | interestingness of topic $t$ as a problem                              |
| $M_t$                       | maturity of topic $t$  |
| $AG$                        | applicability graph  |
| $A_t$                       | applicability of topic $t$   |
| $IT_t$                      | interestingness of topic $t$ as a tool                                 |
| $T'$                        | set of known topics  |
| $Q_{t_1 \rightarrow t_2}$   | interesting question   |
| $IQ_{t_1 \rightarrow t_2}$  | interestingness of question $Q_{t_1 \rightarrow t_2}$                  |
| $\mathbb{F}$                | unknown frontier   |

$\mathbb{S}$  new topics area

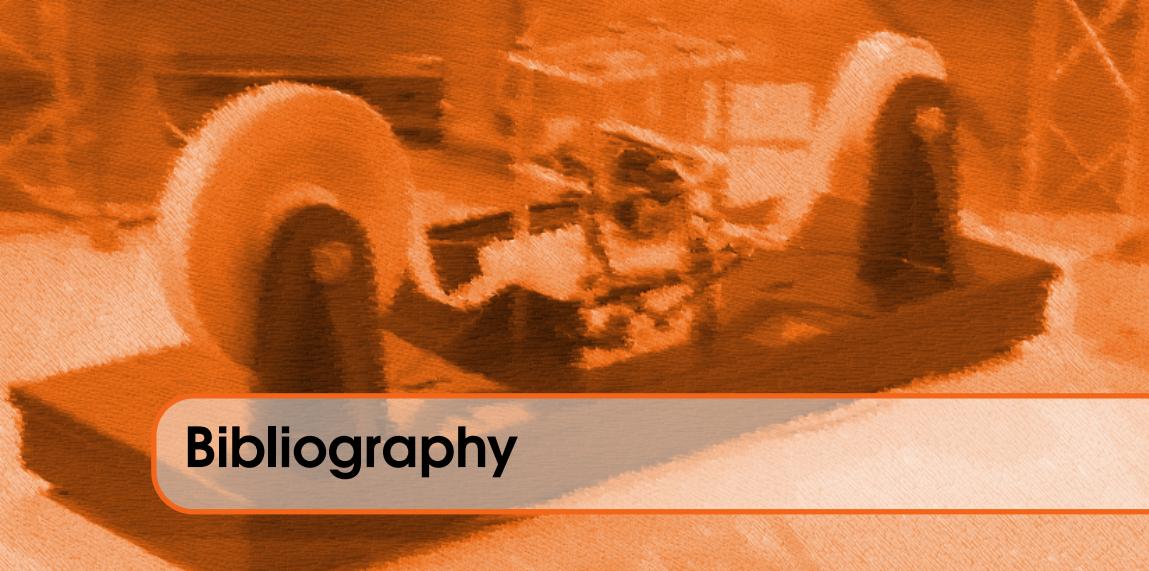
$S_{\{t_1, t_2\}}$  new topic

$IS_{\{t_1, t_2\}}$  interestingness of a new topic

$IT_A$  interestingess of an area as tool

$IP_A$  interestingess of an area as problem

$\hat{t}(\hat{\mathbf{y}}, \mathbf{y})$  inaccuracy of predicted values



# Bibliography

## Books

- [Abr63] Norman Abramson. *Information theory and coding*. 1963 (cited on pages 118, 377).
- [Cal02] Cristian S Calude. *Information and randomness: an algorithmic perspective*. Springer Science & Business Media, 2002 (cited on page 394).
- [Cha13] Alan F Chalmers. *What is this thing called science?* Hackett Publishing, 2013 (cited on page 48).
- [Chi13] Timothy Childers. *Philosophy and probability*. Oxford University Press, USA, 2013 (cited on page 334).
- [Coo03] S Barry Cooper. *Computability theory*. CRC Press, 2003 (cited on page 353).
- [Cor+90] Thomas H Cormen et al. *Introduction to algorithms mit press*. MIT Press, 1990 (cited on page 279).
- [CT12] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012 (cited on pages 48, 377, 394).
- [DeG+86] Morris H Morris H DeGroot et al. *Probability and statistics*. 04; QA273, D4 1986. 1986 (cited on page 334).

- [Fer09] Maribel Fernández. *Models of Computation: An Introduction to Computability Theory*. Springer Science & Business Media, 2009 (cited on page 353).
- [GG12] Allen Gersho and Robert M Gray. *Vector quantization and signal compression*. Volume 159. Springer Science & Business Media, 2012 (cited on page 377).
- [GV13] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013 (cited on page 279).
- [Grü07] Peter D Grünwald. *The minimum description length principle*. MIT press, 2007 (cited on page 209).
- [HH10] Ifan Hughes and Thomas Hase. *Measurements and their uncertainties: a practical guide to modern error analysis*. OUP Oxford, 2010 (cited on page 104).
- [Jec13] Thomas Jech. *Set theory*. Springer Science & Business Media, 2013 (cited on page 78).
- [Joh09] Richard Johnsonbaugh. *Discrete mathematics*. Pearson, 2009 (cited on page 279).
- [LV13] Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications*. Springer Science & Business Media, 2013 (cited on pages 48, 78, 394).
- [Mie12] Kaisa Miettinen. *Nonlinear multiobjective optimization*. Volume 12. Springer Science & Business Media, 2012 (cited on page 436).
- [Pop14] Karl Popper. *Conjectures and refutations: The growth of scientific knowledge*. routledge, 2014 (cited on page 105).
- [Rob15] Borut Robič. *The foundations of computability theory*. Springer, 2015 (cited on page 78).
- [Sip12] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2012 (cited on pages 48, 279, 353).
- [Soa16] Robert I Soare. *Turing computability*. Springer, 2016 (cited on page 353).
- [Van80] Bas C Van Fraassen. *The scientific image*. Oxford University Press, 1980 (cited on page 91).
- [Wal05] Christopher S Wallace. *Statistical and inductive inference by minimum message length*. Springer Science & Business Media, 2005 (cited on pages 209, 436).

---

## Articles

- [CAO+05] Manuel Cebrián, Manuel Alfonseca, Alfonso Ortega, et al. “Common pitfalls using the normalized compression distance: What to watch out for in a compressor”. In: *Communications in Information & Systems* 5.4 (2005), pages 367–384 (cited on page 242).
- [Cha69] Gregory J Chaitin. “On the simplicity and speed of programs for computing infinite sets of natural numbers”. In: *Journal of the ACM (JACM)* 16.3 (1969), pages 407–422 (cited on page 394).
- [Cha95] Gregory J Chaitin. “The berry paradox”. In: *Complexity* 1.1 (1995), pages 26–30 (cited on page 78).
- [Gar+13] Salvador Garcia et al. “A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 25.4 (2013), pages 734–750 (cited on page 209).
- [Göd31] Kurt Gödel. “Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I”. In: *Monatshefte für mathematik und physik* 38.1 (1931), pages 173–198 (cited on page 78).
- [Hua13] Keguo Huang. “Three hundred years of the St. Petersburg paradox”. In: (2013) (cited on page 334).
- [Ioa05] John PA Ioannidis. “Why most published research findings are false”. In: *PLoS medicine* 2.8 (2005), e124 (cited on page 91).
- [Kol65] Andrei N Kolmogorov. “Three approaches to the quantitative definition of information”. In: *Problems of information transmission* 1.1 (1965), pages 1–7 (cited on page 394).
- [LSW13] Bruno Latour, Jonas Salk, and Steve Woolgar. “Laboratory life: The construction of scientific facts”. In: (2013) (cited on page 91).
- [Li+04] Ming Li et al. “The similarity metric”. In: *IEEE transactions on Information Theory* 50.12 (2004), pages 3250–3264 (cited on page 242).
- [Llo82] Stuart Lloyd. “Least squares quantization in PCM”. In: *IEEE transactions on information theory* 28.2 (1982), pages 129–137 (cited on page 377).

- [McM56] Brockway McMillan. “Two inequalities implied by unique decipherability”. In: *IRE Transactions on Information Theory* 2.4 (1956), pages 115–116 (cited on page 377).
- [Pos46] Emil L Post. “A variant of a recursively unsolvable problem”. In: *Bulletin of the American Mathematical Society* 52.4 (1946), pages 264–268 (cited on page 353).
- [QR89] J Ross Quinlan and Ronald L Rivest. “Inferring decision trees using the minimum description lenght principle”. In: *Information and computation* 80.3 (1989), pages 227–248 (cited on page 209).
- [Shm+10] Galit Shmueli et al. “To explain or to predict?” In: *Statistical science* 25.3 (2010), pages 289–310 (cited on page 209).
- [Sol64] Ray J Solomonoff. “A formal theory of inductive inference. Part I and II”. In: *Information and control* 7.1 (1964), pages 1–22 (cited on page 394).
- [Sup02] Patrick Suppes. “Representation and invariance of scientific structures”. In: (2002) (cited on page 90).
- [TAY22] JR TAYLOR. “An introduction to error analysis: The study of uncertainties in physical measurements (3rd ed. edition).” In: (2022) (cited on page 104).
- [Tur36] Alan Mathison Turing. “On computable numbers, with an application to the Entscheidungsproblem”. In: *J. of Math* 58.345–363 (1936), page 5 (cited on page 353).
- [Tur39] Alan Mathison Turing. “Systems of logic based on ordinals”. In: *Proceedings of the London Mathematical Society* 2.1 (1939), pages 161–228 (cited on page 78).
- [WB68] Chris S Wallace and David M Boulton. “An information measure for classification”. In: *The Computer Journal* 11.2 (1968), pages 185–194 (cited on page 436).
- [WP93] Chris S Wallace and JD Patrick. “Coding decision trees”. In: *Machine Learning* 11.1 (1993), pages 7–22 (cited on page 209).
- [YW09] Ying Yang and Geoffrey I Webb. “Discretization for naive-Bayes learning: managing discretization bias and variance”. In: *Machine learning* 74.1 (2009), pages 39–74 (cited on page 209).



# Index

- $P \stackrel{?}{=} NP$ , 337  
 $\sigma$ -algebra, 285
- Abductive reasoning, 451  
Absolute frequency, 413  
Absolute value, 268  
Abstract entities, 442  
Adjacency matrix, 276  
Adjacent vertices, 276  
Alphabet, 268  
Analogy, 452  
Ancestors of a vertex, 277  
Anti-realism, 443  
Antisymmetric relation, 266  
Applicability, 147  
Applicability graph, 147  
Applicability of mathematics, 448  
Areas in decay, 154  
Average interestingness of an area, 154  
Axiom of comprehension, 53
- Axiom of Separation, 53  
Axiomatic interpretation of probability, 284
- Backus–Naur form, 271  
Balanced tree, 278  
Bayes theorem, 404  
Bayes' theorem, 290, 304  
Bayesian approach, 461  
Bayesian inference, 404  
Bayesian interpretation of probability, 283  
Bernoulli distribution, 316  
Bernoulli process, 317  
Bernoulli trial, 317  
Berry paradox, 32, 65  
Big-O notation, 349  
Bijective function, 268  
Binary relation, 266  
Binary sequence, 269  
Binary tree, 278  
Binomial coefficient, 273

- Binomial distribution, 317  
Bipartite graph, 277  
Blank Symbol, 338, 347  
Branches of a tree, 277  
Breadth-first tree traversal, 279  
Burrows–Wheeler algorithm, 217  
bzip2, 217
- Cantor’s theorem, 27, 53  
Cardinality of a set, 265  
Cartesian product, 266  
Categorical attribute, 408  
Causal relevance, 453  
Causal-Mechanical model, 456  
Causality, 462  
Ceil, 268  
Central limit theorem, 330  
Certain event, 284  
Certificate, 350  
Characteristic function, 268  
Chebyshev’s inequality, 327  
Child of a vertex, 277  
Chomsky hierarchy, 271  
Church-Turing thesis, 336  
Class interval, 414  
Class interval width, 414  
Class limits, 414  
Class mark, 414  
Class P, 350  
Classical interpretation of probability, 282  
Classification problems, 409  
Code, 356  
Code alphabet, 356  
Code efficiency, 367  
Code extension, 357  
Code word, 356  
Codomain, 267  
Column matrix, 274  
Combinations, 273  
Combinatorics, 272
- Compact code, 366  
Complement of a set, 265  
Complete code, 367  
Complexity class, 349, 350  
Composition, 268  
Computable function, 345  
Computable procedure, 335  
Computable set, 346  
Computably enumerable set, 346  
Computation, 341  
Computational complexity, 336, 349  
Concrete entities, 442  
Conditional inaccuracy, 97  
Conditional independence, 290  
Conditional independence of random variables, 305  
Conditional Kolmogorov complexity, 387  
Conditional model, 71, 73  
Conditional probability, 288  
Conditional probability mass function, 302  
Conditional surfeit, 113  
Configuration, 340  
Configuration yields configuration, 340  
Confounding factors, 463  
Connected graph, 276  
Context sensitive grammar, 271  
Context-free-grammar, 271  
Continuous attribute, 408  
Convergence in probability, 327  
Correlation, 313  
Countable set, 268  
Countably many set, 268  
Covariance, 312  
CRISPR, 232  
Criteria for Interesting Questions, 137  
Cumulative frequency, 415  
Cycle, 276

- De Morgan's laws, 265  
Deductive-Nomological model, 455  
Deep learning, 232  
Degree of a vertex, 276  
Degree sum formula, 277  
Demarcation problem, 45, 235, 465  
Depth of a vertex, 277  
Depth-first tree traversal, 278  
Descendant of a vertex, 277  
Description, 32, 65  
Description function, 66  
Diagonal matrix, 274  
Directed graph, 277  
Discrete attribute, 408  
Discrete normal distribution, 319  
Discrete random variable, 292  
Discrete random vector, 298  
Disjoint sets, 265  
Domain, 267  
Dominant, 122  
Dutch book, 283
- Edges of a graph, 276  
Element of a set, 264  
Empirical adequacy, 453  
Empiricism, 439  
Empty language, 270  
Empty set, 265  
Empty string, 269  
Endpoints of an edge, 276  
Entities, 52  
Entity, 27  
Entity class, 62  
Epistemology, 440  
Equal sets, 265  
Equivalence class, 267  
Equivalence relation, 267  
Equivalent elements, 267  
Estimate, 400  
Estimator, 400, 410  
Event, 284
- Expected length of a code, 365  
Expected value, 306  
explanandum, 455  
explanans, 455  
Explanatory depth, 454
- Fiction view of models, 450  
Field of sets, 266  
Final State, 338, 347  
Fixed-length code, 360  
Floor, 268  
Free monoid, 269  
Frequency distribution, 414  
Frequentist interpretation of probability, 283  
full k-ary tree, 278  
Function, 267
- Generalizability, 439  
Grammar, 270  
Graph, 68, 276  
Graphene, 232  
Gödel numbering, 54
- Halting problem, 343, 344  
Handshaking theorem, 276  
Height of a tree, 277  
Heuristic programming, 451  
Holism, 443, 464  
Hypothesis, 121
- Identity function, 268  
Identity matrix, 274  
Impossible event, 284  
In-degree of a vertex, 277  
In-order tree traversal, 278  
Inaccuracy, 27, 37, 95, 172  
Inaccurate description, 37, 94  
Incident edge, 276  
Inclusion-exclusion principle, 272, 287  
Incompressible string, 392

- Independent and identically distributed, 322  
Independent events, 289  
Independent random variables, 301  
Individuals, 408  
Inferential conception, 449  
Infinite graph, 276  
Information distance, 389  
Initial State, 338, 347  
Initial vertex, 277  
Injective function, 268  
Input Symbol, 338, 347  
Interdisciplinary new topic, 153  
Interdisciplinary question, 151  
Interestingness of a new topic, 153  
Interestingness of a question, 150  
Interestingness of a topic as a problem, 146  
Interestingness of a topic as a tool, 149  
Intersection of sets, 265  
Intradisciplinary new topic, 153  
Intradisciplinary question, 151  
Invariance Theorem, 382  
Inverse function, 268  
Isolated vertex, 276  
Joint Kolmogorov complexity, 385  
Joint probability distribution, 297  
Joint probability mass function, 298  
Joint Representation, 62  
Joint representation, 83  
k-ary tree, 278  
Kleene closure, 269  
Knowable entity, 58, 441  
Knowledge frontier, 45  
Known subset of an area, 74  
Kolmogorov Complexity, 381  
Kolmogorov complexity, 54  
Kolmogorov's axioms, 285, 288  
Kraft's inequality, 361  
Labeled graph, 277  
Lambda calculus, 336  
Language, 270  
Law of Large Numbers, 328  
Leaf vertex, 277  
Leibniz's series, 36  
Length of a string, 269  
Likelihood function, 401  
Log-likelihood function, 401  
Logic, 440  
Logical coherence, 453  
Lower class limit, 414  
Luminiferous ether, 59  
Machine State, 338, 347  
Marginal probability mass function, 300  
Markov's Inequality, 326  
Matrix, 273  
Matrix determinant, 275  
Matrix main diagonal, 274  
Matrix rank, 275  
Matrix transpose, 274  
Maturity, 149  
Maximal element, 267  
Maximum element, 267  
Maximum Likelihood Estimator, 402  
McMillan's Inequality, 363  
Median, 308  
MediaWiki, 213  
MediaWiki API, 214  
Metaphysics, 440  
Michelson-Morley experiment, 59  
Minimal element, 267  
Minimum element, 267  
Miscoding, 27, 34, 81  
MNIST, 172  
Model, 409  
Model-based reasoning, 452  
Modulo, 268  
Multi-layer perceptron, 56

- Multinomial coefficient, 273  
Multiplication rule, 272, 289  
Multitape Turing machine, 339
- n-tuple, 266  
Natural kinds, 444  
Neighborhood of a vertex, 276  
Nescience, 27  
Neural network, 68  
New research entity, 45  
New topic, 153  
New topics area, 152  
Newton's second law, 38  
Nominal attribute, 408  
Non-knowable entity, 442  
Non-observable entities, 443  
Non-singular code, 356  
Non-terminal symbol, 270  
Normalized Aapplicability, 147  
Normalized compression distance, 391  
Normalized information distance, 391  
Normalized relevance, 145  
Normalized simplified applicability, 148
- Observable entities, 443  
Occam's razor principle, 66  
Ontology, 440  
Open class interval, 414  
Oracle Turing machine, 30, 36, 55, 336, 347  
Order of a function, 268  
Ordered pair, 266  
Ordinal attribute, 408  
Out-degree of a vertex, 277  
Outcome, 284  
Overfitting, 173
- P vs. NP, 99  
P=NP Problem, 352  
Padding lemma, 342
- Paradigm shift, 451  
Parameter of a probability distribution, 399  
Parameter space, 399  
Parametric random variable, 399  
Parent of a vertex, 277  
Pareto frontier, 122  
Pareto optimal, 122  
Pareto optimality, 42  
Pareto point, 103, 117  
Parsimony, 453  
Partial computable function, 345  
Partial function, 267  
Partial order, 267  
Partially ordered set, 228, 267  
Partition of a set, 265, 266  
Path, 276  
Pendant vertex, 276  
Perfect conditional description, 72, 73  
Perfect description, 66  
Perfect description of an area, 75  
Perfect knowledge, 43  
Permutations, 272  
Philosophy of science, 437  
Phlogiston, 36  
Pigeonhole orinciple, 273  
Pleonastic description, 68  
Polywater, 61  
Population, 408  
poset, 267  
Possibility of misrepresentation, 448  
Post Correspondence Problem, 344  
Post-order tree traversal, 278  
Posterior distribution, 404  
Posterior probability, 291  
Power set, 265, 266  
Pragmatic theories, 456  
Pre-order tree traversal, 278  
Prefix free set, 269  
Prefix string, 269

- Prefix-free code, 358  
Prefix-free universal Turing machine, 381  
Principle of indifference, 283  
Prior distribution, 404  
Prior probability, 291  
Probability, 285  
Probability mass function, 288, 294  
Probability space, 285  
Problem of ontology, 55, 448  
Problem of style, 56, 447  
Production rule, 270  
Proper subset, 265  
  
Qualitative attribute, 408  
Quantitative attribute, 408  
Question, 150  
Quotient set, 267  
  
Random sample, 322  
Random string, 393  
Random variable, 292  
Randomized controlled trials, 463  
Range, 267  
Realism, 443  
realism, 464  
Recursive function, 336, 345  
Recursively enumerable grammar, 271  
Reductionism, 443, 464  
Redundacy, 111  
Redundancy, 217  
Reflexive relation, 266  
Regression problems, 409  
Regular grammar, 271  
Relative frequency, 414  
Relative variation of inaccuracy, 100  
Relative variation of surfeit, 115  
Relevance, 144  
Relevance Graph, 144  
Relevance graph, 144  
Representation, 28  
Representation function, 55  
Representation of an area, 74  
Representation-as, 450  
Representational demarcation, 55, 447  
Requirement of directionality, 57, 448  
Research area, 74  
Reverse string, 269  
Root of a tree, 277  
Row matrix, 274  
Russell's Paradox, 53  
Russell's paradox, 28  
  
Sample covariance, 325  
Sample mean, 323  
Sample size, 322  
Sample space, 284  
Sample standard deviation, 324  
Sample variance, 324  
Sardinas-Patterson, 358  
Science, 438  
Scientific anti-realism, 464  
Scientific method, 33, 121, 438  
Scientific realism, 464  
Scientific representation, 447  
Self delimited string, 269  
Self-correction, 439  
Sequence of symbols, 268  
Set, 264  
Set difference, 265  
Set formation notation, 264  
Set of conditional descriptions, 71  
Set of descriptions, 65  
Set of integers, 264  
Set of natural numbers, 264  
Set of positive integers, 264  
Set of positive reals, 264  
Set of rationals, 264  
Set of real numbers, 264  
Set of representations, 58

- Shortest-path problem, 350  
Shortlex ordering, 270  
Sibling to a vertex, 277  
Sicence problem, 121  
Similarity conception, 449  
Simple cycle, 276  
Simple path, 276  
Simplified applicability, 148  
Simplified applicability graph, 148  
Singular value decomposition, 275  
Source alphabet, 356  
Source symbol, 356  
Square matrix, 274  
St. Petersburg Paradox, 334  
Standar deviation, 311  
Standard of Accuracy, 56  
Standard of accuracy, 447  
Standarized discrete random variable, 318  
Start symbol, 270  
State diagram, 339  
Statistic, 399  
Statistical inference, 398, 461  
Statistical model, 399  
Statistical relevance, 455  
Statistical-Relevance model, 455  
Stipulative fiat, 449  
Stock market, 175  
String, 269  
String concatenation, 269  
String quotient, 269  
Strongly connected graph, 277  
Structuralist conception, 449  
Subgraph, 277  
Subjective interpretation of probability, 283  
Submatrix, 274  
Subset, 265  
Substring, 269  
Subtree, 278  
Surfeit, 27, 40, 108, 216  
Surjective function, 268  
Surrogate reasoning, 57, 88, 448  
Symbols, 268  
Symmetric matrix, 274  
Symmetric relation, 266  
Tape Symbol, 338, 347  
Target variable, 409  
Targetless models, 55, 448  
Targetless representation, 87  
Terminal symbol, 270  
Terminal vertex, 277  
Testability, 439  
Theoretical framework, 439  
Theory of relativity, 59  
Time complexity, 349  
Total order, 267  
Total relation, 266  
Totally ordered set, 267  
Training set, 410  
Transition Function, 338, 347  
Transitive relation, 266  
Traveling salesman problem, 351  
Tree, 277  
Tree traversal, 278  
Turing machine, 30, 32, 335, 337  
Turing reducibility, 352  
Turing's thesis, 341  
Unbalance dataset, 175  
Uncountable set, 268  
Unificationist account, 456  
Uniform distribution, 316  
Union of sets, 265  
Uniquely decodable code, 357  
Universal Turing machine, 336, 343  
Universe of a set, 264  
Unknowable entity, 58  
Unknowable unknown, 62  
Unknown frontier, 152  
Unknown unknown, 45  
Upper bound of a function, 268

Upper class limit, 414  
utility function, 42  
  
Valid Representation, 60  
Variance, 308  
Variation of inaccuracy, 100, 101  
Variation of surfeit, 114  
Venn diagram, 265  
Verifier, 350  
Vertices of a graph, 276  
  
Wallis' product, 36  
Weakly dominates, 123  
Weakly Pareto optimal, 123  
Weakly Paretor frontier, 123  
Weight of an edge, 277  
Wikipedia, 212  
Wikitextparser, 213  
Words of a language, 270  
  
Yield, 340