Health Informatics

Gyorgy J. Simon
Constantin Aliferis *Editors*

Artificial Intelligence and Machine Learning in Health Care and Medical Sciences

Best Practices and Pitfalls

OPEN ACCESS

Springer

Health Informatics

This series is directed to healthcare professionals leading the transformation of healthcare by using information and knowledge. For over 20 years, Health Informatics has offered a broad range of titles: some address specific professions such as nursing, medicine, and health administration; others cover special areas of practice such as trauma and radiology; still other books in the series focus on interdisciplinary issues, such as the computer based patient record, electronic health records, and networked healthcare systems. Editors and authors, eminent experts in their fields, offer their accounts of innovations in health informatics. Increasingly, these accounts go beyond hardware and software to address the role of information in influencing the transformation of healthcare delivery systems around the world. The series also increasingly focuses on the users of the information and systems: the organizational, behavioral, and societal changes that accompany the diffusion of information technology in health services environments.

Developments in healthcare delivery are constant; in recent years, bioinformatics has emerged as a new field in health informatics to support emerging and ongoing developments in molecular biology. At the same time, further evolution of the field of health informatics is reflected in the introduction of concepts at the macro or health systems delivery level with major national initiatives related to electronic health records (EHR), data standards, and public health informatics.

These changes will continue to shape health services in the twenty-first century. By making full and creative use of the technology to tame data and to transform information, Health Informatics will foster the development and use of new knowledge in healthcare.

Gyorgy J. Simon • Constantin Aliferis
Editors

Artificial Intelligence and Machine Learning in Health Care and Medical Sciences

Best Practices and Pitfalls

Springer

Editors
Gyorgy J. Simon
Institute for Health Informatics
University of Minnesota
Minneapolis, MN, USA

Constantin Aliferis
Institute for Health Informatics
University of Minnesota
Minneapolis, MN, USA

ISSN 1431-1917 ISSN 2197-3741 (electronic)
Institute for Health Informatics
ISBN 978-3-031-39354-9 ISBN 978-3-031-39355-6 (eBook)
https://doi.org/10.1007/978-3-031-39355-6

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Paper in this product is recyclable.

I am dedicating this book to my wife, Mayuko, and our children, George and Klara, for their love and for their patience with my long hours writing this book. I could not have succeeded either in my career or with this book without Mayuko's dedication and support.

Gyorgy J. Simon

I most affectionately dedicate this work to my wife Alla, for the inspiration she gives me daily and for her love.

Constantin Aliferis

Foreword 1

Imagine a world where you receive treatment that is precisely targeted to give you the best outcome possible, where care is efficient and safe, and where you have guidance in managing your health at home and are a partner in your healthcare. We dream of a world like this, and you as a reader of this book probably see a role for AI in making that dream come true.

Rear Admiral Grace Hopper, developer of the first compiler for a computer language, repeated a motto coined by John Augustus Shedd: "A ship in port is safe; but that is not what ships are built for. Sail out to sea and do new things."[1] As a reader of this book, you are leaving a safe harbor to do something hard and something risky.

I left that safe harbor in 1994 when I pivoted from my humanities university degree in linguistics and Chinese to enroll in a graduate program in medical informatics at the University of Utah. My husband introduced me to the field, and my love of language steered me to the new world of natural language processing. In classrooms, I raised my hand to ask about the meaning of basic words like "algorithm," "heuristic," and "ML," and as the only female in the campus computer lab at 2 am, I sat with my 2-year-old son asleep at my feet trying to get my linked list to compile. I was an outsider in the world of computers and AI, and I was an outsider in the world of healthcare. But like each of you, I brought unique experience, and embracing that, I have been able to both develop methodological innovations and apply those to problems like disease surveillance and creation of research cohorts from electronic health record data.

The department where I studied was chaired by Homer R. Warner,[2] who founded the department in 1964.

> Homer Warner … developed in 1961 the first computerized program for diagnosing disease. Basing it on 1,000 children with various congenital heart diseases, Warner showed that Bayes' [theorem] could identify their underlying problems quite accurately. "Old cardiologists just couldn't believe that a computer could do something better than a human," Warner recalled.[3]

[1] "Grace Hopper: The Youthful Teacher of Us All" by Henry S. Tropp in Abacus Vol. 2, Issue 1 (Fall 1984) ISSN 0724-6722.

[2] https://en.wikipedia.org/wiki/Homer_R._Warner.

[3] https://yalebooks.yale.edu/book/9780300188226/the-theory-that-would-not-die/.

Homer sailed from a safe harbor, and, in doing so, he developed one of the first electronic medical records to collect data not only to improve access to information but ultimately to transform the diagnostic and patient care process through building intelligence into the system.[4]

And now you are here, in this open sea. It has been over half a century since Homer Warner and other pioneers developed and implemented AI systems in healthcare. The age of AI seems to be upon us. The big tech industry sees the opportunity and is making unprecedented investments in healthcare.

So, what is the risk?

When creating innovative healthcare solutions using AI, challenges will arise in at least three steps: development, application, and implementation. In *developing* ML and AI models, you will not be able to fully trust the output of the tools, because the learnings and predictions only represent a potentially deceptive proxy of the real situation,[5] and the algorithms are most likely learning from biased data and biased healthcare delivery practices. In *applying* ML and AI models to healthcare problems, you will encounter unintended consequences—your predictive model may lead to a rapid upsurge in overdiagnoses, for example. In *implementing* your tools, you may discover that the reality does not match the hype: most never even make it to the real world, and when they do, the results are often disappointing.[6] When you put a new technology into the complex system of healthcare, everything around it changes. You will shift relationships, you will shift workflows, and you will shift power differentials. And those changes can cause harm.

Given the risks and the difficulty you will face, should you even launch your boat into the sea of healthcare AI? Yes! We need you. We need the smartest brains tackling problems more consequential than how to get people to click on ads.[7] And that is where this book comes in. *Artificial Intelligence and Machine Learning in Health Care and the Health Sciences: Pitfalls and Best Practices* will be your compass to help you chart a more successful path.

The lead authors of this book have decades of experience teaching and doing research on this topic. Dr. Aliferis has dedicated his career to the responsible use of AI to improve human lives and support scientific discovery. He is an innovator of high-performing and reliable methods with a goal of improving safety and effectiveness. Dr. Simon has extensive experience in data mining, machine learning, statistical analysis, and biostatistics. He has a solid background in hands-on software development in academic and commercial settings. The editors have pulled together a star-studded cast of content contributors with both knowledge and experience.

[4] Paul D. Clayton, PhD, Presentation of the Morris F. Collen Award to Homer R. Warner, MD, PhD: "Why Not? Let's Do It!", Journal of the American Medical Informatics Association, Volume 2, Issue 2, March 1995, Pages 137–142, https://doi.org/10.1136/jamia.1995.95261907.

[5] https://www.youtube.com/watch?v=cDAXiq-at5M.

[6] https://khn.org/news/a-reality-check-on-artificial-intelligence-are-health-care-claims-overblown/.

[7] https://web.archive.org/web/20150202014230/http://www.bloomberg.com/bw/magazine/content/11_17/b4225060960537.htm.

Never before has so much wisdom about best practices in health and biomedical AI been compiled into one place.

This book is comprehensive but also extremely practical. It provides diverse and reliable best practices through a wide range of illustrative applications of machine learning and AI. The book will detail specific requirements and adaptations that are necessary to successfully tailor general algorithms and techniques to healthcare and health sciences discovery. Authors describe common pitfalls that plague research and commercial attempts at applying AI to healthcare and will give advice on how to avoid repeating the mistakes of the past. Through this book, you will learn how to develop AI applications that can be trusted and therefore have a higher likelihood of success to achieve the goal of high performance, safety, and cost-effectiveness in healthcare.

Pedro Domingos said, "People worry that computers will get too smart and take over the world, but the real problem is that they're too stupid and they've already taken over the world."[8] If you come from outside of healthcare, you may be surprised at how "stupid" their computers are and how long the path may be to apply your cutting-edge innovation. If you come from within healthcare, you understand why it has taken so long to bring the innovations you see around you to healthcare, but you may not understand the technical aspects well enough to bridge the gap. Studying this book will make us all more informed partners and will accelerate our journey to improved health and discovery through AI.

Digital Health and Informatics
Centre for Digital Transformation of Health
University of Melbourne
Parkville, VIC, Australia

Wendy Chapman

[8] Domingos, P. 2015, The master algorithm: How the quest for the ultimate learning machine will remake our world, Basic Books, New York, NY, USA.

Foreword 2

The emergence of advanced technologies impacts healthcare. For several years now, anticipation has been building that artificial intelligence (AI) and machine learning (ML) will lead to a paradigm shift in healthcare, promoting seamless, safe, and convenient access to healthcare services including disease management and prevention. AI/ML technologies can support the diagnosis of complex medical conditions, help clinicians make more informed decisions, and ultimately improve patient outcomes through timely diagnosis and tailored treatment. These technologies can facilitate the collection and analysis of diverse data including clinical data and behavioral, genomic, and environmental datasets that all can provide unique insights into individual healthcare needs in the context of precision health.

In my own work, I have focused on the use of technology to support aging and studied the use of AI and ML in the space of gerontology and geriatrics. I lead the Penn Artificial Intelligence and Technology (PennAITech) Collaboratory for Healthy Aging, which has the goal to identify, develop, evaluate, commercialize, and disseminate innovative technology and artificial intelligence (AI) methods and software to support older adults and those with Alzheimer's disease (AD) and Alzheimer's disease and related dementias (ADRD) in their home environment. These technologies cover a broad spectrum ranging from home-based monitoring technologies and smart home sensors to robotic applications, conversational agents, wearables, and other digital phenotyping tools. The Collaboratory is motivated by the need for a comprehensive pipeline across technology-based monitoring of older adults in the home, collection and processing of monitoring data, integration of those data with clinical data from electronic health records, analysis with cutting-edge AI methods and software, and deployment of validated AI models at point of care for decision support.

In this work, it becomes clear that AI and ML technologies can create new opportunities for monitoring and supporting older adults and their families in a variety of settings. However, for these technologies to be widely adopted and trusted by healthcare providers, patients, and families, it is essential to not only generate solid evidence of their effectiveness but also establish best practices in their development and use.

Such best practices are important to ensure the safety and efficacy of AI and ML technologies in patient care. As has been argued numerous times, AI and ML algorithms are only as good as the data they are trained on, and if the data are biased or

inaccurate, the algorithms will produce biased or inaccurate results. The consequences can be significant; incorrect diagnoses can lead to harm or even death, and bias in treatment selection may affect outcomes and exacerbate existing inequities. Informed efforts such as using large, diverse, and representative datasets can in many cases help to mitigate these risks and eliminate algorithmic bias. We have to ensure the transparency and accountability of such technologies. Healthcare providers and patients need to understand how AI and ML technologies "make decisions" and what factors influence these decisions.

Establishing best practices in the development and use of AI and ML technologies in patient care and more broadly in biomedicine is crucial for ensuring the safety, efficacy, transparency, accountability, and ethical use of these technologies. However, to date, data scientists in health sciences, clinicians, and administrators do not have concrete frameworks to help them navigate this landscape. As computational advances accelerate the growth of AI and ML, the healthcare industry appears to try to catch up with methodological, policy, and clinical guidelines. The book by Drs. Simon and Aliferis, "*Pitfalls and Best Practices in AI/ML for Healthcare and Health Science*," addresses this gap as it provides an interdisciplinary perspective and an in-depth insight into reliable AI/ML methods and their properties, approaches for benchmarking, best practices for transparency and dissemination, and broad range of tools allowing data scientists, informaticians, and clinicians to develop or utilize AI/ML while maximizing effectiveness and safety and avoiding well-documented pitfalls. This timely book delves into the application of AI and ML in healthcare, exploring benefits and limitations and providing insights into the future of AI; the book features interdisciplinary and evidence-based perspectives that will lead to more accurate, efficient, and personalized patient care.

George Demiris
Department of Biostatistics, Epidemiology and Informatics, Perelman School of Medicine and School of Nursing
University of Pennsylvania
Philadelphia, PA, USA

Preface 1

We are living in an interesting time. Broad adoption of AI is underway with very significant achievements, including chatbots with humanlike language skills and autonomous taxi services operating in many parts of the country, and we are at the cusp of broad adoption of AI in clinical care and health sciences.

AI is not without its risks. We have seen AI dish out bad advice, be racist, and be sometimes utterly incompetent. In the summer of 2019, Constantin and I sat down to write a paper about the "Knowledge Cliff" problem in AI/ML in biomedicine. While researching this paper, we realized that merely pointing out problems with AI, while it is useful, is not particularly constructive or actionable. Drawing on our decades of teaching and research, we refocused the paper into explaining the pitfalls of certain approaches and providing actionable best practice recommendations in six (or so) different areas. As such, one paper had to grow into six. It did not take long to recognize that our goal requires an entire book. We also recognized the dearth of AI/ML textbooks that focus on healthcare, its special characteristics, and requirements, with sufficient technical depth to enable a meaningful discussion about pitfalls and best practices of building AI/ML systems for biomedicine.

This book is meant for a broad audience; virtually anyone can benefit from reading it. Although this book can be read cover to cover in sequential order of the chapters, we aimed to make chapters self-contained (sometimes at the cost of minor repetitions), because certain audiences benefit from some chapters more than others. My recommendations on how to use this book for certain audiences follow.

First, AI/ML professionals with extensive training in computer science and general-purpose AI/ML, who consider getting into healthcare or health sciences. I am a computer scientist who made this transition into biomedicine; thus, the book reflects on my personal experience. For this audience, the book offers overviews of aspects of method and model development that are often neglected in general-purpose AI/ML (Chapters "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" and "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models"), an introduction to data design (Chapter "Data Design"), detailed (more detailed than others) description of modeling methods for time-to-event and longitudinal data (Chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"), introduction to model evaluation from the perspective of health benefits and health economics (Chapter "Evaluation"),

description of healthcare standards, terminologies, and ontologies (Chapter "Data Preparation, Transforms, Quality, and Management"), etc. The goal of our book for this audience is to help them transition into healthcare seamlessly.

Second, analysts without a rigorous computer science training who are already building health AI/ML models. To this audience, we cover the foundational concepts in computer science, AI and ML (Chapter "Foundations and Properties of AI/ML Systems"), a 10,000-foot view of a broad range of modeling methods (Chapters "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" and "Foundations of Causal ML"), and other important aspects of health modeling (Chapters "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems," "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models", "Regulatory Aspects and Ethical Legal Societal Implications (ELSI)", and "Reporting Standards, Certification/Accreditation, and Reproducibility"). The goal of our book for this audience is to entice and enable them to be more rigorous with their work.

Third, decision makers, who wish to gain a better understanding of the capabilities and limitations of AI in healthcare and health sciences. To this audience, I recommend Chapters "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs" (historic case studies and lessons learnt), "Characterizing, Diagnosing and Managing the Risk of Error of ML & AI Models in Clinical and Organizational Application", "Regulatory Aspects and Ethical Legal Societal Implications (ELSI)", and "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" for appraising existing methods, and any other chapter that discusses topics that they wish to deepen their knowledge of.

It is my opinion that everybody will find something new. Even readers with minimal technical background will find Chapters "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs" fascinating, and on the other end, experts in AI/ML may find new nuggets of knowledge in Chapter "Overfitting, Underfitting and General Model Overconfidence and under-Performance Pitfalls and Best Practices in Machine Learning and AI" or about human cognitive biases (Chapter "From 'Human versus Machine' to 'Human with Machine'") or design biases (Chapter "Data Design"). Chapters are self-contained; please feel free to read any chapter you find interesting.

Inspired by my co-author's, mentors', and collaborators' passion for patient care and safety, my ultimate goal with this book, and more broadly with my career, is to promote better patient care, safety, and ethics. This book aims to lay the foundations and instill the scientific rigor necessary for the successful implementation of helpful and safe AI in clinical care and for making fruitful discoveries in health sciences. I wish the reader great success in the exciting and growing field of biomedical AI.

Minneapolis, MN, USA Gyorgy J. Simon

Preface 2

The promise and challenges of health artificial intelligence (AI) and machine learning (ML). After a long history of exploration and experimentation, brilliant successes interleaved with some striking failures, springs of optimism, and winters of disappointment, AI has arrived and is here to stay. This realization is engraved in the minds of the scientific community and lay public alike.

It is not hyperbolic to recognize that the AI and ML methods and technologies at the service of health sciences and healthcare of our times are nothing short of amazing. As remarkable as the problem-solving tools we have are, in order to be deployed for clinical and other mission-critical tasks, they have to overcome significant gaps in performance and safety, however.

The comparison with other areas of applied technology is both striking and worrisome. Consider a humble and low-tech device such as an oven toaster (or any other electric appliance operating with high voltage): in order to be approved for the consumer market, rigorous testing must be done to ensure that it is not a fire hazard or that it will not electrocute its operator. Bridges are not opened to public use unless the civil engineers that built them provide plans that undergo tremendous scrutiny and establish, for example, how much load they can withstand, what wind forces or earthquakes they can tolerate without collapsing, and what maintenance they need and how often. Similarly, cars, mechanical tools, children's toys, drugs, and so on across the whole gamut of human activity are closely scrutinized and become available for consumer use only when sufficient measures have been established to ensure safe application. Public and individual safety is of paramount consideration and enforced everywhere.

Everywhere, except in health AI and ML, it seems. Poorly constructed, evaluated, deployed, or monitored AI and ML models, apps, and systems have the potential to cause great benefit or grave harm at a massive scale. This is true for healthcare medical applications, healthcare business decisions, and health sciences. Regulation is only now emerging, and AI/ML products have been offered for years (often at huge financial costs) to the trusting healthcare providers and find their ways to profoundly affect human lives, without providing guarantees of effectiveness and safety.

Another related deficiency of the present state of the health AI/ML market is that of efficiency. It would be unthinkable for a car manufacturer to market, for example, a four-passenger automobile for everyday commuting purposes with fuel

consumption of 1 mile per gallon. Yet, we routinely see commercial and noncommercial AI/ML offerings that are worse offenders than this example in terms of the computing costs per unit of output when perfectly capable (or better) alternatives exist with orders of magnitude lower costs of use.

From another viewpoint, and to use an electrical engineering analogy, the ML models of today function as components of a larger system that presently lacks protection against overloading the system. Well-designed consumer electrical devices are routinely engineered in such a way that their input and output obey specifications ensuring that a system of interconnected such units will function properly. Nothing like this exists in health AI/ML at present.

But even if a very high standard of accountability was put in place (and steps are being taken recently in this direction by the FDA, NIST, EU, etc. as described in the present volume), it would be of little value, unless we could equip the data scientists and the organizations adopting AI/ML solutions with the technical means by which to achieve (and verify) the standard's expectation.

Pitfalls and Best Practices: A Personal Perspective. This book aims precisely to contributing to solving these problems by providing comprehensive information in the form of identifiable pitfalls and practical best practices supporting the effective, safe, efficient, cost-effective, science-driven, rigorous, informed, rational, derisked, trust-inspiring, and accountable health AI and ML.

Some of the book's concepts started forming in my mind many years ago when I was a beginning graduate student in AI. I have written elsewhere[1] how my personal journey started and why I believed early on that AI could make medicine more scientific and ultimately more effective. A formative experience of particular relevance to this book has been the NSF-funded ML Pneumonia Prediction project led by my graduate advisor Greg Cooper (connected with the broader Pneumonia Patient Outcomes Research Team (PORT) cohort study). The ML project aimed at exploring many cutting-edge methods of the era to create the most accurate and practical models possible for predicting community-acquired pneumonia mortality. Several word-class labs and AI and machine learning luminaries from the University of Pittsburgh and CMU participated in the effort (i.e., the Bruce Buchanan lab, Greg Cooper lab, Tom Mitchel lab, Peter Spirtes lab, Clark Glymour lab), along with Dr. Michael Fine, the leader of the PORT study, which ended having very significant impact nationally. Everyone strived to produce the best models working off a single discovery dataset. In these circumstances, it would have been very easy to totally overfit the models, but thanks to Greg's scientific foresight and rigor, a nested cross-validation design and other safeguards were put in place to eliminate the risk of overfitting and taught me a career-defining lesson about the value of pursuing high-stakes modeling with utmost rigor.

As I started my career as a faculty, 24 years ago, I placed significant emphasis on the modeling challenges of the very novel at the time gene expression microarray, mass spectrometry, and other high-dimensional data. I was surprised and occasionally dismayed at what appeared to be technical abuse and overinterpretation of

[1] https://imia-medinfo.org/wp/history-book/

methods with widespread lack of rigor in the bioinformatics field fueled by exuberant expectations. In one of many memorable incidents during my early faculty career, a cancer biology professor, whom I respected very much for his scientific accomplishments, reached out to me one day and asked me if I could "cluster his gene expression data to predict cancer." I asked how many patients he had assayed, and to my astonishment, he said "three." I asked him what made him believe that clustering three patients in two groups would have any statistical validity and predictive modeling usefulness, and to my continuing amazement, he showed me a recent copy of one of the two most respected biology journals. "Can't you just do it like they did it in this article?" he asked, pointing to the cover article. This incident was representative of the whole field of genomics of the time at the infancy of high-throughput technology and showed me that applying complex AI/ML to genomic datasets had a long way to go in terms of educating many of the key practitioners.

Over the years, I watched many scientific accomplishments and breakthroughs in pure AI and machine learning and followed the numerous applied biomedical advances that they enabled. I also built new methods and put those and every other major technique invented by others to practice, in many NIH-funded projects. I taught students and early-career faculty AI and ML and watched how this knowledge helped them in their careers. I also organized and oversaw at NYU and my current institution, the UMN, institutional-level research support teams and cores that deployed AI and ML via consulting, and team science for hundreds of projects and thousands of consults between 2008 and today. I also gave invited lectures in both academic and industry settings about the dangers inherent in AI and ML when not used with enough rigor and discipline. I followed closely the literature and developments in regulatory frameworks, best practices, and major failures and case studies in AI/ML. Finally, my group conducted some of the largest (and in some cases, I dare say, authoritative) benchmarks of ML in several fields of biomedical research.

Embarking on a long-overdue project. It was thus about time in 2021 to finally put together a book distilling the above information in a format that would be accessible, backed up by all relevant scientific evidence, and practically useful for working scientists, students, administrators, practitioners, and other stakeholders of health AI. With regard to style, a great influence has always been the late Richard Feynman and his monumental *Lectures on Physics,* in which he managed to condense the full range of physics, sparing no difficult subject, in a way that was accessible to first-year college students, without getting vague or sacrificing accuracy. Tom Mitchell, Sholom Weiss, and one of my most respected role models, Casimir Kulikowski, achieved in their corresponding books on ML similar levels of clarity, accuracy, and accessibility. Gyorgy and I used the above three works as inspiration and our "Northern Star" in terms of clarity and accessibility for the present volume.

Best practices imply necessary conditions for success, or necessary and sufficient conditions in some cases. But it also, unavoidably, involves describing sufficient conditions for failure that need to be prevented. Not uncommonly, when the knowledge in some topic is not enough to delineate with absolute certainty what needs to be done, a best practice is no more than the best possible recommendation

for how to apply AI/ML given the limited knowledge of the time. In Chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML", we explain in detail the sources for best practices and pitfalls presented in this work. We decided early on that we should not expunge from this work our personal experiences or hands-on expertise in specific methods and applications. We resolutely decided, however, that this book should not be a summary of our preferred way of developing and applying AI/ML, and our own personal practices would have a place here if and only if they are solidly backed by the scientific evidence in the literature and if they are paradigmatic of the various messages we wish to convey. In many chapters, this decision has worked particularly well, in my view, because we were able to dive deeply into topics that would be hard to present in a clear manner if we did not describe details stemming from in-depth and first-hand experience (Chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" on method development and appraisal is an example whereby we try to "lift the curtain" and show, from the inside, an example of how rigorously-designed and -executed new methods can come to existence).

Regarding helpfulness and defensibility, we realized that such a book should be constructive but not sugarcoat the facts. This is not a volume about presenting on even ground all methods, systems, or efforts, without taking a position on strengths, weaknesses, and relative performance. The result may not be viewed favorably within circles where dogged devotion to this or the other method supersedes objective performance considerations. This is however a book about *pitfalls* and *best practices.* The notion of pitfalls entails dangers, failures, and risk that we should avoid or minimize. It is imperative that all students, teachers, and practitioners of AI/ML embrace its collective history, learn from it, own any mistakes made, and not repeat them again. Showcasing failures in the field is not intended to diminish the people or organizations behind them, but to learn from these case studies how to do AI/ML in a safe and effective manner. Conversely, it is essential to show what can (and does) happen when AI/ML is not done systematically with sufficient scientific and technical rigor. Naturally, it is entirely possible to disappoint people whose methods or practices are criticized. Gyorgy and I took this seriously into account, but we balanced it with the need to present scientific truth (the best way we can grasp it anyway) and above all to protect patients. AI/ML can critically affect the well-being of patients and of society and will increasingly do so in the future, after all.

In terms of scope, we cast a very wide net, first taking a broad look at all types of health science and healthcare applications of AI/ML. We generally refrain in the book from engaging in analysis of methods with only historical significance, except in cases where historical case studies or other works are highly informative for modern-day AI/ML. We retrieved all health AI/ML best practice and guideline papers as well as meta-analyses and systematic reviews of AI/ML methods and model comparisons across all biomedical fields, in PubMed. Chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML" gives a glimpse of that

space. This material was supplemented with our personal collection of AI/ML books and papers, which comprises thousands of entries plus many more that were identified in PubMed and Google Scholar to help with topics outside our usual scope of work. We synthesized and integrated prior best practices, in many cases invisibly to the reader, and in some cases bringing certain preexisting best practices to the forefront of the book because we saw them as very important, or in need of discussion, extensions, etc. Most importantly, we placed a huge emphasis on principles of operation and properties of AI/ML methods for two reasons: on the one hand, these provide the necessary justification for recommended use (or avoidance). On the other hand, method properties can be applied to infinite situations and the readers can tailor them to their own specific projects, which we could not possibly anticipate in full, whereas guidelines and best practices are by necessity more context and problem sensitive. We also worked to modularize the recommended practices to mirror the stages that AI/ML methods are being created, validated, and deployed. This should make reading the material easier. In the end of the book, we provide a collection of all best practices and annotate them as being high or medium impact and as being highly mature or evolving. High-impact recommendations are ones that following or discarding them will have the gravest consequences across most contexts of use. Lower impact ones may have lesser consequences or significant consequences but in a small portion of application areas. Evolving best practices are ones that will likely improve over time as the field's understanding advances, whereas highly mature ones are so foundational that they will almost certainly continue to be applicable far into the future.

Minneapolis, MN, USA Constantin Aliferis

Acknowledgments

First and foremost, I wish to acknowledge Constantin, with whom I co-authored this book, for convincing me, over a 1-year-plus time period, to write this book. He did a job, worthy of the deepest admiration, with some of the most difficult topics, including lessons learnt and ethics. He managed to keep these difficult chapters, like all other chapters, constructive, actionable, and objective, and I believe that they became the crown jewels of this book. I recommend absolutely everyone to read these chapters. Clearly, this book would not have been possible without his hard work.

Besides co-authoring the book, I owe Constantin a debt of gratitude for his professional mentoring, passion for scientific discovery, and his appreciation of rigor in our work to make those valuable scientific discoveries happen.

I would also like to acknowledge Dr. Vipin Kumar, a well-recognized leader in (general-purpose) data mining and machine learning and a scientific role model to me and many other scientists. I feel fortunate to have had him as my thesis advisor, and he remains one of my professional mentors and collaborators to this day.

At the time I started out as a junior faculty, mentorship for faculty was scarce. I was lucky that Dr. Genevieve Melton-Meaux, a surgeon, NLP expert, and executive at a large health system, offered a helping hand. Drawing on her vast experience in the health system, she channeled my ML knowledge in a direction that is most valuable to clinical care and healthcare delivery research.

Dr. Pedro Caraballo, an MD passionate about patient care and an advocate for ML in clinical decision support, was one of my first collaborators in the ML-based clinical decision support realm who instilled in me the importance of ML in research as well as clinical decision support.

I would also like to thank my past and present co-workers who helped with the book, including Drs. Steve Johnson, Erich Kummerfeld, Sisi Ma, Dennis Murphree, Rui Zhang, Jinhua Wang, Bryan Andrews, Dalton Schutte, Anirudh Choudhary, Puneet Bhullar, and Nneka I. I thank them for the valuable insights they gave me, and especially thank them for the chapters they contributed.

I thank Melissa Malkowski and Beth Madson for their outstanding administrative support and Nithya Sechin from Springer.

Thanks also go to the UMN Office of Academic and Clinical Affairs (Dr. Jakub Tolar) for funding that made the book possible as open access.

I also thank Springer and especially senior editor Grant Weston, who believed and encouraged this project with great passion.

Gyorgy J. Simon

I would be amiss if I did not express my deep gratitude to a number of individuals that made my personal journey in biomedical AI and ML worthwhile and indirectly or directly benefited this book.

The first foundations I received in the data sciences were provided by the late Professor Dimitrios Trichopoulos and his faculty at the University of Athens School of Medicine. Trichopoulos was internationally renowned as an epidemiologist of formidable intellect and scholarship, and he made sure that his students received a wealth of knowledge in research design, biostatistics, and public health methods. Other mentors and collaborators were instrumental in nurturing my curiosity about the quantitative data sciences in the context of several research projects. I am deeply grateful to my former medical school professors and mentors E. Batrinos, E. Geordiadis, N. Katsilambros, and especially E. Georgiou who was my first mentor in biomedical informatics.

Greg Cooper's influence on my AI training was instrumental because of the early formative experience in the years I worked with him as postdoctoral fellow in his Pneumonia Prediction project and the years during which he served as my MS and PhD theses' main advisor. Greg and I became lifelong friends, and he has never stopped giving me valuable guidance for which I am grateful. I am also grateful to all my teachers in the Intelligent Systems Studies Program of the University of Pittsburgh. My other advisors, Bruce Buchanan, Randy Miller, Martha Pollack, Michael Wagner, and Howard Doyle, were outstanding mentors and role models who above all helped me learn to think critically.

I am most grateful to my colleague and co-author Gyorgy Simon for believing that this project could be done, for writing brilliant material, and for working with me in harmony even under a most demanding work schedule and difficult deadlines. Gyorgy is as knowledgeable, and effective, as he is humble and collegial. I thank him for sharing so much of his extensive knowledge in this book and being instrumental in making it a reality.

I am profoundly grateful to Grant Weston, senior editor at Springer, for being a true believer to the need for such a work. His enthusiasm, patience, and support are deeply appreciated.

Frank Harrell has been a great mentor and a fantastic role model both as a distinguished biostatistician and for setting a standard for "speaking the truth that others do not dare say."

Isabelle Guyon has been a wonderful collaborator and inspiring role model in AI/ML. I had the pleasure to work with her as a co-author in several books and papers and in a major ML challenge. I learnt so much from her scientific example and approach to structuring complex tasks and managing teams of diverse competencies.

Perry Miller, Henry Lowe, John Wilbur, and Michael Kahn, along with Greg Cooper, guided me in my early years as informatics leader at NYU and taught me many valuable lessons in leadership. Richard Woodrow, my executive leadership coach at NYU, spent 2 years teaching me how to put honesty above other concerns, and I am very grateful for his wisdom, which I tried to put to practice in this volume. Vivian Lee gave me a big career boost by making me Head of Informatics at NYU Langone and nurtured my developing leadership skills in navigating a complex landscape of avant-garde science in desperate need of high-quality data science and AI/ML modeling. Apostolos Georgopoulos at the UMN has been a most generous mentor and friend and encouraged this volume with great enthusiasm.

Bill Stead and Randy Miller gave me my first faculty academic home at Vanderbilt and enabled my lab there (the Discovery Systems Lab) to exist and make foundational algorithmic advances. I am very grateful for this opportunity. Stead has in addition been an inspiring role model in his unending quest to transform medicine using informatics. I am deeply appreciative of main collaborators in pursuing method development and landmark or other large-scale benchmarking: especially Ioannis Tsamardinos, Alexander Statnikov at Vanderbilt and NYU, and Sisi Ma and Roshan Tourani at the UMN. Had it not been for the extraordinary ability of Alexander Statnikov to execute benchmarking studies of immense complexity and scale in record time, a large body of invaluable benchmarking knowledge would have not existed today. Alexander, my second ever PhD student and later faculty colleague at NYU, has also made very significant methodological contributions with unique capabilities that are discussed in this volume.

I am also thankful to many of my students and mentees who took AI/ML classes from me or sought to receive scientific mentoring because invariably I learnt a lot from them, and they influenced the evolution of my teaching style and provided me with an empirical basis for how certain AI/ML topics can be best conveyed in order to help learners the most. Yindalon Aphinyanaphongs, and Larry Fu in particular, come vividly to mind, in addition, because their work allowed a joint exploration of AI/ML in health information retrieval, scientometrics, translational science, and text categorization that has stood the test of time.

I am also grateful to my extraordinary IP lawyer and friend, Laurence Weinberger, who in the context of preparing >20 patent applications has helped me and other co-inventors immensely toward clarifying and formulating very complicated technical concepts in a way that is accessible to broad audiences. Certainly, the appropriate use of AI/ML comes from theoretical knowledge but equally importantly from successful empirical applications in real-life projects. I have been privileged throughout my career to collaborate with several exceptional clinical, basic, and translational scientists. My main long-term collaborators in AI/ML applications that have most informed the present volume are Glenn Saxe (psychiatry), Boris Winterhoff (cancer precision medicine), Pamala Jacobson (pharmacogenomics), Virginia Kraus (aging and longevity), Bill Kraus (cardiometabolic outcomes), and Laura Niederhoffer (cellular senescence). Several other superb scientists

collaborated with me and my lab in shorter term but highly rewarding applied research projects. Marty Blaser (microbiomics), Ed Fisher (atherosclerosis), Steve Abramson (osteoarthritis), and Doug Hardin (text categorization) in particular gave me the opportunity to put AI to good and challenging use in the corresponding fruitful biomedical areas. Our joint work, for which I am grateful, has been a testing lab and proving ground for the principles and practices espoused in the present volume.

Although the majority of this volume was written by Gyorgy Simon and I, we had the good luck to receive wonderful supplemental material by Erich Kummerfeld, Sisi Ma, Bryan Andrews, Jinhua Wang, Steve Johnson, Dalton Schutte, Rui Zhang, Dennis Murphree, Anirudh Choudhary, Puneet Bhullar, and Nneka I. Comfere. I am very appreciative of their expert contributions to this volume.

Melissa Malkowski and Beth Madson provided us with excellent administrative support at the UMN, and Nithya Sechin's, Hashwini Vytheswaran's and the whole editorial support and production teams' work from the Springer side is much appreciated.

The UMN OACA-Office of Academic and Clinical Affairs (led by VP and Dean Dr. Jakub Tolar) and the UMN CTSI (led by Dr. Bruce Blazar and Daniel Weisdorf) provided the Institute for Health Informatics with an environment where we can explore and put novel ideas to test and practice, typically at large scale (across all of the health sciences institutionally and beyond). I am also grateful to OACA for providing financial resources to make this book open access.

Finally, I am grateful to the two eminent colleagues who graciously provided forewords to this volume. Professor Wendy Chapman and Professor George Demiris are distinguished researchers, scholars, innovators, and academic leaders in health AI. They graced this volume with opening remarks that set the stage for the book in intellectually stimulating, compelling, and insightful ways.

It is my hope that the material here will contribute to developing, validating, evaluating, and applying health AI/ML with rigor, leading to performance and safety guarantees for the benefit of patients, science, and the industry. Hopefully, the reader will find the book to be a valuable companion in what is surely going to be a long and exciting journey for the field.

Constantin Aliferis

Contents

Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML

Constantin Aliferis and Gyorgy Simon

Abstract

In the opening chapter we first introduce essential concepts about Artificial Intelligence and Machine Learning (AI/ML) in Health Care and the Health Sciences (aka Biomedical AI/ML). We then provide a brief historical perspective of the field including highlights of achievements of Biomedical AI/ML, the various generations of AI/ML efforts, and the recent explosive interest in such methods and future growth expectations. We summarize how biomedical AI and ML differ from general-purpose AI/ML. We show that pitfalls and related lack of best practices undermine practice and potential of Biomedical AI/ML. We introduce high-level requirements for biomedical AI/ML and 7 dimensions of trust, acceptance and ultimately adoption, which serve as the driving principles of the present volume. We outline the contents of the volume, both overall and chapter-by-chapter, noting the interconnections. We discuss the intended audience, and differences from other AI/ML books. We finally discuss format, style/tone, and state a few important caveats and disclosures.

Keywords

Machine learning (ML) · Artificial intelligence (AI) · Algorithm · Model · AI trust · Acceptance · Adoption

C. Aliferis (✉) · G. Simon
Institute for Health Informatics, University of Minnesota, Minneapolis, MN, USA
e-mail: constantinaibestpractices@gmail.com

G. J. Simon, C. Aliferis (eds.), *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, Health Informatics,
https://doi.org/10.1007/978-3-031-39355-6_1

What Is Machine Learning? Algorithms, Programs, and Models

A myth that was pervasive in earlier stages of the history of computing was that computers can only solve problems (or perform actions) that a human programmer had specifically instructed them how to tackle. As even the broad lay audience can appreciate circa 2023, computers equipped with Machine Learning (ML) capabilities can learn from data how to perform intelligent tasks and perform complicated problem solving on their own [1–3]. Whereas the ML algorithms are typically programmed by humans, once implemented, these types of software can interpret data in ways that far exceed the capabilities of their human creators, not just in terms of speed but also by making inferences that are qualitatively superior to humans, for example by avoiding human cognitive biases and blind spots and performing inferences that humans do not do at all or are not good at performing (e.g., pattern recognition in very high dimensional spaces sometimes in the 10^6 variables scale or more) [4]. In addition, whereas ML programs are currently typically presented with data prepared by human operators/analysts, it is entirely possible (and in some cases routine) to collect data on their own, or instruct human operators to collect data needed for problem solving [5–7].

Definition
Machine Learning (ML) is the science and technology of computing systems that learn how to solve problems by analyzing data related to the problems.

To go into slightly more detail, ML **algorithms** implemented in ML **programs and systems**, use so-called **training data** from which they build problem-solving **models**. It is useful to understand these important concepts further since there is confusion among the non-technical audience (including biomedical scientists and healthcare providers and administrators).

A ***computer program*** is a set of instructions that a computer can understand and execute toward performing a task intended by the program [8]. For example, a program written in the language *Python* that instructs a personal computer with the ability to execute Python commands, how to sort a set of numbers into descending order.

A software ***computer system*** is a complex set of interconnected programs that perform a number of interrelated functions. For example, an Electronic Health Record (EHR) system comprises a set of programs and databases that manage patient data to support patient care, record actions for compliance, perform billing and reimbursement, etc.

A ***computer algorithm*** is a generalized (programming language-agnostic) set of computer instructions designed to solve a class of problems. Computer algorithms are presented in a form (so called pseudo-code [9]) that is geared towards being interpretable by humans. In contrast, computer programs are written in a computer

language that is interpretable by computers. For example, the "*quicksort" number sorting algorithm* is a set of instructions, written in a format meant for human interpretation, that can be translated to any general-purpose computer programming language. Furthermore, the quicksort algorithm needs to be translated by a programmer into a programming language, a process known as *implementing the algorithm,* before it can be executed by a computer. In another ML example, the *ID3 algorithm creates, from previously-diagnosed patient data, decision tree models* that can be used for diagnosing new patients.

An ***AI/ML model*** is therefore a computable representation of some problem-solving domain so that when informed with a set of inputs describing specific instances of the problem space, outputs solutions to those. These models are created by hand by using AI knowledge and other *knowledge engineering* methods and tools, and in the case of ML fully automatically from training data [10].

Computer algorithms [9, 11] *have a number of distinguishing characteristics from computer programs:*

(a) They (typically, and formally) not need be described in a specific programming language, but in *pseudo-code*, as previously explained.
(b) They represent a potentially infinite set of programs that can be implemented in every applicable programming language and computing environment.
(c) When properly constructed, they have well-defined properties that guarantee performance, error free (or error-acceptable) operation, generalizability etc. (more on this later in the book).
(d) When properly *implemented* (i.e., translated to a specific programming language) they *guarantee that the algorithm properties are imparted in the particular program that implements the algorithm.*

The field of ***Design and Analysis of Algorithms*** studies the properties of algorithms (and associated ***data structures*** [9, 11] (i.e., ways to represent and organize data for storage, retrieval and other operations)) and methods to design algorithms for specific problems so that desired operating characteristics (e.g., speed, memory usage, accuracy etc.) are achieved.

Pitfall 1.1
Very commonly in the commercial healthcare space a computer program or system implements **unspecified, undisclosed or insufficiently-analyzed algorithms**, hence no-one knows what the properties of the program are.

In chapter "Foundations and Properties of AI/ML Systems" but also in several other places of the present volume, we will address the fundamental issue of guaranteed properties of AI/ML systems and best practices enforcing those.

Pitfall 1.2
In healthcare and the health sciences, ***clinical algorithms*** are often confused with ***computer algorithms*** (including ML algorithms). A *clinical algorithm* [12, 13] describes diagnostic, risk assessment, preventative, treatment or other actions needed to care for patients with specific diseases, usually in the context of evidence-based guideline-driven medicine. It can be written in human language or specialized *computable languages*. A clinical algorithm is a human-assistive *decision model* and is not an algorithm that can learn how to solve the problem from data. Finally, a model produced by a ML algorithm can serve as a clinical algorithm in a health care setting.

ML algorithms are therefore implemented in ML computer programs that when presented with training data, learn and output decision models. In chapters "An Appraisal of Operating Characteristics of Major Machine Learning Methods Applicable to Healthcare and Health Sciences", and "Foundations of Causal Machine Learning" we will review major ML families of algorithms and describe the types of models they output. In several other chapters of the present volume we will discuss specific algorithms and models and their characteristics and optimal use.

Well-constructed ML models do have general applicability beyond the training data, otherwise they would be just a catalogue of past problem instances and their solutions, without the ability to be used for new problem instances. **Machine Learning theory** [14, 15] provides results and techniques that enable and ideally guarantee the generalization properties of ML models beyond the training data.

Artificial Intelligence (AI); Types of AI and ML Tasks; on the Pervasive Applicability of ML and AI

The language we adopted on ML algorithms as a means of solving problems, has a deeper significance as it relates to the definition of Artificial Intelligence. AI depending on the context, the era and the author, has been viewed as (a) the field of science and technology that investigates the creation of fully autonomous computer systems (i.e., "Intelligent Systems"); (b) exhibiting intelligence capabilities indistinguishable to those of humans (i.e., so-called "hard AI"); (c) providing the empirical means for putting forth and testing under controlled (computer lab) conditions theories of cognition; or (d) creating programs capable of solving hard (computational, mathematical, cognitive, decision, optimization and other inferential) problems [2]. Operationally we will adopt the following view on AI:

Definition
Artificial Intelligence (AI) is the science and technology of computing systems that can autonomously solve hard inferential problems.

Such problems historically have been associated with the prerequisite of "intelligence".

From a perspective of organization of scientific fields and their relationships, ML is one of the fields of AI which, in turn, is a field of Computer Science (CS). At the same time, ML is a core part and arguably the most important component (along with statistics) of the nascent field of *Data Science*.

Definition
Data Science is the field of science and technology that studies the: (a) design and execution of data measurements, sampling/collection; (b) data representation and management, harmonization, secure storage and transmission; (c) analysis, interpretation, and (d) deployment of results in applied problem-solving settings.

Data Science spans and connects several fields including ML and statistics, as well as parts relevant to data sampling and modeling from applied mathematics, operations research, econometrics, psychometrics, decision sciences, information science, scientometrics and bibliometrics, statistical genetics and genomics, etc. [16, 17].

Figure 1 shows the relationship among Computer Science, AI and ML. As can be seen in the figure, both AI and ML are very diverse and developed, comprising many types of research, systems, algorithms, and applications.

An important pitfall (the importance of which will become abundantly obvious in this volume) is to consider one very narrow subfield, for example Deep Learning, as the totality or the main focus/armamentarium of all of ML and AI, or as another example, considering ML as the totality of AI. This has serious consequences as we will see in this book because it prevents users of AI and ML to have the *right perspective in which a plurality of methods can be brought to bear on solving problems by matching the right method to the problems at hand.*

Pitfall 1.3
Very commonly novice advocates of ML and AI, or vendors promoting certain products will present the whole field as being about one narrow technology or a small set of tools, ignoring the broader spectrum of available options that can solve the problem at hand. The many options available however, have hugely varying performance characteristics that need careful consideration as *no single class of methods is suitable for all biomedical problems*.

In the present book we place a heavy emphasis on data-driven forms versus expert-knowledge-driven AI, for the following reasons: first, modern health AI is predominantly data driven and will continue to be so in the foreseeable future. Second, ML is vastly more scalable than expert knowledge-driven construction of AI systems. Third, ML has many pitfalls and intricacies that require addressing. Fourth, ML is

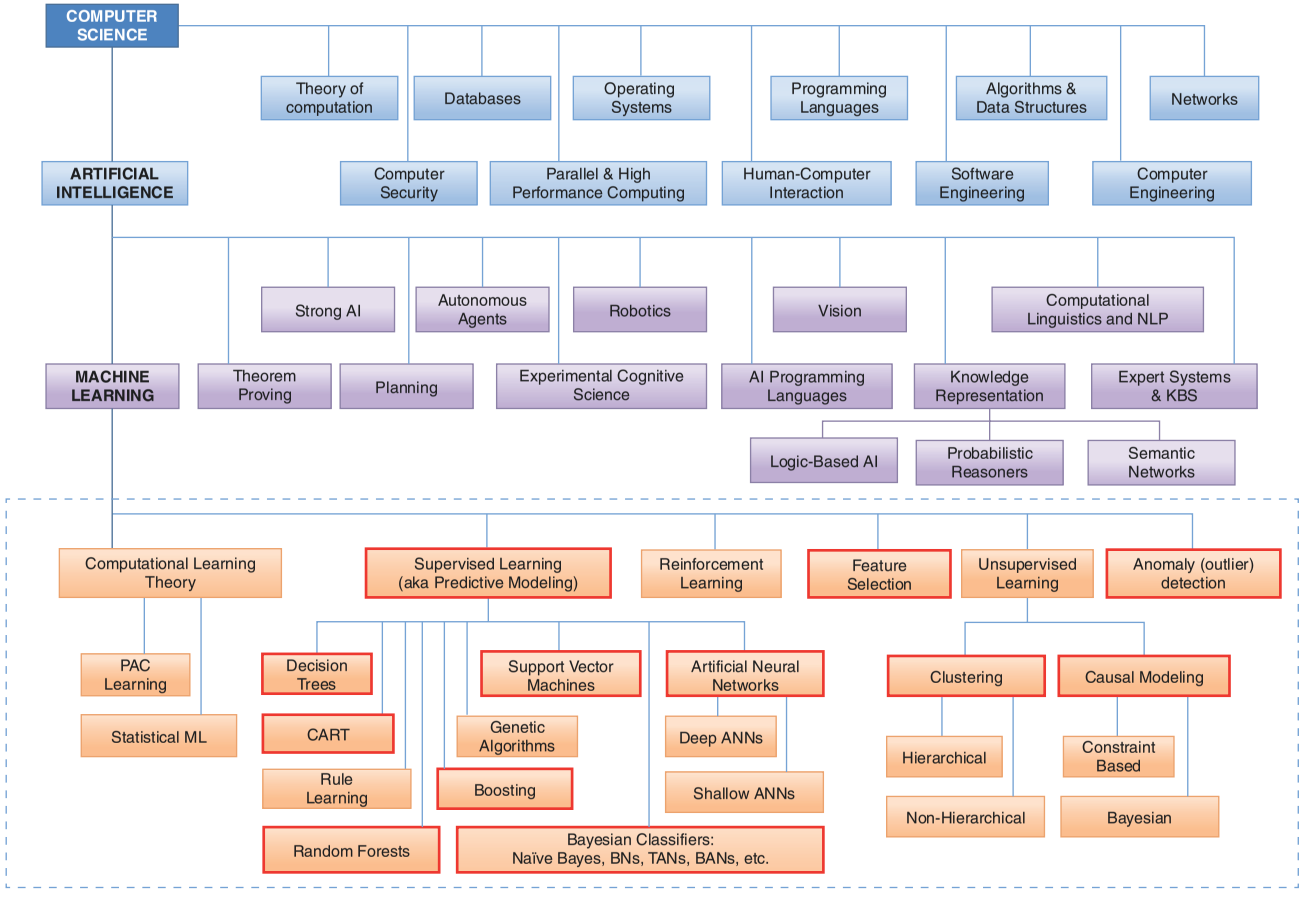


Fig. 1 Relationship and contents of Computer Science, AI and ML. In red rectangles, ML subfields with particular relevance to health sciences and healthcare

also an important component of other forms of AI (e.g., NLP, computer vision, robotics). Finally, the highlighted pitfalls and best practices are often useful for both ML and other forms of AI.

Readers not already deeply familiar with AI/M applications in the health sciences and care delivery are likely to be surprised by the *extraordinarily wide range of applications* of these fields. This *pervasive applicability of ML and AI* is *not accidental,* however. It can be immediately grasped once one considers that both health sciences and care are fundamentally designed to pursue discovery and application of predictive and causal knowledge. **Predictive modeling** encompasses diagnosis, prognosis, forecasting and general pattern recognition [1–4]**. Causal modeling** [18–20] seeks to discover cause-effect relationships, to quantify their effects, and to choose among various interventions those that will maximize some desired outcome. It encompasses discovery of laws of biology, therapeutics, understanding the factors that drive system and patient-level outcomes such as development, treatment and prevention of disease at the individual level. At the level of the system of care, they encompass intervention on factors that affect quality of care, costs, reimbursements, patient experience and all other desiderata of health systems [21].

Neither General AI/ML, Nor Biomedical AI/ML Are New. Highlights of Achievements of Biomedical AI/ML

The general public became aware of AI and ML as a viable technology in very recent years as a result of the emergence of commercial offerings backed by established corporations as well as numerous startups catering to healthcare systems and health research organizations. The scope of adoption and widespread use of AI and ML, is currently breathtaking and includes: autonomous vehicle navigation (cars, airplanes, industrial robots), cybersecurity, fraud and spam detection, financial applications, internet and e-commerce applications, manufacturing, games, education, legal, and numerous other applications [22].

In healthcare and the health sciences, **examples of successful applications** include automated diagnosis, prognosis, treatment selection (using as inputs: coded clinical data, text reports, images, omics data, etc.) [23]; discovery of gene mutations causing specific forms of cancer or other disease [24]; precision medicine tests (e.g., genes' expression level patterns determining response to a treatment used for treatment selection) [25]; automated evaluation of scientific papers to determine whether the research design was good [26]; annotating genomes and other genetics applications [27]; predicting tertiary & quaternary protein structure from amino acid sequence [28]; predicting drug-drug and drug-food interactions [29]; medical imaging [30] and numerous other applications which we will cover in depth in the present volume.

The advent of **big data** in particular, in healthcare and population health (e.g., EHR, sensor, environmental, social networks) and the health sciences (e.g., genomics, proteomics, metabolomics, microbiomics, copy number variation, and other "bulk" and single cell "omics" data, deep sequencing databases, research consortia data, etc.) has simultaneously demanded the development of high-quality scalable

analysis methods and strongly incentivized their deployment at scale [31]. In the last 20 years t*here is a synergistic co-evolution of big data generation/capture and ML-driven analysis and discovery* with **key themes of modern health science and health care** such as: rational drug development [32], modern post-sequencing era genomics precision and personalized medicine [25], learning health systems and care cost/quality/experience improvements [33], to mention just some of the key developments that depend on ML and AI and that are foci of the present work.

To give a sense of the immense scope and rapid maturity with respect to health outcomes the following searches[1] return:

((“outcomes” or “health services”) and “machine learning”)	→ 6255 results (most since 2015)
((“outcomes” or “health services”) and “machine learning”) and “systematic review“	→ 240 results

These systematic reviews (not cited explicitly here for space, but readily retrievable from PubMed with the stated queries) represent broad application areas with significant and diverse bodies of work. They include predictive, prognostic, diagnostic and etiologic outcomes modeling in:

Neurosurgical outcomes, depression, obesity, surgical outcomes, EEG classification, dermatology, urology outcomes, suicide prevention, Covid mortality, autoimmune disease outcomes, stroke, various cancers, dementias, orthopedic surgery, heart failure outcomes, pregnancy outcomes, imaging and radiomics analysis, sepsis in the ICU, managing covid-19, assessing physician competence, hematopoietic stem cell transplantation (HSCT), various infectious diseases, cardiac surgery, management and treatment of burns, infant pain evaluation, management of heart failure patients, bipolar disorder, degenerative cervical and lumbar spine disease, cardiovascular outcomes from wearable data, psychosocial outcomes in acquired brain injury, acute gastrointestinal bleeding, personalized dosing of heparin, Parkinson’s disease, genetic prediction of psychiatric disorders, diabetes, clinical deterioration in hospitalized patients, community-based primary health care, palliative and end-of-life care, hypertension, graft failure following kidney transplantation, outcomes in neonatal intensive care units, degenerative spine surgery, predicting fatal and serious injury crashes from driver crash and offense history data, health care spending, extraction of data from randomized trials, improving medication adherence in hypertensive patients, neighborhood-level risk factors, gait analysis, wearable inertial sensors to quantify everyday life motor activity in people with mobility impairments, outcome prediction of medical litigation, rheumatic and musculoskeletal diseases, analysis of patient online reviews, chronic low back pain, risk of readmission and several other topics.

[1] Conducted on June 2, 2022

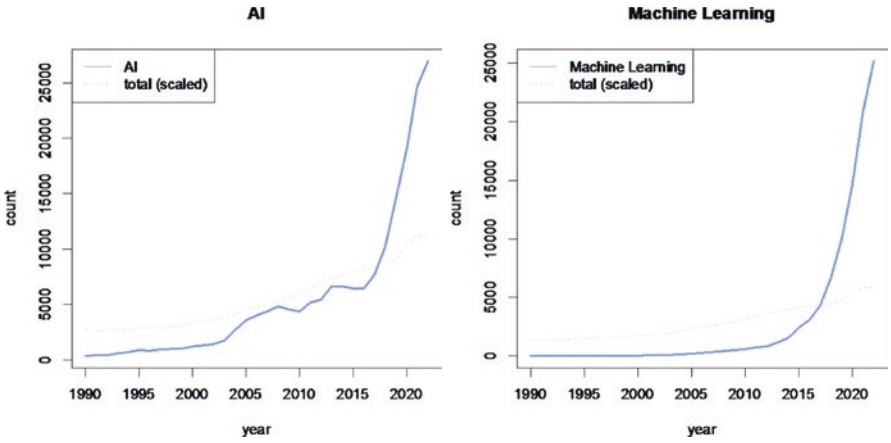


Fig. 2 Number of PubMed publications with MeSH term "Artificial Intelligence" (left) and keyword "Machine learning" (right) in the years between 1990 and 2022. To facilitate the comparison of growth between AI/ML and publications in general, the black dotted line represents (a downward scaled version) of the total number of publications in PubMed

PubMed is also informative on relative literature volumes pertaining to AI/ML methods and applications, and their trends[2]:

Figure 2 illustrates the explosive growth of ML and AI through the number of Pubmed publications over the years between 1990 and 2022. The blue line represents the number of publications for AI [MeSH] (left) and Machine Learning [Keyword] (right); the black dotted line represents the scaled number of total citations (from any field). The rate of growth in AI and ML far outpaces the overall growth rate of publications since 2015.

In Fig. 3, we show how the growth in health AI is distributed over some of its subfields. Machine Learning enjoys most of the growth, with Natural Language Processing (NLP) and Image Analysis following closely. Modern advances in Machine Learning, Deep Learning in particular, serve as an enabling technology for both of these subfields. Other subfields, such as Knowledge Representation exhibited a more modest growth, while Expert Systems appears to have experienced negative growth since they are being replaced by ML. We need to remember that PubMed focuses on biomedicine.

In terms of absolute volume of publications, the following tables provide relevant data (Table 1):

These results are to some degree an artifact of the indexing of articles employed by Pubmed. For example:

"clustering" (which is a form of ML)	→ 489,442 results
"Artificial neural network" (Mesh term)	→ 23,746 results
But:	
"Deep learning" (Mesh term)	→ 40,377 results

[2] conducted on June 2, 2022, and using Mesh index terms when available

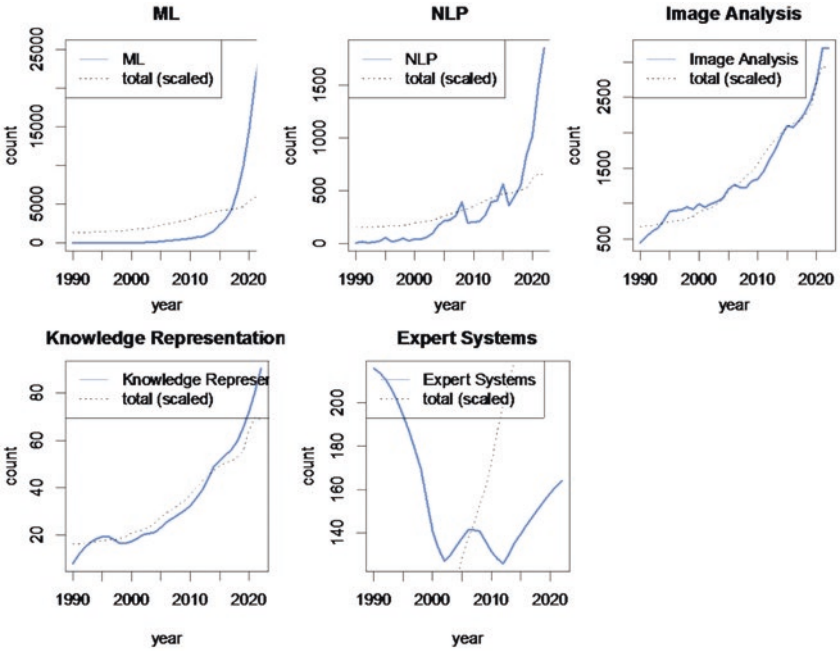


Fig. 3 Trends of publications in various subfields of AI between 1990 and 2022

Table 1 Health AI/ML publication volumes

"machine learning" (Mesh term)	→ 89,260 entries
artificial intelligence (Mesh term)	→ 165,990 results
("artificial intelligence" or "machine learning") (keywords)	→ 113,531 results

Caveat: Deep Learning is a special type of artificial neural network, which entails that if indexed properly the entries indexed by "artificial neural network" should be a strict superset of the entries indexed by "Deep Learning".

As to articles with key types of ML, in addition to the ones above we see:

"Decision tree" (Mesh)	→ 23,206 results
"Support vector machine" (Mesh)	→ 22,675
"Genetic algorithm" (Mesh)	→ 90,728 results
"Random forest" (Mesh)	→ 23,357 results
"Bayesian network" (Mesh)	→ 10,076 results
"Bayesian classifier" (Mesh)	→ 12,503 results
"Granger causality" (Mesh)	→ 3810 results

With regards to major types of AI in addition to the ML ones mentioned we see:

"Autonomous robot"	→ 2743 results (most since 2005)
"Expert systems"	→ 20,627 results (most since 1990)
"Knowledge representation"	→ 12,526 results (most since 1990)
"Semantic network"	→ 6482 results (most since 2005)
"Natural language processing"	→ 9659 results (most since 2005)

The exponential-rate growth of most of these methods in the biomedical literature started and took place for the most part in the last 15-to 30 years. It is worth noting that in the field of Biomedical Informatics (aka Health Informatics) seminal publications in ML and AI appeared as early as in 1959, however. The 1959 article by Ledley and Lusted [34] is particularly important since it anticipated many of the key themes and methods that were rediscovered (and in some cases ignored) by modern commercial vendors and academic or industry adopters of biomedical AI/ML 63 years later.

Similarly, the 1961 article by Warner et al. is [35] is a seminal paper for the field of Medical Informatics and describes a ML-based approach to improving diagnosis in a significant disease, later expanded to many other diseases in the 60s all the way to the 80s by these and other pioneering investigators.

Another important seminal early work, this time in human expert knowledge-driven AI was the work by Miller et al. [36]. This notable AI system employed heuristic knowledge representation and reasoning that managed to perform at a hard reasoning task (challenging diagnostic cases across all of internal medicine) at a level that matched or in some cases exceeded expert physicians. This system was emblematic of the efforts in the 70 s and the 80 s to create AI that was driven by extracting and representing in computable form human expert problem solving. These efforts were followed by newer ML-based systems with the advent of more capable ML algorithms and representations taking advantage of increasing amounts of training data, such as Bayesian Networks and other sophisticated Bayesian classifiers [37, 38], early multi-layered artificial neural networks [39, 40], decision tree learners and other ML algorithms [1–4] that vastly outperformed in ease of use, cost-effectiveness and accuracy early ML algorithms and human expert knowledge.

The "Perfect Storm" for Biomedical AI/ML

The ability to capture massive Big Data (as indicated above) in the 2000s and onward, fueled the explosive application and refinements in kernel-based nonlinear classifiers (e.g., SVMs) [1–4], boosting algorithms, causal discovery and inference algorithms [18–20], deep artificial neural networks [39, 40, 41], significant extensions to decision trees (Random Forests [42]), regularized versions of statistical regression algorithms [43], and other methods that could now manage tens, hundreds and in some cases millions of variables

with modest compute requirements and most importantly with extreme tolerance to low sample sizes without overfitting [44]. These methods exhibited properties that classical statistical science and practice previously considered impossible [4, 14]. Some types of newer algorithms also had the ability to discover causality without experiments which have also been considered previously impossible [18–20] and newer scalable causal algorithms that made application to high dimensional data as well as scalable hybrid predictive and causal modeling feasible [45–48]. This "perfect storm" for biomedical AI/ML led to its current cycle of explosive growth. It is not surprising that the above developments in general AI and ML are closely associated with the work of **9 Turing award recipients** (Marvin Minsky, John McCarthy, Herbert A. Simon, Edward Feigenbaum, Raj Reddy, Judea Pearl, Yoshua Bengio, Geoffrey Hinton, Yan Le Cun), and **7 Nobel Prize recipients** in economics: (Herbert A. Simon, Daniel Kahneman, Clive Granger, Thomas A. Sargent, Christopher A. Sims, Joshua Angirst, Guido Imbens) solidifying thus the scientific credibility and immense importance of these methods.

Yet, despite all of this scientific activity and accomplishments (>three million entries in Google Scholar mentioning ML and > three million mentioning AI as of 2023), these fields have been presented to the general public and the non-experts, as either entirely new, or they have been presented as invented recently in the laboratories of a handful of commercial companies. This brings us to another important pitfall:

Pitfall 1.4
The field of general and biomedical AI and ML is not a new one. Ignoring the vast literature and re-inventing the wheel in some cases, fails to take advantage of a wealth of very substantial prior work that can inform effective, safe and cost-effective use. Methods that have undergone rigorous development, analysis and validation over many years have in general better-understood properties, better performance robustness, and better operating safety characteristics than newer less well-developed methods.

Best Practice 1.1
When considering development or application of AI/ML, ensure that it is informed by the well-developed and evaluated pre-existing science and technology.

Differentiation of Biomedical AI and ML from General-Purpose AI/ML

Another important pitfall we will address in this volume is the distinction between general purpose AI & ML versus biomedically-tailored AI & ML.

Pitfall 1.5
Biomedical AI and ML have specific requirements and adaptations tailored to the goals of healthcare and of health sciences discovery. AI and ML devised and tested in unrelated fields have very different properties and do not ensure the goals of healthcare and health science applications.

A summary of the adaptations and differentiation, to be elaborated further in this volume, is as follows:

Biomedical AI/ML:
(a) Is driven by, and has strong interactions with clinical objectives, health economics, and healthcare delivery within specific health systems.
(b) Requires the ability to handle very large dimensionalities (i.e., number of variables).
(c) Requires the ability to handle very small sample sizes without overfitting.
(d) Must be equipped with the ability to discover and model causality, since it is often necessary to estimate effects of interventions.
(e) Requires specialized data operations and the ability to handle diverse data types including clinical coded data, text, imaging, biomolecular data, and combinations.
(f) Places great emphasis on accuracy, cost-effectiveness, quality control and de-risking.

All of these requirements will be addressed in detail in the present volume.

Future Potential of Biomedical AI/ML

As widespread and rapidly growing biomedical AI/ML is, it has potential for orders of magnitude more growth. For example, compared to classical biostatistics, AI/ML has a smaller data science footprint in biomedical literature as revealed by the following PubMed searches:

"Cox regression" (Mesh)	→ 105,385 results
"Chi square test" (Mesh)	→ 116,546 results
"ANOVA" (Mesh)	→ 522,350 results
"Regression" (Mesh)	→ 1,011,918 results

AI/ML methods are rapidly substituting complex inferential statistics and/or are extending them in substantial ways, however. There are many signals for the forthcoming growth of biomedical AI/ML. We mention a few strong indicators:

(a) In the domain of *molecular profiling for precision medicine* [25], only just a handful of such profiles have been brought to market so far, although, >170,000 molecular signature papers have been published (many of them showing feasibility of clinical signatures). The number of patient-touching precision tests expected to be in use *at any given time* in the future, if estimated as the combination of (*diseases * drugs),* exceeds 100,000.
(b) Other areas where massive biomedical AI/ML growth is expected include *health systems outcomes improvement* [21] with hundreds of thousands of AI/ML models conceivable to be developed and deployed in the future, assuming that at least one model will be deployed for every major decision/disease/outcome combination that is affecting patients, units and systems.
(c) Similarly in the space of *precision clinical trials* [25] currently much less than 1% of all trials are precision trials and migrating to this model of clinical therapeutics validation will necessitate application of AI/ML at scale across the research domain (>20,000 new large new trials annually).
(d) In *radiology*, we can safely expect a massive transition to computer-assisted (and in some cases fully automated) interpretation of clinical or research imaging, across many health science and care domains.
(e) In *single-cell transcriptomics and other omics (including "multiplexed" combinations) and their spatiotemporal extensions*, the use of AI/ML is absolutely necessitated by the immense dimensionalities (> 5000 cells * 10,000 molecular probes with current technology yields dimensionalities of > 50 million variables *per patient/research subject*). Single-cell omics technologies are the successor of bulk deep sequencing technologies (themselves the successor of microarray technologies) and according to all indications, will be driving biological discovery for decades to come. If these precursors are an indication, then 100,000 s of applications of AI/ML single-cell omics are to be expected [49, 50].
(f) The vast majority of models referenced in the hundreds of systematic reviews (covering thousands of modeling studies) mentioned in section "Neither General AI/ML, nor Biomedical AI/ML are New. Highlights of Achievements of Biomedical AI/ML", are pre-clinical or otherwise feasibility efforts as stated in the corresponding systematic reviews. These reviews found very promising results but identified that the models have not yet reached the clinically mature stages needed for broad deployment. Closing this gap will undoubtedly be a large part of the future of health AI/ML.

Pitfalls and Related Lack of Best Practices Undermine Biomedical AI/ML. AI/ML Trust and Acceptance

The strong and sustained trends outlined above in the literature and commercial AI/ML, suggest that *AI/ML will grow to be a science and technology that permanently and irrevocably enables progress across all aspects of health science research and health care delivery. There is an ethical and utilitarian necessity therefore for this science and technology to be executed with an emphasis on meeting performance, safety, and cost-effectiveness requirements.*

Performance requirements entail that AI/ML has to be accurate and minimize false positive and false negative results. For example, the massive application of AI/ML if allowed to generate false positives will drown the research system in noise, rendering the space of scientific investigation a destructively low signal-to-noise environment. Avoidable false negatives due to poorly thought AI/ML represents the space of corresponding opportunity cost.

Safety requirements entail that AI/ML systems applied in clinical care settings as well as preventative policy and other public health settings should not allow for any avoidable errors of either wrong treatment/intervention decisions that incur risk to patients, populations, or systems of care. They should also not allow errors of failing to identify opportunities to improve patient/human subject health (for example, diagnosis of treatable diseases, opportunities to improve cost and quality of the system of care) as such failures translate to decreased life expectancy/quality of life of individuals, populations and negatively affect the health systems that care for them.

Cost-effectiveness requirements entail that AI/ML systems applied in care settings as well as health science discovery should not be wasteful in either time-to-results, or compute requirements, or sample size requirements, or cost of decisions. The costs of such inefficiencies can quickly become unmanageable.

Perspectives on building trust, adoption, and acceptance of technology by humans (as individuals or at the society level) are diverse and encompass performance, economic, legal, accountability, ethical, psychological, social and other factors [51–59]. Operationally we frame the above requirements from the perspective of stakeholders using a Biomedical *AI/ML trust and acceptance framework, comprising the following 7 dimensions:*

1. **Scientific and Technical Trust and Acceptance.** AI/MLmodels must be accurate at deployment (e.g., low error rate, not falling outside their boundaries of strong performance (known as their "knowledge cliff")).
2. **Health System Trust and Acceptance** AI/ML models must be safe, cost-effective and well-embedded in systems of health with clear benefits and without unexpected/unacceptable risks, disruptions or other negative consequences.

3. **System-of-science Trust and Acceptance.** AI/ML models must be safe and cost-effective to operate in the system of science without unexpected/unacceptable risks and consequences.
4. **Beneficiary Trust and Acceptance.** AI/ML models must be accepted by patients and human subjects individually and at the community level.
5. **Delivery and Operator Trust and Acceptance**. AI/ML models must be accepted by clinicians and scientists.
6. **Regulatory Trust and Acceptance.** AI/ML models must be compliant to applicable laws and approved by regulatory bodies.
7. **Ethical Trust and Acceptance.** AI/ML models must be non-discriminatory and must promote health equity and social justice related to health science and care (e.g., by being non-discriminatory on the basis of race, socioeconomic factors, gender, etc.).

In their 2022 program solicitation (NSF 22–502), entitled "*National AI Research Institutes Accelerating Research, Transforming Society, and Growing the American Workforce*", the National Science Foundation (NSF) acknowledged that identifying, prioritizing, and satisfying the fundamental attributes that render an AI trustworthy are open research challenges. Notably the program described trustworthiness through examples from other areas of mature technology such as automobiles or electric lighting. These systems are trustworthy, "*because they are reliable, predictable, governed by rigorous and measurable standards, and provide the expected benefits. Facilitated by basic knowledge of their operation, we are familiar with common faults and how to address them, and there is infrastructure to deal with problems we cannot handle ourselves.*" It's a compelling proposition that health-related AI should have similar characteristics.

The whole purpose of the present volume therefore is to outline a set of **preferred practical requirements and methods ("Best Practices') that will move us forward to biomedical AI/ML that avoids pitfalls and achieves the 7 dimensions of trust, acceptance and eventual adoption**. In order to justify the requirements and assemble/build the proposed best practices we will also need to introduce a body of necessary technical background knowledge.

Intended Purpose and Audience of the Book

AI & ML are extremely popular topics and numerous books are available, generally falling into four categories:

1. *Hands-on instructional texts on how to build a general-purpose AI system*, e.g., using a particular Python software package. Such books are not specific to health care or health sciences and their specific problems; nor do they provide a strong conceptual understanding of how different models work and how this relates to their applicability to different health problems.

2. *General purpose data mining, AI, and ML textbooks*. Such books do not relate to health care or health science and do not give advice on how to develop models specifically for health care or any other area: they focus on a very narrow aspect of model development. Moreover they do not differentiate between feasibility and exploratory analysis from the much more mission-critical clinical and other high-stakes modelling settings that are so prominent in healthcare and the health sciences.
3. *Health care analytics and the promise of AI in health care*. Most works in this category focus on conventional (reporting and compliance) analytics. A few address the new capabilities brought by AI/ML. They are not designed to provide the reader with a deep understanding of what the (primarily) technical challenges are in health care AI, or what the pitfalls are and how specifically and systematically to avoid them.
4. *Bioinformatics and genomics discussing AI/ML approaches in that context.* These are technical books that typically do not focus on systematic methodologies for ensuring appropriateness of various AI/ML methods, or their methodological underpinnings.

From our review of the literature there are more than 100 textbooks in 2023 in press in the above categories. We view them as very useful background for broad fundamentals and/or context of use: from such books readers can learn basic concepts of general machine learning, and can also learn how to build certain types of models; our present effort however focuses on knowledge and practices specific to how health science, clinical, translational, and healthcare AI/ML systems differ from the general-purpose AI/ML. The book aspires to impart comprehensive and in-depth knowledge on how to build robust and safe models for the high-stakes settings in health science and care, and to evaluate the strengths and weaknesses of such models produced by others. We will cover both general (mostly immutable) scientific principles as well as specific technical guidance that may evolve over time.

More precisely, we envisioned the present volume to be the first book in the field to provide guidance for the following concepts/topics:

1. The critical differences between general-purpose AI & ML and medically-applicable AI & ML.
2. Building models that can be applied with minimal risk in high-stakes settings including clinical applications, healthcare system optimization, and discovery of clinical modalities.
3. Models that integrate multi-level, multi-modal clinical and molecular data.
4. The importance of data design and post-modelling safeguards for high-stakes applications.
5. Common limitations and remedies of efforts (commercial and academic) in the field.

6. In-depth presentation of not just predictive but also causal and hybrid causal-predictive methods.
7. A comprehensive summary and critique of operating characteristics of all major AI & ML methods.

This volume emphasizes the need and methods for biomedical AI/ML to:

1. Be intentional, with well-defined and meaningful goals and metrics of success.
2. Effectively manage risk for errors that may affect adversely the health of patients, the effectiveness of health systems, and the effectiveness of the system of science.
3. Operate in real-life (as opposed to idealized and simplified theoretical) health care as well as in health science discovery ecosystems.
4. Develop within a lifecycle that starts from problem statements and needs all the way to successful deployment and continuous iterative improvement.
5. Prevent and overcome the fundamental dangers of over fitting and under fitting as well over confidence in models and under performance of models.
6. Have known properties that guarantee performance and safety.
7. Be based on sophisticated and appropriate data designs.
8. Be differentiated along the levels of systems/stacks, protocols, algorithms, models.

We adopt an interdisciplinary perspective, using and integrating methods from Data Science, Computer science (Machine Learning, AI, predictive analytics), Statistics, Epidemiology (study design), Clinical Decision Support, Bioinformatics, Clinical and Health Informatics, Genomics, Learning Health Systems, and Precision and Personalized Medicine.

Our intended audience comprises all stakeholders to the healthcare and health science ecosystems: (a) *Applied and research Health Data Scientists* working in industry, academia, and healthcare. (b) *Clinicians/Professionals/Practitioners* who are called on to evaluate, select, and use AI&ML based decision support. (c) *Healthcare and translational (e.g. pharmaceutics and biotechnology) industry leaders/ administrators* including but not limited to IT leaders who wish to evaluate and deploy competing technologies in medical AI&ML. (d) *Educators and Students* in informatics, ML & AI, health economics, health business administration, and data science. (e) Funding agency officers. (f) Journals and their editors. (g) Regulatory agency officers. And (h) Community members, representatives and advocates.

We elected to make this book an ***open access*** one, ensuring that all members of our intended audience can access this volume without financial restrictions.

Outline of the Book: Style, Format, and How to Read

The book is organized in three parts (with a total of 18 chapters): Foundations, Modelling, and Implementation. Each chapter typically covers several of the following: technical didactic exposition, case studies (of success and failure varieties), related pitfalls discussion, best practices addressing the pitfalls and serving the trust principles, along with literature references and occasional discussion thereof. We also provide brief chapter abstracts (at the start of each chapter), assignments for classroom use, and recapitulation of concepts, definitions, pitfalls and Best Practices (at the end of each chapter).

Educators may wish to use the book in whole or in part as classroom textbook. Features supporting classroom use include:

1. Consistent structure and tone to the chapters. The two main authors have written the majority of the material and have co-authored or edited the contributing chapters to harmonize the content and style across the volume.
2. Practice questions, discussion topics and assignments. Some of those are more conceptual and open-ended (e.g. appropriate for less technical learners) and some are more technology-oriented (e.g. targeting learners who need to develop technical knowledge and skills).
3. Comprehensive coverage of the topic, not just the methods that the authors have invented, have used, or prefer.
4. In the future we intend to provide an "official" answer key to the assignments and discussion topics of this volume.

Because our intended audience is very diverse, we make every effort to use plain language with minimal jargon and to keep mathematical, statistical and computer science technical details at a minimum. This does not mean that we shy away from presenting formulas, algorithms, and theorems. However, when we do so, we present them only when they are necessary for making sense of the Pitfalls/Best Practices in discussion. We also sought to use the simplest language possible that does not sacrifice validity. We also introduce background we think is required to understand these technical elements and emphasize the intuition and their practical consequences behind them.

The style and level of detail has been ground-tested on our teaching these concepts (for a combined 30+ years) in a variety of settings and audiences (e.g., from undergrad college interns to professional programmers, to graduate students in data science fields, to medical residents, to health sciences faculty, and to national tutorials with mixed health care and health science audiences). As is expected, our writing reflects our own formal training in these fields (spanning 27 years combined). More importantly, both main authors of the present volume are working scientists who have led and are active in many R&D method/technology and applications projects. These have occurred in the health sciences domain (mostly funded by the NIH and the NSF) but also in industry and in health care contexts. These experiences have provided us with a wealth of knowledge about the roadblocks that our intended audience routinely faces, and the ways to overcome them.

At the end, of course, the reader will decide if the approach taken here is as effective as we hope it will be. We caution that audiences with strong technical backgrounds may find the text "hiding" some technical details. We advise these readers to explore the ample references for more technical depth, and to focus their reading of the book on applied aspects that are not covered at all or are not synthesized sufficiently in the primary technical literature.

Audiences without or with incomplete technical backgrounds may find some concepts challenging at first read. Unless otherwise noted, we advise this type of reader to not skip the scientific and technological principles underlying ML/AI, since these are critical for successful use in high stakes tasks and environments.

With regards to the book assignments, we revisit and *incrementally enrich and deepen many of them* as new knowledge is provided by the various chapters. Readers should address them with the knowledge gained up until the chapter they are encountered.

Finally we recommend the independent reader to read the chapters in sequence (possibly only skimming material that the reader has already mastered elsewhere). We made every effort to cross-reference in each chapter concepts with all other parts of the book where they are discussed so even an out-of-sequence reading should be free of confusion.

For in-classroom use, the class instructor is trusted to determine the right components to emphasize or omit, and in the right sequence for her class objectives and learners' background and needs. The incremental structure of assignments and discussion topics is valuable for *developing gradually an increasingly sophisticated understanding of recurring themes and topics*. It can also serve as a record of the students' progress in mastering the related body of knowledge and their ability to integrate and evaluate the material. This will be disrupted unavoidably in any out-of-sequence reading, however, and the instructor has to make adjustments to the assignments in such cases.

We also note that all assignments are motivated by real-life examples of methods development and application challenges. They can be traced to literature and case studies in the public domain as well as to our personal experience as working scientists, teachers, advisors, consultants and administrators. Whenever we felt there was possibility to breach upon privacy or reputation of third parties, we omitted specific references to technology and persons, in all other cases we name methods, products, and scientists, especially when credit was due for important discoveries or other scientific and technological contribution acknowledgment.

Caveats and Disclosures: Sourcing Best Practices

Where Do Best Practices Come from?

The realistic answer is that, circa 2023, biomedical AI/ML Best Practices are not to be found in one place, stated as such, and having fully complete and immutable status. This volume, to the best of our knowledge, is the first book to strive for that goal. Our recommendations originate from a variety of sources and are characterized by different levels of (a) maturity/validation, (b) breadth of applicability, and

(c) technical clarity and depth. We have thus considered and included in the present volume the following sources for the presented Best Practices:

1. **Published guidelines stated as such**, for example the PubMed search ("artificial intelligence" or "machine learning") and "best practices"(e.g., [60]) yields 217 results, several of which contain proposed best practices (of various degrees of validation and usefulness as we will see in subsequent chapters). In some cases important Best Practices and guidelines are contained in articles with a broader scope, for example, guidance issued by the biometrics division of the NCI [61].
2. **Implicit but clear findings and recommendations published by quality control consortia** (e.g., [62]).
3. **Broad and well-designed benchmark studies** that demonstrate the appropriateness and effectiveness of various algorithms in specific settings (e.g., [62, 63]).
4. **AI/ML competitions** (properly designed to prevent biases) e.g., [64].
5. **Criteria used in meta-analytic and systematic review studies** to assess quality, risk of bias etc. (see for example chapter "Reporting standards, Certification/Accreditation & Reproducibility").
6. **Published reporting, regulatory, and certification standards and requirements** (e.g., [65]).
7. **Theoretical properties of AI/ML algorithms, protocols and related methods** that directly suggest proper and improper usage (see for example chapters "Foundations and Properties of AI/ML Systems", "An Appraisal of Operating Characteristics of Major Machine Learning Methods Applicable to Healthcare and Health Sciences", and "Introduction to Causal Inference and Causal Structure Discovery").
8. **Case studies** that inform generalizable types of errors and suggest strategies to avoid them (see for example chapter "Lessons Learnt from Historical Failures, Limitations and Success of Health AI/ML. Enduring Problems and the Role of Best Practices").
9. **Literature reports that have focused on identifying specific types of errors or modeling/analysis problems** and have provided reusable approaches for avoiding or minimizing them (e.g., [66]).

In general this volume avoids offering guidance based on the authors' preferred workflows or methods unless these are falling in one of the above categories.

A key value proposition of the present work therefore is that we have assembled, reviewed, critically analyzed, and synthesized a plurality of sources to inform pitfalls and related best ways currently known for improving AI/ML quality, performance, effectiveness and safety.

We caution the reader that like every other cutting-edge field of scientific endeavor, this is work in progress and some of the currently known Best Practices

in ML/AI will undoubtedly improve and be revised as new methods come into play and the field deepens and widens its knowledge. We welcome reader feedback and criticism and we will make every effort to appraise and incorporate all useful suggestions in future editions. See also "Final Synthesis of Recommendations" for discussion about future evolution of Best Practices.

Outline of the Book: Contents Summary by Part and Chapter

Part I: Foundations

This present chapter entitled *"Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: the need for Best Practices enabling Trust in AI and ML*"), aims to provide introductory concepts about the field, to motivate the need for best practices in biomedical AI and ML, and to map out the book's scope and contents so that readers are well oriented. A small set of high-level pitfalls and guidelines are also included.

Chapter "*Foundations and Properties of AI/ML Systems"* provides a broad introduction to the foundations of health AI and ML systems and includes: (1) Theoretical properties and formal vs heuristic systems; practical implications of complexity for system tractability. (2) Foundations of AI including logics and symbolic vs non-symbolic AI, Reasoning with Uncertainty, AI/ML programming languages. (3) Foundations of Machine Learning Theory.

Chapter *"An Appraisal of Operating Characteristics of Major Machine Learning Methods Applicable to Healthcare and Health Sciences"* provides an outline of how each method works, and in addition we summarize the intended uses, the usual way it is employed in practice, and its known and unknown properties. Readers who have not delved into ML before, will find a useful introduction and review of key methods. Readers who may already know about some or all of these methods will gain additional insights as we critically revisit the key concepts and add to their prior knowledge summary guidance on *whether and when* each technique is applicable or preferred (or not) in healthcare and health science problem solving.

Chapter "*Introduction to Causal Machine Learning*" covers the important dimension of causality. The vast majority of texts in biomedical AI/ML focuses on predictive modeling and does not address causal methods, their requirements and properties. Yet these are essential for determining and assisting patient-level or healthcare-level interventions toward improving outcomes of interest. Causal methods are also indispensable for discovery in the health sciences.

Chapter "*Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems"* outlines a comprehensive process, governing all steps from analysis and problem domain needs specification, to creation and validation of AI/ML methods that can address them. The stages are explained and grounded in many existing methods. The process discussed equates to a generalizable Best Practice guideline applicable across all of AI/ML. An equally important use of this Best Practice is as a guide for understanding and evaluating any ML/AI technology under consideration for adoption for a particular problem domain.

Part II: Modelling

Chapter "*The Process and Lifecycle of a Clinical-Grade AI/ML Model*" introduces the notion of "clinical-grade" models and contrasts such models with feasibility, exploratory, or pre-clinical ones. The main tenet of the chapter is that AI/ML systems and models must be designed and deployed in a manner that is aware of, and seamlessly integrated in healthcare systems or discovery processes (for healthcare and health science discovery, respectively). The steps outlined span from requirements engineering to deployment, monitoring and iterative development and continuous improvement. They also emphasize contextual factors that influence success.

Chapter "*Data Design for Biomedical AI/ML*" addresses the critical aspect of data (or research) design and related best practices. This endeavor is foundational to the success of AI/ML for both clinical care and scientific discovery. Yet to the extent of our review of the literature, a systematic and in-depth treatment of this most important aspect receives little attention in the ML literature. In this chapter (a) we present common designs (e.g., retrospective, cohort, case/control, EHR, time series, RCT, hybrid, etc.) and implications of design choices for the success of modelling; (b) we discuss common data biases (e.g., selection bias, assertion bias, confounding bias, Simpson's paradox, etc.).

Chapter "*Data Preparation, Transforms, Quality, and Management*" introduces guidance for performing data preparations so that the goals of modeling are effectively and efficiently accomplished. It also addresses *data quality, mapping, feature engineering, data transformations, clinical and research data warehousing and management.*

Chapter "*Model Selection and Evaluation"* addresses best practices for finding models that are accurate, and generalize well. Estimation of the generalization error is also addressed both in terms of *error estimator* procedures and their interaction with model selection as well as in terms of *error metrics* and their effect on analysis. In addition to general-purpose performance metrics, this chapter also discusses aspects of model evaluation that are unique to biomedical applications, such as evaluating clinical efficacy, the suitability of a model for clinical decision support, and health economic evaluations.

Chapter "*Overfitting, Underfitting and Model Overconfidence and Underperformance in Machine Learning and AI*" makes a deep dive into overfitting and under fitting which are arguably two of the most far-reaching and impactful challenges in AI/ML with high-dimensional data, modest or small sample sizes, and modern high-capacity learners. Avoiding over and under-fitted analyses and models is critical for ensuring high generalization performance. In modern ML/AI practice these factors are typically interacting with error estimator procedures and model selection, as well as with sampling and reporting biases and thus are considered together in context. These concepts are also closely related to statistical significance and scientific reproducibility. We examine several common scenarios where over confidence in model performance and/or model under performance occur as well as recommended practices for preventing, testing and correcting them.

Chapter "From '*Human vs Machine' to 'Human with Machine'"* addresses: (a) empirical evaluations of healthcare and health science AI/ML decision-making. (b) Empirical comparisons of computer vs human decision making in health sciences and health care. (c) Important human cognitive biases that lead to decision errors.

(d) Summary comparison of human vs computer strengths and limitations that may manifest as errors in medical practice or science discovery settings. (e) Practical considerations in constructing hybrid computer-human problem-solving systems.

Chapter "*Lessons Learned from Historical Failures, Limitations and Successes of Health AI/ML. Enduring Problems, and the Role of Best Practices"* covers a variety of case studies relevant to best practices. Examples include: the infamous "AI winters"; overfitting; using methods not built to purpose; over-estimating the value and potential of early and heuristic technology; developing AI that is disconnected from real-life needs and application contexts; over-interpreting or misinterpreting results from learning theory; failures/shortcomings of literature including the persistence of incorrect findings; failures/shortcomings of modeling protocols, data and evaluation designs; high profile science failures; factors that may render guidelines themselves problematic. These case studies in most cases were followed by improved technology that overcame the limitations. The case studies reinforce, and demonstrate the value of rigorous, science-driven practices for addressing enduring and new challenges.

Chapter "*Characterizing and Managing the Risk of AI/ML Models in Clinical and Organizational Application"* covers practical methods for reviewing the face validity of AI/ML models, and characterizing and managing risk of such models at development and at deployment stages. This chapter also briefly discusses broader methods and practices for detecting and correcting issues with ML modeling and the emerging concept of *debugging* ML models and analyses.

Part III: Implementation

Chapter "*Considerations for Specialized Health AI/ML Modelling and Applications: NLP"* looks into the field- and task-specific best practices for the domain of health NLP.

Chapter "*Considerations for Specialized Health AI/ML Modelling and Applications: Imaging – Through the perspective of Dermatology"* looks into field and task-specific best practices in the specialized domain of Imaging (with a dermatology focus).

Chapter "*Regulatory Aspects and Ethical, Legal, and Societal Implications (ELSI)*" reviews the regulation of AI/ML models, the risk management principles underlying international regulations of clinical AI/ML, discusses the conditions under which AI/ML models in the U.S. are regulated by the Food and Drug Administration (FDA), and reviews FDA's Good Machine Learning Practice (GMLP) principles. In its second part, the chapter provides an introduction to the nascent field of biomedical AI ethics, covering general AI ELSI studies, AI/ML racial bias, and AI/ML health equity principles. The chapter discusses (and gives illustrative examples) of the importance of causality and equivalence classes for practical detection of racial bias in models. It concludes with a series of recommended best practices for promoting health equity and reducing health disparities via the design and use of health AI/ML.

Chapter "*Reporting Standards, Certification/Accreditation & Reproducibility"* covers the interrelated topics enhancing the quality safety and reproducibility of clinical AI/ML via (a) reporting standards; (b) recent efforts for accrediting health care provider organizations for AI readiness and maturity; (c) professional

certification; and (d) education and related accreditation of educational programs in data science and biomedical informatics, specific to AI/ML.

Chapter "*Final Synthesis of Recommendations*" presents a consolidated view of the identified pitfalls and recommended practices across the book. We differentiate between macro-, meso- and micro-levels of pitfalls and corresponding best practices-roughly corresponding to high-level principles, concrete differentiations of the above and granular/detailed tools and techniques for implementation. We discuss the non-uniqueness of best practice frameworks and several open problems. The continued development and dissemination of Best Practices for biomedical AI/ML is certain to become in the years to come a field of inquiry with significant growth and value.

Key Concepts Discussed in Chapter " Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML"

Artificial Intelligence (AI) and Machine Learning (ML)
Data Science
Computer program
Computer system
Computer algorithm
AI/ML model
Data Science
Performance requirements
Safety requirements
Cost-effectiveness requirements
Trust, acceptance, and adoption

Key Messages Discussed in Chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML"

1. AI/ML are long standing disciplines with millions of published articles since the 1960s and with several Turing and Nobel awards linked to them.
2. Biomedical AI/ML has a long history and extensive literature behind them also starting in the 1960s. They have recently exploded in the literature in adoption for discovery and care and as their own fields of study.
3. AI/ML are applied broadly in science and health care because they relate to extremely broad classes of prediction/pattern recognition and causal modeling and problem solving tasks.
4. Biomedical AI/ML has several distinct requirements than general-purpose AI/ML.
5. AI/ML Algorithms, programs and systems must inspire and guarantee trust in their safety, effectiveness and cost effectiveness. Best Practices must be developed, shared and followed to enable trust and acceptance.

6. Known properties are essential for AI/ML trust.
7. Currently known Best Practices originate from a variety of sources, have different levels of maturity or validation and will undoubtedly expand and improve in the future.

Pitfalls Discussed[3] in Chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML"

Pitfall 1.1: Unspecified, undisclosed or insufficiently-analyzed algorithms.

Pitfall 1.2: In healthcare and health sciences, ***clinical algorithms*** are often confused with ***computer algorithms***.

Pitfall 1.3: Viewing the whole field as being about one narrow technology or a small set of tools, ignoring the broader spectrum of available options.

Pitfall 1.4: Ignoring the vast literature or "re-inventing the wheel".

Pitfall 1.5: Ignoring the specific requirements and adaptations tailored to the goals of healthcare and of health sciences discovery.

Best Practices Discussed in Chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML"

Best Practice 1.1 When considering development or application of AI/ML ensure that it is informed by well-developed and evaluated existing science and technology.

Classroom Assignments and Discussion Topics[4] chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: The Need for Best Practices Enabling Trust in AI and ML"

1. If *science is self-correcting via reproducibility studies*, what are the dangers/downsides to producing AI/ML systems/methods and related articles with a high proportion of false results?

[3] To be further elaborated later in the book, including related Best Practices.

[4] Several of these and similar topics will be clarified and elaborated upon in subsequent chapters. However we recommend to class instructors and self-learners to get a first-pass evaluation of where the reader/classroom is (attitude, knowledge, experience) with regards to such problems.

2. Identify from news sources and business publications articles about past industry failures in health AI/ML. Summarize and draw your conclusions about how to remedy and avoid such problems.
3. What, in your view, is the ideal relationship (i.e., rules of engagement and assignment of responsibilities/foci) of industry and academia in developing and delivering health AI/ML?
4. What are areas where health AI/ML cannot reach human problem solving? What about the reverse?
5. The so-called *No Free Lunch Theorem (NFLT)* states (in simplified language) that all ML and more broadly all AI optimization methods are *equally accurate over all problems on average*. Discuss the implications for choice of AI/ML methods in practical use cases.
6. "It is not the tool but the craftsman". Does this maxim apply to health AI/ML?
7. How would you go about identifying and measuring/documenting the impact that AI/ML has had on specific health science discoveries?
8. Is AI confined to computer systems? Can other artificial intelligent agents such as corporations be viewed as AI? Discuss implications of such a broader view.
9. Construct a "pyramid of evidence" for health AI/ML similar to the one used in evidence based care practice. Consider two pyramids: one focusing on clinical healthcare and another on health science discovery.
10. You are part of a university/hospital evaluation committee for a vendor offering a patient-clinical trial matching AI product. Your institution strongly needs to improve the patient-trial matching process to increase trial success and efficiency metrics.

    The sales team makes the statement that "this is a completely innovative AI/ML product; nothing like this exists in the market and there is no similar literature; we cannot at this time provide theoretical or empirical accuracy analysis, however you are welcome to try out our product for free for a limited time and decide if it helpful to you". The product is fairly expensive (multi $ million license fees over 5 years covering >1000 trials steady-state).

    What would be your concerns based on these statements? Would you be in position of making an institutional buy/not buy recommendation?
11. A company has launched a major national marketing campaign across health provider systems for a new AI/ML healthcare product based on its success on playing backgammon, reading and analyzing backgammon playing books and human games, extracting novel winning strategies from matches, answering questions about backgammon, and teaching backgammon to human players.

    How relevant is this impressive AI track record to health care? How would you go about determining relevance to health care AI/ML? How your reasoning would change if the product was not based on success in backgammon but success in identifying oil and gas deposits? How about success in financial investments?

12. Your university-affiliated hospital wishes to increase early diagnosis of cognitive decline across the population it serves. You are tasked to choose between the following AI/ML technologies/tools:
    (a) AI/ML tool A guarantees optimal predictivity in the sample limit in distributions that are multivariate normal.
    (b) AI/ML tool B has no known properties but is has been shown to be very accurate in several datasets for microarray cancer-vs-normal classification.
    (c) AI/ML tool C is a commercial offshoot of a tool that was fairly accurate in early (pre-trauma) diagnosis of PTSD.
    (d) AI/ML tool D is an application running on a ground-breaking quantum computing platform (Quantum computing is an exciting and frontier technology that many believe has potential to make AI/ML with hugely improved capabilities in the future).
    (e) AI/ML tool E runs on a novel massively parallel cloud computing platform capable of Zettascale performance.

    What are your thoughts about these options?

13. The same question as #12 but with the following additional data:
    (a) AI/ML tool A sales reps are very professional, friendly and open to offering deep discounts.
    (b) AI/ML tool B is offered by a company co-founded by a widely-respected Nobel laureate.
    (c) AI/Ml tool C is offered by a vendor with which your organization has a successful and long relationship.
    (d) AI/Ml tool D is part of a university initiative to develop thought leadership in quantum computing.
    (e) AI/Ml tool E will provide patient-specific results in 1 picosecond or less.

    How does this additional information influences your assessment?

References

1. Hart PE, Stork DG, Duda RO. Pattern classification. Hoboken: Wiley; 2000.
2. Russell, S.J., 2010. Artificial intelligence a modern approach. Pearson Education, Inc.
3. Weiss SM, Kulikowski CA. Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems. Morgan Kaufmann Publishers Inc.; 1991.
4. Statnikov A. A gentle introduction to support vector machines in biomedicine: theory and methods, vol. 1. world scientific; 2011.
5. Sverchkov Y, Craven M. A review of active learning approaches to experimental design for uncovering biological networks. PLoS Comput Biol. 2017;13(6):e1005466.
6. Statnikov A, Ma S, Henaff M, Lytkin N, Efstathiadis E, Peskin ER, Aliferis CF. Ultra-scalable and efficient methods for hybrid observational and experimental local causal pathway discovery. J Mach Learn Res. 2015;16(1):3219–67.

7. Guyon I, Cawley GC, Dror G, Lemaire V. Results of the active learning challenge. In: Active learning and experimental design workshop in conjunction with AISTATS 2010. JMLR Workshop and Conference Proceedings; 2011, April. p. 19–45.
8. Tanenbaum AS. Structured computer organization. Prentice Hall; 1984.
9. Cormen TH, Leiserson CE, Rivest RL, Stein C. Introduction to algorithms. MIT press; 2022.
10. Brookshear, J.G. Computer science: An overview. Benjamin-Cummings Publishing Co., Inc; 1991.
11. Sedgewick R. Algorithms in c++, parts 1–4: fundamentals, data structure, sorting, searching. Pearson Education; 1998.
12. Margolis CZ. Uses of clinical algorithms. JAMA. 1983;249(5):627–32.
13. Grimshaw J, Russell I. Achieving health gain through clinical guidelines. I: developing scientifically valid guidelines. Qual Health Care. 1993;2(4):243–8.
14. Vapnik, V. The nature of statistical learning theory. Springer science & business media. 1999.
15. Kearns MJ, Vazirani U. An introduction to computational learning theory. MIT press; 1994.
16. Donoho D. 50 years of data science. J Comput Graph Stat. 2017;26(4):745–66.
17. Cao L. Data science: a comprehensive overview. ACM Comput Surv. 2017;50(*3*):*1*–42.
18. Spirtes P, Glymour CN, Scheines R, Heckerman D. Causation, prediction, and search. MIT press; 2000.
19. Glymour CN, Cooper GF, editors. Computation, causation, and discovery. AAAI Press; 1999.
20. Pearl J. Causality. Cambridge university press; 2009.
21. Roski J, Bo-Linn GW, Andrews TA. Creating value in health care through big data: opportunities and policy implications. Health Aff. 2014;33(7):1115–22.
22. https://en.wikipedia.org/wiki/Applications_of_artificial_intelligence.
23. Cohen TA, Patel VL, Shortliffe EH, editors. Intelligent Systems in Medicine and Health: the role of AI. Springer Nature; 2022.
24. Szymczak S, Biernacka JM, Cordell HJ, González-Recio O, König IR, Zhang H, Sun YV. Machine learning in genome-wide association studies. Genet Epidemiol. 2009;33(S1):S51–7.
25. Adam T, Aliferis C. Personalized and Precision Medicine Informatics. Health Informatics Series. Basel, Springer Nature Switzerland. 2020.
26. Aphinyanaphongs Y, Tsamardinos I, Statnikov A, Hardin D, Aliferis CF. Text categorization models for high-quality article retrieval in internal medicine. J Am Med Inform Assoc. 2005;12(2):207–16.
27. Libbrecht MW, Noble WS. Machine learning applications in genetics and genomics. Nat Rev Genet. 2015;16(6):321–32.
28. Cheng J, Tegge AN, Baldi P. Machine learning methods for protein structure prediction. IEEE Rev Biomed Eng. 2008;1:41–9.
29. Ryu JY, Kim HU, Lee SY. Deep learning improves prediction of drug–drug and drug–food interactions. Proc Natl Acad Sci. 2018;115(18):E4304–11.
30. Erickson BJ, Korfiatis P, Akkus Z, Kline TL. Machine learning for medical imaging. Radiographics. 2017;37(2):505–15.
31. Andreu-Perez J, Poon CC, Merrifield RD, Wong ST, Yang GZ. Big data for health. IEEE J Biomed Health Inform. 2015;19(4):1193–208.
32. Vamathevan J, Clark D, Czodrowski P, Dunham I, Ferran E, Lee G, Li B, Madabhushi A, Shah P, Spitzer M, Zhao S. Applications of machine learning in drug discovery and development. Nat Rev Drug Discov. 2019;18(6):463–77.
33. Olsen L, Aisner D, McGinnis JM. The learning healthcare system: workshop summary. Institute of Medicine (US). National Academies Press (US); 2007. ISBN 978-0-309-10300-8.
34. Ledley RS, Lusted LB. Reasoning foundations of medical diagnosis: symbolic logic, probability, and value theory aid our understanding of how physicians reason. Science. 1959;130(3366):9–21.
35. Warner HR, Toronto AF, Veasey LG, Stephenson R. A mathematical approach to medical diagnosis: application to congenital heart disease. JAMA. 1961;177(3):177–83.

36. Miller RA, Pople HE Jr, Myers JD. Internist-I, an experimental computer-based diagnostic consultant for general internal medicine. N Engl J Med. 1982;307(8):468–76.
37. Pearl J. Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan kaufmann; 1988.
38. Shwe MA, Middleton B, Heckerman DE, Henrion M, Horvitz EJ, Lehmann HP, Cooper GF. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. Methods Inf Med. 1991;30(04):241–55.
39. Rumelhart DE, McClelland JL, PDP Research Group. Parallel distributed processing, vol. 1. New York: IEEE; 1988. p. 354–62.
40. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44.
41. Saba L, Biswas M, Kuppili V, Godia EC, Suri HS, Edla DR, Omerzu T, Laird JR, Khanna NN, Mavrogeni S, Protogerou A. The present and future of deep learning in radiology. Eur J Radiol. 2019;114:14–24.
42. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
43. Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.
44. Aliferis CF, Statnikov A, Tsamardinos I, Mani S, Koutsoukos XD. Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation. J Mach Learn Res. 2010;11(1):171–234.
45. Tsamardinos I, Brown LE, Aliferis CF. The max-min hill-climbing Bayesian network structure learning algorithm. Mach Learn. 2006;65(1):31–78.
46. Tsamardinos I, Aliferis CF, Statnikov AR, Statnikov E. Algorithms for large scale Markov blanket discovery, vol. 2. FLAIRS conference; 2003. p. 376–80.
47. Tsamardinos I, Aliferis CF, Statnikov A. Time and sample efficient discovery of Markov blankets and direct causal relations. In: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining; 2003, August. p. 673–8.
48. Aliferis CF, Tsamardinos I, Statnikov A. HITON: a novel Markov blanket algorithm for optimal variable selection. In: AMIA annual symposium proceedings, vol. 2003. American Medical Informatics Association; 2003. p. 21.
49. Wang X, Fan J. Spatiotemporal molecular medicine: a new era of clinical and translational medicine. Clin Transl Med. 2021;11(1):e294.
50. Wu Y, Cheng Y, Wang X, Fan J, Gao Q. Spatial omics: navigating to the golden era of cancer research. Clin Transl Med. 2022;12(1):e696.
51. Glikson E, Woolley AW. Human trust in artificial intelligence: review of empirical research. Acad Manag Ann. 2020;14(2):627–60.
52. Hengstler M, Enkel E, Duelli S. Applied artificial intelligence and trust—the case of autonomous vehicles and medical assistance devices. Technol Forecast Soc Chang. 2016;105:105–20.
53. Siau K, Wang W. Building trust in artificial intelligence, machine learning, and robotics. Cut Bus Technol J. 2018;31(2):47–53.
54. Winfield AF, Jirotka M. Ethical governance is essential to building trust in robotics and artificial intelligence systems. Philos Trans R Soc A Math Phys Eng Sci. 2018;376(2133):20180085.
55. Jacovi A, Marasović A, Miller T, Goldberg Y. Formalizing trust in artificial intelligence: prerequisites, causes and goals of human trust in AI. In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency; 2021. p. 624–35.
56. Asan O, Bayrak AE, Choudhury A. Artificial intelligence and human trust in healthcare: focus on clinicians. J Med Internet Res. 2020;22(6):e15154.
57. Matheny M, Israni ST, Ahmed M, Whicher D. Artificial intelligence in health care: the hope, the hype, the promise, the peril. Washington, DC: National Academy of Medicine; 2019.
58. Rigby MJ. Ethical dimensions of using artificial intelligence in health care. AMA J Ethics. 2019;21(2):121–4.
59. Bates DW, Auerbach A, Schulam P, Wright A, Saria S. Reporting and implementing interventions involving machine learning and artificial intelligence. Ann Intern Med. 2020;172(11_Supplement):S137–44.

60. Makarov VA, Stouch T, Allgood B, Willis CD, Lynch N. Best practices for artificial intelligence in life sciences research. Drug Discov Today. 2021;26(5):1107–10.
61. Dupuy A, Simon RM. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. J Natl Cancer Inst. 2007;99(2):147–57.
62. Shi L, Campbell G, Jones WD, et al. The MicroArray quality control (MAQC)-II study of common practices for the development and validation of microarray-based predictive models. Nat Biotechnol. 2010;28(8):827–38.
63. Statnikov A, Aliferis CF, Tsamardinos I, Hardin D, Levy S. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. Bioinformatics. 2005;21(5):631–43.
64. Guyon I, Aliferis C, Cooper G, Elisseeff A, Pellet JP, Spirtes P, Statnikov A. Design and analysis of the causation and prediction challenge. In: Causation and prediction challenge. PMLR; 2008. p. 1–33.
65. https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-aiml-enabled-medical-devices.
66. Aliferis CF, Statnikov A, Tsamardinos I, Schildcrout JS, Shepherd BE, Harrell FE Jr. Factors influencing the statistical power of complex data analysis protocols for molecular signature development from microarray data. PLoS One. 2009;4(3):e4922.

Foundations and Properties of AI/ML Systems

Constantin Aliferis and Gyorgy Simon

Abstract

The chapter provides a broad introduction to the foundations of health AI and ML systems and is organized as follows: (1) Theoretical properties and formal vs. heuristic systems: computability, incompleteness theorem, space and time complexity, exact vs. asymptotic complexity, complexity classes and how to establish complexity of problems even in the absence of known algorithms that solve them, problem complexity vs. algorithm and program complexity, and various other properties. Moreover, we discuss the practical implications of complexity for system tractability, the folly of expecting Moore's Law and large-scale computing to solve intractable problems, and common techniques for creating tractable systems that operate in intractable problem spaces. We also discuss the distinction between heuristic and formal systems and show that they exist on a continuum rather than in separate spaces. (2) Foundations of AI including logics and logic based systems (rule based systems, semantic networks, planning systems search, NLP parsers), symbolic vs. non-symbolic AI, Reasoning with Uncertainty, Decision Making theory, Bayesian Networks, and AI/ML programming languages. (3) Foundations of Computational Learning Theory: ML as search, ML as geometrical construction and function optimization, role of inductive biases, PAC learning, VC dimension, Theory of Feature Selection, Theory of Causal Discovery. Optimal Bayes Classifier, No Free Lunch Theorems, Universal Function Approximation, generative vs. discriminative models; Bias-Variance Decomposition of error and essential concepts of mathematical statistics.

C. Aliferis (✉) · G. Simon
Institute for Health Informatics, University of Minnesota, Minneapolis, MN, USA
e-mail: constantinaibestpractices@gmail.com

G. J. Simon, C. Aliferis (eds.), *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, Health Informatics,
https://doi.org/10.1007/978-3-031-39355-6_2

Keywords

Properties of AI/ML models · Computational complexity (time, space) · Tractable vs intractable computer solutions · Symbolic and non-symbolic AI · Shallow vs. ontologically rich AI · Bayesian networks · Machine Learning theoretical foundations

Theoretical AI/ML Properties and Formal Vs Heuristic Systems

We will first address a few key concepts regarding studying and understanding, but also designing, AI systems by way of their formal properties. By **formal properties** we mean theoretical properties that are mathematical or computational, and technical and objective in nature.

Computability/Provability and Turing-Church Thesis

The most foundational property for any computer system (not just AI/ML systems) is computability, that is the fundamental question of whether there can even exist a computer program or system that achieves the computation needed for the inferences that we want this system to perform. Goedel [1] proved a theorem that shook the mathematical and computer science worlds.

Goedel's celebrated "incompleteness theorem" shows that any non-trivial mathematical system for making deductive inferences is either complete or consistent but not both. Or stated differently, statements can be formed in this system that are true but cannot be proven if we wish to maintain the correctness of deductions.

A **complete system** is one that can deduce (or prove) from the axioms of the system all statements that are true.

A **consistent system** is one that does not produce contradictory conclusions (which entails false conclusions).

Correspondingly, in the realm of computing there are **functions that are not computable**, that is there is no computer program that can compute them.

These two results (provability and computability) are essentially mirroring each other because there is a close correspondence relationship between a "proof" in a mathematical system and a "program" in an equivalent computing system implementing the mathematical system. Non-computable functions are the ones that cannot be proven and vice versa.

Notice that the existence of *non-computable functions/non-provable statements is with reference to a specific computing system*. A different system may be able to

prove certain statements at the expense of not being able to prove others that the first system can. Also, we note that in systems involving a finite number of domain elements, we do not face restrictions in computability. However, this is of small consolation if we realize that even systems as "basic" as common arithmetic, for example, involve many non computable functions.

What is the relationship of computability/provability in the computational/mathematical realm with that in human intelligence and reasoning?

The **Turing-Church thesis** posits that everything that the human mind can infer can also be inferred by a computer/mathematical system [1]. According to the thesis, there are no special functions of human intelligence that a computer or mathematical system cannot emulate. This thesis is axiomatic, meaning not proven. From what we know so far from neuroscience, cognitive science etc., there is nothing in the human brain that a computer system cannot model in principle, and the vast majority of AI scientists accept the Turing-Church thesis.

Computational Complexity of a Problem, Algorithm of Program

Computational complexity of a program refers to the efficiency of running a computer program that solves a particular problem according to a specific algorithm (that the program implements). In other words, it describes (for problems that can be solved by computer), how expensive is to solve the problem. Computational complexity is in the form of a function that typically takes the **size of the problem instance** as inputs.

Computational complexity of an algorithm applies the same rationale to algorithms instead of programs. Typically we analyze computational complexity at the level of algorithms assuming that programs will be the most efficient implementation of the algorithm (when exceptions happen in practice, we state upfront that a particular implementation of an algorithm is not as efficient as it can be).

Computational complexity of a problem is then analyzed at the level of the most efficient algorithm known (or *that could be devised but not yet known—we will see later how this is accomplished*) for solving this problem.

Space complexity refers to how much space the computer program/algorithm/problem class requires to reach a solution. **Time complexity** refers to how much time the computer program/algorithm/problem class requires to reach a solution. Because different computers differ greatly in the time needed to execute the same basic operation (e.g., one addition or one access of a random access memory location, etc.) we often measure time complexity *not in units of time but in numbers of some essential operation* (and then we can translate these units to time units for available computer systems). Because the differences between computers are within constant factors, this does not make a difference in an asymptotic sense.

Worst, average, and best-case complexity. Often, not all problem instances require the same amount of resources (space or time) to be solved by the same

program/algorithm. **Worst case complexity** refers to the cost of the worst (most expensive) instance of the problem when solved by the best possible algorithm (or, alternative for a specific algorithm of interest). **Best case complexity** refers to the cost of the best (least expensive) instance of the problem. **Average case complexity** refers to the cost averaged over all instances of the problem.

Exact complexity refers to a precise complexity for example:

$$Cost\left(x\right) = x^2 \tag{1}$$

where x is the size of the i^{th} problem instance. In this example, the cost of solving the problem is exactly the square of the size of the problem instance.

Asymptotic complexity refers to complexity as an asymptotic *growth function, i.e.,* that is how fast the complexity grows as input size grows. For example,

$$Cost\left(x\right) = O\left(f\left(x\right)\right) \tag{2}$$

The **"Big O" notation *O(f(.))*** denotes that there is a problem instance size *k,* above which the complexity (cost) of all problem instances of size at least as large as *k,* is bounded from above within a positive constant from the value of function f(.), or more compactly stated:

$$\exists k \text{ s.t.}, \forall x \geq k : Cost\left(x\right) \leq cf\left(x\right)$$

- Where ∃ is *the existential oper*ator (denoting that the quantity in the scope of the operator exists)
- ∀ is the *universal operator* denoting that for all entities in the scope of the operator a statement that follows is true
- x is the size of the i^{th} problem instance
- *Cost (x)* is the computational cost of running the algorithm for input size x
- k is a input size threshold above which the complexity statement holds
- c is a positive constant
- "S.t." is the common abbreviation "such that".

We often use asymptotic cost complexity for two reasons: (a) It eliminates confusion created by differences in the speeds of various computer systems since in practice these are all within a small constant factor of each other. (b) It shifts the attention to the broad classes of rates of cost growth (e.g. linear, quadratic, exponential, etc.) and not the precise cost formulas that can be convoluted. Mathematical analysis can accordingly be greatly simplified.

To understand the implications of asymptotic growth contrast the polynomial asymptotic complexity of formula (1) with the one below:

$$Cost\left(x\right) = O\left(2^x\right) \tag{3}$$

The following Table 1 shows how quickly these cost functions grow (assuming, for illustration purposes, $c = 1$). For input sizes above 100, the cost in terms of space and time complexity grows to sizes comparable to the size of the universe

Table 1 Demonstration of the practical significance of asymptotic computational complexity

	Quadratic Cost	Exponential Cost	*Related to complexity O* (2^X)		
Size of problem instance	*If cost of computation grows as O* (x^2)	*Cost of computation grows as O* (2^x)	Moore's Law (here: If speed doubled every 4 years) How many years needed until CPUs catch up starting at size 100 and cost = 2^{100}?	Parallelize (linearly by using *m* CPUs) How many CPUs needed? (within a constant factor)	Other comments
1	1	2	T	2	T
2	4	4	R	4	R
3	9	8	A	8	A
4	16	16	C	16	C
5	25	32	T	32	T
6	36	64	A	64	A
7	49	128	B	128	B
8	64	256	L	256	L
9	81	512	E	512	E
10	100	1024		1024	
20	400	1,048,576		1,048,576	$\geq 10^6$
30	900	1,073,741,824		1,073,741,824	$\geq 10^9$
100	10^4	2^{100}	Wait for 280 years	2^{100} CPUs	Comparable to number of atoms in the universe
1000	10^6	2^{1000}	Wait for 3680 years	2^{1000} CPUs needed	> > than size of known universe
10^6	10^{12}	$2^{1,000,000}$	Wait for ~40 million years	$2^{1,000,000}$ CPUs Needed	

(measured in atoms) and quickly becomes much larger than the size of the universe. This means that there is not enough physical space or time to solve these problems!

The fallacy/pitfall that we will "use a big enough cluster" (or other high-performance computing environment) to solve a high-complexity problem is addressed in the parallel column where it is shown that the number of CPUs needed would quickly exceed the size of the universe. The fallacy/pitfall that Moore's law (e.g., computing power doubles every few years) will provide enough power is addressed in the Moore's law column where is shown that millions of years would be needed to address problems of any significant size, and after some point the space and time requirements exceed the size of the known universe.

We will refer to problems, algorithms, programs and systems exhibiting such exorbitant complexities, as **intractable.** The following pitfalls and corresponding best practices need be taken into consideration:

Pitfall 2.1
From a rigorous science point of view, an AI/ML algorithm, program or system with intractable complexity does not constitute a viable solution to the corresponding problem.

Pitfall 2.2
Parallelization cannot make an intractable problem, algorithm or program practical.

Pitfall 2.3
Moore's law improvements to computing power cannot make an intractable problem algorithm or program practical.

Best Practice 2.1
Pursue development of AI/ML algorithm, program or systems that have tractable complexity.

Best Practice 2.2
Do not rely on parallelization to make intractable problems tractable. Pursue tractable algorithms and factor in the tractability analysis any parallelization.

Best Practice 2.3
Do not rely on Moore's law improvements to make an intractable problem algorithm or hard program practical. Pursue tractable algorithms and factor in the tractability analysis any gains from Moore's law.

It is very common in modern AI/ML to be able to address problems that have *worst case* exponential (or other intractable) complexity and routinely tackle, for example, analyses of datasets with $>10^6$ variables for problems with worst-case exponential cost by using a number of strategies that we will summarize below. First we round up the introduction to complexity properties with an overview of complexity classes.

Reduction of Problems to Established Complexity Class

Earlier we mentioned that computational complexity of a problem can be analyzed at the level of the most efficient algorithm known, or *that could be devised but not yet known*. How is this possible? One ingenious way to achieve this was discovered by Cook who proved a remarkable theorem (and received a Turing award for the work) [2]. Karp, based on Cook's result, showed how to prove that several other problems were in the same complexity class (and also won a Turing award for this work) [3].

The above constitute a generalizable methodology, very widely used in computer science and AI/ML, comprising two steps:

1. First establish via mathematical proof that a problem class P1 has an intrinsic minimum complexity regardless of the algorithm or program that has been devised or could be devised to solve it (i.e., intrinsic to the problem and independent of algorithm, in the sense that no Turing machine can exist that could do better). This part does not require the knowledge of a conventional algorithm that solves P1.
2. Second, in order to prove that problem P*i* at hand belongs to the same or harder complexity class as P1, it suffices to establish that a **fast reduction** (e.g., with polynomial-time complexity) exists that maps problems and their solutions in P1 to problems and solutions in P*i*, such that when a problem solution to a P*i* problem instance is found then it can be converted fast to a problem solution for P1.

"Fast" in this context means that: cost of the reduction + cost of solving the P1 version of the P*i* problem, will be no costlier (asymptotically) than solving P*i*. For example, if P*i* has cost $O\ (2^X)$, a reduction with cost $O\ (x^2)$ satisfies the requirement since $O\ (2^x + x^2) = O\ (2^x)$.

Step 1 has to be accomplished only *once for a prototypical problem class* and is of the greatest mathematical difficulty. Step 2, which is typically considerably easier, is done each time a new method is introduced and is conducted once for the new method, with reference the prototypical problem class.

Cook's discovery provided exactly step 1 and opened the flood gates via the reduction methods of Karp (step 2) for assigning whole problem classes to complexity cost classes ***regardless of the algorithm or problem used to solve it and regardless of whether even a single algorithm is currently known for solving the problem.***

AI/ML and computer scientists often use prototypical complexity classes to study and categorize problems and the algorithms solving them, the most common ones being:

The P complexity class: contains problems that can be solved in polynomial time. These are considered as tractable (assuming, as is typically the case, that the polynomial degree is small).

The NP complexity class: contains problems that have the property that a solution can be verified as correct in polynomial time.

The NP-Complete complexity class: These are problems that are in NP and moreover if any of the problems in this class can be solved in polynomial time, then all other problems in the class can also be solved in polynomial time.

NP Hard problems. Are problems that are as hard as those in NP but it is unknown whether they are in NP.

Several other classes exist and are subject to study and exploration (as to what problems belong to them or what relationships exist among them).

The practical significance of the complexity classes is as follows:

- Problems in **P** are considered as tractable (assuming, as is typically the case, that the polynomial degree is small).
- Problems in **NP-Complete** or **NP Hard** classes are considered very hard and it is extremely unlikely that algorithms that solve such problems tractably in the worst case, can be created.

A fundamental property of AI/ML problem solving is that it usually operates in problem spaces belonging to the very high complexity/worst-case intractable classes. Many strategies have been invented to circumvent these theoretical difficulties and guide creation of efficient algorithms and systems, however (discussed later in the present chapter).

A List of Key and Commonly Used Formal Properties of AI/ML (Table 2)

Many **additional special-purpose or ancillary formal properties** can also be studied and established such as: whether performance estimators are biased, statistical decision false positive and false negative errors when fitting models, whether scoring rules or distance metrics used are proper or improper, various measures of statistical certainty, etc. We emphasize that the properties listed in Table 2 have immediate and obvious relationship with, and consequences for, the common

Table 2 Commonly-considered important formal (theoretical) properties that characterize all AI/ML algorithms, programs and systems

1. **Representation power**: Can the models produced by the method represent all problem instances of interest and their solutions?
2. **Semantic clarity and transparency**: Do the programs and the corresponding models exhibit clarity based on precisely understood semantics (i.e., formally defined meaning)? Are the models produced by the method easy to understand (i.e., are they **"transparent box"**) and can they be easily understood by human inspection (i.e., are they human interpretable, aka **explainable**)?
3. **Soundness**: When the methods output a solution to a problem instance, is this solution correct? If there is a degree of error (measured on some scale of loss, risk or other scale) how large is the error and its uncertainty?
4. **Completeness**: Does the method produce correct answers to all problem instances? If only a fraction, how large is the fraction?
5. **Computational complexity.** What is the exact or asymptotic computational complexity of running the method to produce solutions as a function of the input size?

   For AI/ML methods that produce models as intermediate step in producing solutions, we differentiate
   (a) Computational complexity of producing problem-solving models: What is the exact or asymptotic computational complexity of running the method to produce models as a function of the input size (e.g., number of variables, or sample size)? And
   (b) Computational complexity of executing problem-solving models: What is the exact or asymptotic computational complexity of running the models to produce solutions as a function of the input size (e.g., number of variables, or sample size)?
6. **Space complexity.** What is the exact or asymptotic space complexity of running the method to produce solutions as a function of the input size? For AI/ML methods that produce models as intermediate step in producing solutions we can differentiate:
   (a) Space complexity of producing problem-solving models: What is the exact or asymptotic space complexity of running the method to produce models as a function of the input size (e.g., number of variables, or sample size)? And
   (b) Space complexity of executing problem-solving models: What is the exact or asymptotic space complexity of running the models to produce solutions as a function of the input size (e.g., number of variables, or sample size)?
7. **Additional cost functions**: For example, *financial costs* to obtain and store input data and run analyses on a compute environment, either at model discovery or at model deployment time? *Compliance risks. Ethical, litigation or reputational risks*, etc.
8. **Sample complexity, learning curves, power-sample requirements**: How does the error of the produced models vary as function of sample size of the discovery data? How much sample size is needed in order to build models with a specific degree of accuracy and statistical error uncertainty, and (separately) to establish statistically superiority to random or alternative models and performance levels?
9. **Probability and decision theoretic consistency**: Is the ML/AI method compatible with probability and utility theory?

objectives of health AI/ML. In the present volume we will refrain from study of properties that do not have strong relevance to the success or failure of AI/ML modeling. For example, the accuracy of a predictive model has immediate consequences for its usefuleness. By contrast, the centrality measures of network science models

say very little about their predictive (or causal) value. Similarly, the use of perplexity measure to study the degree by which a Large Language Model has learned (essentially the grammar underlying) a text corpus, does not indicate the clinical error severity resuting from output errors made by the model, which may of much higher importance for health applications.

Formal (aka theoretical) properties are "hard" technical properties (i.e., mathematical, immutable). There exist "softer" properties (i.e., less technical, more transient, or even harder to establish objectively) such as compliance to regulatory or accreditation guidance, reporting standards, ethical principles, etc.

An additional category with special significance is that of **empirical performance** properties. These are obtained using methods of empirical evaluation (chapters "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems", "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models", "Evaluation", and "Characterizing, Diagnosing and Managing the Risk of Error of ML & AI Models in Clinical and Organizational Application").

Importance of theoretical and empirical properties. Taken together these characterizations of AI/ML systems provide an invaluable framework for:

(a) Understanding the strengths and limitations of AI/ML methods, models and systems;
(b) Improving them;
(c) Understanding, anticipating, and effectively managing the risks and benefits of using AI/ML; and
(d) Choosing the right method for the problem at hand, among the myriad of available methods and systems.

We will see many examples of these formal, empirical and ancillary properties in the chapters ahead, considered in context. Chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML" describe properties of main AI/ML methods and chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" provides a summary table with the main properties of all main health AI/Ml methods.

Principled Strategies to Achieve Practically Efficient Algorithms, Programs and Systems for Worst-Case Intractable Problems

Since most intractability results pertain to worst-case complete and sound problem solving, a number of strategies can be used to achieve tractability, by trading off computational costs with reduction in soundness, completeness or worst-case complexity. Such common example strategies are listed below.

(a) **Focus on portions of the problem space that admit tractable solutions and ignore the portions with intractable solutions.** In problem domains where the worst case complexity diverges strongly form the average case complexity, such an approach is especially appealing. For example, in ML problems, focus on restricted data distributions or target function sub-classes that lead to tractable learning. Or focus on sparse regions of the data-generating processes and ignore dense (and commonly intractable) regions.

(b) **Exploit prior domain knowledge to constrain and thus speed-up problem solving**. For example, in discovery problems, avoid generating and examining many possible solutions that are incompatible with prior biomedical knowledge about the credible solution space. This may viewed as a case of "knowledge transfer" from this or similar problem domains. This is also often called *pruning*, where large branches, that are guaranteed to not contain the correct solution, are eliminated from vast solution search trees.

(c) **Instead of an intractable complete solution, provide a tractable localized part of the full solution that is still of significant value**. For example, when pursuing a causal model of some domain, focus on the partial causal model around some variables of interest (i.e., biological pathway discovery involving a phenotype instead of full network discovery).

(d) **Instead of an intractable complete solution, provide a tractable non-local portion of the full solution that is still of significant value**. For example, when pursuing a causal model of some domain allow discovery of a portion of correct relationships of interest (i.e., biological causal relationship discovery involving factors *across the data generating network* instead of full network discovery; or recovering a correct but unoriented causal network instead of the oriented one).

(e) Instead of producing perfectly accurate but intractable solutions **focus on more tractable but acceptable approximations of the true solutions.**

(f) **Do not solve harder problems than what is needed by your application.** A classic example demonstrating this principle is to prefer *discriminative models over generative ones in predictive modeling*. In plain language, this means that we can often solve a hard problem (e.g., what treatment to give to a patient with a kidney stone?) by building simple decision functions describing only relevant facts and not a full computational theory of the domain (e.g., a full theory of the function of the kidneys from the nephron up, and the interaction of the kidneys with the rest of the body are *not* needed to conclude that removing the stone or breaking it up with ultrasounds will be sufficient for curing the patient with a kidney stone).

(g) **Perform operations on compact representations.** This strategy involves replacing intractable large data structures with declarative and highly compact representations. For example, "every person who suffers from a mental health disorder" encompasses an estimated 10^9 people globally but does not enumerate or even identify them. This approach also involves operating on classes of the problem space simultaneously rather than each member in the class. This is

particularly evident in several forms of ML modeling where astronomical numbers of model structures are scored at once and represented compactly.

Best Practice 2.4
When faced with intractable problems, consider using strategies for mitigating the computational intractability by trading off with less important characteristics of the desired solution.

Heuristic and Ad Hoc Approaches and the Prescientific-to Scientif ic Evolutionary AI/ML Continuum

The term **"Heuristic" AI/ML methods or systems** refers to several types of systems or strategies: First, *rules of thumb* that may give a good solution sometimes but do not guarantee this. Second, *functions used inside AI search methods* to accelerate finding problem solutions. Third, *ad hoc* systems, i.e., that are not designed based on a formal frameworks for AI/ML, and do not guarantee strong or safe performance in a generalizable sense. Fourth, methods and systems *applied outside their scope of guaranteed safe, effective, or efficient use* (i.e., hoping that an acceptable solution may be produced).

To clarify these concepts consider the following examples (note: all of the mentioned methods and systems will be thoroughly discussed in this and subsequent chapters):

- "We need at least 10 samples per variable when fitting an ordinary least squares regression model" is an example of the first type of heuristic. Another example of the first heuristic type is "choosing 100 genes at random from a cancer microarray dataset will yield predictor models with very high accuracy for diagnosis, often near the performance of special gene selection algorithms" (for surprised readers, not familiar with such data, this is because there is large information overlap and redundancy among genes with respect to clinical cancer phenotypes).
- The "Manhattan distance" as an estimate for the spatial distance between the current location and the goal location in a robot navigation problem is a heuristic than when used inside the A* search algorithm (see later in present chapter) allows the algorithm to find a path with minimum cost to the goal. This is an example of the second type of heuristic.
- The well-known INTERNIST-I system for medical diagnosis in internal medicine was an example of the third type of heuristic AI. It lacked a formal AI foundation both in knowledge representation and inference. It was shown to be highly accurate in certain tests of medical diagnosis problems, however [4].

- Examples of the fourth type are: (1) using Naïve Bayes (a formal ML method that assumes very special distributions in order to be correct) in distributions where the assumptions are known to not hold, hoping that the error will be small. (2) Using Propensity Score (PS) techniques for estimating causal effects, without testing the distributional assumption that makes PS correct (i.e., "strong ignorability", which is not testable within the PS framework). (3) Using Shapley values, a Nobel-Prize winning economics tool devised for value distribution in cooperative coalitions to explain "black box" ML models (a completely different task, for which the method was not designed or proven to be correct; as we will see later in the present volume, it can fail in a wide variety of models). (4) Using IBM Watson, a system designed and tested in an information retrieval task (Jeopardy game) for health discovery and care (for which it had no known properties of correctness or safety). (5) Using Large Language Models (LLMs), e.g., ChatGPT and similar systems (designed for NLP and conversational bot applications) for general-purpose AI tasks (not supported by the known properties of LLMs).

For the purposes of this book, the third and fourth type are most interesting and we will focus on them in the remainder of this section.

In earlier times in the history of health AI/ML as well as broad AI/ML, proponents of ad hoc (heuristic type 3) systems argued that as long as heuristic systems worked well empirically they should be perfectly acceptable especially if more formally-constructed systems did not match the empirical performance of heuristic systems or if constructing formal systems or establishing their properties was exceedingly hard. Proponents of formal systems counter-argued that this ad-hoc approach to AI was detrimental since one should never feel safe when applying such systems, especially in high-stakes domains. At the same time many proponents of the formal approach engaged in practices of the fourth type of heuristic (not testing assumptions, or using a system designed for task A, in unrelated task B).

From a more modern scientific perspective (with substantial benefit of hindsight) of performant and safe systems operating in high-risk domains such as health, the above historical debate is more settled today than in the earlier days of exploring AI. **Heuristic systems and practices represent pre-scientific approaches** in the sense that a true scientific understanding of their behavior does not exist (yet) and that with sufficient study in the future, a comprehensive understanding of a heuristic system of today can be obtained. In other words, the heuristic system of today will be the well-characterized, scientific, non-heuristic system of tomorrow.

In this book we adopt the sharp distinction:

> *AI/ML systems with well understood theoretical properties and empirical performance vs. systems that lack these properties (aka Heuristic systems).*

A further distinction can be made regarding whether a system is based on formal foundations or being ad hoc. The importance of formal systems is that they make the transition to well-understood systems faster and easier. In the absence of formal

foundations, it is hard to derive properties and expected behavior. If formal foundations exist, often many of the properties are immediately inherited from the general properties of the underlying formal framework. In any case, deriving formal properties of methods with strong formal foundations is vastly easier than of ad hoc methods.

In addition, there is a **strong practical interplay between theoretical properties and empirical performance**. If theory predicts a certain behavior and empirical tests do not confirm it, this means that errors likely occurred in how the models/systems have been implemented and debugging is warranted. Alternatively, it may suggest that we operate in a domain with characteristics that are different from the theoretical assumptions of our model (and we need to change modeling tools or strategy). If model A empirically outperforms model B on a task for which A is not built but B is theoretically ideal, this suggests that there are implementation errors or evaluation data/methodology errors in model B, and so on. In other words, a strong theoretical understanding bolsters, and is enhanced by, the empirical application and validation. What is not working well is lacking one or both of these important components (theoretical base + empirical base). More on the interplay of theorical properties and empirical performance can be found in chapter "Principles of Rigorous Development and Appraisal of ML and AI Methods and Systems".

It is also significant to realize that there is an **evolutionary path** from pre-scientific informally-conceived systems, to partially-understood (theoretically or empirically) systems, and finally to fully-mature and well-understood AI/ML.

In earlier related work Aliferis and Miller [5] discussed the "grey zone" between formal systems with known properties but with untestable or unknown preconditions for correctness in some domain, and ad hoc systems with unknown properties across the board. Their observation was that both classes required a degree of faith (with no guarantees) for future success. This early work can be elaborated taking into account the following parameters: *formal or ad hoc foundation; known theoretical properties or not; whether the known properties are testable and have been tested vs not; known empirical performance or not; and whether empirical performance is satisfactory and what alternatives may exist.*

The following table (Table 3) distills the above multi-dimensional space to its essential cases and describes this landscape and developmental journey from pre-scientific systems (lacking properties, rows 1, 3) to intermediate level systems (with partial properties, rows 2, 4, 5, 6), to mature reliable science-backed systems (with known properties, rows 7, 8). The table also points to pitfalls and BPs of building and using systems of the listed characteristics.

It is worth emphasizing that systems with known properties are not automatically optimal or even suitable for solving a problem. Knowledge of properties of various methods and approaches can be used to find the best solution for a task, however.

Table 3 Classification of AI/ML systems based on their formal foundation and properties. The development spectrum from pre-scientific to mature science-backed systems

	Built on formal theory	Known theoretical properties	Known empirical performance	Comments and further dimensions/considerations
1	No	No	No	• Ad hoc with unknown theoretical properties and performance. • Using such systems is a major pitfall and use should be avoided until they are better understood.
2	No	No	Yes	• Ad hoc with unknown properties but known empirical performance in a number of empirical evaluations. • Using such systems is a major risk and needs to take into account the range of evaluation, how good the performance is, whether the evaluation matches the application domain, and whether better alternatives exist.
3	Yes	No	No	• Systems with formal underpinnings but with unknown theoretical properties and empirical performance. • Examples of those systems are systems built on established mathematical frameworks but being poorly mapped to a biomedical problem of interest. • Using such systems is a major pitfall and should be avoided until they are better understood.
4	Yes	No	Yes	• Systems with some formal underpinnings: built on formal foundations but with unknown properties and known empirical performance. • Examples of those systems are systems built on established mathematical or computational frameworks but being poorly mapped to a biomedical problem of interest. • Using such systems entails major risks and should be avoided until risks are better understood.
5	No	Yes	No	• Systems initially starting as ad hoc that eventually evolved to having known properties but yet unknown empirical performance. • This case is in paractice equivalent to formal systems of type (6).
6	Yes	Yes	No	• Theoretically understood but poorly tested (not yet mature) formal systems. • Lack of empirical performance data leaves open the possibility for misalignment of theory with the application domain. Potentially high risk of empirical failure indicates the need for empirical validation before deployment.
7	No	Yes	Yes	• Systems initially starting as ad hoc that eventually evolved to having known properties and known performance. • This case is in practice equivalent to formal systems of type (8).
8	Yes	Yes	Yes	• Fully-realized, fully mature formal systems with known properties and empirical performance. • They can immediately inform whether they can solve the problem at hand (in absolute terms and compared with alternatives).

Chapters "Foundations of Causal ML" and "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" further elaborate on how these concepts can be implemented in practice during the practical development of performant and safe AI/ML.

Pitfall 2.4
Believing that heuristic systems can give "something for nothing" and that have capabilities that surpass those of well-characterized systems. In reality heuristic systems are pre-scientific or in early development stages.

Best Practice 2.5
As much as possible use models and systems with established properties (theoretical + empirical). Work within the maturation process starting from systems with unknown behaviors and no guarantees, to systems with guaranteed properties.

Foundations of AI: Logics and Other Symbolic AI and Non symbolic Extensions

Symbolic vs. Non-Symbolic AI

Logic is a staple of science and the cornerstone of all types of so-called *symbolic AI.*

By **symbolic AI** we refer to AI formalisms that focus on representing the world with symbolic objects and logical relations, and making inferences using logical deductions.

Symbolic systems typically contain **deep, structured representation of the problem solving domain.**

Examples include *production systems, rule-based systems, semantic networks, deductive reasoners, causal modeling with detailed causal relations,* and other types of systems discussed later in this chapter.

By contrast, **non symbolic AI** encompasses various formal systems that focus on uncertain and stochastic relationships using various forms of inference that either rely on probability theory or can be understood in terms of probability.

Non-symbolic systems are typically **shallow representations** of input-output relationships without a detailed model of the structure of the problem solving domain.

Terminology Caution: Deep Learning neural networks are designated as such because they have many hidden node layers (as opposed to shallow ANNs that have few). However they are both shallow AI systems because they lack a rich representation of the domain and its entities (i.e., they are **ontologically shallow**).

Examples of non-symbolic AI (in the ontological shallowness sense) include connectionist AI that approaches AI from an artificial neural network point of view, probabilistic AI that uses a probability theory perspective, shallow causal models, genetic algorithms that adopt an evolutionary search perspective, reinforcement learning, predominantly within the data-driven ML which is the currently dominant paradigm of AI, and most recently Large Language Models (LLMs).

There exist also **formalisms that transcend and attempt to unify symbolic and non-symbolic AI** such as causal probabilistic graphs, probabilistic logics or ANN-implemented rule systems.

See chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring problems, and the role of Best Practices" for discussion of this important class of AI/ML.

Propositional Logic (PL)

Propositional logic [1, 6] is the simplest form of logic allowing the construction of sentences like:

$$\begin{pmatrix}((\text{Symptom\_positive\_A} = \text{True}) \wedge (\text{Test\_positive\_B} = \text{False})) \\ \vee \neg(\text{Test\_positive\_C} = \text{True})\end{pmatrix} \\ \rightarrow \text{Diagnosis\_Disease1} = \text{True}$$

Or in words, if the patient has symptom A and test B is negative, or if she does not have a positive test C, then she has Disease1.

As can be seen, PL uses *propositions* (statements) that can take values in {*True, False*}, and logical *connectives* (*and, or, implication, negation, equivalence, parentheses*). By combining the above based on the straightforward *syntax* of PL, we create complex sentences that may be valid or not. Other than the (tautological) meaning of the truth values {*True, False*}, the precise meaning (*semantics*) of the propositions is embedded in them (i.e., it is not explicit in the PL language).

A *PL Knowledge Base* (KB) contains a set of sentences that are stated as *axioms* (true propositions or valid complex sentences) by the user and then other sentences can be *constructed and proven t*o be valid or not using the **truth table** of a PL sentence. The correspondence between the validity of propositions and sentences in the KB and the real world is provided by the notion of a *model* for that KB which is some part of the world (e.g., a biomedical problem domain) where the KB truth assignments hold. Syntactic operations (e.g., by a computer) on the KB prove the validity of sentences in all models of that KB. Inferring that some manipulation of computer symbols has automatically a valid interpretation in the real world (originating from the validity of axioms) is commonly referred by the expression "the computer will take care of the syntax and the semantics will take care of themselves".

Truth *tables* can be used to show that a sentence is valid or not by examining if the sentence is true for all truth assignments of the propositions involved (hence valid), otherwise it is not valid. Sentences are decomposed to smaller parts in a truth table so that truth values for the sentence can be determined. Commonly-used inference steps are encapsulated in *inference rules* such as *Modus Ponens, And-elimination, And-introduction, Resolution*, etc. These are used to avoid constructing very large/complex truth tables. The computational complexity of proving that a sentence in PL is valid is worst-case intractable but quite manageable in small domains [1].

First Order Logic (FOL)

FOL is a vastly more powerful form of logic than PL and can represent:

- Objects (e.g., patients, genes, proteins, hospital units)
- Properties of objects (e.g., alive, up-regulated, secondary structure, full)
- Relations among objects and terms(e.g., patient 1 has a more severe disease than patient 2, gene 1 determines phenotype 1, protein 1 catalyzes chemical reaction 2, hospital unit 1 is less utilized than unit 2)
- Functions (e.g., BMI of a patient P, length of a gene G, molecular weight of a protein P, bed capacity of a hospital unit U)

The **syntax of FOL uses**:

- Objects,
- Constants,
- Variables,
- Predicates used to describe relations among constants or variables,
- Functions over constants, variables or other functions
- Connectives: equivalent, implies, and, or
- Parentheses
- Negation

- Quantifiers: there exists, for all (defined over objects)
- Terms formed from constants, variables and functions over those
- Atomic sentences defined over predicates applied on terms
- Complex sentences defined using atomic sentences, connectives, quantifiers, negation and parentheses.

For a technical syntax specification see [1].

Higher order logics allow expression of *quantifiers applied over functions and relations* (not just objects). These logics are more powerful, but inference is much harder, thus logic-based AI typically deals with FOL or simpler derivative formalisms.

The application of FOL (or derivatives) to build a Knowledge Base (KB) useful for problem solving in some domain is an instance of **Knowledge engineering**. It involves: (a) **ontology engineering,** that is identifying and describing in FOL (or other appropriate language chosen during ontology engineering) the **ontology** (objects, types of relationships) in that problem domain; and (b) **knowledge acquisition** that is identifying and describing in FOL the relevant axioms (facts) describing key aspects of the domain, and from which inferences can be drawn to solve problems.

In the health sciences and healthcare a number of ontologies have been created and are widely used. A most significant component of those are the common data models used to describe entities and variables. These are of essential value for both symbolic and data driven ML methods and for harmonizing data and knowledge across health care providers, studies, and scientific projects. See chapter "Data Preparation, Transforms, Quality, and Management" for a discussion of the most commonly used common data models and standards.

Knowledge engineering could be substituted for ordinary programming however the fundamental advantage of Knowledge engineering is that it is a **declarative approach to programming** with significant advantages (whenever applicable) such as: ability to represent compactly facts and complex inferences that may be very cumbersome to conventional procedural or functional programming methods. Moreover, once the AI knowledge engineer has constructed the knowledge base, *then a myriad of inferences can be made using the pre-existing inferential mechanisms of FOL.* In other words, no new problem solving apparatus is needed, because it is provided by FOL. Declarative programming needs only a precise statement of the problem.

Logical Inference

FOL has a number of sound inference procedures that differ in their completeness. Such procedures are ***Generalized Modus Ponens (that can be used in Forward-Chaining, Backward-Chaining directions), and Resolution Refutation*.**

Forward chaining is an algorithm that starts from the facts and generates consequences, whereas Backward chaining starts from what we wish to prove and works backward to establish the necessary precedents. The "chaining" refers to the fact that as new sentences are proven correct, they can be used to activate new rules until no more inferences can be made.

As a very simple example consider a KB with:

Axioms:	A, B
Rules of the type x → y:	A → C, and C ∧ B → D

From this KB,

The **Forward Chaining** algorithm will perform the following sequence of operations:

1. From A and A → C it will infer C and add it to the KB
2. From B, C and C ∧ B → it will infer D and add to the KB
3. Will terminate because no new inferences can be made

The **Backward Chaining** algorithm from the same original KB, and user request to prove D, will:

1. First see that D is not an axiom
2. Identify that C ∧ B → D can be used to try and prove D
3. Will seek to prove C and B separately
4. B is an axiom so it is true
5. C is not an axiom but rule A → C can be used to prove it
6. Will seek to prove A
7. A is an axiom thus true
8. Thus (by backtracking to (5) C is true
9. Thus (by backtracking to (2) D is true
10. Terminate reporting success in proving D

Forward and backward chaining strategies are widely used in biomedical symbolic AI expert systems. They are not FOL-complete however! Recall that Goedel proved that in sufficiently complex reasoning systems (such as FOL) there are true statements that cannot be proven from the axioms of the system. He also proved that if there are provable sentences, then there exists an algorithm to generate the proof. Robinson [7] discovered such an algorithm (Resolution Refutation) which operates by introducing the negation of a sentence we wish to prove in the knowledge base

and deriving a contradiction. Resolution Refutation is complete *with respect to what can be proven in FOL.* For technical details of the algorithm refer to [1, 7].

The **Resolution Refutation** algorithm in the KB of the previous example, will:

1. Add ¬ D to the KB
2. From A and A → C it will infer C and add it to the KB
3. From C and C ∧ B → D it will infer D and add to the KB
4. From D and ¬ D it will derive a contradiction and will terminate declaring success in proving D

The above examples are *hugely simplified* by not addressing predicates, variables, functions, quantification, conversion to different canonical forms and their matching, which are all needed for the general case algorithms operation. For technical details see [1, 6, 7]. Nilsson [7] in particular gives a definitive technical treatment of rule-base systems and their properties.

Logic-Derivative Formalisms and Extensions of FOL

FOL is almost never used in its pure form in biomedical AI applications. Instead, it serves as a foundation for other more specialized and invariably simplified formalisms. Occasionally researchers have extended ordinary FOL to accommodate reasoning with probabilities, or time. The following Table 4 lists important FOL derivatives and extensions.

Table 4 Types of logic-based systems (FOL derivatives)

- **Decision Trees** which are very widely used both in the construction of clinical guidelines and as a language for ML (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare & Health Science").
- **Rule based systems** for discovery or clinical care based on forward or backward chaining algorithms and extensions for reasoning under uncertainty e.g., the highly influential systems DENDRAL, META-DENDRAL and MYCIN [8, 9]
- **Logic-based programming** e.g., the widely-used PROLOG language [10]
- **Non chaining rule based decision support** e.g., the widely-used ARDEN SYNTAX for clinical decision support [11]
- **Semantic networks/"slot-and-filler" representations, semantic WWW, and taxonomies, e.g.,** [12, 13]
- **Boolean networks** e.g., widely used for biological pathway discovery and modeling [14]
- **Symbolic NLP systems** (see chapter "Considerations for Specialized Health AI and ML Modelling and Applications: NLP" for details and references)
- **Ontologies and declarative representations e.g.,** BIOPORTAL [15]
- **Fuzzy logic e.g.,** [16]
- **Non monotonic logic e.g.,** [17]
- **Probabilistic, and temporal logics e.g.,** [18]
- **Planning systems e.g.,** used for therapy planning or for industrial and operations planning purposes [7, 19]

Although FOL is not used without major modifications and simplifications in the above, it remains the most important theoretical framework for understanding the structure, capabilities and limitations of such methods and systems. Chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring problems, and the role of Best Practicess" discuss present-day concerns in the AI/ML science and technology community that a drastic departure from symbolic AI (e.g., in favor of purely statistical and ontologically shallow input-output representations), does not bode well for the ability of the field to successfully address the full range and complexity of health science and health care problems.

Non-Symbolic AI for Reasoning with Uncertainty

Numerous non-symbolic methods exist and the most important ones in current practice are covered in detail in chapters "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" and "Foundations of Causal ML". Here we will address two methods of great importance in the modern practice of healthcare and health science research: Decision Analysis and Bayesian networks.

Methods that have predominantly historical significance will not be addressed, in order to preserve reader and book bandwidth and focus more on techniques that are part of modern practices.

Decision Analysis (DA) and Maximum Expected Utility (MEU)-Based Reasoning

Decision Analysis using Maximum Expected Utility stems for the fundamental work of Von Neumann and Morgenstern dating back to 1947. This theory provides a model of *prescriptive* decision making designed to limit the risk of a decision agent facing uncertainty. Whereas the theory may not be universally applicable in all situations involving biomedical decisions with uncertainty, they still describe a powerful model with wide applicability.

The principles of MEU and DA can be readily grasped with a simple example. Consider the hypothetical case of a patient facing the dilemma of whether to undergo a surgery for a condition she has, or to opt for the conservative treatment. Assume that either decision cannot be followed by the other (e.g., a failed surgery precludes improvement by the conservative treatment, whereas the conservative treatment exceeds the time window when the surgery is beneficial).

Furthermore let the probability of success of surgery in such patients be p(surgical success) = 0.9 and probability of success of non-surgical treatment in such patients be p(nonsurgical success) = 0.6. Finally let the quality of life (measured in a *utility scale* ranging in [0,1]) after successful surgery be 0.8, after failed surgery be 0.2, after successful conservative treatment be 1, and under failed conservative treatment be 0.5.

Utility assessment protocols designed to identify a patient's preferences and map them to a utility scale exist. Expected utility defines four axioms describing a

rational decision maker: completeness; transitivity; independence of irrelevant alternatives; and continuity [57]. The principle of MEU decision making based on these axioms, designates the optimal decision as the one that maximizes the expected utility over all possible decisions:

$$Optimal\ decision = \text{argmax}_i\ \mathbf{E}\left(U\left(decision_i\right)\right)$$

where

$$\mathbf{E}\left(U\left(decision_i\right)\right) = \sum_j \mathbf{E}\left(U\left(outcome_{ij}\right)\right)$$

and:

- $U\ (decision_i)$ is the expectation of the utility of the i^{th} decision,
- $U\ (outcome_{j,i})$ *is the patient-specified utility of the* j^{th} *outcome based on decision* i and
- $\boldsymbol{E}\ (U(outcome_{j,i})$ *is the expected patient-specified utility of the* j^{th} *outcome based on decision* i

In our hypothetical example, we can easily see that

$$\mathbf{E}\left(U\left(decision_{surgery}\right)\right) = 0.9 * 0.8 + 0.1 * 0.2 = 0.74$$

whereas

$$\mathbf{E}\left(U\left(decision_{conservative}\right)\right) = 0.6 * 1 + 0.4 * 0.5 = 0.80$$

The decision that maximizes expected utility is thus the non-surgical treatment.

In graphical form the above scenario is captured by the following **Decision Analysis** Fig. 1.

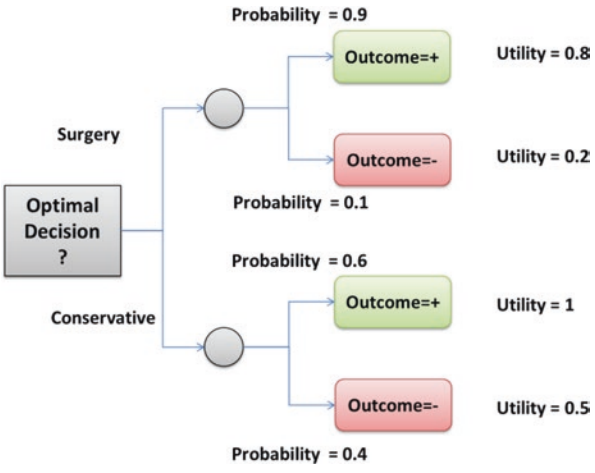


Fig.1 A decision analysis tree augmented with probabilities and utilities corresponding to the hypothetical example in the text

We note that MEU DA, whenever applicable, is a powerful and principled way to make decisions that maximize benefit. Pitfalls in MEU-DA based reasoning are:

Pitfalls 2.5
Decision Analysis (DA) and Maximum Expected Utility (MEU)-based reasoning

1. Errors in the estimation of probabilities for various events.
2. Errors in eliciting utility estimates in a way that captures patients' true preferences (including using the care providers' utilities rather than the patients').
3. The structure or complexity of the problem setting defies analyst's ability to completely/accurately describe it.
4. Developing a DA for one population and applying in another with different structure of the problem, different probabilities for action-dependent and action-independent events, or with different preferences.

The corresponding best practices are addressing these sources of errors that can lead a decision analysis astray.

Best Practice 2.6
Decision Analysis (DA) and Maximum Expected Utility (MEU)-based reasoning

1. Ensure that the structure of the problem setting is sufficiently/accurately described by the DA tree. Omit known or obvious irrelevant factors.
2. Elicit utility estimates in a way that captures patients' true preferences using established utility-elicitation methods.
3. Accurately estimate probabilities of action-dependent events and action-independent events.
4. In most conditions, and whenever applicable, data-driven approaches should be preferred to subjective probability estimates. Use probability-consistent statistical or ML algorithms to estimate the probabilities.
5. Ensure that the decision analysis is applied to the correct population.
6. Conduct sensitivity analyses that reveal how much the estimated optimal decision is influenced by uncertainty in the specification of the model.
7. Whenever possible, produce credible intervals/posterior probability distributions for the utility expectations of decisions.

These cover only the most salient aspects of the art and science of MEU driven decision analysis. A more detailed introduction is given in [20] and a comprehensive applied treatment in [21].

Reasoning with Uncertainty: Probabilistic Reasoning with Bayesian Networks

Bayesian Networks (BNs) are an AI/ML family of models that can describe the probabilistic (or deterministic, or hybrid) relationships among variables. They have extensive usability, range of application, attractive properties and thus high practical value. They can support several use cases and types of problem solving (which can be combined):

- **Use 1:** Overcome the limitations of intractable (brute force Bayes), or unduly restrictive (Naive Bayes), classifiers.
- **Use 2:** They are very economical to describe (i.e., they have space complexity that closely mirrors the distribution complexity).
- **Use 3:** They can be created both from expert knowledge and from data (or with hybrid sources).
- **Use 4:** They can be used for MEU DA (providing probability estimates for DAs or in their "influence diagram" form).
- **Use 5:** They can perform flexible classification and other sophisticated probabilistic inferences.
- **Use 6:** They can be thought of as probability-enhanced logical rules and combine forward and backward, as well as forward-backward inferences in a way that is probabilistically accurate.
- **Use 7:** They can be used (with very mild additional restrictions) to reason causally including: (1) Distinguishing between *observing* passively a variable's value vs. applying interventions that cause the variable to take that value, and reason accordingly. (2) Reasoning from causes to outcomes, from outcomes to causes, and simultaneously in both directions over multiple and overlapping causal pathways.
- **Use 8:** Their causal variants can be used to discover causality, not just perform inferences with existing causal models.
- **Use 9:** They have close relationship to the Markov Boundary theory of optimal feature selection.

Because of these properties we touch upon various forms and derivatives of BNs in several chapters and contexts in this volume: AI reasoning under uncertainty (chapter "Foundations and Properties of AI/ML Systems"), Bayesian classifiers (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"), Markov Boundary based feature selection (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science") and causal discovery and modeling (chapter "Foundations of Causal ML").

We caution that not every graph or every probabilistic graph is a BN and the BN properties derive from very specific requirements. Because there is confusion in parts of the literature (where some authors derive models that are not BNs but present them as such), we will provide here, for clarity, an unambiguous technical description of this family of AI/ML models.

BN Definitions

Definition. Bayesian Network. A BN comprises (1) a directed acyclic graph (DAG); (2) a joint probability distribution (JPD) over variable set ***V*** such that each variable corresponds to a node in the DAG; and (3) a restriction of how the JPD relates to the DAG, known as the Markov Condition (MC).

Definition. Directed Graph. directed graph is a tuple <***V***,***E***>, where ***V*** is a set of nodes representing variables 1-to-1, and ***E*** is a set of directed edges, or arrows, each one of which connects an ordered pair of members of ***V.***

Definition: Two nodes are **adjacent** if they are connected by an edge. Two edges are adjacent if they share a node.

Definition: A ***path*** is any set of adjacent edges.

Definition: A **directed path** is a path where all edges are pointing in the same direction.

Definition: A **directed acyclic graph** (DAG) is a directed graph that has no cycles in it, that is, there is no directed path that contains the same node more than once.

Definition: The **joint probability distribution** (JPD) over ***V*** is any proper probability distribution (i.e., every possible joint instantiation of variables has a probability associated with it and the sum of those is 1).

Definition: Parents, children, ancestors, and descendants: In a directed graph, if variables X,Y share an edge X→ Y then X is called the parent of Y, and Y is called the child of X. If there is a directed path from X to Y then X is an ancestor of Y and Y is a descendant of X.

Definition: Spouses: In a directed graph, the spouses of a variable Vj is the set comprising all variables that are parents to at least one of the children of Vj.

Definition: The **Markov Condition (MC)** states that every variable V is independent of all variables that are non-descendants of V given its direct causes.

Definition: If *all dependencies and independencies* in the data are the ones following from the MC, then the encoded JPD is a **Faithful Distribution** to the BN and its graph.

Definition: Degree of a node is the number of edges connected to it. In a directed graph, this can be further divided into **in-degree** and **out-degree**, corresponding to the number of parents (edges oriented towards the node) and children (edges oriented away from the node) that the node has.

Definition: A **collider** is a variable receiving incoming edges from two variables. For example in: X → Y ← Z, Y is the collider. A **collider is either "shielded" or "unshielded"** iff the corresponding parents of the collider are connected by an edge or not, respectively. Unshielded colliders give form to the so-called "**v-structures**".

Definition: A **trek** is a path that contains no colliders.

Definition: The **graphical Markov Boundary** of a variable Vj is the union of its parents Pa(Vj), its children Ch(Vj) and the parents of its children Sp(Vj).

Definition: The **probabilistic Markov Boundary** of a variable Vj is the set of variable S that renders Vj conditionally independent of all other variables, when we condition on S, and is minimal.

Key Properties of Bayesian Networks

Unique and Full Joint Distribution Specification. If the Markov Condition (MC) holds, then the conditional probabilities of every variable given its parents specifies a well-defined and unique joint distribution over variables set ***V.***

Any Joint Probability Distribution Can be Represented by a BN. If we wish to model JPD J1 by a BN, we can order the variables arbitrarily, connect with edges every variable Vj with all variables preceding it in the ordering, and define the conditional probability of Vj given the parents in the graph equal to the one calculated from J1. Then the implied JPD J2 of the BN will be J2 = J1. Note: the outline constructive proof presented here is a large sample result. Much more sample-efficient procedures exist for small sample situations.

The Joint Distribution of a BN Can Be Factorized Based on Parents. The joint distribution is factorized as a product of the conditional probability distribution of every variable given is direct causes set

$$probability\left(V_1, V_2, \ldots, V_k\right) = \prod_j probability\left(V_j \mid Pa\left(V_j\right)\right) \tag{4}$$

Where j indexes the variables in **V**, and $Pa\ (V_j)$ is the set of parents of variable V_j.

Because of this factorization, we only need to specify up to |V| conditional probabilities in order to fully specific the BN (where |V| is the number of variables. When all variables have a small number of parents, the total number of probabilities is linear to |V|. By comparison in a Brute Force Bayes classifier we always need specify $2^{|V|}$ probabilities.

Similarly whenever we need to compute the joint probability of Eq. 4, for a particular instantiation of the variables involved, this is a linear time operation in the number of variables.

Definition: D-separation

1. Two variables X, Y connected by a path are d-separated (aka the path is "blocked") given a set of variables S, if and only if on this path, there is
   (a) A non-collider variable contained in S, or
   (b) A collider such that neither it nor any of its descendants are contained in S.
2. Two variables, X and Y, connected by several paths are d-separated given a set of variables S, if and only if in all paths connecting X to Y, they are d-separated by S.
3. Two disjoint variable sets X and Y are d-separated by variable set S iff every pair <Xi,Yj > is d-separated by S, where Xi and Yj are members of X, Y respectively.

Inspection of the Graph of a BN Informs us About all Conditional Independencies in the Data. By inspection **(by eye or algorithmically)** of the causal graph (and application of *d-separation*) we can infer all valid conditional independencies in the data, *without analyzing the data* as follows:

- If variable sets X and Y are d-separated by variable set S then they will be conditionally independent given S in the JPD encoded by the BN.

Inspection of the Graph of a BN Encoding a Faithful Distribution, Informs us about all Conditional Independencies and Dependencies in the Data. A BN encoding a faithful distribution entails that all dependencies *and* independencies in the JPD can be inferred by the DAG by application of the d-separation criterion as follows:

- If variable sets X and Y are d-separated by variable set S in the BN graph, then they will be conditionally independent given S in the JPD encoded by the BN. Otherwise they will be dependent.
- Equivalently:
- Variable sets X and Y are conditionally independent given S in the JPD encoded by the BN, iff they are d-separated by variable set S in the BN graph.

Therefore in faithful distributions, the BN graph becomes a map (so-called **i-map**) of dependencies and independencies in the data JPD encoded by the BN. Conversely, by inferring dependencies and indecencies in the data we can construct the BN's DAG and parameterize the conditional probabilities of every variable given its parents, effectively recovering the unoriented causal process that generates the data. This is a **fundamental principle of operation of causal ML methods (**discussed in more detail in chapter "Foundations of Causal ML").

A Variable in a BN is Independent of all Other Variables Given its Graphical Markov Boundary and Equivalently a variable in a JPD is independent of all other variables given its Probabilistic Markov Boundary in Faithful Distributions.

Relationship of Markov Boundary and Causality. This can be used to obtain optimal feature sets for predictive modeling when the BN is known or is inferred from data. Because the graphical and probabilistic Markov Boundary are identical in faithful distributions, in causal BNs, there is a close connection of the local causal network around a variable and its probabilistic Markov Boundary (see chapter "Foundations of Causal ML").

BNs Allow Flexible Inference

We will illustrate flexible inference with an example depicted in Fig. 2.

In Fig. 2 part (1) we see a BN model for some problem solving domain. In part (2) we query the BN with the question: what is the probability of F (grey node) given that we have observed the values of variables {C, B, H} (green nodes)? The inference algorithms propagate and synthesize information upward (e.g., from C and B to A) and downward from A and B to F. Notice that given B, H is irrelevant to F.

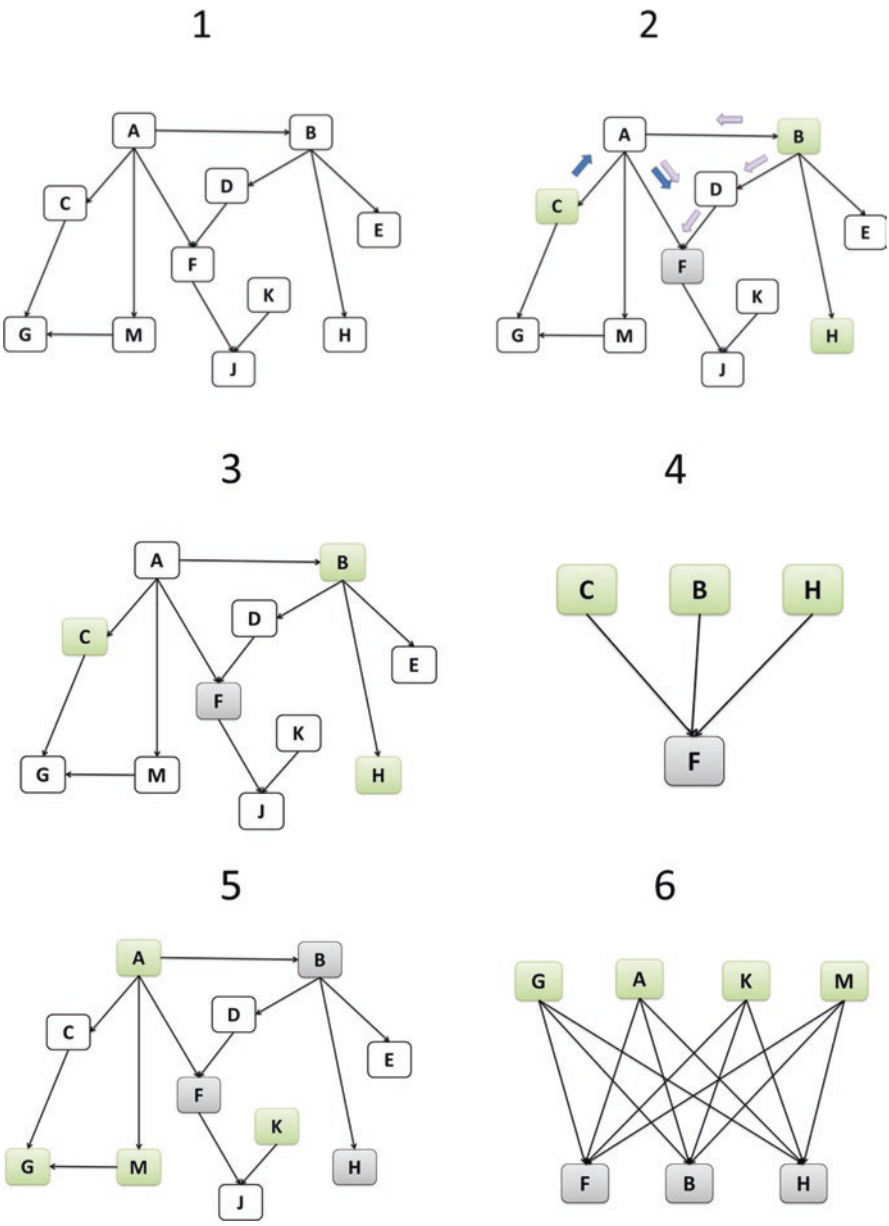


Fig. 2 Flexible predictive modeling and forward/backward reasoning in BNs

If we wish to set up a conventional classifier (of any type, Logistic Regression, Boosting, Deep Learning, SVM, Random Forest etc., it has to obey a fixed input-response structure depicted in (4)); in other words in order to answer this question we need to approximate a function of the probability (F | C, B, H) and train it *from scratch for that query*. The BN (3) requires no modification however.

If we wish to answer next what is the joint probability of {F, B, H} given that we observe {G, A, K, M}, again the same BN (5) can give us the answer. Other predictive modeling methods however (6) will need to be trained *from scratch* to estimate: probability (F, B, H | G, A, K, M).

Because the number of such queries grows exponentially to the number of variables, it is essentially impossible to answer all the answerable queries by a BN by training specialized classifiers. Moreover *any* subdivision of the variables sets as observed, unknown, or query variables is allowed and needs not be known a priori.

We now examine (Fig. 3) how a causal BN (see chapter "Foundations of Causal ML" for formal definitions) can answer causal questions. Consider the query: what is the probability of F (grey node) given that we have observed the values of variables {B, H} (green nodes) and we have manipulated C to take a specific value (via intervention denoted by *do(C)*)? The causal BN (left) knows that when we manipulate a variable, nothing else can affect it. Thus the Arc: A→C is effectively eliminated by the manipulation in the context of the query. Consequently, information does not travel from C to F via A as in the case of observing C. The predictive modeling models lacking causal semantics (e.g., Logistic Regression, Boosting, Deep Learning, SVM, Random Forest etc.) will propagate information from C to F thus arriving at a wrong conclusion. Incidentally this problem cannot be fixed in the conventional predictive modelers by eliminating C from the model, since valid causal/information paths may exist from C to F than need be considered even if we manipulate C (and indeed the causal BN will do so).

Computational complexity for both learning BNs and for conducting inference with them is intractable in the wort case. However highly successful mitigation strategies have led to super-efficient average case or restricted-purpose algorithms (see chapters "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Lessons Learned from Historical Failures, Limitations and Successes of ML in Healthcare and the Health Sciences. Enduring Problems and the Role of Best Practices" for details). Key references for properties of BNs discussed here are [22–25].

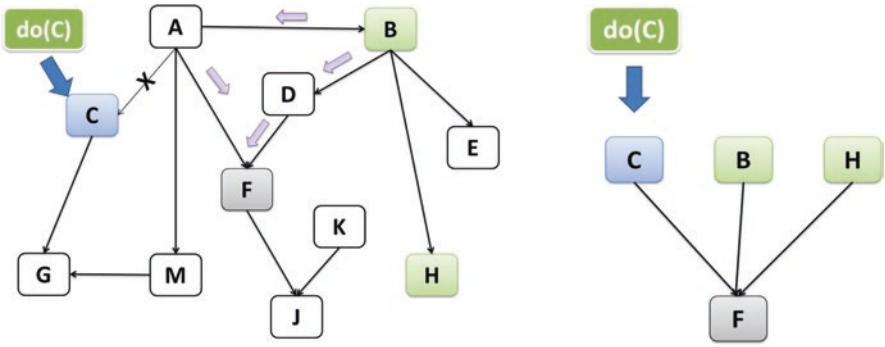


Fig. 3 Causally Consistent Inference with BNs AND Forward/Backward reasoning

AI Search

Search as a General Problem Solving Strategy. Conventional Search Versus AI Search

Search is a general problem solving methodology in which many (if not most) problems can be reduced to. Somewhat similar to physical search of an object or a location inside a physical space, AI search constructs a state space with each state representing a possible solution or partial solution to a problem. The search algorithms then traverse this state space trying to discover or construct a solution. For example, in a ML context, the state space could comprise models fit from data, such that each model has a different structure and corresponding estimated generalization predictivity. AI search in the ML context seeks then to find ML models that achieve the highest or sufficient high predictivity. In an autonomous navigation context, AI search would seek to find a navigation path that achieves smallest traversed distance, smallest cost of trip, or other objectives. In a scheduling context AI search may seek to schedule patients and operating room personnel to operating rooms so that cancelations are minimized. The diversity of problems that can be solved with search is infinite and covers the full space of computable functions.

General AI Search Framework: "Best" First Search (BeFS) Family and Variants

Whereas search is also accomplished outside AI, most notably with linear and non linear optimization methods, ad hoc search algorithms, and Operations Research algorithms, AI search has distinctive qualities:

- AI search can use state spaces that are infinite in size.
- AI search can attempt to solve problems in the hardest of complexity classes.
- AI search admits any computable goal, not just a small space of computable functions.
- AI search can operate with symbolic and non-symbolic representations.

Table 5 outlines a very general framework for AI search.

Table 5 High-level operation of general AI search:

- The general AI search algorithm maintains a priority list of states that will be explored.
- This list is initialized with a starting search state, and an iterative loop begins:
- If the first state in the priority list it is not the goal state then the algorithm
  Creates all successor states from it and updates the list contents and then prioritizes those according to a prioritizing function. Else the algorithms returns the goal state and terminates
- The process will stop iterating when
  – A goal state has been found or
  – No more states can be reached or
  – Another termination criterion is met.
  Each state also stores the path to reach that state and the cost of that solution path. Thus when a goal state is returned, the solution path can be extracted

This prima facie very simple procedure has immense power and flexibility and can be instantiated in a variety of ways leading to different behaviors and properties. For example, the following variants can be had as follows (Table 6):

Table 6 Notable instantiations of general BeFS AI search:

Instantiation of prioritizing function	Resulting type of search	Properties
Ranks states in priority list in ascending order according to order that states were generated (i.e. first in first out, or FIFO)	Depth first search	Will terminate and find a solution if search space is finite and enough computation time and space are allowed Worst case complexity = O (state space size). May not find optimal solution For finite state spaces organized as trees of depth *d* and breadth *b:* Worst case time complexity = O (b^d). Space complexity is O (m * *b*) where *m* is the maximum depth of a path. Preferable if many optimal solutions are arranged in same search tree depth level
Rank states in priority list in descending order according to order that states were generated (i.e. last in first out, or LIFO)	Breadth first search	Will find a solution and terminate if search space is finite and enough computation time and space are allowed. May not find optimal solution Worst case complexity = O (state space size). For finite state spaces organized as trees of depth *d* and breadth *b:* Worst case space and time Complexity = O (b^d) Preferable if many solutions are arranged in same search path
Rank states in priority list in ascending order according to cost of path to each state	Uniform cost search (aka branch and bound)	Will find a solution if search space is finite and enough computation time and space are allowed. Will find optimal solution if path cost is non decreasing Worst case complexity = O (state space size). For finite state spaces organized as trees of depth *d* and breadth *b:* Worst case complexity = O (b^d)
Rank states in priority list in ascending order according to estimated cost of state to goal state	Greedy Search (ie., most locally promising next step *with* backtracking)	Will find a solution if search space is finite and enough computation time and space are allowed. May not find optimal solution For finite state spaces with max depth path *m:* Worst case time complexity = O (b^m) If estimates of cost to goal are good then it may reach a solution very fast

Table 6 (continued)

Instantiation of prioritizing function	Resulting type of search	Properties
Rank states in ascending order according to: Estimated cost of state to goal state. Also eliminate from priority list all but the most locally promising next state.	Hill climbing Search (most locally promising next step *without* backtracking)	May not find a solution or if it finds one it may not be optimal. It may not terminate if search space is infinite For finite state spaces with max depth path *m:* Worst case time complexity = O (m * b) and space complexity is = 1 Has tendency to be trapped in locally optima that are not globally optimal solutions If, however, search space is convex or concave then it finds the optimal solution with time complexity = depth of solution and space complexity = 1. This fact is exploited by many mathematical convex optimization algorithms [26]
Rank states in priority list in ascending order according to: (cost of path to that state + estimated cost of state to goal state) with the constraint that estimated cost to goal cannot exceed true cost.	A*	Will find a solution if search has finite branching factor, and enough computation time and space are allowed Will find optimal solution No other search algorithm using the same estimated cost to goal state function can outperform A* For max depth path m: Worst case space complexity = O (b^m) Worst case time complexity is polynomial if the error of the heuristic cost estimate to goal state will not grow faster than the logarithm of the "perfect heuristic" h* that returns the true distance to the goal

The fundamental general search algorithm and its instantiations can be readily extended to cope with infinite size search spaces and computing environments with limited space using *depth-limited*, i*terative deepening* and *simplified memory-bounded A** versions. For details see [1].

Other Notable AI Search Methods

In addition to the above "classical" AI search algorithms notable search methods include **Simulated Annealing, Genetic Algorithms, Ant Colony Optimization and search procedures applicable to rule based systems and resolution refutation.**

Simulated annealing [27] is inspired by metallurgy and the annealing process. It comprises a classic hill climbing method modified to incorporate a randomized

jump to one of nearby states so that local optima have a larger chance (but no guarantee) of not trapping the algorithm in suboptimal solutions.

Genetic algorithms [28, 29] are inspired by biological evolution which they mimic. Genetic algorithms represent solutions in digital chromosome representation on which they perform, just like evolution does on actual organisms, random mutations and crossover operations. This is their way to generate successor solution states. Multiple states are maintained at each stage of the algorithms with the best-fit ones having a smaller chance to be discarded. The algorithm has the attractive property that an exponentially increasing portion of better performing states are considered in each step. It can be applied in domains where the data scientists now nothing about the domain (i.e., they are "black box optimizers"). On the other hand, they have important limitations: they are not guaranteed to reach an optimal solution (i.e., can be trapped in local optima), it has been proven that they cannot learn certain classes of functions (e.g. epistatic functions); and they are not efficient when compared to non-randomized algorithms solving the same problems.

Ant Colony Optimization (ACO) and **"Swarm intelligence"** is inspired by the foraging behavior of some ant species. These ants deposit pheromone on the ground in order to mark some favorable path that should be followed by other members of the colony. ACO can be used for graph searching, scheduling problems, classification, image recognition and other problems. For several ACOs, convergence has been proven (albeit at an unknown number of iterations). Finally empirical results in >100 NP-hard problems has shown competitive and ocasionaly excellent performance compared to best known algorithms [30].

Specialized search procedures include AO* (suitable for searching AND/OR graphs used in decomposable rule based systems), MINIMAX and ALPHA-BETA search (suitable for game tree search), and Resolution Refutation search strategies (e.g., unit preference, set of support identification, input resolution, linear resolution, subsumption etc.) designed to make the resolution refutation algorithm reach a proof faster [1, 6, 7].

AI/ML Languages

Whereas statement of algorithms and theoretical analysis is typically conducted using ***pseudocode***, practical development depends on choice of programming languages. A few languages are particularly suitable for AI/ML and their properties are summarized in the next Table 7:

Table 7 Notable AI/ML Languages

Language	Properties and suggested use
Pseudocode	• Comprises generally-stated data structures, code modularization (e.g., functions, procedures), and control structures that are programming language independent. • May also contain declarative and logical statements that can be implemented in any applicable programming language. For example, a universally quantified statement can be converted to a conventional loop. • It is widely understood by all computer scientists. • Especially appropriate for stating algorithms and conducting theoretical analysis. • They can be passed to programmers that will implement them in a programming language of choice.
LISP	• The most powerful and flexible language for AI and possibly the most powerful programming language ever created. • Systems of immense complexity and capabilities can be implemented very compactly and easily. • Particularly suitable to symbolic AI. • Incorporates procedural, functional, and object oriented paradigms. • Uses lists (with dynamic memory management) as primary data structures. More efficient or special purpose data structures can readily be implemented. • Compiled and interpreted. • LISP programs and data are interchangeable. Programs are themselves data that can be generated or modified by programs. Programs can modify themselves or other programs at runtime or off line. • The language symbols can be assigned to different functions. Thus the language itself can be modified by programs. • These immense powers may create interpretability problems since programmers do not immediately know what a program will do, e.g., they have to understand how the usual language features are modified at runtime.
Prolog	• Designed for symbolic AI. • Uses backward chaining rule-based programming paradigm. • Declarative programming. • (Surprisingly) is resolution-refutation complete. • Not as widely used anymore.

(continued)

Table 7 (continued)

Language	Properties and suggested use
Matlab	• Extremely powerful language and development environment especially suitable for ML. • Uses matrices and matrix operations as fundamental building blocks. • Extremely optimized operations can produce immensely efficient programs. • Very rapid development. • Interpreted and compiled. • Can be interfaced with all major languages. • Numerous toolboxes (libraries) cover all major mathematics, machine learning, engineering, imaging, bioinformatics, etc. types of development project needs. • On the downside it is a commercial product that requires paid educational or commercial licenses.
R	• Flexible and powerful language well suited for statistical and ML development. • Numerous open source libraries. • No license costs. • Many libraries and codes are unoptimized and may also have implementation errors. • Does not scale as well as other languages.
Python	• Flexible language especially well-suited to text processing and ML. • Interpreted and compiled. • Numerous open source, no-cost libraries. • Varying degrees of quality of available free codes.
A sample of other languages commonly encountered in the AI/ML space	• C, C++ and Objective C. • Pascal and variants (e.g. Delphi). • Pearl. • Ruby. • Basic and variants (e.g., Visual Basic). • Older languages (FORTRAN, MODULA, COBOL) are seldomly used for new development. • Assembly language: for specialized applications where speed optimization is of paramount importance. • MUMPS for EHR-focused programming. • SQL: very useful for relational database querying.

Foundations of Machine Learning Theory

AI and ML as Applied Epistemology

Epistemology is the branch of philosophy concerned with knowledge: its generation and sources, nature, its achievable scope, its justification, the concept of belief as it relates to knowledge, and related issues [31]. From the perspective of this volume it is worthwhile to notice first that AI formalizes knowledge so that it can be used in

applied settings. AI can also inform what types of knowledge are computable and what inferential procedures can be applied and with what characteristics. ML in particular, by virtue of being able to *generate knowledge from data,* on one hand obtains its justification not just by empirical success but by epistemological principles of science. On the other hand, ML puts to test epistemological hypotheses and theories about how knowledge *is, can, or should* be generated. The following sections provide a concise outline of the key theories that provide the firm scientific ground on which ML is built and in particular for fortifying its performance and generalization properties. They also summarize a few related pitfalls and high level best practices that will be developed further in other chapters.

ML Framed as AI search and the Role of Inductive Bias

We showed earlier in this chapter how AI search can be used to solve hard problems. ML itself can be cast as a search endeavor [29, 32, 33]. In this framework, ML search comprises:

(a) A model language ***L*** in which a *family* of ML models can be represented. For example, decision trees, logic rules, artificial neural networks, linear discriminant equations, causal graphs etc. Typically the model language will come with associated procedures that enable models to be built when data ***D*** are provided (i.e., model fitting procedure ***MF***).

(b) A data-generating or design procedure ***DD*** that creates data (typically by sampling from a population or other data-generating process) from which models are fit.

(c) A hypothesis space ***S***. The language ***L*** implicitly defines a space ***S*** comprising all models expressible in the language and that can be fit with ***MF*** applied on ***D***, with each model $\boldsymbol{M_i}$ representing a location or state in ***S.*** For example, the space of all decision trees, neural networks, linear discriminant functions, boosted trees, etc. that can be built over variable set ***V*** using ***MF*** on ***D.***

(d) A search procedure ***MLS*** that navigates the space in order to find a model representing an acceptable solution to the ML problem as defined by a goal criterion. For example, a steepest ascend hill climbing search procedure over the space of decision trees over ***V*** *fit by* **MF** *given data* **D.**

(e) A goal (or merit) function ***GM*** that examines a search state (i.e., a model $\boldsymbol{M}$i) and decides whether it is a solution, or how close to a solution it is (its merit function value). For example, whether $\boldsymbol{M_i}$ has acceptable predictivity, uncertainty, generalizability etc. or what is the merit value (e.g., difference of its properties to the goal ones).

The tuple: < ***L, MF, DD, S, MLS, GM*** > defines the **architecture of a ML method.**

Every ML method can be described and understood in these terms (although additional perspectives and analytical frameworks are also valuable, and in some cases necessary).

The tuple: < ***L, MF, S, GM*** > describes what is commonly referred to as a **ML "algorithm",** whereas **MLS** describes the **model selection procedure** that ideally will incorporate an **error estimator procedure** for the final (best) model(s) found. **GM** and its estimators from data may or may not be identical to the *error function* and its estimators (see chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models").

The tuple: < ***L, MF, S, MLS, GM*** > describes the **Inductive Bias** of that ML method.

The inductive bias of a ML method is the **preference ('bias") of that method for a class of models** over other models that are not considered at all or are not prioritized by the method.

Notice that in practice there are *two search procedures* in operation:

MF is a search procedure in the space of model parameter values once a model family and its model family parameter values (aka "**hyperparameters"**) have been visited by the **second (top-level, or over-arching) search procedure MLS which searches over possible model families and their hyper parameters**.

For example, the search procedure in decision tree induction algorithm is a greedy steepest ascent while a hyper parameter may be the minimum number of samples allowed for accepting a new node or leaf. In an SVM model the search procedure is quadratic programming and hyper parameters may be the cost C and the kernel functions and their parameters. The model selection procedure that decides over these two model families and the right hyperparameter values for them may be *grid search* using a cross validation error estimator, or other appropriate model selection process (see chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" for details).

The ML search framework above readily entails important properties of ML:

1. The choice of model language affects most major model properties like error, tractability, transparency/explainability, sample efficiency, causal compatibility, generalizability etc.
2. The data generating procedure implements the principles and practices of data design which is a whole topic by itself (see chapter "Data Design for Biomedical AI/ML"). Because the whole operation of ML as search is so dependent on the data ***D***, the *data design/data generating procedure strongly interacts with the other components and determines the success of the ML model search*.
3. The search space typically has infinite size, or finite but astronomically large size.

4. The ML search procedures MF typically have to find sparse solutions in the infinite/practically infinite search space. Therefore they are often custom-tailored or optimized for the specific ML algorithm.
5. The MLS search procedures are designed to operate over several ML algorithm families and their hyper-parameters. They are typically much less intensive and typically informed by prior analyses in the problem domain of similar data, giving guidance about which hype-parameter ranges will likely contain the optimal values.

   Worth noting in particular with respect to the inductive bias:
6. The match of the inductive bias of a ML method to the problem one wishes to solve (hence the data generating function to be modeled and the data design procedure that samples from the data generating function) determines the degree of success of this ML method.
7. It also follows, that if a ML method does not have restrictions on inductive bias, it cannot learn anything useful at all, in the sense that it would accept any model equally as well as any other (i.e., accept good and bad models alike) and in the extreme it would amount to random guess among all conceivable models).
8. At the same time, a successful ML method must not have a too restrictive inductive bias because this may cause lack the ability to represent or find good models for the task at hand.
9. Taken together (6), (7), and (8) show that a successful ML method must find the right level of restriction or "openness" of the inductive bias.

We note that *the inductive bias of ML is a useful bias* and **should not be used as a negative term** (as for example, ethical, social, or statistical estimator biases which are invariably negative).

Pitfall 2.6
Using the wrong inductive bias for the ML solution to the problem at hand.

Pitfall 2.7
Ignoring the fit of the data generating procedure with the ML solution to the problem at hand.

Best Practice 2.7
Pursue ML solutions with the right inductive bias for the problem at hand.

Best Practice 2.8
Create a data generating or design procedure that matches well the requirements of the problem at hand and works with the inductive bias to achieve strong results.

ML as Geometrical Construction and Function Optimization

As will be elaborated in chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" ML methods are in some important cases cast as geometrical constructive solutions to discriminating between objects. Figure 4 below shows a highly simplified example of diagnosing cancer patients from healthy subjects on the basis of 2 gene expression values. The ML method used (SVMs in the example) casts this diagnostic problem as geometric construction of a line (in 2D space, and hyperplane in higher dimensions) so that the cancer patients are cleanly separated from health subjects (and subject to a maximum gap achieved between the two classes).

Such geometrical formulations of ML can be analytically and algebraically described and then operationalized using linear algebra and optimization mathematical tools and codes. See chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" for several examples and details on the mathematical formulations and the ensuing properties.

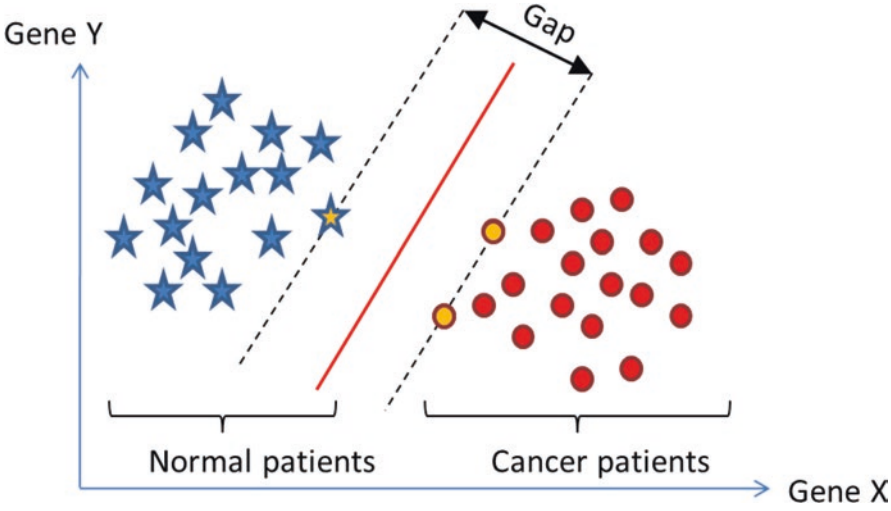


Fig. 4 *Geometrical constructive formulation of ML.* See chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" for mathematical formulation

Computational Learning Theory (COLT): PAC Learning, VC Dimension, Error Bounds

Computational Learning Theory (COLT) , formally studies under which conditions learning is feasible and provides several bounds for the generalization error depending on the classifier used, the definition of error to be minimized (e.g., number of misclassifications), and other assumptions. While theoretical results in classical statistics typically make distributional assumptions about the data (i.e., the probability distribution of the data belongs to a certain class of distributions), COLT results typically make assumptions only about the class of discriminative model considered. Notice though, that it may be the case that an optimal discriminative model never converges to the data generating function of the data.

COLT research has defined several mathematical models of learning. These are formalisms for studying the convergence of the errors of a learning method. The most widely-used formalisms are the **VC (Vapnik-Chervonenkis) and the PAC (Probabilistically Approximately Correct) analyses**. A VC or PAC analysis provides bounds on the error given a specific classifier, the size of the training set, the error on the training set, and a set of assumptions, e.g., in the case of PAC, that an optimal model is learnable by that classifier. Typical PAC bounds, for example, dictate that for a specific context (classifier, training error, etc.) the error will be larger than epsilon with probability less than delta, for some given epsilon or delta. Unlike bias variance decomposition, COLT bounds are independent of the learning task. From the large field of COLT we suggest [34–36] as accessible introductions.

The VC (Vapnik-Chervonenkis) dimension (not to be confused with the VC model of learning above) is (informally) defined as the *maximum* number of training examples that can be correctly classified by a learner *for any possible assignment of class labels*. The VC dimension of the classifier is a quantity that frequently appears in estimation bounds in a way that all else being constant, higher VC dimension leads to increased generalization error. Intuitively, a low complexity classifier has low VC dimension and vice-versa. An example of VC bound follows: if VC dimension h is smaller than l, then with probability of at least 1-n, the generalization error of a learner will be bounded by the *sum of its empirical error (i.e., in the training data) and a confidence term* defined as:

$$\sqrt{\frac{h\left(\log\frac{2l}{h}+1\right)-\log\left(n/4\right)}{l}}$$

Where $0 < n \leq 1$. Notice how this error bound is independent of dimensionality of the problem [37]. The number of parameters of a classifier does not necessarily correspond to its VC dimension. In [38] (examples are given of *a classifier with a single parameter that has infinite VC dimension and classifiers with an unbounded number of parameters but with VC dimension of 1*.

Thus, a classifier with a large number of parameters (but a low VC dimension) can still have low error estimates and provide guarantees of non-over-fitting. In

addition, some of these bounds are non-trivial (i.e., less than 1) even when the number of dimensions is much higher than the number of training cases.

Such results prove unequivocally that learning is possible (when using the right learning algortihms) in the situation common in modern health science and healthcare data where the number of observed variables is much higher than the number of available training sample. Many popular classical statistical predictive modeling methods in contrast break down in such situations.

The mentioned COLT results also justify the assertion that over-fitting is not equivalent to a high number of parameters. Unfortunately, many of the estimation bounds provided by COLT are not tight for the number of samples available in common practical data analysis. In addition, COLT results often drive the design of classifiers with interesting theoretical properties, robust to the curse of dimensionality, and empirically proven successful, such as Support Vector Machines (discussed in detail chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science").

ML Theory of Feature Selection (FS)

Traditional ML theoretical frameworks (e.g., PAC and VC frameworks of COLT) focus on generalization error as a function of the model family used for learning, sample size and complexity of models. The theory of feature selection is a newer branch of ML that addresses the aspect of selecting the right features for modeling. It aims to guide the design and proper application of principled feature selection (as opposed to heuristic FS with unknown and suboptimal properties).

Table 8 summarizes key areas and example of results in the theory of feature selection. In the remainder we will discuss two formal feature selection frameworks (i.e., Kohavi-John and Markov Boundary) and will describe certain classes of feature selection problems that are commonly addressed.

The *standard feature selection problem.* Consider variable set V and a data distribution J over V, from which we sample data D. Let T be a variable which we wish to predict as accurately as possible by fitting models from D. The *standard feature selection problem* is typically defined as [40]:

- *Find the smallest set of variables S in V s.t. the predictive accuracy of the best classifier than can be fit for T from D, is maximized.*

Kohavi-John framework for Standard predictive feature selection problem. Kohavi and John [39] decompose the standard feature selection problem as follows:

Table 8 Summary of major topics and examples in the theory of feature selection

Topic	Notes
Filter-wrapper and embedded-explicit feature selection taxonomy	*Wrapper algorithms* conduct a heuristic search in the space of feature subsets and evaluate them using a classifier and loss function of choice. *Filters* examine the data distribution and infer desirable and undesirable features independent of classifier. *Embedded feature selection* removes undesired features as part of fitting a model. The removal may be *implicit* (.e.g. regularization where features with small coefficients may stay in the model but do not influence it much) or *explicit* (i.e., features are dropped out of the model). See [39–41] and chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" for details.
Ugly duckling and no free lunch theorems	Proofs that the choice of feature selection must be tied to a specific class of target functions, learners and loss functions. See [32, 42] and NFLT section below.
Bespoke characterization of individual FS or classifier models	This body of work examines theoretically (and tests empirically) whether specific ML algorithms and feature selectors are capable of solving specific feature selection problems (e.g., [43–45])
Kohavi-John framework of relevancy	Defines what are necessary/indispensable, or useful but redundant, or useless features in a general sense (without reference to algorithms) [39].
Markov Boundary framework of relevancy (and intersection with other forms of ML)	Defines what are necessary/indispensable, useful but redundant, and useless features in a general sense and leads to construction of specific algorithms. Also allows extensions for causality, equivalence classes, and guided experimentation [24, 41, 46, 47].

A feature X is **strongly relevant** if removal of X alone will result in performance deterioration of the Optimal Bayes Classifier using the feature. Formally:

X is strongly relevant iff: $X \not\perp T \mid \{V - X, T\}$

A feature X is **weakly relevant** if it is not strongly relevant and there exists a subset of features, S, such that the performance of the Optimal Bayes Classifier fit with S is worse than the performance using $S \cup \{X\}$. Formally:

X is weakly relevant iff: X is not strongly relevant and $\exists\ S \subseteq \{V\text{-}S, T\}$ s.t. $X \not\perp T \mid S$.

A feature is **irrelevant** if it is not strongly or weakly relevant.

The strongly relevant feature set solves the standard feature selection problem.

Intuitively, choosing the strongly relevant features provides the minimal set of features with maximum information content and thus solves the standard feature selection problem since a powerful classifier in the small sample or the Optimal Bayes Classifier in the large sample will achieve maximum predictivity. The Kohavi-John framework does not provide efficient algorihms for discovery of the strongly relevant feature set, however.

Markov Boundary framework for Standard predictive feature selection problem. Recall from the section of Bayes Networks (BNs) that a set S is the Markov Boundary of variable T (denoted as S = MB(T)), if S renders T independent on every other subset of the remaining variables, given S, and S is minimal (cannot be reduced without losing its conditional independence property). This is the MB(T) *in the probabilistic sense*. Tsamardinos and Aliferis [24] connected the Kohavi-John relevancy concepts with BNs and Markov Boundaries as follows:

In faithful distributions there is a BN representing the distribution and mapping the dependencies and independencies so that:
1. The strongly relevant features to T are the members of the MB(T).
2. Weakly relevant features are variables, not in MB(T), that have a path to T.
3. Irrelevant features are not in MB(T) and do not have a path to T.

Thus in faithful distributions, the Markov boundary MB(T) is the solution to the standard feature selection problem and algorithms that discover the Markov boundary implement the Kohavi-John definition of strong relevancy.

Local causally augmented feature selection problem and Causal Markov Boundary. In faithful distributions with causal sufficiency (see chapter "Foundations of Causal ML") there is a causal BN that is consistent with the data generating process and can be inferred from data in which: strongly relevant features = members of MB(T), and also comprise the solution to the local causally augmented feature selection problem of finding:
1. The direct causes of T.
2. The direct effects of T.
3. The direct causes of direct effects of T.

Thus in faithful distributions with causal sufficiency, the causal graphical MB(T) set is connected with the probabilistic MB(T). Inducing the probabilistic MB(T) then: 1. Solves the standard predictive feature selection problem, and 2. Solves the *local causally augmented feature selection problem* [24, 41].

Equivalency-augmented feature selection problem and Markov Boundary Equivalence Class. In faithful distributions the MB(T) exists and is unique [22]. However, in non-faithful distributions where variables or variable sets exist that have the same information for the target variable (i.e., target information equivalences exist in ("*TIE distributions*") and we may have more than one MB(T) [46]). The number of Markov Boundaries can be exponential to the number of variables [46] and in empirical tests with real life genomic data Statnikov and Aliferis extracted tens of thousands of Markov boundaries before terminating the experiments [48].

In TIE distributions:

1. The Kohavi John definitions of relevancy break down since there are no Kohavi-John strongly relevant features any more, only weakly relevant and irrelevant ones. This is because if S1, S2 are both in the MB equivalence class {MBi(T)} then: S1 $\perp$ T | S2 *and* S2 $\perp$ T | S1.
2. *The 1-to-1 causal and probabilistic relationship of the probabilistic and graphical MB(T) breaks down.* A variable can be a member in some MBi(T) without having a direct causal or causal spouse relationship with T.
3. The standard predictive feature selection problem is solved by the smallest member in the equivalence class of MBi(T).
4. The *Equivalency-augmented feature selection problem* is to find the equivalence class of all probabilistic MBi(T).

Chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" provides further details of the above 3 fundamental feature selection problem classes, organizes them into a hierarchy of increasing difficulty, shows examples, and presents and contrasts practical algorithms based on their ability to solve these problems.

Theory of Algorithmic Causal Discovery and of Computational Properties of Experimental Science

The theory of causal discovery extends traditional ML theoretical frameworks that focus on generalization error, by investigating the feasibility, complexity, and other properties of causal discovery algorithms from passive observational data, of experimental interventional approaches (e.g., RCTs, biological experiments, etc.) and hybrid experimental-observational algorithmic approaches. Pearl provides a comprehensive modern theory of causality [49] and Spirtes et al., a historically influential algorithmic framework for its discovery [50]. Chapter "Foundations of Causal ML" presents an extensive introduction to the function of causal discovery algorithms from non-experimental data, and their properties under specific assumptions. Chapter "Foundations of Causal ML" also lists several algorithms used for causal discovery, including more modern and scalable ones. Chapter "An Appraisal and

Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" as well as section of "ML Framed as AI search and the Role of Inductive Bias" of the present chapter reference theory and algorithms at the intersection of feature selection and causality. Moreover we mention here selected additional fundamental results that will round the readers' understanding of causal discovery from a theory perspective:

Eberhardt et al. showed that under assumptions: *if any number of variables are allowed to be simultaneously and independently randomized in any one experiment, then log2 (N) + 1 experiments are sufficient and in the worst case necessary to determine the causal relations among N ≥ 2 variables when no latent variables, no sample selection bias and no feedback cycles are present* [51]. Bounds are provided when experimenters can't intervene on more than K variables simultaneously. These results point to fundamental limitations of RCTs and biological experiments conducted with small number of variables manipulated at a time, and is further discussed in chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring problems, and the role of Best Practices".

The same researchers showed that: *by* ***combining experimental interventions with causal algorithms*** *for graphical causal models under familiar assumptions of causal induction, with perfect data, N - 1 experiments suffice to determine the causal relations among N > 2 variables when each experiment randomizes at most one variable* [52]. These results require that all variables are simultaneously measured, however.

Statnikov et al. [47] showed that in TIE distributions (i.e., with multiple equivalent Markov Boundary sets with respect to the response variable T), an algorithm exists that guides experimentation combined with causal discovery from observations, so that at most *k* single-variable experiments are needed to learn the local causal neighborhood around T where *k* is the size of the union of all Markov Boundaries of T.

Mayo-Wislon showed [53] that *for any collection of variables V, there exist fundamentally different causal theories over V that cannot be distinguished unless all variables are simultaneously measured. Underdetermination can result from piecemeal measurement, regardless of the quantity and quality of the data*.

The same investigator in [54] found that *when the underlying causal truth is sufficiently complex, there is a significant possibility that a number of relevant causal facts are lost by trying to integrate the results of many observational studies in a piecemeal manner*. Specifically, he shows that as the graph gets large, if the fraction of variables that can be simultaneously measured stays the same, then the proportion of causal facts (including e.g., who mediates what relationships) that can be learned *even with experiments*, approaches 0.

Optimal Bayes Classifier

The optimal Bayes Classifier (or OBC for short) is defined by the following formula:

$$argmax_{(i)} \sum_{j} P\left(T = i|Mj\right) * P\left(Mj|\, D\right)$$

Where *i* indexes the values of the response variable T and *j* indexes models in the hypothesis space where the classifier operates. In plain language, the OBC calculates the posterior probability that a model has generated the data (i.e., it is the data generating function) given the data, for every model in the hypothesis space. It also calculates for each of the response variable's values, the probability for that value's probability given each model. The predictions are summed over all models, weighted by the probabilities of the models given the data, and the value with the higher value is the one that the classifier outputs.

Because the hypothesis space can be infinite or intractably large, the calculations involved are also intractable. Also, if we calculate the conditional probabilities using Bayes' rule we also have to deal with the problem of prior probability assignment over the model space members; in case of very biased priors, the calculated posteriors will converge slowly to the large sample correct ones. These issues place the application of the OBC outside the realm of the practical. However, it turns out that the error of the OBC is optimal in the large sample. Hence the OBC is a *valuable analysis tool* when we consider the errors of various learning algorithms by comparing them to the OBC error (as we will see in chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science") [29, 32].

No Free Lunch Theorems (NFLTs)

NFLTs is a general class of theorems each one applying to optimization, search, machine learning and clustering (in the last case referred to as Ugly Duckling Theorem or UDT for short) [32, 42].

The crux of these theorems is that under a set of conditions intended to describe all possible application distributions, there is no preferred algorithm, and that by implication the right algorithm should be chosen for the right task, since there is no dominant algorithm irrespective of task. This particular interpretation is commonsensical and useful. It is also stating in different terms essentially the notion that a well-matched inductive bias to the problem at hand will led to better solutions.

This is especially important for clustering algorithms and the UDT. The UDT entails that in the absence of external information, there is no reason to consider two patterns P1 and P2 more or less similar to each other than P3. Over all possible functions associated with such patterns (and the features that define them) any grouping is as good as any other. This implies that similarity/distance functions that define the behavior of clustering algorithms *must be tailored to specific use contexts of the resulting clusters* (which in turn entails a restriction on the class of functions modeled).

The problems with common use of clustering are three-fold: (a) Per the UDT, clustering by algorithm X is as good as random clustering over all possible uses of

the clusters. Unless we select or construct a distance/similarity function designed to solve the specific problem at hand, clustering will not provide any useful information. (b) There is no useful unbiased clustering. Researchers who present clustering results as "unbiased" (meaning "hypothesis free" - a practice very common in modern biology research and literature) fail to realize that any practical clustering algorithm has an inductive bias implemented as a distance function and as a grouping/search strategy. And as we saw, in the absence of (a well-chosen) inductive bias little can be accomplished. (c) Finally *clustering should not be used, for predictive modeling* [55]. Clustering algorithms can only know something about a classification problem, e.g. of response T as T+ or T-, if and only if we design a similarity function that distinguishes between T+ and T- and embed it in the clustering algorithm. But this function is precisely a predictive modeling classifier, rendering the whole clustering-for-prediction endeavor, redundant.

In chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" we give examples of the above as well as recommendations for goal-specific clustering.

Pitfall 2.8
Probably more than other theoretical results, the NFLT for ML has the largest risk to be misunderstood and misapplied.

In summary form the NFLT for ML states that *all learning methods have on average the same performance over all possible applications,* as a mathematical consequence of 3 conditions:

(a) The algorithm performance will be judged over all theoretically possible target functions that can conceivably generate data.
(b) The prior over these target functions is uniform.
(c) Off Training Set Error (OTSE) will be used to judge performance [32, 42].

This result has been misinterpreted to suggest that we could use models that have low instead of high accuracy according to unbiased error estimators and do as well as when choosing the high accuracy models. In this (mis)interpretation random classification is as good overall as classification using sophisticated analytics and modeling. The mathematics of the NFLT derivation are impeccable but the results are problematic because of the flaws of the 3 underlying assumptions:

(a) In real life a tiny set of data generating functions among infinite ones are the ones that generate the data. *Nature is highly selective to its distributions*.
(b) The prior distribution over these data generating functions is highly skewed.

Taken together assumptions (a) and (b) of the NFLT, are *mathematically equivalent with a label random reshuffle procedure* (see chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models"). This procedure distorts the relationship between inputs and response variable and creates a distribution of target

functions that on average have zero signal. Because on average there is no signal, this target function space is on average random and thus unpredictable. Therefore no learning algorithm exists that can do better than random and NFLT, naturally, finds and states as much. If such an algorithm existed then the distribution would be predictable and thus non-random. The NFLT for ML then just says that when there is no signal (on average), every algorithm will fail (i.e., will be as good as the random decision rule on average) and thus all algorithms will be equally useless (on average).

(c) In addition, by OTSE excluding the input patters that have been seen by the algorithm during training, an artificially low biased performance estimate is obtained for future applications. By contrast statistical theory and all branches of statistical ML and of science adopt for purposes of validation OSE (Off Sample Error) which is just a random sample from the data generating function.

In the present volume we specifically discussed at some length the dangers in overinterpreting the NFLT for ML because of published claims that the theorem somehow entails that choosing the models with best cross validation error (or best independent validation error, or best reproducibility of error) are just as good as choosing the model with worst reproducibility or independent validation error [42].

Best Practice 2.9
Cross validation and independent data validation, as well as their cousin reproducibility, are robust pillars of good science and good ML practice and are not, in reality, challenged by the NFLT.

Best Practice 2.10
Clustering should not be used for predictive modeling.

Best Practice 2.11
A very useful form of clustering is post-hoc extraction of subtypes from accurate predictor models.

Universal Function Approximators (UFAs) and Analysis of Expressive Power

UFAs are ML algorithms that *can represent any function that may have generated the data*. UAF theorems establish that certain ML algorithms have UAF capability [29].

Pitfall 2.9
If a ML algorithm cannot represent a function class, this outright shows the inability or sub- optimality of this algorithm to solve problems that depend on modeling a data generating function that is not expressible in that algorithm's modeling language.

For example, clustering (i.e., grouping objects or variables into groups according to similarity or distance criteria) does not have the expressive power to represent the causal relationships among a set of variables or entities. Thus the whole family of clustering algorithms is immediately unsuitable for learning causal relationships. Similarly, simple perceptron ANNs cannot represent non-linear decision functions and that places numerous practical modeling goals and applications outside their reach.

By contrast, Decision Trees can represent any function over discrete variables. Similarly, ANNs can represent any function (discrete or continuous) to arbitrary accuracy by a network with at least three layers [29]. BNs can represent any joint probability distribution as we show in the present chapter. AI search can be set up to operate on model languages that are sufficiently expressive to represent any function as well. Genetic Algorithms, being essentially search procedures share this property.

Pitfall 2.10
UAF theorems should not be over-interpreted. While it is comforting that e.g., algorithm A can represent *any function in the function family* F (i.e., the model space and corresponding inductive bias are expressive enough), learning also requires effective (space and time-tractable, sample efficient, non-overfitting etc.) model search and evaluation in that space.

For example, Decision Trees (DTs) do not have practical procedures to search and learn every function in the model space expressible as a DT, since practical (tractable) DT induction involves highly incomplete search of the hypothesis space. Similarly, ANNs can represent any function however, the number of units needed and the time needed for training are intractable and the procedures used to search in the space of ANN parameters are not guaranteed to find the right parameter values.

Generative vs. Discriminative Models

Generative models are typically considered the ones that can model the full joint distribution of the variables in an application domain. Discriminative models, by contrast, are ones that only model a decision function that is sufficient to problem of interest in that domain. Consider as example the SVM hyperplane model in Fig. 4. This model solves the diagnostic problem stated perfectly without modeling the probability distribution of the variables involved.

The optimal choice of generative vs. discriminative model entirely depends on the application domain. For example, for general predictive modeling as well as other pattern recognition such as text categorization, the use of discriminative models confers practical advantages and better performing models than generative models, in many datasets. For causal modeling, simulation, natural language understanding, density estimation, or language generation, generative models are necessary or advantageous. We also wish to clarify a **terminology confusion** (especially in non-technical literature) between generative modeling at large vs "Generative AI". The latter refers to a small number of specific classes of algorithms that generate data (e.g., Generative Adversarial Networks, Large Language Models) with established or unknown properties. Generative modeling on the other hand includes all methods that model the data generating distribution and in typical usage the term refers to algorithms that have guarantees for correct modeling of the data generating distribution (e.g., BNs, Logistic Regression, Density estimator algorithms).

Best Practice 2.12
The choice of generative vs. discriminative modeling affects quality of modeling results and has to be carefully tied to the problem domain characteristics. All else being equal discriminative models confer efficiency (computational and sample) advantages.

Bias-Variance Decomposition of Model Error (BVDE)

The concept of BVDE is one that originates from statistical machine learning but has broad applicability across ML and all of data science. It is pervasively useful, yet not as widely known as it deserves among non-technical audiences, so we will present it here in some detail. A detailed treatment can be found in [56]. While the whole idea of BVDE is that other than noise in measurements (which is intrinsic in the data and independent of modeling decisions) or inherent stochasticity of the data generating function (which is intrinsic in the data generating process and independent of modeling decisions), the remaining modeling error of any ML (or for that matter any statistical or quantitative data science) model has two components: a component due to the inductive bias mismatch with the problem vs. the data at hand; and another component due to sample variation in small sample settings.

In the terminology of BVDE, the error due to inductive bias mismatch is referred to as "bias" with "high bias" indicating a severe mismatch, toward simplicity (aka small complexity, or small capacity) of the model language (and related search and fitting procedures) with respect to the data generating function. The error due to sampling variation is referred to as "variance" with variance increasing as sample size decreases. More precisely, the bias is the error (in the sample limit, relative to the data generating function) of the best possible model that can be fitted within the class of models considered. The variance is the error of the best model (in the small sample, within the class of models considered) relative to the error of the best model that can be fitted in the large sample within the class of models considered. The bias

then is a function of the learner used and the data generating function; the variance, for a fixed data generating function, is a function of the learner and the sample size.

Implications for modeling: When the modeling bias is fixed one can reduce total model error by increasing the sample size (reduce variance), and when the sample is fixed one can reduce total error by optimizing the bias. More importantly, when both sources of error are under analyst control, BVDE explains that there is an ideal point of balance of error due to bias and error due to variance. The optimal error will be found when these two sources of error are balanced for a particular modeling setting. Moreover high bias models have smaller variance (i.e., are more stable in low samples) but on average over many samples will approximate the target function worse. Low bias models have higher variance, hence are unstable in small samples but on average(!) approximate the target function better. We now delve into BVDE with a concrete example.

Figure 5 depicts a two-dimensional data set, based on one input variable *x,* plotted along the horizontal axis and response *y* along the vertical. The black points

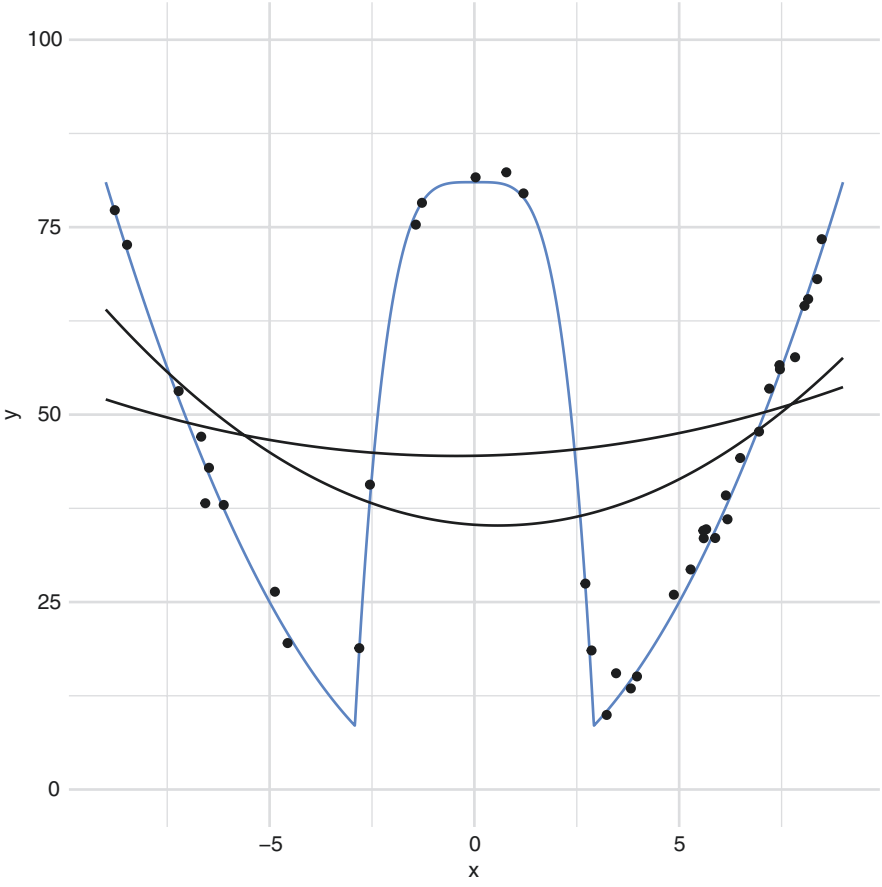


Fig. 5 Illustration of the bias-variance tradeoff. The x-axis shows models' input values and the y-axis is the response. The training data observations are the black points, and the true data generating function is depicted in blue. The two black curves represent two models

represent the observed data points and the blue line *f* depicts the true generating function, which in this example is:

$$f(x) = \begin{cases} x^2 & when\, x < -3 \\ 81 - x^2 & when -3 \le x \le 3 \\ x^2 & when\, x > 3 \end{cases}$$

The black lines represent quadratic models fit to random samples from this data.

Consider the expected generalization error from this procedure when predicting the point at (say) *x = 0*. The error has three components. First, the observed dependent variable can differ from the true value of the generating function, that is measurement **noise**. Visually, noise is the difference between the black points and the blue line**.** The second component, **variance**, is the variability of the model due to the specific sample, which is visually represented by the spread of the predictions from the different models (black lines) at x = 0. In this example, with only two models, this ranges from 35 to 45. Finally, the third component is **bias**, which is the difference between the expectation of the prediction (expectation is taken over the different models built on the different samples) and the true value of the dependent variable (the blue line at x = 0). In this example, the expectation of the prediction from different models appears to be approximately 40, while the true value is 81.

The generalization error expressed as MSE at any x can be written as:

$$\mathbf{E}\left[\left(y - \hat{f}(x)\right)^2\right] = \mathbf{E}\left[\left(y - f(x)\right)^2\right] + \mathbf{E}\left[\left(f - E\hat{f}(x)\right)^2\right] + \mathbf{E}\left[\left(\hat{f}(x) - E\hat{f}(x)\right)^2\right]$$

The three terms correspond to noise, bias and variance of $\hat{f}$.

Figure 6 shows the bias (orange), variance (blue) and mean squared error (MSE) (gray) of models of increasing complexity on a test set. Model complexity is controlled by the degree of x the model is allowed to use and how far the optimizer can optimize the training MSE. Complexity increases from left to right. As the model complexity increases, variance increases while bias decreases. For improved readability, bias, variance and MSE are scaled to the same range in the figure. MSE is thus a weighted sum of bias (squared) and variance. The optimal fit is achieved where MSE is minimal (in the middle of the complexity range). Lower complexity leads to underfitting, which is characterized by lower variance and higher bias, while increased complexity leads to overfitting, which is characterized by higher variance and lower bias (compared to the bias and variance at the optimal complexity).

These concepts are critical in helping analysts create models with maximum predictivity (see chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI").

Essential Concepts of Mathematical Statistics Applicable to ML

Mathematical Statistics is the subfield of Statistics that studies the theoretical foundations of statistics. At the same time, many of the concepts and tools of

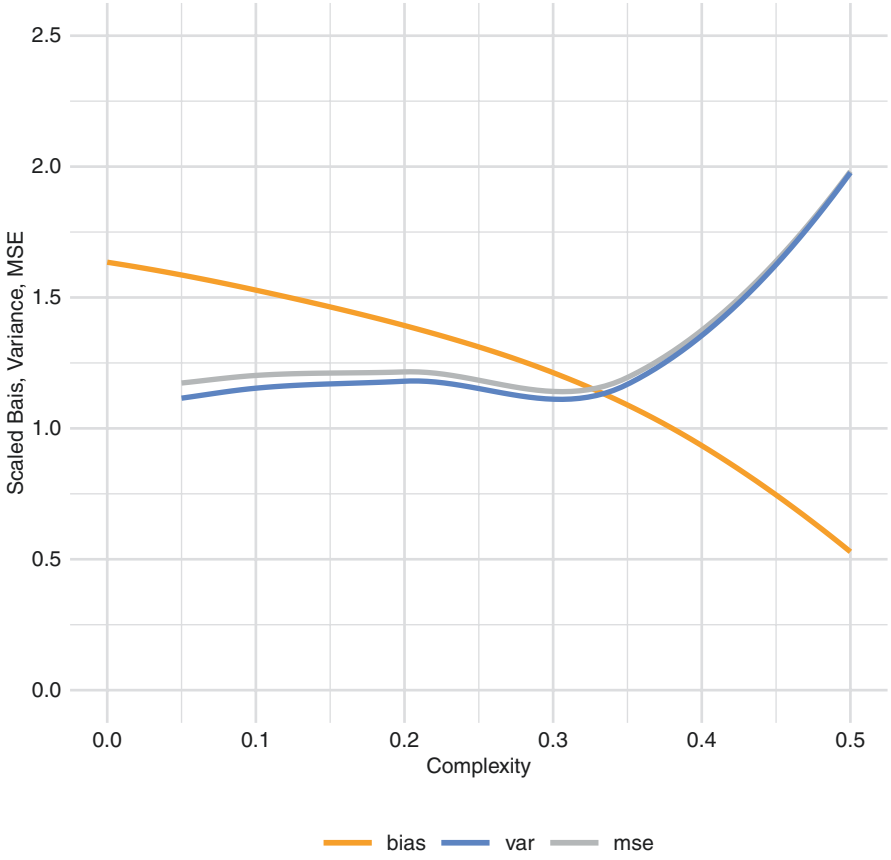


Fig. 6 The relationship between the bias-variance components and complexity in the example (fixed noise is considered). The horizontal axis is complexity (of quadratic models) and the vertical axis is the various bias/variance components scaled to the same range. Orange corresponds to bias, blue to variance and gray is total MSE

mathematical statistics are useful across data science broadly and for ML more specifically. Table 9 provides examples of important areas and analytical tools developed within this field that have value for understanding, advancing, and practicing ML [58].

Techniques and results from mathematical statistics (and its applications) are present throughput the chapters of the present volume.

Table 9 Key concepts and areas of mathematical statistics useful for ML

Area	Subject matter and importance for ML
Special probability distributions and density functions	E.g., Uniform, Bernoulli, Binomial, Hypergeometric, Poisson, Multinomial, etc. distributions; Uniform, Normal, Exponential, Chi-Square, Beta, etc. density functions Uses: • Design and analysis of modeling methods by tailoring them to data characteristics • Instrumental in inference by describing sampling distributions • Also valuable in understanding model errors and other aspects of modeling
Sampling distributions	E.g., Distribution of the mean, Chi-square, t distribution, F distribution Uses: • Measuring the uncertainty of parameters of models, model predictions • Hypothesis testing of whether a sample or model statistic value comes by sampling from a null hypothesis distribution • Estimating the uncertainty of model parameters, structure, and predictions
Estimation and estimators	E.g., Properties of estimators such as unbiasedness, efficiency, consistency, robustness Uses: • Estimators for model construction • Estimators for model error and other properties • Embedding estimators in model selection and construction
Hypothesis testing	E.g., Losses and risk for testing statistical hypotheses. Neyman-Pearson Lemma. Power functions of tests. Control of false positives and false negatives in statistical hypothesis testing decisions Uses: • Deciding whether model properties and function are due to random sampling variation or reliable and generalizable • Deciding on minimum data sample size needed for analysis • Reducing errors when large number of hypotheses are tested when data mining and conducting hypothesis-free discovery

Conclusions

The successful design of problem-solving AI/ML models and systems can be guided by and evaluated according to well specified technical properties. Systems that lack properties are pre-scientific and used heuristically, whereas systems with well-established properties and guarantees provide more solid ground for reliably solving health science and healthcare problems. The nature of the properties listed matches well the practical applications of AI/ML. Properties disconnected from practical implications are not subject of study in the present volume.

AI has both symbolic and non-symbolic methods as well as hybrid variants. Foundational methods in the symbolic category are logics and logic-based systems such as rule based systems, semantic networks, planning systems, NLP parsers and certain AI programming languages. In the non-symbolic category, ML, probabilistic, connectionist, decision-theoretic formalisms, systems and languages dominate.

Among health AI methods capable of reasoning with uncertainty, Bayes Nets and Decision Analysis stand out for their ability to address a variety of uses cases and problem classes.

A major distinction is between shallow systems that essentially are equivalent to function relating outputs to inputs (e.g., most of ML predictive modeling), and systems with rich ontologies and elaborate models of the physical world in the application domain. Non-symbolic AI systems tend to fall in the former category, whereas symbolic ones, in the latter.

The framework of AI search is especially powerful both as a problem-solving technology but also as an analytical tool that helps us understand and architect successful methods and systems. AI search cuts across the symbolic vs. non-symbolic, shallow vs. rich, and the data-driven (Ml) vs. knowledge driven distinctions.

ML has solid and extensive theoretical foundations that include: Computational Learning Theory, ML as AI search, ML as geometrical construction and function optimization, COLT (PAC learning, VC dimension), Theory of feature selection, Theory of causal discovery, Optimal Bayes Classifier, No Free Lunch Theorems, Universal Function Approximation, Generative vs. Discriminative models; Bias-Variance Decomposition of error, and an extensive set of tools borrowed from the field of mathematical statistics.

Key Concepts and Messages Chapter "Foundations and Properties of AI/ML Systems"

- The critical importance of knowing or deriving the properties of AI/ML models and systems.
- The main technical properties of AI/ML systems.
- Tractable vs. intractable problems and computer solutions to them.
- The various forms of Logic-based (symbolic) AI.
- Non-symbolic AI, reasoning under uncertainty and its primary formalisms.
- AI search.
- Foundations of Machine Learning Theory

Pitfalls and Best Practices Chapter "Foundations and Properties of AI/ML Systems"

Pitfall 2.1. From a rigorous science point of view an AI/ML algorithm, program or system with intractable complexity does not constitute a viable solution to the corresponding problem.

Pitfall 2.2. Parallelization cannot make an intractable problem, algorithm or program practical.

Pitfall 2.3. Moore's law improvements to computing power cannot make an intractable problem algorithm or hard program practical.

Pitfall 2.4. Believing that heuristic systems can give "something for nothing" and that have capabilities that surpass those of formal systems. In reality heuristic systems are pre-scientific or in early development stages.

Pitfall 2.5 in Decision Analysis (DA) and Maximum Expected Utility (MEU)-based reasoning

1. Errors in the estimation of probabilities for various events.
2. Errors in eliciting utility estimates in a way that captures patients' true preferences (including using the care providers' utilities rather than the patients').
3. The structure or complexity of the problem setting defies analyst's ability to completely/accurately describe it.
4. Developing a DA for one population and applying in another with different structure of the problem, different probabilities for action-dependent and action-independent events, or with different preferences.

Pitfall 2.6. Using the wrong inductive bias for the ML solution to the problem at hand.

Pitfall 2.7. Ignoring the fit of the data generating procedure with the ML solution to the problem at hand.

Pitfall 2.8. Probably more than other theoretical results, the NFLT for ML has the largest risk to be misunderstood and misapplied.

Pitfall 2.9. If a ML algorithm cannot represent a function class, this outright shows the inability or sub- optimality of this algorithm to solve problems that depend on modeling a data generating function that is not expressible in that algorithm's modeling language.

Pitfall 2.10. UAF theorems should not be over-interpreted. While it is comforting that e.g., algorithm A can represent *any function in the function family* F (i.e., the model space and corresponding inductive bias are expressive enough), learning also requires effective (space and time-tractable, sample efficient, non-overfitting etc.) model search and evaluation in that space.

Best Practices Discussed in Chapter "Foundations and Properties of AI/ML Systems"

Best Practice 2.1. Pursue development of AI/ML algorithm, program or systems that have tractable complexity.

Best Practice 2.2. Do not rely on parallelization to make intractable problems tractable. Pursue tractable algorithms and factor in the tractability analysis any parallelization.

Best Practice 2.3. Do not rely on Moore's law improvements to make an intractable problem algorithm or hard program practical. Pursue tractable algorithms and factor in the tractability analysis any gains from Moore's law.

Best Practice 2.4. When faced with intractable problems, consider using strategies for mitigating the computational intractability by trading off with less important characteristics of the desired solution.

Best Practice 2.5. As much as possible, use models and systems with formal and established properties (theoretical + empirical). Work within the maturation process starting from systems with unknown behaviors and no guarantees, to systems with guaranteed properties.

Best Practice 2.6. Decision Analysis (DA) and Maximum Expected Utility (MEU)-based reasoning

1. Ensure that the structure of the problem setting is sufficiently/accurately described by the DA tree. Omit known or obvious irrelevant factors.
2. Elicit utility estimates in a way that captures patients' true preferences using established utility-elicitation methods.
3. Accurately estimate probabilities of action-dependent events and action-independent events.
4. In most conditions, and whenever applicable, data-driven approaches should be preferred to subjective probability estimates. Use probability-consistent statistical or ML algorithms to estimate the probabilities.
5. Ensure that the decision analysis is applied to the correct population.
6. Conduct sensitivity analyses that reveal how much the estimated optimal decision is influenced by uncertainty in the specification of the model.
7. Whenever possible, produce credible intervals/posterior probability distributions for the utility expectations of decisions.

Best Practice 2.7. Pursue ML solutions with the right inductive bias for the problem at hand.

Best Practice 2.8. Create a data generating or design procedure that matches well the requirements of the problem at hand and works with the inductive bias to achieve strong results.

Best Practice 2.9. Cross validation and independent data validation, as well as their cousin reproducibility, are robust pillars of good science and good ML practice and are not in reality challenged by the NFLT.

Best Practice 2.10. Clustering should not be used for predictive modeling.

Best Practice 2.11. A very useful form of clustering is post-hoc extraction of subtypes from accurate predictor models.

Best Practice 2.12. The choice of generative vs. discriminative modeling affects quality of modeling results and has to be carefully tied to the problem domain characteristics. All else being equal discriminative models confer efficiency (computational and sample) advantages.

Discussion Topics and Assignments, Chapter "Foundations and Properties of AI/ML Systems"

1. Revisit questions (10–13) of chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: the Need for Best Practices Enabling Trust in AI and ML**"** from the perspective of which properties of the proposed systems are known.
2. Use Table 3 to classify the proposed systems in questions (10–13) of chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: the Need for Best Practices Enabling Trust in AI and ML"
3. Which of the following are heuristic systems (and in what category of the classification of Table 3 in this chapter:
   (a) INTERNIST-I
   (b) MYCIN
   (c) QMR-BN
   (d) A classical regression model for which we do not know if data is normally distributed. Compare to a classical regression model for which we know that data is not normally distributed.
   (e) A Large Language Model implementing an EHR "ChatBot" tool answering queries about the patients' medical history.
   (f) IBM Watson Health

4. Based on your findings in question 3, how would you go about next steps toward putting these systems into practice from a perspective of accuracy and safety?
5. Discuss: are BNs deep or shallow representations?
6. Consider a population with age distribution as in the table below:

Age →	0–10	11–20	21–30	31–40	41–50	51–60	61–70	>70
% →	20	20	10	10	10	15	10	5

   (a) What would be a good 2-way clustering (grouping) of individuals in this population?
   (b) For a pediatrician: what would be a good 2-way clustering (grouping) of individuals in this population?
   (c) For a gerontologist: what would be a good 2-way clustering (grouping) of individuals in this population?
   (d) For an obstetrician: what would be a good 2-way clustering (grouping) of individuals in this population?
   (e) What can you conclude about the value of a priori clustering without any reference to use of the produced groups?

7. Occam's Razor is the epistemological principle that says that given two explanations that fit the data equally well, we should choose the simplest one. Analyze this proposition from a BVDE viewpoint.

References

1. Russell SJ. Artificial intelligence a modern approach. Pearson Education, Inc; 2010.
2. Cook S. The complexity of theorem proving procedures. In: Proceedings of the Third Annual ACM Symposium on Theory of Computing; 1971. p. 151–8.
3. Karp RM. Reducibility among combinatorial problems. In: Miller RE, Thatcher JW, editors. Complexity of computer computations. New York: Plenum; 1972. p. 85–103.
4. Miller RA. A history of the INTERNIST-1 and quick medical reference (QMR) computer-assisted diagnosis projects, with lessons learned. Yearb Med Inform. 2010;19(01):121–36.
5. Aliferis CF, Miller RA. On the heuristic nature of medical decision-support systems. Methods Inf Med. 1995;34(1–2):5–14.
6. Rich, E. and Knight, K., Artificial Intelligence. 1991. Ed.
7. Nilsson NJ. Principles of artificial intelligence. Springer Science & Business Media; 1982.
8. Buchanan BG, Shortliffe EH. Rule based expert systems: the mycin experiments of the Stanford heuristic programming project (the Addison-Wesley series in artificial intelligence). Addison-Wesley Longman Publishing Co., Inc.; 1984.
9. Buchanan BG, Feigenbaum EA. DENDRAL and meta-DENDRAL: their applications dimension. Artif Intell. 1978;11(1–2):5–24.
10. Clocksin WF, Mellish CS. Programming in PROLOG. Springer Science & Business Media; 2003.
11. Hripcsak G, Clayton P, Pryor T, Haug P, Wigertz O, Van der Lei J. The Arden syntax for medical logic modules. In proceedings. Symposium on computer applications in medical care; 1990. p. 200–4.
12. Machado CM, Rebholz-Schuhmann D, Freitas AT, Couto FM. The semantic web in translational medicine: current applications and future directions. Brief Bioinform. 2015;16(1):89–103.
13. McCray A. The UMLS semantic network. In: Proceedings. Symposium on computer applications in medical care; 1989, November. p. 503–7.

14. Shmulevich I, Dougherty ER. Probabilistic Boolean networks: the modeling and control of gene regulatory networks. Society for Industrial and Applied Mathematics; 2010.
15. Noy NF, Shah NH, Whetzel PL, Dai B, Dorf M, Griffith N, Jonquet C, Rubin DL, Storey MA, Chute CG, Musen MA. BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Res. 2009;37(suppl_2):W170–3.
16. Torres A, Nieto JJ. Fuzzy logic in medicine and bioinformatics. J Biomed Biotechnol. 2006;2006:1–7.
17. McDermott D, Doyle J. Non-monotonic logic I. Artif Intell. 1980;13(1–2):41–72.
18. Haddawy P. A logic of time, chance, and action for representing plans. Artif Intell. 1996;80(2):243–308.
19. Langlotz CP, Fagan LM, Tu SW, Sikic BI, Shortliffe EH. A therapy planning architecture that combines decision theory and artificial intelligence techniques. Comput Biomed Res. 1987;20(3):279–303.
20. Pauker SG, Kassirer JP. Decision analysis. In: Medical uses of statistics. CRC Press; 2019. p. 159–79.
21. Sox HC, Blatt MA, Marton KI, Higgins MC. Medical decision making. ACP Press; 2007.
22. Pearl J. Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan kaufmann; 1988.
23. Neapolitan RE. Probabilistic reasoning in expert systems: theory and algorithms. John Wiley & Sons, Inc; 1990.
24. Tsamardinos I, Aliferis CF. Towards principled feature selection: relevancy, filters and wrappers. In: International workshop on artificial intelligence and statistics. PMLR; 2003. p. 300–7.
25. Cooper GF. The computational complexity of probabilistic inference using Bayesian belief networks. Artif Intell. 1990;42(2–3):393–405.
26. Boyd S, Boyd SP, Vandenberghe L. Convex optimization. Cambridge university press; 2004.
27. Rutenbar RA. Simulated annealing algorithms: an overview. IEEE Circuit Devices Magazine. 1989;5(1):19–26.
28. Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl. 2021;80:8091–126.
29. Mitchell TM. Machine learning, vol. 1, No. 9. New York: McGraw-hill; 1997.
30. Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Comput Intell Mag. 2006;1(4):28–39.
31. Audi R. Epistemology: a contemporary introduction to the theory of knowledge. Routledge; 2010.
32. Duda RO, Hart PE, Stork DG. Pattern classification. New York: John Wiley & Sons. Inc.; 2000. p. 5.
33. Weiss SM, Kulikowski CA. Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems. Morgan Kaufmann Publishers Inc.; 1991.
34. Anthony M, Biggs NL. Computational learning theory: an introduction. Cambridge University Press; 1992.
35. Kearns MJ, Vazirani U. An introduction to computational learning theory. MIT press; 1994.
36. Langford J. Tutorial on practical prediction theory for classification. J Mach Learn Res. 2005;6(Mar):273–306.
37. Schölkopf C, Burges JC, Smola AJ. Advances in kernel methods—support vector learning. Cambridge, MA: MIT Press; 1999.
38. Herbrich R. Learning kernel classifiers: theory and algorithms, (2002). Cambridge: MA, USA MIT Press; 2002.
39. Kohavi R, John GH. Wrappers for feature subset selection. Artif Intell. 1997;97(1-2):273–324. Prediction (Vol. 2, pp. 1-758). New York: springer.
40. Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res. 2003;3(Mar):1157–82.
41. Guyon I, Aliferis C. Causal feature selection. In: Computational methods of feature selection. Chapman and Hall/CRC; 2007. p. 79–102.

42. Wolpert DH. What the no free lunch theorems really mean; how to improve search algorithms, vol. 7. Santa Fe Institute; 2012. p. 1–13.
43. Hardin D, Tsamardinos I, Aliferis CF. A theoretical characterization of linear SVM-based feature selection. In: Proceedings of the twenty-first international conference on machine learning; 2004. p. 48.
44. Statnikov A, Hardin D, Aliferis C. Using SVM weight-based methods to identify causally relevant and non-causally relevant variables. Signs. 2006;1(4):474–84.
45. Zou H. The adaptive lasso and its oracle properties. J Am Stat Assoc. 2006;101(476):1418–29.
46. Statnikov A, Lemeir J, Aliferis CF. Algorithms for discovery of multiple Markov boundaries. J Mach Learn Res. 2013;14(1):499–566.
47. Statnikov A, Ma S, Henaff M, Lytkin N, Efstathiadis E, Peskin ER, Aliferis CF. Ultra-scalable and efficient methods for hybrid observational and experimental local causal pathway discovery. J Mach Learn Res. 2015;16(1):3219–67.
48. Statnikov A, Aliferis CF. Analysis and computational dissection of molecular signature multiplicity. PLoS Comput Biol. 2010;6(5):e1000790.
49. Pearl J. Causality. Cambridge university press; 2009.
50. Spirtes P, Glymour CN, Scheines R, Heckerman D. Causation, prediction, and search. MIT press; 2000.
51. Eberhardt F, Glymour C, Scheines R. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among N variables. In: Bacchus F, Jaakkola T, editors. Proceedings of the 21st conference on uncertainty in artificial intelligence (UAI); 2005. p. 178–84.
52. Eberhardt F, Glymour C, Scheines R. N-1 experiments suffice to determine the causal relations among N variables. In: Holmes D, Jain L, editors. Innovations in machine learning, theory and applications series: studies in fuzziness and soft computing, vol. 194. Springer-Verlag; 2006. See also Technical Report CMU-PHIL-161 (2005).
53. Mayo-Wilson C. The problem of piecemeal induction. Philos Sci. 2011;78(5):864–74.
54. Mayo-Wilson C. The Limits of Piecemeal Causal Inference. Br J Philos Sci. 2014;65(2):213–49.
55. Dupuy A, Simon RM. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. J Natl Cancer Inst. 2007;99(2):147–57.
56. Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction, vol. 2. New York: springer; 2009. p. 1–758.
57. Von Neumann J. and Morgenstern O. 2007. Theory of games and economic behavior. In: Theory of games and economic behavior. Princeton university press. Original in Wald, A., 1947. Theory of games and economic behavior.
58. Wackerly D. Mendenhall W. and Scheaffer, RL. 2014. Mathematical statistics with applications. Cengage Learning.

An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science

Gyorgy Simon and Constantin Aliferis

Abstract

This chapter provides an outline of most major biomedical ML methods in a manner suitable for both readers who have not delved into ML before, and readers who may already know about some or all of these methods. The former will find here a useful introduction and review. The latter will find additional insights as we critically revisit the key concepts and add summary guidance on whether and when each technique is applicable (or not) in healthcare and health science problem solving. Toward that end, for each technique, we introduce a "Method Label", akin to a drug label, which provides distilled information about the techniques at a glance. The method labels present the primary and secondary uses of each technique, provide context of use, describe the principles of operation, and summarize important theoretical and empirical properties.

Keywords

Method label · Predictive modeling · Feature selection · Exploratory analysis

G. Simon · C. Aliferis (✉)
Institute for Health Informatics, University of Minnesota, Minneapolis, MN, USA
e-mail: constantinaibestpractices@gmail.com

G. J. Simon, C. Aliferis (eds.), *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, Health Informatics,
https://doi.org/10.1007/978-3-031-39355-6_3

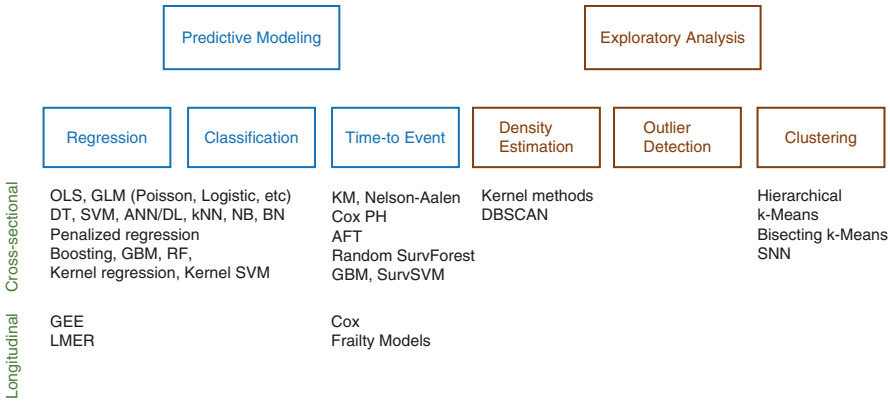


Fig. 1 Lay of the land of biomedical ML. Tabular categorization of major machine learning methods based on modeling task (columns) and predictor type (rows). See the text for abbreviations and details

Introduction

A vast number of machine learning techniques has been proposed for solving a rich set of problems. As we discussed in the Introduction, many of the clinical problems fall into a few categories, some of which are more heavily researched than others. We call these categories **analytic tasks** and, in this chapter, we consider six tasks, which fall into two broader categories: predictive modeling and exploratory analysis.

Figure 1 depicts the lay of the land for predictive and exploratory analysis tabulating the most common techniques. We will address causal modeling separately in chapter "Foundations of Causal ML".

In **predictive modeling**, the goal is to assign values to one or more variables, called outcomes (aka response, or dependent variables), using the known values of other variables (aka predictor variables, independent variables, or features). Somewhat abusing ordinary language, "predictive" in the context of "ML predictive modeling", does not necessarily imply forecasting future events. Any pattern recognition falls under the category including future forecasting, prognosis, diagnosis and recognizing past events (e.g., retroactive diagnosis). Also, "predictor" and "predictor variable" are often used interchangeably although from context it may be clear whether a predictor refers to a variable (feature) or a full model.

In contrast, **exploratory analysis** aims to model the relationships among many variables, none of which is designated as an outcome or predictor variable. For example, predicting patients' risk of mortality (outcome) based on current diagnoses and laboratory results (predictor variables) is a predictive modeling task because variables corresponding to mortality are designated as outcome, variables corresponding to diagnoses and laboratory results are designated as predictors, and we predict the future unknown value of an outcome using known values of the predictor variables. Conversely, understanding a patient population in terms of common

comorbidities that co-occur with diabetes is an exploratory analysis task, because there is no particular outcome to predict.

Predictive Modeling Tasks

Within predictive modeling, we distinguish between several tasks based on the outcome type. In the rest of this chapter, we focus on three of them: classification, regression and time-to-event modeling. These are the outcome types and corresponding tasks most frequently encountered in biomedical ML.

Continuous outcomes. Continuous outcomes are measured on a continuous scale. Continuous variables can be **ratio**s (variables that do not have a well-defined 0 point) or **interval**s (which have a well-defined 0 point). For example, lengths are intervals and a length of zero indicates that the object does not have length. Whether we measure length in inches, centimeters, or miles, 0 length is the same. Conversely, temperature is a ratio, because 0 °F or 0 °C does not mean that the object has no temperature. Furthermore, 0 temperature depends on the scale we use: 0°F and 0°C do not designate the same temperature.

Another relevant distinction from a modeling perspective is the distribution of the continuous variable. Commonly used distributions include Gaussian, Poisson, exponential, negative binomial, etc. Prediction problems with a continuous outcome are referred to as **regression** problems.

Categorical variables take a value from a set of finite distinct values. For example, color (red, amber, green), grade (A, B, C, F), or risk category (low, medium, high) are categorical variables. **Binary** (also known as **binomial**) variables are categorical variables that have exactly two levels (they can take one of two values); while **multinomial** variables have more than two levels.

Categorical variables with multiple levels can be further classified as nominal or ordinal variables. In case of **ordinal** variables, the levels are ordered (e.g. good, better, best), while for **nominal** variables, the levels are not ordered (e.g. colors). Prediction problems with categorical outcomes are referred to as **classification** problems. If the outcome is binary, we have **binary classification**; if the outcome is multinomial, we have **multi-class classification** (aka n-ary or polychotomous classification).

Time-to-event outcomes. The measurement of interest is the time between a particular time point (known as **index date or index time**) and an event of interest. The quintessential example is survival, where the measure of interest is the time between the start of the study (index date) and death (the event of interest). The predictive modeling task that predicts time-to-event outcomes is referred to as *time-to-event modeling* or **survival analysis** (when the outcome is survival).

Sequence outcomes. Sequences are ordered sets of observations and when the outcome of interest is a sequence, we have a **sequence prediction** problem. Examples include genomic sequence (ordered set of nucleotides) prediction, *text synthesis or translation* (predicting an ordered set of words), or **trajectory mining** (predicting future sequences of e.g., disease states).

Structured outcomes. The outcome of the predictive model can also be a complex structure such as a graph or the actual structure of an entity (e.g. protein structure prediction). For techniques to discover *causal* structure, see the chapter "Foundations of Causal ML".

Exploratory Analysis Tasks

Density estimation. The goal of density estimation (also encompassing discrete probability functions) is to infer the (often multi-dimensional) probability distribution underlying observed data. The simplest form of density estimation is unidimensional scaled histograms. For example, one might be interested in describing the probability distribution of blood glucose in a population. Density estimation can also be performed on multi-dimensional data and techniques exist for both low and high-dimensional data. Density estimation has several natural uses, including discovery of multiple modes of data, clustering and outlier detection.

Clustering. Clustering creates a grouping of the observations in a data set such that observations that belong to the same group (cluster) are more similar to each other than to observations that belong to a different group. Clustering can be used, for example, for subpopulation discovery, where well-separated groups of observations can represent subpopulations at different states of health or groups of patients with different disease etiology.

Clustering can be achieved based on many principles, one of which is based on data density. In that case, clusters are high-density regions in the data, separated from each other by low-density regions.

Outlier detection. Outliers are observations that are dissimilar to most other observations. Outliers may either fall into low-density regions, or they may behave very differently from model-based expectations (model-based outlier). For example, in a hospitalized population, outliers can be patients who have an unusually long hospital length of stay (LoS), say, above 9 days; or alternatively, they may have a LoS of less than 9 days, but unusually long for the disease that they got admitted for. The first example is clearly patients who fall into a low-density region (very few patients in a patient population stay hospitalized for more than 9 days), while the latter patients are in a low-density region among patients who got admitted with the same disease.

Temporal Characteristics of the Data

A further categorization of methods is based on the temporal characteristics of the data. It is very common for healthcare data to be temporal, thus several AI/ML as well as classical statistical techniques have been developed specifically to take advantage of various temporal characteristics.

Cross-sectional data. This data captures the state of a sample from a population at a specific point in time. The state information often contains temporal information about the past implicitly in the definitions of variables - but not explicitly modeled. The implicit temporal information is typically abstracted to different time scales and granularities. The vast majority of the machine learning techniques expect cross-sectional data.

Cross-sectional data can also be used for predictive modeling in which the outcome occurs at a future time relative to the the index date of the predictor variables. For example, when modeling the 7-year risk of diabetes, the outcome, diabetes, must occur or not within 7 years, but the predictor variables they have been evaluated at a particular point in time (the index date) and changes to them over the 7 years are not of interest.

Longitudinal data. Measurements for a patient population is taken repeatedly over time. Measurements are not necessarily taken at the same time for everyone and not all measurements are taken each time. Routinely collected clinical test data, falls into this category. At most encounters with the health system, some aspect of a patient's health is measured and recorded. Most patients have more than one encounter and at each encounter, different measurements (e.g., lab tests) can be taken.

Time-series data. Similarly to longitudinal data, in time series data, several measurements are taken over time, but unlike longitudinal data, time-series data focuses on a single sampling unit. If we aim to model the glucose trajectory of a single individual over a long period of time, then we are solving a time-series modeling problem; if we aim to model the glucose control of a population of patients over time, then we have a longitudinal data modeling problem.

Figure 1 tabulates some of the techniques from this chapter. The columns correspond to the various analytic tasks, while the rows correspond to the temporal characteristics of the methods. Naturally, several methods can be used for multiple tasks (with appropriate modifications) and with data sets having multiple temporal characteristics. We either put the methods into the categories where they are most prominent (e.g. SVM into classification), or into shared categories (e.g. many techniques that are used for classification can also be used for regression).

Method Labels

In the following sections, we are going to describe the major machine learning methods in terms of their primary use, additional secondary uses, key operating principles, operating characteristics and properties, and provide a context for their use that helps assess their appropriateness for different modeling tasks. We also mention when and why the use of a method is not recommended.

For each method (or family of methods), we are going to present highly digested and operationally-oriented information in what we call a ***Method Label***. A Method Label is similar to a drug label, presenting the most vital information about a method at-a-glance.

Format of method labels	
Main Use	This entry describes the main purpose of the model. What kind of tasks can it solve? Within that task, are there specific problems that this method is best suited for? For example, linear regression solves predictive analytic problems with continuous outcomes.
Context of use	In practice, when is this method used? This can be a subset of the intended use or a superset of the intended use. For example, ordinary least square regression is designed for Gaussian outcomes, but is often used for a wide range of continuous outcomes.
Secondary use	This entry describes potential situations where the method can also be used. This may not be the primary intended use of the methods; or this may not be the model that is most appropriate for the use case. For example, SVM can be used for regression, although its primary use is classification.
Pitfalls	**Pitfall 3.1.4.1.** This entry lists negative consequences of using this method under certain conditions. For example, SVM when used for causal problems, leads to wrong causal effect estimates.
Principle of operation	A short description of how the method works. For example, linear regression is approximating a regression using maximum likelihood estimates of the regression parameters.
Theoretical properties and Empirical evidence	This entry describes any known theoretical properties the method may have and the assumptions linked to them, as well as empirical evidence for the method performance.
Best practices	**Best practice 3.1.4.1.**This entry provides prescriptive practice recommendations about when and how to use or not the method.
References	Key literature related to the above.

Readers that cannot delve into technical details can still benefit greatly by the information provided in the Method Labels.

Chapter Layout

We begin (in section "Foundational Methods") with describing the foundational methods for predictive modeling of cross-sectional data. Section "Ensemble Methods" is devoted to ensemble methods which use foundational techniques from "Foundational Methods" to addresses issues related to model stability and performance; and section "Regularization" is devoted to regularization, which addresses high dimensionality, or more broadly, constrains the model complexity. The subsequent three sections address feature selection and dimensionality reduction, time-to-event outcomes, and longitudinal data, respectively. We close the chapter with a brief mention of a few more methods that the reader should be aware of. As the reader will observe we weave classical and modern statistical methods with mainstream ML methods since this reflects modern ML practice and there are significant mathematical, conceptual and computational commonalities between the fields.

Foundational Methods

Foundational methods in the chapter will refer to first-order methods that:
(a) Are of high theoretical and/or practical value on their own, or
(b) Are of high theoretical and/or practical value in conjunction with other higher-order methods.

Ordinary Least Square (OLS) Regression

OLS regression was invented by Sir Francis Galton in 1875 as he described the relationship of the weight of sweet pea seeds and the weight of the seeds from their mother plants. This experiment also gave rise to the correlation coefficient: Karl Pearson, Galton's biographer, developed the mathematical formulation for the Pearson correlation coefficient in 1896 [1].

Given a matrix X of predictor variables (independent variables) and outcome (dependent) variable y, the ordinary linear regression model is

$$y \sim Normal\left(X\beta, \sigma^2\right)$$

where β is a vector of **coefficients**. The outcome is assumed to be normally distributed with mean $X\beta$ and variance σ^2. In other words, the outcome has a deterministic component $X_i\beta$ for the i^{th} observation, and a random component, which is Gaussian noise with mean 0 and variance σ^2. The objective is to find the coefficient vector β, which makes the observations y the most likely, that is to maximize the Gaussian log likelihood

$$\ell\left(\beta\right) = \text{const} - \sum_i \frac{\left(y_i - X_i\beta\right)^2}{2\sigma^2},$$

where i iterates over the observations. The coefficient vector β that maximizes the log likelihood is the same vector that minimizes the least square error $\sum_i (y_i - X_i\beta)^2$, hence the name Ordinary Least Square regression.

As a least square estimator, OLS is "BLUE" (best linear unbiased estimator) [2]. The coefficient estimates are normally distributed, allowing for a Wald-type test for their significance. The least square problem is convex, thus when a solution exists, it is the global solution.

Assumptions. The assumptions follow from the model: (1) for all observations, the noise component has constant variance (σ^2). Having uniform variance across the observations is referred to as homoscedasticity. (2) The errors of the observations are independent. (3) Observations are identically distributed. (4) The mean of the observations is a linear combination of the predictor variables. The effect of the predictors on the outcome is thus linear and additive.

Expressive capability. OLS, in its native form, is only able to express linear and additive effects. By explicitly including transformations of the original variables, the linear effect assumption can be relaxed. Explicitly including interactions terms of the predictor variables can relax the additivity assumption. OLS has no ability to

automatically discover interactions or nonlinearities, so these need be hand-crafted by the data scientist.

Dependent and outcome variable types. Ordinary linear regression assumes that the observations (rows of X) are independently sampled and thus it is more appropriate for dependent variables that can be represented as regular tabular data and a continuous outcome variable that follows a Gaussian distribution.

Sample size requirement. Ordinary linear regression is not appropriate for high-dimensional data, where the number of predictor variables is similar to the number of observations or exceeds them. As a rule of thumb, 10 observations per predictor variable is recommended. This is one of the least sample-intensive techniques, thus when the number of observations is low and the number of predictor variables is low, ordinary linear regression may be the most appropriate modeling technique.

Main use and its context. OLS is intended to solve regression problems with Gaussian outcomes. Guarantees about the solution hold true only for this use case.

Intepretability. For a covariate (predictor variables) X_i with coefficient β_i, every unit increase in X_i is associated with an increase of β_i in the outcome, if all other predictor variables are fixed.

As a result, OLS is highly interpretable, and also fit for use with causal estimation once the causal structure is known (see chapter "Foundations of Causal ML").

We recommend using OLS as a "default" algorithm in low dimensional data unless a generalized linear regression model with a different linkage is more appropriate (see GLM). Building an OLS model, even if its performance is expected to be inferior to more advanced regression techniques, is recommended, because the cost of building an OLS model is minimal, the model is highly interpretable, and it can reveal data problems, biases, design problems and potentially other issues. As we will see the potentially higher predictive performance from other methods needs to be evaluated from the perspective of trading interpretability for performance, and in some applications, higher interpretability can balance out some performance deficit.

Optimality. The coefficients found by maximizing the likelihood are unbiased. Also the log likelihood function is convex, thus the global maximum is easy to find.

Method label: ordinary least squares regression	
Main Use	• Regression problems • Continuous, preferably Gaussian outcomes • Cross-sectional data
Context of use	• First choice, most common regression method in low dimensional data • When highly interpretable model is required
Secondary use	• May achieve acceptable performance for some non-Gaussian continuous outcomes
Pitfalls	**Pitfall 3.2.1.1.** In high-dimensional data, coefficients may be biased or cannot be estimated **Pitfall 3.2.1.2.** OLS is negatively affected by high collinearity
Principle of operation	• Least square estimation (or equivalently, maximizing the Gaussian log likelihood)

Method label: ordinary least squares regression	
Theoretical properties and empirical evidence	• OLS is a consistent, efficient estimator of the coefficients • The coefficients are asymptotically normally distributed • Minimizing the least squares is a convex optimization problem. If a solution exists, numerical solvers will find the global solution. • Highly interpretable and causally consistent models • Vast literature in health sciences documenting successful applications • Because it is a simple model, risk of overfitting in low sample is lower than complex models • By the same token it can fail to capture highly non-linear data generating functions
Best practices	**Best practice 3.2.1.1.** Unless a generalized linear model is more appropriate, OLS is a good default technique. **Best practice 3.2.1.2.** Building an OLS, even if it is known not to produce optimal predictive performance, can reveal data problems, biases, etc.
References	• Numerous good textbooks describe OLS regression. Below is one example. Tabachnick & Fidell. Using Multivariate Statistics. Pearson, 2019

Generalized Linear Models (GLM)

Generalized linear models (GLM) were first introduced by Nelder and Wedderburn in 1972. GLM was born out of the desire to model a broader range of outcome types than Gaussian outcomes and was enabled by advancements in statistical computing [3]. The defining characteristic of GLMs is that the data generating functions is not linear and a **link function** linearizes the relationship between an outcome and the predictor variables, where the outcome is distributed by an exponential family distribution. A fully specified GLM has the following components:

1. The distribution of y
2. A linear predictor $\eta = X\beta$
3. The link function $g(\mu)$ that links the expectation of $E(y) = \mu$ to the linear predictor: $\eta = g(\mu)$; or equivalently, $\mu = g^{-1}(\eta)$.

As an example, let us consider logistic regression, linear regression for outcomes with binomial distribution. The distribution of y_i is Bernoulli with parameter η_i, and the link function is

$$\text{logit}\left(y_i\right) = \log\frac{\Pr\left(y_i\right)}{1-\Pr\left(y_i\right)} = X_i\beta.$$

The objective is to find the coefficient vector β that maximizes the likelihood of observing y, which in case of the logistic regression is the binomial likelihood.

GLMs are frequently used for modeling outcomes that follow other exponential family distributions, including multinomial, Poisson, and negative binomial [4].

Expressive capability. The link function does not change the model's expressive capability; the relationship between η and the predictor variables is linear, the only difference from OLS is that the linear predictor is transformed through the link function so that GLM can model specific families of non-linear functions.

Dependent and outcome variable types. GLM are best suited for cross-sectional data (with independent and identically distributed observations), but the outcome types have to be distributed in accordance with the link function.

Prediction task. GLM can solve classification problems (logistic and multinomial link) and regression problems where the continuous outcome can be distributed following any of the exponential family distributions.

Theoretical properties. The GLM is a maximum likelihood estimator for the exponential family distributions. Some instances of GLM, such as an overdispersed GLM, does not correspond to an actual exponential family distribution. In such cases, a variance function can be specified and GLM becomes a quasi-likelihood estimator [5]. Both estimators (maximum likelihood and quasi-likelihood) are consistent and efficient. They yield coefficient estimates that are normally distributed and thus the Wald test can be used for testing their significance. Both the likelihood and quasi-likelihood are convex, thus when a solution exists, it is a global solution and solvers can typically find it efficiently.

Method label: generalized linear models	
Main Use	• Predictive modeling with outcomes that follow a distribution from the exponential family (i.e., relationship of outcome and predictor variables can be non-linear) • Most common applications are classification (logistic regression), estimating count outcomes (Poisson regression) and exponential outcomes • Cross-sectional data
Context of use	• First-pass/comparator classifier in low dimensional problems with limited need for input interaction modeling • Highly interpretable model
Secondary use	• Applicable also to deviations from the exponential family, most typically where the sample variance is higher than theoretically expected under the corresponding exponential family distribution (over-dispersion) • Logistic regression may offer acceptable performance in classification problems, where the linear additive assumption is mildly violated
Pitfalls	**Pitfall 3.2.2.1.** In high-dimensional data, coefficients may not be estimatable **Pitfall 3.2.2.2.** Tendency to overfit in the presence of high collinearity
Principle of operation	• When the outcome follows an exponential family distribution, GLM is a maximum likelihood estimator • When the outcome does not follow an exponential family distribution, GLM can be a quasi-likelihood estimator

Method label: generalized linear models	
Theoretical properties and empirical evidence	• GLM provides consistent, efficient estimates of coefficients • The coefficients are asymptotically normally distributed • Minimizing the likelihood or quasi-likelihood is a convex optimization problem. If a solution exists, numerical solvers will find the global solution efficiently. • Highly interpretable models • GLM can have high performance even when the assumptions are violated. Chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs" discusses comparisons between logistic regression and more modern ML techniques
Best practices	**Best practice 3.2.2.1.** Use GLM as first pass, or main comparator classifier **Best practice 3.2.2.2.** Building a GLM, even if it is known not to produce optimal predictive performance, can reveal data problems, biases, etc.
References	• Walter Stroup. Generalized Mixed Linear Models, CRC Press, 2003 • P. McCullagh, JA Nelder. Generalized Linear Models, CRC Press, 1989

Ordinal Regression Models

There are two main strategies for modeling ordinal outcomes using GLMs. The first one is cumulative logits and the second one is proportional odds [6].

Consider an ordinal outcome variable with J levels, $1 < 2 < \ldots < J$. Under the **cumulative logits** strategy, J-1 logistic regression models are fit. The j^{th} model is a classifier distinguishing $y \leq j$ verus $y > j$. Under the **proportional odds** strategy, again, *J-1* models are built, but these models share all coefficients except for the intercept. The j^{th} model is

$$\text{logit}\left(y \leq j\right) = \alpha_j + X\beta,$$

where α_j is the level-specific intercept and β are the slopes shared across the *J-1* models.

Notice, that the cumulative logits model can use any binary component classifier; while the proportional odds GLM is a special case of multi-task learning.

Key reference: Agresti A. Categorical Data Anlaysis, second edition. Chapter 7.2. Wiley Interscience, 2002.

Artificial Neural Networks (ANNs)

For main milestones in the development of ANNs see chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs". In this section, we focus on the general form of ANNs for cross-sectional data. Image and language model applications are discussed in Chapters "Considerations for Specialized Health AI

and ML Modelling and Applications: NLP" and "Considerations for Specialized Health AI and ML Modelling and Applications: Imaging—Through the perspective of Dermatology".

Artificial neural networks (or neural networks, NN, for short) can be thought of as regression models stacked on top of each other, and each regression model is thus called a **layer**. Each of these layers are *multiple regression models*, meaning they have potentially multivariate inputs as well as multivariate outputs. The outputs from each layer, are transformed using nonlinear functions, called the **activation functions**, and then passed to the subsequent layer as their input. The final layer (aka the output layer) provides the network's output(s).

Figure 2 shows an example NN with two hidden (aka encoding) and an output layer. The output from this network is

$$\hat{y} = f_3\left(b_3 + W_3 f_2\left(b_2 + W_2 f_1\left(b_1 + W_1 X\right)\right)\right),$$

where the $f_i(\cdot)$ is the activation function, b_i are the biases (or bias vectors) and W_i are the weights (weight matrices) of connections coming into layer i. In layer i, the input is multiplied by W_i, the biases b_i are added and the result is passed through the activation function $f_i()$, producing the output from layer i. The input to the first layer is X, and the output from the topmost layer, the third layer in this example, is the outcome $\hat{y}$. Common activation functions include the sigmoid function (logit function), ReLU (rectified linear unit), and softmax.

Expressive power. NNs are universal function approximators, they can express any relationship between the predictors and the output without distributional restrictions. This includes non-linear relationships as well as interactions.

Predictor and outcome types. The basic form of NNs introduced here is most appropriate for cross-sectional data without any special structure. However, when the data has special structure, corresponding network architectures have been proposed for many of them. For example, Convolutional Neural Networks (CNN) [7] have been proposed for image data, Recurrent Neural Networks (RNN) [8]and Long Short-Term Memory (LSTM) [9] for sequence data, Transformers [10] for language models, Graph Neural Networks [11] for graph data, etc. We discuss some of these architectures in chapters "Considerations for Specialized Health AI and ML Modelling and

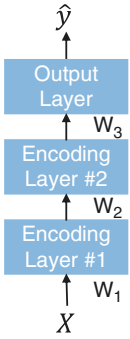


Fig. 2 An example NN with three layers

Applications: NLP" and "Considerations for Specialized Health AI and ML Modelling and Applications: Imaging—Through the perspective of Dermatology".

Sample size. The sample size required for NNs can be very large. The deeper the network (the more layers it has), the more parameters it has and the larger the required sample size. Many practical applications of deep learning use millions of parameters. Although the traditional statistical rule of the thumb, that the number of required observations is approximately 10 times the number of parameters, does not hold for deep learning—they can operate on data with fewer samples—the required sample size is still very large.

The largest GPT-3 language model, which has 175 billion parameters, was trained on 45 TB of text data taking 355 GPU-years [12]. Certain structures (most notably convolution) and regularization can alleviate the sample size requirement to some degree.

Interpretability. The key to NNs is to automatically transform the original data space into a new representation that is more amenable to the predictive modeling task at hand. A side effect of this automatic transformation is that the meaning of the original space is lost and the meaning of the resulting variables are often unknown. Thus, NNs are considered black-box (uninterpretable). One way to interpret them is by "local approximation" of the NN using an interpretable model, such as a multivariate regression model, fitted over input-output pairs sampled from the NN model.

NNs in Less Data-Rich Environments

Two significant shortcomings of NN is the sample size requirement and the training cost. Several strategies exist aiming to alleviate these shortcomings.

Transfer learning. Training neural networks, especially highly performant, large networks, is very expensive not only in terms of CPU time but also in terms of required sample. Large pre-trained *generic* models (so-called *foundational* models) are available in many application areas, including language processing and computer vision. These generic models transform the input space (say written English text) into a representation that is more amenable to carrying out language modeling tasks than the original representation. To solve a *specific* language-related task, such as distinguishing patients with and without dementia, a foundational language model (with many pre-trained encoding layers) can become extended by a task-specific layer so that the new model performs the actual classification task. Only the task-specific layer needs to be trained.

Incorporating domain knowledge. Another avenue to reduce training cost is to incorporate domain knowledge as follows: when synthetic data for a domain can be generated, NNs can be pre-trained using synthetic data to learn a representation of the application. Then this pre-trained model can be further refined using real data to solve specific problems in that domain. Beside pre-training, several other methods exist, which include augmenting the input space of a NN with output from physical models (e.g. climate models) or ascribing meaning to hidden nodes and constraining the connections among such hidden nodes based on what is possible in the real world. See [13] for a survey of incorporating domain knowledge into machine learning models.

Method label: artificial neural networks (including deep learning)	
Main Use	• Solving predictive modeling problems. Classification, including classification with very many classes, is most common, but it can also solve regression and time-to-event problems • In data-rich environments, ANNs can produce highly performant models
Context of use	• ANNs are recommended when sample size is very high and a very complex function needs be modeled • Neural networks work best for specific applications, with network architectures specifically designed for that application. Such applications include image analysis, text and audio modeling, text synthesis, etc.
Secondary use	• Modeling distributions using GANs and auto-encoder variants
Pitfalls	**Pitfall 3.2.4.1.** ANNs can fail when sample size is not large **Pitfall 3.2.4.2.** ANNs are innately black-box models. Their use in applications where transparency is important may be problematic **Pitfall 3.2.4.3.** ANNs do not reduce the number of features needed for prediction and this may be an important requirement in many biomedical problems. However, using strong feature selectors before running the ANN modeling may be a good combination for some problems (but can be detrimental in others) **Pitfall 3.2.4.4.** Training cost is high due to (a) large cost of training a single model, and (b) large number of models that need be trained to explore the immense hyperparameter space **Pitfall 3.2.4.5.** ANNs do not have either formal or empirically competitive causal structure discovery capabilities **Pitfall 3.2.4.6.** Even when a causal structure is known, estimating causal effects with ANNs leads to biased results because the ANN is not designed to condition on known confounders and may introduce other effect estimation biases (e.g., due to blocking mediator paths and opening M-structure paths)
Principle of operation	• Minimizing a penalized loss • Linear combinations of inputs transformed through a non-linear activation function layer-by-layer yields a non-linear model
Theoretical properties and empirical evidence	• ANNs with at least two hidden (encoding) layer and unbounded number of units, can be universal function approximators • NNs are most commonly solved using gradient optimizers. Because the objective function of NNs can be arbitrarily complex with multiple local optima, the optimizers may fail to reach the globally optimal solution • In several biomedical problems deep learning and other ANN learners have exhibited superior accuracy (especially in image recognition). There is also significant evidence that in several clinical domains not involving images they do not outperform vanilla logistic regression (see Chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs")
Best practices	**Best practice 3.2.4.1.** Deep learning is most recommended for predictive modeling in large imaging datasets. Other domains may also be good candidates. In all cases additional (alternative and comparator) methods should be explored at this time within the same error estimation protocols (see chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models") **Best practice 3.2.4.2.** At this time ANNs are not suitable for causal discovery and modeling. Formal causal methods should be preferred (chapter "Foundations of Causal ML") **Best practice 3.2.4.3.** ANNs are not suitable for problems where explainability and transparency are required, or when large reduction of the feature space is important to model application

Method label: artificial neural networks (including deep learning)	
References	• Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press, 2016 • Discussion and references in Chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs"

Support Vector Machines

Support Vector Machines (SVMs) is a family of methods that can be used for classification, regression, outlier detection, clustering, feature selection and a special form of learning called transductive learning [14–16]. SVMs use two key principles (1) regularization and (2) kernel projection.

SVM regularization: SVMs cast the classification or regression problems as a non-linear quadratic optimization problem where the solution to predictive modeling is formed as a "*data fit loss + parameter penalty*" mathematical objective function. Intuitively and as depicted in Fig. 3, each object used for training and subsequent model application is represented as a vector of measurements in a space of relevant dimensions (variable inputs). We will discuss regularization in the general (non-SVM) setting in section "Regularization".

Binary classification is formulated in SVMs as a geometrical problem of finding a hyperplane (i.e., the generalization of a straight line from 2 dimensions to n dimensions) such that all the instances above the hyperplane belong to one class and all subjects below the hyperplane to the other. Translating this geometrical problem into

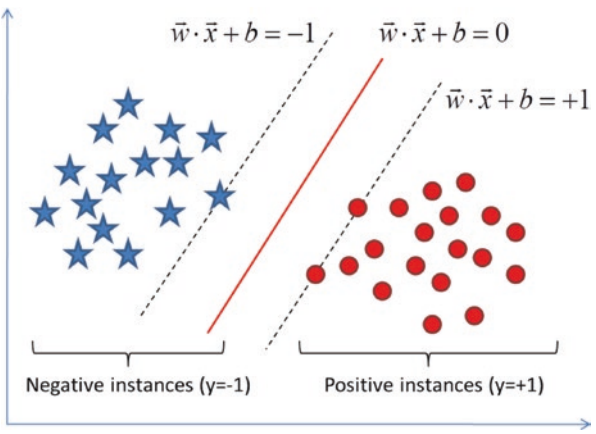


Fig. 3 SVMs and the classification problem as geometrical separation. In the top panel, a geometrical representation of a 2-class predictive modeling (classification problem) with 2 input dimensions (x_1, x_2) is depicted. Each subject is represented by a dot (i.e., a 2-dimensional vector). Blue dots are negative instances and red dots are positive ones. The line that separates negatives from positives—while maximizing the distance between classes—is the solution to the SVM problem. The instances at the border of each class are the "support vectors". In the figure we also see the mathematical expression of the classifier hyperplane and its instantiation for the three support vecotrs of the example. Such problems are easily solved by modern software

linear algebra constraints is a straightforward algebraic exercise. Every variable has a weight and collectively these weights determine the hyperplane decision function.

To ensure a model with good generalization performance and resistance to over fitting, the SVM learning procedure requires two elements: (a) That the hyperplane must be such that the number of misclassified instances is minimized (the "data fit loss" part of the objective function to be minimized). (b) A generalization-enforcing constraint, the so-called "**regularizer**", that is the total sum of squared weights of all variables, must also be minimized. Specifically, the regularizer must be minimized subject to the locations and labels of training data fed into the algorithm. This is an instance of **quadratic program** non-linear optimization function that can be solved exactly and very fast.

$$Minimize\ \frac{1}{2}\sum_{i=1}^{n} w_i^2 \quad \text{subject to} \quad y_i\left(\vec{w}\overrightarrow{x_i}+b\right)-1\geq 0 \quad \text{for}\, i=1,\ldots,N$$

A "soft margin" formulation of the learning problem in SVMs allows for handling noisy data (and to some degree non-linearities). The primary method for modeling non-linear decision functions is **kernel projection** which works pictorially as follows (Fig. 4).

In a non-linearly separable problem, there is no straight line (hyperplane) that accurately separates the two classes. The SVM (and other kernel techniques) use a mapping function that transforms the original variables (x_1, x_2, in Fig. 4) into SVM-constructed features, such that there exists a straight line (hyperplane) that separates the data in the new space. Once the solution is found in the mapped space, it is reverse-transformed to the original input variable space. Once projected back to the original input space the solution is a non-linear decision surface. Because the mapping function is very expensive to compute, special kernel functions are used that allow solving the SVM optimization *without incurring the expense of calculating the full mapping*. In mathematical terms the above take the following form:

$$f\left(x\right)=sign\left(\vec{w}\vec{x}+b\right)$$

$$\vec{w}=\sum_{i=1}^{N}\alpha_i y_i \overrightarrow{x_i}.$$

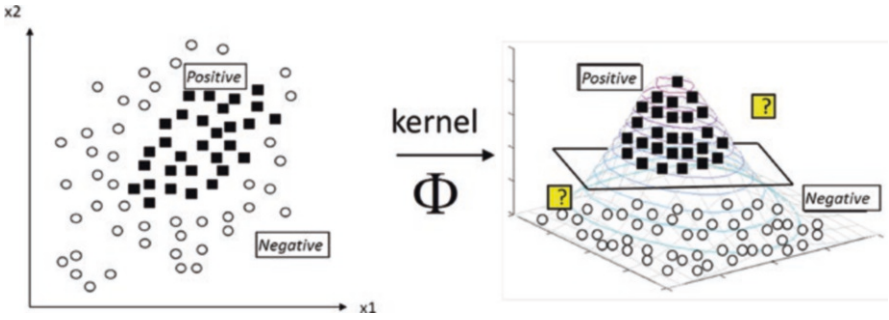


Fig. 4 Non-linearly separable classification by mapping from an original space (x_1, x_2) to a different space (with commonly higher number of dimensions) by using kernel functions

When data is mapped into higher-dimensional features space $\Phi(\vec{x})$,

$$f(x) = sign\left(\vec{w}\Phi(\vec{x}) + b\right)$$

$$\vec{w} = \sum_{i=1}^{N} \alpha_i y_i \Phi\left(\overrightarrow{x_i}\right)$$

Combining them into a classifier yields

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i y_i \Phi\left(\overrightarrow{x_i}\right) \circ \Phi(\vec{x}) + b\right) = sign\left(\sum_{i=1}^{N} \alpha_i y_i \mathrm{K}\left(\overrightarrow{\mathrm{x_i}}, \vec{x}\right) + b\right).$$

The above computations are extremely quick to execute and are solved effectively allowing the SVM to explore an astronomical-size space of non-linear interaction effects in quadratic running time and without over-fitting. Let's demonstrate the remarkable computational and sample efficiency that the kernel projection affords using an example where we will compare the number of parameters that need to be estimated and the sample size needed for a relatively simple non-linear SVM with polynomial kernel degrees of 3 or 5 and number of variables to be modeled ranging from 2 to 100. We will compare with the sample needed and interactions effects that need be constructed and estimated by the corresponding regression model under a conservative requirement of 5 sample instances per parameter.

As can be seen from Table 1. (adapted from [15] for a dataset with 100 variables with up to fifth degree polynomial interaction effects, classical regression would need >96 million parameters to be explicitly constructed and > 482 million sample size in order to estimate the model's parameters. By comparison the SVM algorithm explores the same space in time quadratic to the number of variables (i.e., in practice in seconds in a regular personal computer). Moreover, the SVM generalization error is independent of the number of variables and is bounded by a function of the number of support vectors which is smaller or equal to the available sample size (see chapter "Foundations and Properties of AI/ML Systems" for more details).

One way to think of the effects of regularization is that by forcing weights to be as small as possible, all variables that *are not relevant or are superfluous* to the predictive modeling will tend to have zero or near zero weights and are effectively "filtered" out of the model. Equivalently, the minimization of weights entails that the separation between classes is geometrically maximized and statistical machine learning theory

Table 1 Comparison of non-linear SVM vs classical regression in terms of number of parameters. N denotes the (often very small) sample size available in practice

Number of variables	Polynomial degree	Number of parameters in the SVM model	Number of parameters in the Regression model	Required sample by Regression
2	3	≤N	10	50
10	3		286	1430
10	5		3003	15,015
100	3		176,851	884,255
100	5		96,560,646	482,803,230

shows that this often leads to more generalizable models. In yet another view, regularization entails that the target function is smooth, in the sense that small differences in the input variables result in small changes to the response variable's values.

Additional aspects of SVMs include: **primary and dual formulations** of the learning problem (suitable for low dimensionality/high sample, or high dimensionality/small sample situations respectively), **using only dot product representations of the data**, **Structural Risk Minimization** (i.e., the model complexity classes are neatly organized in embedding classes so that model selection can be orderly and efficient), and **known bounds of error**. These bounds are not dependent on the number of input variables but only on the support vectors (which are at most equal to the sample size N) thus demonstrating the power of SVMs to self-regulate their complexity and avoid overfitting (see also chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI" for comprehensive discussion of overfitting and underfitting). SVM **scores can be converted to probabilities** in a post-hoc manner and can also be used to **perform feature selection for other classifiers.** While SVMs output scores and not probabilities, these scores can be converted to calibrated probabilities [15].

Method label: support vector machines	
Main Use	• Solving classification and feature selection problems • SVMs can produce highly performant models in low sample size/ large dimensionality settings • Dominant performance in certain domains (e.g., gene expression and other omics and multi-modal based classifiers, text classification)
Context of use	• SVMs are suitable for clinical data, omic data, text and other unstructured data as well as combinations • Especially suited to non-linear, noisy and small sample data. • Can be combined with other classifiers and feature selectors to form strong analysis stacks and protocols • Are very fast to train and to run model inferences
Secondary use	• Applicable, but not as first choice, to regression, clustering, and outlier detection
Pitfalls	**Pitfall 3.2.5.1.** SVMs are unsuitable for causal discovery. The features they select are not interpretable causally. SVM variable weights are not causally valid even if true causal features only are included in the model **Pitfall 3.2.5.2.** Linear SVMs are easily interpretable. Non-linear SVMs require additional steps for explanation and are innately "black box" models **Pitfall 3.2.5.3.** Error bounds are loose and typically cannot be used to guide model selection
Principle of operation	• Maximum margin (gap) classifiers with hard or soft margins. • Regularization • Quadratic Programming formulation of learning problem guarantees optimal solution in tractable time • Kernel projection enables fast exploration of immense spaces of non-linearities • Structured risk minimization ensures that kernel hyper-parameters relate monotonically to error

Method label: support vector machines	
Theoretical properties and empirical evidence	• Can model practically any function • Known error bounds • Extremely sample and computationally efficient • Overfitting resistant • Best of class performance in several domains
Best practices	**Best practice 3.2.5.1.** Primary choice for omics, text classification, and combined clinical/molecular/text tasks **Best practice 3.2.5.2.** Secondary choice for feature selection (with Markov boundary methods being first choice). In very small sample situations where Markov boundary methods may suffer, SVM feature selection can be first choice **Best practice 3.2.5.3.** SVM weights features or models should not be interpreted causally **Best practice 3.2.5.4.** Explain SVMs by converting them to interpretable models via meta-learning or other approaches; and convert scores to probabilities when needed
References	• Statnikov A, Aliferis CF, Hardin DP, Guyon I. A gentle introduction to support vector machines. In: Biomedicine: Theory and methods (Vol. 1). World scientific. 2011 • Statnikov,A, Aliferis, CF, Hardin DP, Guyon I. A gentle introduction to support vector machines.In: Biomedicine: Case studies and benchmarks (Vol. 2). World scientific. 2012 • Vapnik V. the nature of statistical learning theory. SpringerScience & Business Media. 2013 • Aphinyanaphongs, Y., Tsamardinos, I., Statnikov, A., Hardin, D. and Aliferis, C.F., 2005. Text categorization models for high-quality article retrieval in internal medicine. Journal of the American medical informatics association, 12(2), pp.207–216 • Statnikov, A., Aliferis, C.F., Tsamardinos, I., Hardin, D. and levy, S., 2005. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, *21*(5), pp.631–643

Naïve Bayesian Classifier (NBC) and Bayesian Networks (BNs)

BNs and the theoretical Optimal Bayes Classifier (OBC) are covered in the AI reasoning under uncertainty, and in the machine leaning theory sections of chapter "Foundations and Properties of AI/ML Systems". Causal BNs are instrumental for causal discovery and modeling. They are covered in Chapter "Foundations of Causal ML". The Markov Boundary feature selection methods have their origins in BN theory and are covered in the previous chapters and in feature selection section "Feature Selection and Dimensionality Reduction" of the present chapter. We thus defined here the Naïve Bayes (NB) classifier and then provide a unified method label that spans NB and BN classification techniques.

Naïve Bayes (NB)

NB is a highly restricted simplification of the complete (brute force) application of Bayes' Theorem in classification.

For an observation vector x_i and corresponding outcome y_i, which takes values from one of the values $c_1, c_2, \ldots, c_m$, the predicted probability that the outcome has value c_j can be computed through the Bayes formula

$$\Pr\left(y_i = c_j \mid x_i\right) = \frac{\Pr\left(x_i \mid y_i = c_j\right)\Pr\left(y_i = c_j\right)}{\Pr\left(x_i\right)}.$$

where $\Pr(x_i) = \sum_j \Pr(x_i \mid y_i = c_j) \Pr(y_i = c_j)$. Suppose we constructed a probability table for $\Pr(x_i \mid y_i)$, for binary x_i it would be a table of size exponential to the number of predictor variables. This would also lead to difficulties in estimating the large sample probabilities from a small sample dataset, because the dataset size would have to be large enough to contain sufficient number of observations to estimate every single x_i combination, however low the probability. To reduce this burden, Naïve Bayes classifier makes the assumption that the predictor variables are conditionally independent of each other given the outcome value. This simplifies the computation of $\Pr(x_i \mid y_i)$ to

$$\Pr\left(x_i \mid y_i\right) = \prod_k \Pr\left(x_{ik} \mid y_i\right)$$

where x_{ik} is the k^{th} element (component) of the vector x_i. Under the Naïve Bayes assumption, we only need to estimate $\Pr(x_{ik} \mid y_i)$ for each each variable x_k, which reduces the sample size and compute time required for the estimation from exponential to linear in the number of variables.

The problem with NB is that by adopting unrealistic assumptions of conditional independence, it introduces serious errors in distributions where features are not independent given the target class.

Also, if we allow outcomes to not be mutually exclusive (e.g. a patient may have serval diseases simultaneously), then we need to incorporate $2^{|m|}$ values in the outcome variable which exponentially increases (to the number m of outcome values, e.g., diagnostic categories) the number of probabilities that need be estimated and stored. Hence it is common to see the added NB assumption of mutually exclusive and exhaustive values of the target variable. Of course in medical domains this assumption is very commonly violated as well.

In summary, the problems with brute force Bayes is that of intractability, and of high error in the estimates of joint instantiation (because invariably the real-life sample size is never enough to estimate an exponential number of parameters). The problems with NB is that assumptions rarely hold in numerous biomedical domains.

In early years of AI/ML and before the advent of BNs (that can decompose the joint distribution and store only the smallest number of probabilities needed to accurately represent it), NB was used widely. All modern benchmarks suggest however that NB is no longer a competitive classifier (unless we can tolerate large departures from predictive optimality in order to save storage space). Finally, it is worth noting the work of [17] that show that under specific target functions and loss functions, NB can perform well even though its nominal assumptions are violated. These types of distributions are rare in biomedicine however, and better alternatives exist.

Method label: Naïve Bayes (NB), Bayesian Networks (BNs), causal BNs (CBNs) and Markov Boundary methods	
Main Use	• Classification, causal structure discovery, feature selection
Context of use	• Classification under a range of sufficient assumptions that guarantee asymptotic correctness: Naïve Bayes (NB) have highly restrictive assumptions, while BNs (see chapter "Foundations and Properties of AI/ML Systems") have no restrictions on functions and distributions they can model • Flexible classification (chapter "Foundations and Properties of AI/ML Systems") where at model deployment time the inputs can be any subset of variables and the output can also be any subset of variables (while the rest are unobserved) • Causal structure discovery (under specific broad assumptions—chapter "Foundations of Causal ML") • Causal effect estimation (under specific broad assumptions—chapter "Foundations of Causal ML") • Modeling equivalence classes of full or local models (see chapter "Foundations of Causal ML" and "Foundations and Properties of AI/ML Systems") • Closely related to Markov boundary feature selection algorithms (see section on "Feature Selection and Dimensionality Reduction")
Secondary use	• Optimal Bayes classifier (OBC) (see chapter "Foundations and Properties of AI/ML Systems"): can be used for theoretical analysis of optimality of the large-sample error of any classifier • NB is often used as a minimum baseline comparator in benchmark studies • Modeling the full joint distribution of data with BNs (e.g., for simulation or re-simulation purposes) • Guiding experiments with hybrid causal discovery and active experimentation
Pitfalls	**Pitfall 3.2.6.1.** Using NB simply because it is computationally fast and has small storage requirements without paying attention to whether data properties match assumptions may lead to high error **Pitfall 3.2.6.2.** Not every probabilistic graphical model is a BN (see chapter "Foundations and Properties of AI/ML Systems") **Pitfall 3.2.6.3.**Not every BN is causal and not every BN learning algorithm guarantees valid causal discovery **Pitfall 3.2.6.4.** Bad or uninformed assignment of priors may lead Bayesian algorithms astray **Pitfall 3.2.6.5.** Discovery algorithms for BNs and CBNs vary widely in output quality and efficiency **Pitfall 3.2.6.6.** BNs and CBNs properties hold in the large sample. In small samples results may be suboptimal **Pitfall 3.2.6.7.** Approximating OBC with Bayesian model averaging with a small number of models, maybe far from the theoretically ideal OBC performance **Pitfall 3.2.6.8.** BN predictive inference with unconstrained models is computationally intractable in the worst case although average case algorithms exist with good performance chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs") **Pitfall 3.2.6.9.** Discrete BNs are better developed in practice than continuous-function BNs

Method label: Naïve Bayes (NB), Bayesian Networks (BNs), causal BNs (CBNs) and Markov Boundary methods	
Principle of operation	• Application of Bayes' theorem combined with various assumptions about the data, heuristic model search procedures, and algorithms designed to infer from data the most likely model that generated it or estimate model-averaged predictions over a set of models • Causal discovery in addition relies on distributional assumptions such as the causal Markov condition (CMC), the faithfulness condition (FC) and causal sufficiency (CS) • Newer algorithms can discover local or partial models, and overcome violations of CS and FC and worst case complexity
Theoretical properties and empirical evidence	• All Bayesian classifiers have well understood properties ensuring valid predictions, reliable causal structure discovery and unbiased causal effect estimation when the assumptions hold • Closely tied to Markov boundary and causal feature selection • CBNs are the backbone of learning causal models in a sound and scalable manner (see chapter "Foundations of Causal ML") • Large body of validation in causal discovery. Large body of applications and benchmarks for classification and feature selection
Best practices	**Best practice 3.2.6.1.** NB has limited utility in modern health applications and is not a recommended method in usual circumstances **Best practice 3.2.6.2.** Use BNs when flexible classification is needed **Best practice 3.2.6.3.** Use CBNs when causal structure discovery and causal effect estimation are needed **Best practice 3.2.6.4.** Use when modeling equivalence classes of full or local causal or Markov boundary models is needed **Best practice 3.2.6.5.** Markov boundary (identified via specialized algorithms) is typically the feature selection method of choice **Best practice 3.2.6.6.** Use for modeling full joint distributions (e.g., for simulation or re-simulation purposes) while also preserving causal structure **Best practice 3.2.6.7.** Use for guiding experiments in the presence of information equivalences
References	• See references (and discussions thereof) in chapters "Foundations and Properties of AI/ML Systems," "Foundations of Causal ML," "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems," "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models," and "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML in Healthcare and the Health Sciences. Enduring Problems, and the Role of BPs"

K-Nearest Neighbor

The k nearest neighbor (k-NN) classifier and regression was introduced in 1951 [18]. K-NN is categorized as a "lazy" classifier/regressor, in the sense that it does not actually construct an explicit model from the data. Decisions are made based on the training data set rather than by a model trained on the data set. Specifically, the class of an instance is the (weighted) majority class of its *k* nearest neighbors, where *k* is a user-supplied hyperparameter. In case of k-NN regression, the estimated value of an instance is the (possibly weighted) average of the values of the k nearest neighbors.

The critical component of the k-NN classifier/regressor is the similarity function used to determine the *k* nearest neighbors. Defining an appropriate similarity functions is non-trivial, and is particularly difficult when (i) the data is high-dimensional or (ii) the importance of the variables varies greatly. The similarity function can be often learned from data. The problem of learning a similarity function from data is known as the *distance metric learning*.

When we consider the applicability of a method, we usually think about sample size or the form of the decision boundary between the positive and negative classes in a classification problem. In the case of k-NN, we may need to consider the local density of positive and negative instances. If a clear (low-density) separation exists between the positive and negative classes, kNN can be successful, regardless of the shape of the decision boundary; if a clear separation does not exist, kNN will likely not perform well (Fig. 5).

As we said earlier, kNN classifiers do not build an explicit model, instead, they have to determine the *k* nearest neighbors at classification time. This makes "training" the model cheap (there is no training), but the actual model application can become expensive. Additionally, this makes the classification less robust, as the classification of a new instance depends on the training sample, especially for small *k*. Increasing *k* can reduce the dependence on the specific training sample and can also improve robustness in the presence of noise, however, it makes the classification less local, which is the essence of kNN modeling.

The success of kNN classification depends on the amount of noise in the training sample, the choice of *k* (which again depends on the noise in the training sample), and a distance function that defines the neighborhood of the instances to be classified. Large-sample analysis of k-NN shows that in the sample limit with k=1 the error is at most double that of the optimal Bayes Classifier, whereas with larger *k* it can approximate the optimal Bayes Classifier [19].

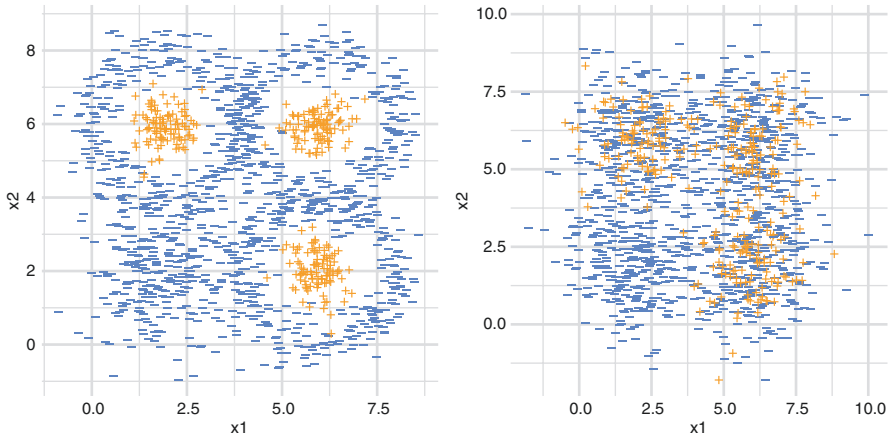


Fig. 5 Two hypothetical two-dimensional problems that can be solved using k-NN classification. Blue '-' signs indicate instances of the negative class; orange '+' the positive class. In the left pane, the positive clusters have a much higher density of positive instances than negative instances, making this an easy problem. In the right pane, there are three positive clusters, and the density of positive (orange) instances in each cluster is similar to that of the negative instances, which makes the problem more difficult

Method label: k-Nearest Neighbor (kNN)	
Main Use	• Classification and regression problems
Context of use	• Classification or regression based on similarity between instances. Its use is most appropriate when a similarity function can be easily constructed
Secondary use	• Density estimation
Pitfalls	**Pitfall 3.2.7.1** kNN asymptotic error is strongly influenced by the choice of *k* **Pitfall 3.2.7.2** kNN error convergence to the large sample error as a function of sample size is not well characterized [19] **Pitfall 3.2.7.3** in high dimensional problems the distance metric values become similar for all instance pairs and this affects accuracy **Pitfall 3.2.7.4** kNN application is computationally intensive unless special data structures are used **Pitfall 3.2.7.5** needs to store the entire training data set **Pitfall 3.2.7.6** kNN will produce poor results when the wrong similarity function is used.
Principle of operation	• Prediction is based on the local density of the k nearest neighbors of the problem instance
Theoretical properties and empirical evidence	• kNN is non-parametric • It can handle arbitrary decision boundaries • In the sample limit, the error of the 1-NN classifier (k = 1), is no more than twice the error of the Optimal Bayes Classifier [18] • Typical similarity functions used in kNN do not perform well in high-dimensional problems • Usually underperforms most modern classifiers in most applications

Method label: k-Nearest Neighbor (kNN)	
Best practices	**Best practice 3.2.7.1.** Use as comparator not as primary classifier. **Best practice 3.2.7.2.** Optimize k with model selection. **Best practice 3.2.7.3.** Use adaptive kNN for high dimensional data. **Best practice 3.2.7.4.** Explore via model selection the right distance metric for the data at hand
References	• Several textbooks, including [19–21] • Cover, T. and Hart, P., 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory*, *13*(1), pp.21–27

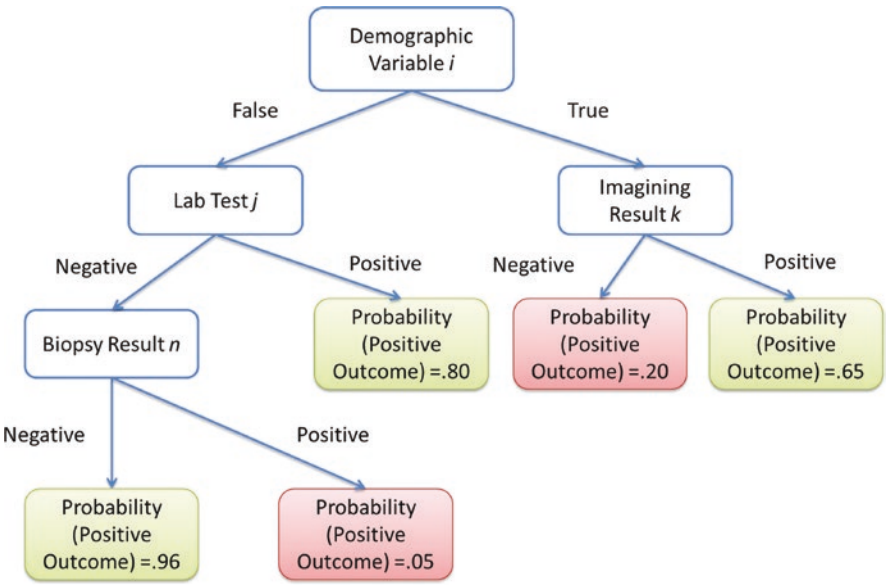


Fig. 6 Illustration of a decision tree classifying patients as having a positive outcome (e.g., successful treatment) depicted as green, or negative outcome (red). Nodes depicted with blue outline are **internal nodes** corresponding to observable variables (e.g., demographic, lab test, biopsy, or imaging test results). **Branches** correspond to the values of each variable. Nodes without children are called "**leaves"** and correspond to DT classifier decisions. Each decision has an associated probability for the values of the response (here, positive patient outcome)

Decision Tree Learning

A decision tree is a predictive model that can be used for classification or regression. Figure 6 depicts an example decision tree for classification built over discrete variables.

Learning a decision tree from data is called *decision tree induction*. Induction algorithms partition the input space into a number of hyperrectangles such that the hyperrectangles are statistically homogeneous for one or the other target class. For

example, in Fig. 6, the leaf with probability of a positive outcome =0.80 corresponds to patients with Demographic variable i = False and Lab Test j = Positive.

Induction of the optimal decision tree is NP-hard [20]. Algorithms used in practice are computationally efficient by trading off tree optimality with time complexity. Commonly used algorithms (e.g., ID3 [21]) work on the principle of *recursive partitioning*. Starting with the entire data set, a splitting attribute is selected as root of the tree, and the dataset is split into multiple partitions based on the value of the splitting attribute such that the partitions are maximally enriched in instances of one class or another (equivalently, they have optimal predictivity up to that point). Then each partition is further split using the same strategy recursively until no more partitioning is possible (i.e., the algorithm runs out of sample or informative features). The various decision tree induction algorithms differ in the way the splitting criterion is selected, the stopping criterion, and the way they handle categorical, multi-level categorical and continuous attributes. Often, DT induction algorithms do not have a global objective function, they operate greedily and offer no guarantees of optimality.

Main Properties of Decision Trees

1. **Expressive power**:
   (a) DTs can model any discrete function.
   (b) Predictions in discrete space are made at the leaf nodes and each node corresponds to a hyperrectangle. Therefore, the decision boundary is complex, non-parametric, and the sides of the hyperrectangles are parallel with the coordinate system that spans the input space.

2. **Logic-based and concept-learning interpretation of DTs**. Each path from the root of the tree to each leaf represents a conjunction (logical AND) of conditions. A tree thus can be translated into a set of conjunctive sentences, that collectively define the target as a logical concept. Therefore there is a close correspondence of DTs with logic and concept learning.
   E.g., in the tree of Fig. 6 the Concept "(most likely) Positive Outcome" is defined by the DT as:
   (Positive Outcome = True) iff:
   (((Demographic variable i = False) AND (Lab Test j = Positive)) OR
   ((Demographic variable i = False) AND (Lab Test j = Negative) AND (Biopsy Result n = Negative)) OR
   ((Demographic variable i = True) AND (Imaging Result k = Positive)))

3. **Rule set interpretation of DTs**. A DT can also be thought of as a collection of rules of the type: "if the variables on a path have the observed instantiations depicted by the tree, then the decision is determined by the corresponding leaf's probability". Hence a DT is a system of rules, each one sufficient (when applicable) to establish a decision.
   E.g., in the tree of Fig. 6 the Concept "Positive Outcome" is defined by the DT as:

   RULE 1
   IF ((Demographic variable i = False) AND (Lab Test j = Positive)) THEN

Outcome = Positive with probability 0.80

RULE 2
IF ((Demographic variable i = False) AND (Lab Test j = Negative) AND (Biopsy Result n = Negative)) THEN Outcome = Positive with probability 0.96

RULE 3
IF ((Demographic variable i = True) AND (Imaging Result k = Positive)) THEN Outcome = Positive with probability 0.65

RULE 4
IF ((Demographic variable i = False) AND (Lab Test j = Negative) AND (Biopsy Result n = Positive)) THEN Outcome = Positive with probability 0.05

RULE 5
IF ((Demographic variable i = True) AND (Imaging Result k = Negative)) THEN Outcome = Positive with probability 0.20.

- The rules corresponding to a decision tree, **are individually correct** when they match problem instances, hence fully modular (i.e., one rules does not affect the other and *can be applied in isolation*).
- Moreover the rules **do not need be chained**: each inference made by the tree on a patient is the result of applying the applicable rule.
- The rules are **mutually exclusive**.
- Finally **the order of the variables in each path/rule, does not matter** *after* the tree is constructed, and can be applied in any order. The order of variables during the DT construction from data, however, is very important for the quality of the DT induced.

Notice that such rule sets are trivial to understand. By comparison rule-based systems where the rules have to be chained in complex forward/backward sequences have logic that cannot be understood easily by examining individual rules.

4. **Subpopulation discovery/clustering interpretation of DTs**. Each path of a DT describes a subpopulation that has a defined probability for the target variables and the group members are homogeneous in their probability for the target and in their values for the features along the path. Thus a DT provides a grouping/clustering of the feature space/individual observations that is tied to the outcome. In our running example:
   GROUP 1: everyone who has:
   ((Demographic variable i = False) AND (Lab Test j = Positive))
   GROUP 2: everyone who has:
   ((Demographic variable i = False) AND (Lab Test j = Negative) AND (Biopsy Result n = Negative))
   Etc. for the other paths.
   From the tree:
   Group1 will have 0.80 probability of Outcome = Positive
   Group2 will have 0.96 probability of Outcome = Positive
   Etc. for the other groups.

Because DTs can be understood in several ways (rules, concepts, groups) and in a modular manner they are an exemplary model of interpretable and explainable ML (as long as DTs are of modest size).

5. A key problem with interpretability is *subtree replication*. Especially in the presence of noise, the exact same subtrees can appear under multiple nodes across the tree, which creates redundancy, hindering interpretation.
6. Decision trees are susceptible to overfitting. During tree induction, some method of protection against overfitting needs to be applied. Such methods include empirical testing on a validation set, regularizers (e.g., maximum allowed number of instances in a leaf, or maximum allowed tree depth, maximum allowed model complexity as part of the stopping criterion). Some of these can be enforced during learning, or after a tree is created.

Method label: Decision Trees	
Main Use	• Classification with maximum interpretability • As component of ensemble and boosted algorithms
Context of use	• DTs are highly interpretable: a decision tree can be translated into a set of rules. Interpretation is the main reason to use a decision tree • DTs are non-parametric, non-additive models that can represent linear and nonlinear relationships. • DTs are commonly used as a base learner in an ensemble • Baseline comparator
Secondary use	• Regression problems (regression trees) • Explaining black box ML models by converting them to functionally equivalent DTs
Pitfalls	**Pitfall 3.2.8.1.** A single DT on its own typically does not have very high performance **Pitfall 3.2.8.2.** DTs depending on how they are used can be unstable (which makes them a good choice for bagging) **Pitfall 3.2.8.3**. DTs if not regularized have a tendency for overfitting **Pitfall 3.2.8.4.** High dimensionality is a problem when feature selection has not been pre-applied
Principle of operation	• Recursive partitioning of the data. The inductive algorithm may build a DT following a partitioning strategy, or may follow other procedures (e.g. genetic algorithms or other search). Once a DT is built, however, it encodes a partitioning of the data
Theoretical properties and empirical evidence	• DT induction is NP-hard. Greedy algorithms are used, which provide no guarantees of optimality • Unless regularization and feature selection are applied, they have a tendency to overfit • DTs are highly interpretable • In an ensemble learning context, DTs can handle very high dimensionality, noise, and can identify interaction terms

Method label: Decision Trees	
Best practices	**Best practice 3.2.8.1.** Use DT for interpretable modeling alone or in conjunction with other methods **Best practice 3.2.8.2.** Use for target variable-specific subpopulation discovery **Best practice 3.2.8.3.** Use as baseline comparator method **Best practice 3.2.8.4.** Use in ensembling (boosting or bagging (Random Forest)) algorithms
References	Quinlan, J. R. C4.5: *Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993 Several textbooks e.g. [19–21]

Clustering

The family of clustering techniques deals with grouping objects into meaningful categories (e.g., subjects into disease groups or treatment-response groups). Typically, the produced clusters contain objects that are similar to one another but dissimilar to objects in other clusters [19, 20]. As has been shown mathematically, there is no single measure of similarity that is suitable for all types of analyses nor is clustering the most powerful method for all types of analysis even when applicable. Thus this generally useful set of methods is known to be often abused or misused [22]. We demonstrate these dangers with the following two figures.

Figures 7 and 8 show that *it is not possible for a clustering algorithm to anticipate predictive use of the clusters if the clustering algorithm does not have information about the labels (or other assumptions that amount to such information).*

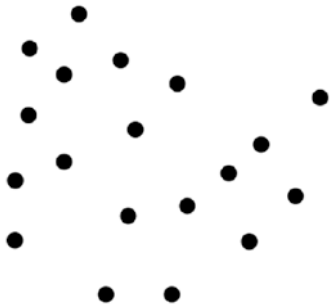


Fig. 7 Demonstration of the fallacy of using unsupervised clustering for predictive modeling. Question: what is a good clustering of the above data?

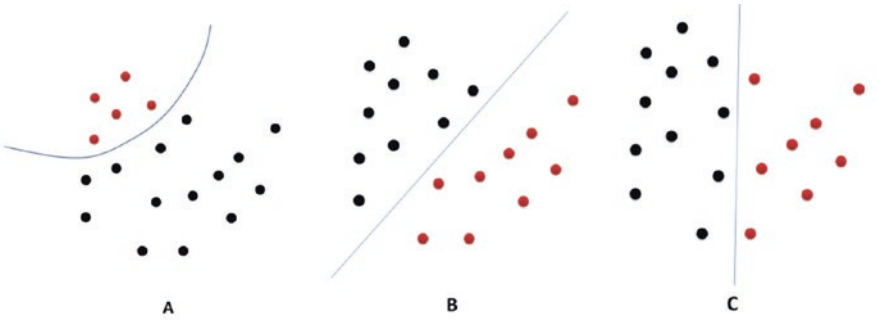


Fig. 8 Demonstration of the fallacy of using unsupervised clustering for predictive modeling, continued. Effect of the target function that generates data labels. As shown, identifying "good" (in the predictive sense) clusters of the above (and any) data, absolutely depends on the values of the target function. Case (**a**) left, case (**b**) middle, and case (**c**), right, involve exactly the same data in terms of input space but with different target functions defined over the input data (positive class depicted in red, negative in black). As a result, classifiers (blue lines) have to the class labels take into account in order to be accurate. The resulting classifiers are radically different and cannot be identified without reference to the target function that generates the class assignments

Pitfall 3.2.9
Use of clustering for predictive modeling will lead to under-performing models.

Best Practice 3.2.9
Do not use unsupervised clustering to produce groups that you intend to use predictively. Use predictive modeling, instead. Once accurate and interpretable classifiers have been built, subpopulations or other useful clusters can be extracted (see section on Decision Trees for a detailed example).

Clarifying Misconceptions About Unsupervised/Supervised Learning, Similarity/Distance-Based Learning, and Clustering

(a) As explained in chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI" a common way that unsupervised clustering apparently yields decent predictive performance (e.g., in some parts of the high-throughput based genomics literature), is because a pre-selection of features was performed on the data based on how strongly the features correlated with the outcome. The

analysts in these cases *inductively biased* the clustering toward classification of the desired response variable. This inductive bias is not enough to lead a clustering algorithm to optimal accuracy levels, however. Moreover, if such feature pre-selection is not done in a nested cross-validated manner (see chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI") the resulting classification accuracy estimates will be highly biased (in the error estimation sense) and generalization will suffer.

(b) Because clustering is often defined in terms of *similarity*, it may be tempting to think that all similarity-based classification is predictively flawed. In reality, powerful classifiers exist that use distance and similarity functions (e.g., KNN, SVMs, etc. see section in the present chapter) however they are *supervised* (i.e., they approximate a particular target function that generates the data) which allows them (along with the rest of their design) to be capable of accurate predictive modeling.

(c) Conversely, and to re-iterate the point previously made, because good similarity-based predictive modeling exists, that does not mean that an unsupervised method, such as clustering, can be successful for predictive modeling.

(d) Finally, users of learning methods such as BNs and Causal Probabilistic Graphical Models (CPGMs) do not specify a target response variable and some may confuse them with unsupervised methods. In reality, because they model the joint probability distribution (and underlying causal generating function for CPGMs) they are *supervised learners for all variables* which can then be used as potential target responses at model execution time (see also chapter "Foundations and Properties of AI/ML Systems", section on flexible modeling with BNs).

Importance of Feature Selection for Similarity-Based Classification

If the features *containing all information about the response variable* are not included in the training data, then the predictive accuracy of similarity-based (*and any other sufficiently powerful classifier* family) will be compromised relative to the best classifier that can be built with this data. Compromised performance may also happen because of using feature selectors that allow large numbers of redundant or irrelevant features. This can overwhelm classifiers that, for example, lack sufficiently strong regularization or other anti-overfitting measures. In high dimensional spaces it is critical to apply sound feature selection algorithms so that choice of features enhances rather than hinders classification.

The following table summarizes essential properties of clustering algorithms.

Method label: clustering	
Main Use	• Group data for exploratory analysis, summarization and visualization purposes
Context of use	• Summarize, visualize and explore data (e.g., outliers, apparent distribution mixtures, etc.)
Secondary use	• Often used *inappropriately* for causal discovery and classification purposes
Non-recommended Uses and Pitfalls	**Pitfall 3.2.9.1.** Clustering variables should not be used to infer that they are causally related (a common mistake in genomics literature) **Pitfall 3.2.9.2.** Clustering individuals should not be used to build classifiers (also a common mistake in genomics literature) **Pitfall 3.2.9.3.** Choice of similarity metric, algorithm, and parameters inductively biases results toward specific groupings. There is no such thing as "unbiased" clustering analysis
Principle of operation	• Typically unsupervised method (i.e., clustering algorithms do not have access to response variable values) • Group together data instances that are similar and apart dissimilar instances • In other versions, cluster variables or cluster *simultaneously* variables and instances • Use similarity metrics and algorithms that employ those to create clusters • Clusters can be overlapping or mutually exclusive (soft vs hard clustering) • Clusters can be hierarchically organized or distinct
Theoretical properties and Empirical evidence	• Across all possible uses all clustering algorithms are on average equivalent ("Ugly Duck Theorem") • Clustering lucks the inductive bias, information capacity and the computational complexity to be compatible with causal discovery
Best practices	**Best practice 3.2.9.1.** Do no use clustering to discover causal structure **Best practice 3.2.9.2.** Do not use clustering to create accurate classifiers **Best practice 3.2.9.3.** Tailor the use of clustering algorithm and metric to the problem at hand **Best practice 3.2.9.4.** Derive predictive subgroups from properly-built and validated classifiers (Decision Trees are particularly good candidates—See "Decision Tree Learning") **Best practice 3.2.9.5.** Perform sensitivity analysis to study the impact of the choice of parameters, metrics and algorithms **Best practice 3.2.9.6.** Repeat and summarize multiple runs of randomized clustering algorithms **Best practice 3.2.9.7.** Either start from causal and predictive algorithms and use clustering to summarize, visualize, etc. their results, or start from clustering analysis in preliminary data to inform the design and analysis with focused techniques

Method label: clustering	
References	• Dupuy, A. and Simon, R.M., 2007. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. *Journal of the National Cancer Institute*, *99*(2), pp.147–157. • Several textbooks [19–21]

Ensemble Methods

So far, a single predictive model has been trained on a training data set and the prediction from this model is the final prediction. In **ensemble learning**, an ensemble, i.e., a set of models, is trained on the training data and their output is combined into a final prediction. Figure 9. illustrates the ensemble learning process. A set of m models, called **base learners**, $L_1, \ldots, L_m$ are trained on their corresponding data sets $X_1, \ldots, X_m$. The data set X_i can be the data set X itself or a sample from it. Predictions from the m base learners are combined using the **meta learner** L^*. The meta learner can be as simple as majority voting or as complex as a neural network. The different ensemble learning methods depend on (i) how they generate the data set X_i from X, (ii) the base learners they use and (iii) the meta learner.

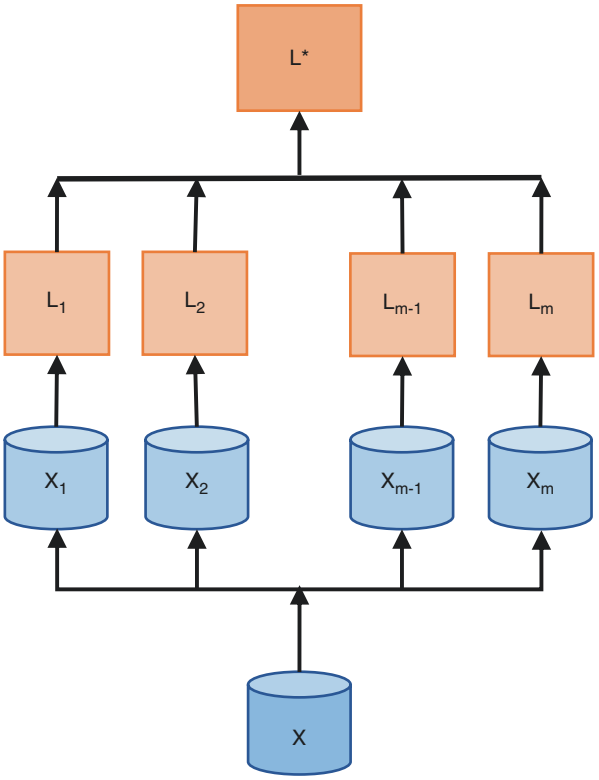


Fig. 9 Generic form of Ensemble Learning

The key motivation for using ensembles is improving predictive performance. This is achieved in four possibly overlapping ways. First, if we assume that the base learners make mistakes independently, as long as the base learners achieve better (lower) than random error rate, the ensemble will have a lower error rate than the individual base learners. In practice, the base learners do not make errors independently, but the ensemble still tends to achieve better performance than the base learners. Second, the ensemble learning framework allows us to combine models of different characteristics, potentially allowing for compensating for biases inherent in some of the methods. Third, ensembles of various types of base learners can expand the base learners' expressive capabilities: the ensemble can express much more complex relationships than base learners. Fourth, the ensemble may reduce the variance of a collection of models built on small samples.

Model Stacking

Model stacking is essentially the generic form of ensemble learning. The data sets X_i can be the original data set (X), a bootstrap resample of X, or a subsample of X. For high-dimensional data sets, X_i can be a random projection of X, which helps most when X consists of highly redundant features. For low dimensional X, X_i can consist of random linear combinations of the features in X. The base learners can be of any type and no uniformity is required: the ensemble can contain different types of models. Finally, the meta learner can be any sufficiently powerful ML algorithm; as of late, a common choice is neural networks.

Expressive ability. The relationship that ensembles can model between the inputs and the output depends on the base learners' expressive ability. In practice, model stacking can increase the base learners' expressive ability.

Example. We show a stack of logistic regression models, where both the base learners and the meta learner are logistic regression models. The ensemble is applied to the problem depicted in Fig. 10.

This is a two-class classification problem using a two-dimensional data set with strong interactions (almost an exclusive OR). The two colors (orange and blue) represent the two classes. The ensemble consists of 10 logistic regression models, each trained on a random subsample of the original data X. The meta learner is also a logistic regression model. The solid line in the figure is the contour line corresponding to 20% probability of positive, the dotted line to 50%, and the dashed line to 80% probability of positive class. Although the individual logistic regression models cannot solve this problem, the ensemble can.

Note on the difference between stacked linear regression and neural networks. This example of stacked logistic regressions appears similar to a 2-layer neural network with sigmoid activation, however, in a 2-layer neural network the "logistic regression models" on the first layer are optimized together with the second layer, while in stacked regression, the first layer models are constructed first (on random subsamples) with the actual output as the dependent variable, their coefficients are

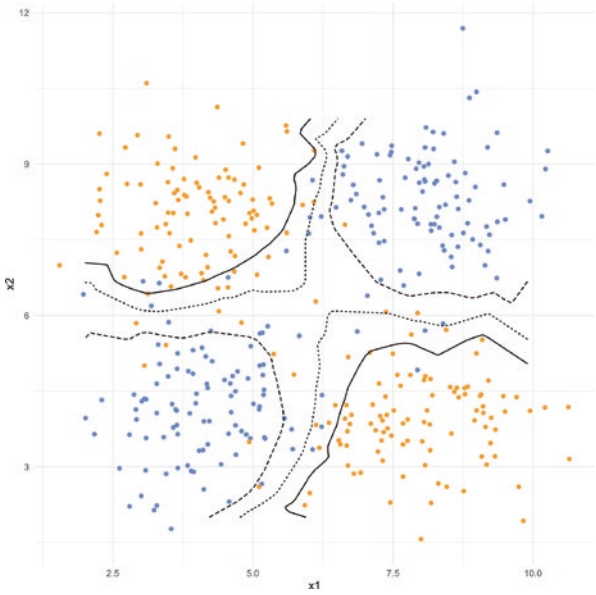


Fig. 10 Illustration of an ensemble of 10 logistic regression models with a logistic regression meta learner on a two-dimensional classification problem

fixed, and only then is the second layer model (the meta learner) constructed. While the second layer logistic regression is being fitted, the coefficients of the first layer are not modified.

Method label: model stacking	
Main Use	• Classification or regression
Context of use	• This is a generic form of ensemble learning • There are four main reasons for using ensemble learning: – Ensembling can reduce error by taking advantage of independent errors of the base learners – Base learners have different inductive biases – The ensemble can expand the base learners' hypothesis spaces by combining their spaces – The ensemble may reduce variance of base learners
Secondary use	• Other types of learning (not only classification or regression) can be ensembled
Pitfalls	**Pitfall 3.3.1.1.** Interpretability suffers **Pitfall 3.3.1.2.** Increased computational cost **Pitfall 3.3.1.3.** Potentially additional data may be needed for training the ensemble
Principle of operation	• Multiple base learners are built and then a meta learner is trained on the output of the base learners and the actual outcome. The prediction from the meta learner is the prediction of the ensemble
Theoretical properties and empirical evidence	• Stacking draws its formal foundations from fundamentals of ML theory plus the theory of weak learner boosting and bagging • Several stacking algorithms exhibit best of class performance in various domains. [23, 24] individual algorithms (e.g., Adaboost, random forests) have distinct characteristics (described below)

Method label: model stacking	
Best practices	**Best practice 3.3.1.1.** Consider stacking as a high priority choice of algorithm when high performance is needed, base algorithms do not perform well, and interpretability is not a strong requirement
References	• Wolpert (1992). "Stacked Generalization". Neural Networks. 5 (2): 241–259 • Breiman, Leo (1996). "Stacked regressions". *Machine Learning*. **24**: 49–64. • Textbooks [20]

Bagging

Bagging, also known as *bootstrap aggregation*, forms the data sets for the base learners as a bootstrap resample (i.e., sample with replacement) of the original dataset. The base learner can be any learning algorithm and the "meta learner" is simply majority voting (or average in the continuous outcome case).

The key benefit of bagging is reducing the variance of the base learner. If a classifier's predictions are sensitive to minor perturbations in the data, this means that the generalization error is negatively affected by the variance of the base learning. Bagging can help reduce random fluctuations across possible training samples. If the base learner is robust to small perturbations, then the generalization error is caused mainly by bias in the base learner, and bagging cannot help. Thus, bagging is most useful in conjunction with base learners that have high variance (such as decision trees).

Figure 11 illustrates bagging. The left panel shows a one-dimensional data set, where the horizontal axis shows the data ("x") and the vertical axis is simply a

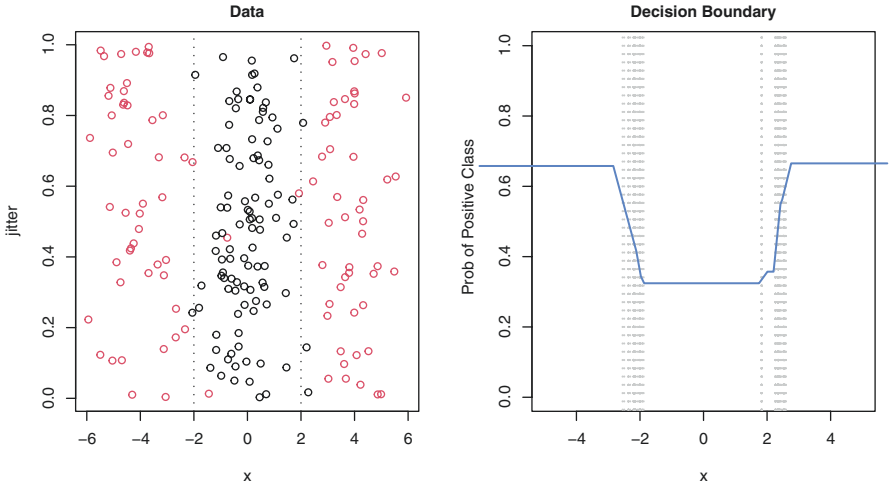


Fig. 11 Illustration of bagging

jitter-plot visualization (i.e., the y axis spreads randomly the corresponding x points so that data points on the x axis are not plotted on top of each other). Red points represent the positive class and black points the negative class. The true decision boundary for the positive class is x < = −2 or x > 2; this is shown as two dashed lines. We constructed 200 decision stumps (decision tree with depth 1, with a single binary split of the input dimension) on bootstrap samples of this data set. Such stumps can accurately learn the target function in $x \geq 2$ or in $x \leq -2$. The right panel shows 200 decision boundaries from these 200 decision trees as dotted gray lines. The horizontal axis is the data dimension ("x") and the vertical axis is the probability (from the trees) of an instance with the corresponding x value belonging to the positive class. The blue line corresponds to **the bagged prediction** using the previously constructed 200 trees.

Bagging can also help increase the expressive capability of the base learner. Fig. 11 demonstrates that by bagging weak learners, improved accuracy can be achieved in a simple example. Bagging decision stumps, for example can form a decision boundary that can only be achieved by deeper (more than one-level) trees without bagging. Also it can be seen in Fig. 11 that the bagged classifier smooths the decision surface.

Bagging a set of models is more resilient to overfitting than overfitting any single model. Overfitting that arises as a result of a predictor aligning with an outcome randomly (i.e. in a specific sample), is not likely to happen in a different sample. Therefore, only one (or very few) trees are affected by this particular random alignment. Other random alignments can also occur in other samples, and they will tend to average out. Thus overfitting is still much possible on the individual model level, but less so at the level of bag of models.

Random Forests

A Random Forest (RF) is an ensemble classification method using decision trees as the base learning algorithm combining the following four ideas: (a) bagging decision trees, (b) random feature projection, (c) off training sample error estimation, and (d) model complexity restrictions.

The RF generates X_i, the data set on which the *i*th tree is grown by bootstrapping once at the start of modeling. The features are randomized at each tree's node expansion step. Feature randomization increases the independence among the trees in the forest, while bagging aims to reduce the variance of the predictions by each tree. The predictions from the tree in the forest are combined using majority voting or averaging (for continuous outcomes).

Method label: fandom forest	
Main Use	• Classification
Context of use	• RFs achieve high predictive performance
Secondary use	• Regression, time-to-Event (Random Survival Forest)

Method label: fandom forest	
Pitfalls	**Pitfall 3.3.3.1.** As with all ensemble techniques interpretability suffers **Pitfall 3.3.3.2.** If not restricting the size of trees RFs can overfit
Principle of operation	• Bagging trees • Random projection of the features • Off-training sample error estimation • Model complexity restrictions
Theoretical properties and Empirical evidence	• Overfitting is controlled • In practice, RFs can handle very high dimensional problems • Excellent empirical performance observed in benchmarks across biomedical domains [25]
Best practices	**Best practice 3.3.3.1.** Use as primary or high priority choice when high predictivity is required and interpretability is not of high importance **Best practice 3.3.3.2.** Do not rely on internal error estimation but use an independent unbiased error estimator **Best practice 3.3.3.3.** When feature selection is important, combine with an external feature selection algorithm **Best practice 3.3.3.4.** Control DT size using as starting point the recommendations of the inventor in original publication [24]
References	• Breiman L (2001). "Random Forests". Machine Learning. 45 (1): 5–32 • Hastie, Friedman, Tibshirani. Elements of statistical learning second edition, Chpater 15. 2009, springer • Tan, Steinbach, Karpatne, Kumar. Introduction to data mining, second edition. Chapter 4.10.6., 2018, Pearson

Boosting

Boosting creates an ensemble of base models sequentially, where the *i*th base model is aimed at correcting errors made by the ensemble of the previous *i-1* base models. The resulting ensemble is an additive model, where the final prediction is the sum of the predictions from the base models. The data set X_i for the *i*th model can be the data set X itself, a bootstrap resampled version of X, or a weighted version of X to emphasize difficult-to-classify instances.

Consider a boosted ensemble of m models. The prediction for an instance X_i is

$$y_i = f\left(\sum_{j=1}^{m} m_j\left(X_i\right)\right)$$

where $f(\cdot)$ is the link function and $m_j(\cdot)$ is *j*th model. Similarly to GLM, $f()$ is identity for gaussian outcomes, logit/expit for binomial outcomes, exp for counts and survival outcomes, etc. The ensemble is built up iteratively, following a gradient descent procedure

$$M_{j+1} = M_j + \gamma \frac{\partial l}{\partial M_j},$$

where M_j is the ensemble after the jth iteration, l is the log likelihood of M_j, and γ is the learning rate. For the exponential family of distributions, including the Gaussian, binomial (and multinomial) and Poisson (survival) outcomes, the derivative of the log likelihood is the residual. Thus the $(j + 1)^{st}$ model is the model fitted to the residual of the ensemble M_j. This leads to a very straightforward interpretation that the subsequent model is built to the errors (residual) of the ensemble M_j thus the $(j + 1)^{st}$ base model aims to correct the mistakes made by the previous j base models.

Gradient boosting can overfit, but typically only slowly to the number of modeling iterations. The number of base models in a boosted model (the number of iterations to perform) is a hyperparameter h. It has been observed that models consisting of substantially more base models than optimal still overfit only minimally. A potential reason for this is that when a model reaches overfitting, adding further base models has very minimal impact. However, boosting will eventually overfit, so overfitting should be controlled by h.

Gradient Boosting Machines (GBM) are a special case of boosting, where the base learners are decision trees, often decision stumps (1 level-deep trees). Similar to bagging, boosting can also expand the base model's expressive capability.

AdaBoost is another special case of boosting which relates to the generic gradient boosting the same way as Fisher scoring (or Newton Raphson) optimization algorithm relate to gradient descent. AdaBoost is specifically designed for binomial (multinomial) outcomes.

The $(j + 1)^{st}$ model is added to the ensemble using

$$M_{j+1} = M_j + \gamma \left(\frac{\partial^2 l}{\partial M_j^2} \right)^{-1} \frac{\partial l}{\partial M_j},$$

where $\frac{\partial^2 l}{\partial M_j^2}$ is a diagonal matrix containing the second derivatives of the log likelihood l. For binomial outcomes, the second derivative is $p(1\text{-}p)$, where $p_i = expit\left(M_j\left(X_i\right)\right) = \frac{\exp\left(M_j\left(X_i\right)\right)}{1+\exp\left(M_j\left(X_i\right)\right)}$.

AdaBoost is implemented by weighing the i^{th} training instance by $(p_i(1 - p_i))^{-1}$ and using the residuals $r_i = y_i - p_i$ as the dependent variable.

Method label: gradient boosting machines (GBM)	
Main Use	• Classification
Context of se	• GBMs offer high predictive performance • Interpretability is limited to variable importance
Secondary use	• Exponential family distributions, including time-to-event
Pitfalls	**Pitfal 3.3.4.1.** As with all ensemble techniques interpretability suffers **Pitfal 3.3.4.2** If not controlling the complexity parameter it can overfit

Method label: gradient boosting machines (GBM)	
Principle of operation	• Gradient Boosting with decision stumps as base learners • Gradient descent (GBM) or Fisher-scoring (AdaBoost) in model space • Can be thought of as successively reducing the residual errors from previous cycles of modeling.
Theoretical properties and empirical evidence	• Very expressive • Overfitting can be controlled • Can handle very high dimensional data • For AdaBoost, theoretical (but loose) error bounds are proven. • Certain boosting algorithms (e.g., Adaboost) have sensitivity to noise • Top performer in several types problems/data
Best practices	**Best practices 3.3.4.1.** Use as primary or high priority choice when high predictivity is required and interpretability is not of high importance **Best practices 3.3.4.2.** When feature selection is important, combine with an external feature selection algorithm **Best practices 3.3.4.3.** Control overftting by restricting number of iterations (number of trees) **Best practices 3.3.4.4.** If data is noisy, prefer noise-robust variants. **Best practices 3.3.4.5.** Select appropriate link function for exponential family outcomes
References	• Schapire, R.E., 1990. The strength of weak learnability. *Machine learning*, *5*, pp.197–227 • Freund, Y., 1999, July. An adaptive version of the boost by majority algorithm. In *proceedings of the twelfth annual conference on computational learning theory* (pp. 102–113) • G. Ridgeway (1999). "The state of boosting," Computing Science and Statistics 31:172–181 • Long, P.M. and Servedio, R.A., 2008, July. Random classification noise defeats all convex potential boosters. In *proceedings of the 25th international conference on machine learning* (pp. 608–615)

Regularization

A problem is considered high-dimensional when the number of predictors is comparable to or exceeds the number of observations. When the number of predictors equals the number of observations, an OLS regression fit can be an exact fit, with 0 training error. Such a model will be most likely overfitted. When we consider other types of models, with the number of predictors very close to or exceeding the number of observations, overfitting becomes highly likely. We start our discussion with an explanation of regularization, a general solution to the high-dimensionality problem, and next, we discuss ways in which regularization can be added to various modeling techniques.

Regularization from a Bias/Variance Perspective

The material builds on the BVDE in chapter "Foundations and Properties of AI/ML Systems". When a model is overfitting and has high variance and low bias, it can be advantageous to increase bias provided that it reduces variance. One way to achieve this is to reduce complexity and another way, the subject of this section, is regularization [26].

We previously saw how SVMs perform regularization and that the resulting RFE-SVM feature selector is one of the strongest feature selectors (second in performance to Markov Boundary methods in empirical performance across various domains). See section "Support Vector Machines" for SVM regularization, and section "Feature Selection and Dimensionality Reduction" for feature selectors. See also chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring Problems, and the Role of Best Practices" when feature should be interpreted causally or not.

Next we examine the Elastic net family of regularizers which we will call Maximum Likelihood (ML) Regularizers.

Let $l(\beta)$ denote the log likelihood function, with β representing the model parameters. Model fitting without regularization solves

$$\max_{\beta} l(\beta).$$

ML regularization adds a penalty term $P(\beta)$

$$\max_{\beta} l(\beta) - \lambda P(\beta),$$

where λ controls the amount of regularization. Different ML regularization methods differ in the form of $P()$. Table 2. shows the most common regularization terms.

Ll regularization modifies the coefficients by pulling them away from maximum likelihood estimate. The maximum likelihood estimate is unbiased, thus regularization introduces bias, in hope of reducing variance. The various ML regularization methods differ in the way they modify the coefficients. Lasso is shrinking the coefficients towards 0 and it has the ability to set some of the coefficients exactly to 0. This allows Lasso to be used as a feature selector. Therefore, Lasso penalty is in principle most useful when some of the features are not truly related to the outcome.

Table 2. Common penalty (regularization) terms

Name	Penalty term	Remark
Lasso	$\|\|\beta\|\|_1 = \sum_j \|\beta_j\|$	Can set the coefficient β_j to exactly 0
Ridge	$\|\beta\|_2^2 = \sum_j \beta_j^2$	
Elastic net	$(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1$	Balance between Ridge and Lasso
Adaptive Lasso	$\sum_j w_j \|\beta_j\|$	w_j reduces the penalty on important variables. A common choice is $\frac{1}{\beta_j^*}$, , where β_j^* is the OLS estimate.

In contrast, Ridge penalty simply shrinks coefficients towards zero without actually setting them to 0. Elastic net allows to blend these two penalties together.

Adaptive lasso [27] also addresses the situation where some of the variables are not related to the outcome. However, by weighing the penalty on the individual coefficients, it aims to shrink important variables less and eliminate variables that are not related to the outcome (shrink their coefficients all the way to 0). The property that the adaptive lasso can set non-zero coefficients to variables that are relevant with probability approaching one is called the *oracle property*. Feature selection is discussed in more detail in the section "Feature Selection".

The procedure for the adaptive lasso proceeds in two steps. First, a regular lasso model is constructed. In the second step, an adaptive lasso model is constructed, with the weights being the inverse of the coefficients from the first lasso.

We will discuss overfitting in chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI". There are many reasons for overfitting, including high dimensionality, with many irrelevant features and highly correlated features in a problem with moderate dimensionality. In the former case, lasso or adaptive lasso is most appropriate: lasso will discard some of the features. In the latter case, Lasso and Ridge will have different effect. When overfitting occurs as a result of high collinearity, the OLS estimator tends to set the coefficients of collinear variables to very high positive and similarly high negative values. Lasso will select one of the correlated features and set the coefficients of the others to zero, while Ridge will set the coefficients to similar values across the correlated features. Ridge prevents the coefficient from taking the extremely high values.

Bias and correctness. Since the penalties aim to trade bias for variance, the estimates are likely biased. Lasso has feature selection ability, however, it is not guaranteed to find the correct support (the exact set of variables that are predictive of the outcome). Adaptive lasso in theory finds the correct support and may help reduce or even eliminate the bias.

Method label: penalized regression	
Main Use	• Predictive modeling with exponential family outcomes
Context of use	• High-dimensional data • Data with collinear variables
Secondary use	• Lasso can be used for variable selection
Pitfalls	**Pitfall 3.4.1.1.** The models assume linearity and additivity. They are not appropriate if these assumptions are violated **Pitfall 3.4.1.2.** Does not have the same interpretability as unregularized regression models. Specifically cannot be used to estimate effects of an exposure on outcomes controlling for confounders given to the model **Pitfall 3.4.1.3.** The theory of optimality of the ML regularized regression does not address the selection of strongly vs weakly relevant vs irrelevant features which is essential to feature selection **Pitfall 3.4.1.4.** Extensive benchmark results show weak empirical feature selection performance across many datasets, loss functions and comparator algorithms
Principle of operation	• Regression models • Biases the coefficient estimates to reduce variance (bias-variance tradeoff)

Method label: penalized regression	
Theoretical Properties and empirical evidence	• They tend to give biased estimates (on purpose) • BVDE give theoretical support to the means of operation • "Oracle property" (as defined by [27])
Best practices	**Best practice 3.4.1.1.** Penalized regression can operate in high dimensional datasets that classic regression cannot handle at all **Best practice 3.4.1.2.** Use as comparator method along with others as appropriate for the application domain **Best practice 3.4.1.3.** May be useful for feature selection, but not as first-choice methods **Best practice 3.4.1.4.** When non-linear regularized models are needed, consider link functions that model the non-linearity as well as kernel SVMs, kernel regression
References	Hastie, T., Tibshirani, R., Friedman, J.H. and Friedman, J.H., 2009. *The elements of statistical learning: Data mining, inference, and prediction* (Vol. 2, pp. 1–758). New York: Springer

Regularizing Groups of Variables

In the previous section, Lasso was used for variable selection, primarily in the context when many variables could be assumed irrelevant to the outcome. Variables may be related to each other and form groups. It may be useful to select variables on a per-group basis [28].

Assume that variables form K groups with p_k variables in the kth group. Let $\beta^{(k)}$ denote the coefficients of the variables in group k. With l denoting the log likelihood function, the group lasso is formulated as

$$\max_{\beta} l(\beta) - \lambda \sum_{k=1}^{K} \sqrt{p_k} \left\| \beta^{(k)} \right\|_2 .$$

The sparse group lasso [29] formulation

$$\max_{\beta} l(\beta) - \lambda\alpha \left\| \beta \right\|_1 - \lambda(1-\alpha) \sum_{k=1}^{K} \sqrt{p_k} \left\| \beta^{(k)} \right\|_2$$

also has a (regular) lasso penalty, allowing for selecting variables on a per-group basis and further selecting variables within the groups.

Regularizing Partial Correlations

The precision matrix, that is the inverse of the variance-covariance matrix, is a key parameter of a multivariate normal distribution. Regularization is necessary to estimate the precision matrix when sufficient observations are not available.

Let Θ denote the precision matrix and S the sample covariance matrix. The regularized estimate of Θ is computed as

$$\max_{\theta \geq 0} \log\det \Theta \; - tr(S\Theta) - \lambda \sum_{j \neq k} \Theta_{jk} .$$

The inverse covariance matrix contains the partial correlations. Two random variables X_i and X_j are independent if their covariance is 0; and they are *conditionally* independent (conditioned on all other variables) if the *ij*th element of the inverse covariance matrix is 0 [30].

Regularization to Constrain the Search Space of Models

So far, we have shown examples of regularization to avoid overfitting, either by computing a sparse solution (some parameters/coefficients are set to exactly 0) or by introducing bias to reduce variance. We have done so mostly in the context of regression.

Regularization is more general. It can be broadly viewed as a means to constrain the search space of models to confer some desirable property on the model. It is not limited to regression, but it is often used in conjunction with likelihoods. This is not a requirement, it can be used with any kind of quasi-likelihood or arbitrary loss functions. Constraining the search space of models through regularization is arguably the most common use and providing an exhaustive overview is impractical as new applications are continuously developed. In this section, we simply show some examples.

Knowledge distillation. Deep learning models are regarded complex highly performing models. Very complex DL models have been trained in many areas that can be further modified for particular applications (See chapters "Considerations for Specialized Health AI and ML Modelling and Applications: NLP," "Considerations for Specialized Health AI and ML Modelling and Applications: Imaging—Through the perspective of Dermatology"). These modified models are still very large, and deploying such models into resource-limited environment is difficult. The teacher-student paradigm, consist of a large teacher model used to train a small student model. Knowledge distillation is the process of generating smaller student models (models with fewer parameters), deployable in limited-resource environment, that are trained based on the more complex teacher models. Knowledge distillation is often implemented using regularization: the student model is regularized so that it resembles the teacher model in various aspects [31].

Learning DAGs. Traditionally, learning DAGs is a combinatorial optimization problem. However, the NOTEARS method introduced a penalty term, applicable to weight matrices, that enforces DAG-ness when this weight matrix is used as an adjacency matrix [32].

Dropout (Neural Networks)

Dropout layers in Neural Networks aim to reduce overfitting due to noise. In dropout, a pre-defined (as hyperparameter) proportion (**dropout portion**) of nodes are dropped from the hidden layers and possibly from the input layers. "Dropping from the network" means that the inputs and outputs of these nodes are severed and thus these nodes no longer influence the prediction. The set of nodes to be dropped is selected at random in each epoch. After the epoch, the nodes are restored [33].

Dropout layers have properties both from regularization as well as from ensemble learning. Clearly, they are similar to regularization, because they constrain the network architecture by dropping some nodes.

The ensemble perspective can be explained as follows. One way to reduce the risk of overfitting in a neural network would be to build an ensemble of neural networks, i.e. multiple neural network models with different parameterizations. Given the high cost of training neural networks, this approach is impractical. Instead, dropout re-configures the network temporarily, which means that the network being trained in each epoch has a (slightly) different architecture. Over the training epochs, a range of different network architectures are explored.

An alternative explanation of the dropout layers is, that the potentially high number of parameters a network has over possibly many layers, makes the network susceptible to **co-adaptation**, where multiple parts (sets of nodes) of the network get optimized so that some parts can correct for errors made by other parts. This co-correction allows for easely fitting noise, which is undesirable. Dropping nodes out of the network at random, breaks these co-adaptation patterns.

Dropouts can be defined on the input layer, as well. In this case, the network temporarily ignores some of the input features. This is similar to introducing noise into the data to make the model more robust.

When a neural network model is used to make predictions, all nodes are used for the prediction, i.e. no dropout is used for prediction. Since the nodes were trained with some of the nodes missing, the weights of the nodes may be too high. To correct for this, the weights are scaled down by the dropout portion at prediction time.

Why Regularized Models and Other Predictive Modeling Should Not Be Used For Reliable Causal Modeling

Regularization, with the exception of penalizing the precision (partial correlation matrix), has profound impact on a modeling technique's ability to condition on variables.

Figure 12 depicts a dataset with two variables A and B and an outcome (target) T. The target T is binary, points in red correspond to positive outcome and the points in black to negative. The data generating causal relationships are $A \rightarrow T$ and $A \rightarrow B$. This means that B is independent of T given A; or in other words, A alone is sufficient to classify the instances, B contains no additional information. From a causal perspective, the association of B with the target is confounded by A (their

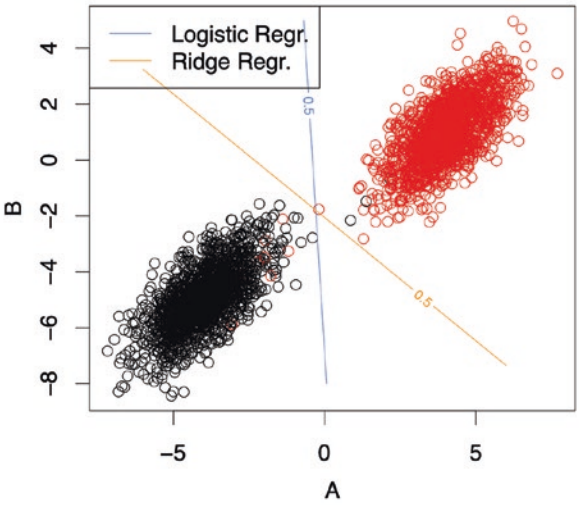


Fig. 12 Illustration of the ability and inability to condition on a variable

common cause). Any valid causal method must be able to differentiate correlation from causation, i.e., determine which correlations are causal and which are confounded. This is not happening in the example since the decision surface (orange line) of the (logistic) ridge regression gives the same weights to both A and B. Ridge regression is not designed to realize that B and T are conditionally independent given (conditioned on) A. Contrast this with the decision surface (blue line) produced by unpenalized Logistic Regression, which assigns almost zero weight to the confounded variable (B) and nearly all the weight to the true cause (A) since LR is capable of correctly conditioning on any set of confounders (thus correctly estimating direct causal effects).

Ridge regression is not the only method that has this problem. The maximal margin decision boundary that SVMs would select, is similar to the orange line; and most classifiers including modern regularized regression methods, principal component and other classical dimensionality reduction methods, as well as all predictive modeling without causal properties, will make similar errors.

Figure 13 illustrates another example where Lasso penalized regression (and other non-causal techniques) will fail to correctly condition on variables. In this example, we have 7 variables, A, B, …, E, S and the target variable T. A, B, …, E are direct causes of T. Variable S which is not causal for T (but confounded with it via A,....E) synthesizes information from causal variables A, B, …, E. In this setup, S and T are independent conditioned on A, B, …, E. However, since S synthesizes information from A, B, …, E, it can contain more information about T than any subset of A, B, …, or E. Thus, when building a predictive model for T, a penalized model, like Lasso, can prefer a set of predictor variables that contains S over the correct set of A, B, …, E. In contrast, unpenalized logistic regression (and any sound

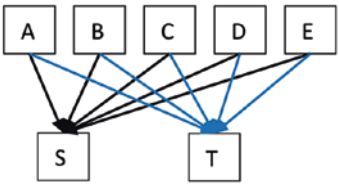


Fig. 13 Simple example where a confounded correlate synthesizes information from multiple true causes

causal algorithm) will correctly identify that *S* is independent of *T* given the other variables and will assign a zero coefficient to *S* given the true causes.

For large scale benchmark studies comparing all modern predictive modeling algorithms with causal algorithms see refs. [34, 35]. In these references, it is explained why various additional non-causal methods fail to model causality.

Implications for interpreting models. Not being able to condition on variables has two important model interpretation implications. The interpretability of the penalized models differs from unpenalized models. (a) In an (unpenalized) regression model, if a variable has non-zero coefficient, it is not conditionally independent of the target. However, in a regularized model, in general, having non-zero coefficient does not imply that the variable is not conditionally independent of the target.

(b) The broader implication is that because discovering causal structure and estimating effects require sound conditional independence tests and conditional association estimation, regularized regression and other purely predictive methods cannot be used for causal structure discovery or causal effect estimation even for simple questions (e.g. direct causal effect estimation) and even when the complete set of confounders is measured and are included in the model. Although unpenalized regression can be used for causal effect estimation, note that this has to be guided by knowledge of the causal structure (or elicitation of it using complex causal modeling algorithms).

See chapter "Foundations of Causal AI/ML" for more details and best practices.

Feature Selection and Dimensionality Reduction

Variable selection (aka feature selection) for predictive modeling has received considerable attention during the last three decades in a variety of data science fields [36, 37]. Feature selection and dimensionality reduction are techniques of choice to tame high dimensionalities in diverse big data applications. Intuitively, **feature**

selection for prediction aims to select a subset of variables for constructing a diagnostic or predictive model for a given classification or regression task. Ideally this selected set should include all variables with unique information and discard variables the information of which is subsumed by the selected set (since they add no information to the classifier). **Dimensionality reduction** maps the original data on a smaller number of dimensions so that fitting models is faster, less prone to overfitting and less sample intensive (all problems created by the high dimensionality and collectively known as "**Curse of Dimensionality**").

Feature Selection

Key concepts of the theory of feature selection were introduced in chapter "Foundations and Properties of AI/ML Systems". Here we will extend that material with a refinement of the types of feature selection problems typically addressed in practice, their relative complexity, examples, and algorithms that are used for feature selection.

Motivation and *Standard Feature Selection Problem*

In practice, reducing the number of features in clinical predictor models can reduce costs, and increase ease of deployment. Second, a more parsimonious model can be easier to interpret. Third, sometimes, identifying the most impactful features helps gain an understanding of the underlying process that generated the data. Finally, many classifiers do not perform well with very high dimensional data: they may overfit, exhibit too long compute times, or even fail to fit models.

Feature selection can be approached from three high-level theoretical perspectives. (1) The first one is that of overfitting/underfitting. In high-dimensional data sets, feature selection can reduce overfitting, but in lower dimensions if not conducted properly it can introduce both overfitting and underfitting (not both at the same time). It can underfit relative to a model that has more features (i.e., resulting model does not have enough capacity); and it can overfit if it is overly influenced by random variations in the data; a model with the same number of features could perform worse on the training set but perform better on a test set. (2) Feature selection in moderate or low dimensions can be used specifically to produce a parsimonious (and thus potentially more interpretable and practical) model, but this may induce underfitting if feature selection is not optimal. (3) Suboptimal feature selection can introduce instability when it selection of features is influenced by random perturbations in the data. To what extent such instability affects predictivity depends on whether the unstable features share the same information for the target or not. It is therefore important to deploy feature selection methods which are both theoretically sound and empirically strong.

The *standard feature selection problem* (chapter "Foundations and Properties of AI/ML Systems") is typically defined as:

> *Find the smallest set of variables St in the training data such that the predictive accuracy of the best classifier than can be fit for response T from the data is maximized*

A commonly used classification of feature selection considers three primary categories of methods: wrapper methods, filter methods and embedded methods [36]. We will further elaborate this taxonomy by considering whether the feature selection methods have a strong theoretical framework and properties (formal) vs not (heuristic).

Heuristic Algorithms

Wrapper methods. These methods conduct a heuristic search in the space of all subsets of the variables to assess each subset's predictive information with respect to the selected predictive model. Wrapper methods treat the classifier as a black box, fit a model to that data with a particular feature set and evaluate the model. Then they build a model with a different feature set and re-evaluate the new model. This process continues until some stopping criterion is met. This approach is computationally expensive, often overfits and underperforms, and also has great variation in performance depending on methods.

a. *Stepwise Regression*. Historically it has been used broadly for statistical regression modeling but its use is reduced because (a) standard implementations have been shown to overfit and are unstable [38] and (b) regularized regression has alleviated the need to use them substantially.

These methods can start from an empty model and use a forward single variable inclusion step, iterated with a backward single variable elimination step method until no improvement can be made. Backward elimination starts with a full model (a model contains all predictor variables) and eliminates one feature at a time. It eliminates the feature with least statistical significance (or equivalently, the one that improves the objective criterion the least). Stepwise feature selection stops when all features in the model achieve a certain level of significance (e.g. p-value of 0.05) and no other significant feature can be added. Alternatively, a penalized objective criterion (e.g. AIC—described later) can be used and the stepwise feature selection process terminates when this penalized objective is maximized.

b. *SVM-RFE* (recursive feature elimination) is an example of a more recent (and surprisingly powerful) wrapper method. SVM-RFE builds an SVM model and examines the contribution of features in the model. In each iteration, 50% of features with the lowest importance are eliminated, a model is re-fit and its accuracy evaluated in a test set. The process iterates recursively until predictivity drops. Due to SVM's resilience to high dimensions, SVM-RFE can produce a stable, highly accurate and non-overfitted set of features. What it typically lacks is minimality (although in practice it often selects parsimonious models).

c. *Bagging for feature selection.* In an attempt to improve the stability of feature selection methods and reduce their bias, bagging can be used. Models are constructed on bootstrapped versions of the training data using a base feature selection technique. Features that appear in some percentage (e.g. 50%) of the models are selected. The tendency of the feature selection method to select a feature because it randomly appears better than another can be mitigated by using multiple samples.

Univariate Selection Filtering. Filter methods select (or pre-select) features before the learning algorithm is applied to the data. Unlike wrapper methods, filter

methods do not use a predictive model for evaluating the features, but rather use statistical criteria to select features. The advantage of this approach is that it is agnostic of the learning algorithm and can be much more computationally efficient than fitting the model to the data.

Univariate variable screening (aka Univariate Association Screening, UAS or univariate association filtering, UAF) is the most commonly-used filter method for pre-selecting a set of variables that have significant association with the outcome at a predefined significance level; or in the case of UAF, variables are ranked based on their univariate association with the outcome and the top k variables are selected. Any common measure of association (e.g. correlation, signal-to-noise, G2, etc.) can be used. The rationale for univariate variable screening is that variables without a univariate association with the outcome are (often) not relevant to the outcome. A key advantage of variable screening is saving computational effort since the dimensionality of problems can be reduced early in the analysis.

Embedded methods. In embedded methods, the modeling technique itself incorporates a method to reduce the influence of irrelevant variables. Examples of this approach are regularization techniques such as SVMs, LASSO and similar methods. These methods and the mechanism by which they eliminate features is described in section "Regularization".

Feature Selector Algorithms Based on Formal Theories of Relevancy

The previously described feature selection methods are essentially heuristic because they do not utilize a principled framework for optimal feature selection. In the remainder we will discuss formal feature selection frameworks (i.e., Kohavi-John and Markov Boundary) and will describe algorithms that conduct provably optimal feature selection.

Kohavi-John and Markov Boundary framework for Standard predictive feature selection problem. Kohavi and John [37] decompose the standard feature selection problem as follows:

- A feature X is **strongly relevant** if removal of X alone will result in performance deterioration of an optimal Bayes classifier built on all data.
- A feature X is **weakly relevant** if it is not strongly relevant and there exists a subset of features, S, such that the performance of the Optimal Bayes Classifier fit with S is worse than the performance using S U {X}.
- A feature is **irrelevant** if it is not strongly or weakly relevant.

Intuitively, choosing the strongly relevant features provides the minimal set of features with maximum information content and thus solves the standard feature selection problem (since a powerful classifier in the small sample or the Optimal Bayes Classifier in the large sample) will achieve maximum predictivity.

Recall from chapter "Foundations and Properties of AI/ML Systems" that a set S is the Markov Boundary of variable T (S = MB(T)), if S renders T independent on every other subset of the remaining variables, given S, and S is minimal (cannot be reduced without losing its conditional independence property). This is the MB(T) in the probabilistic sense.

Tsamardinos and Aliferis connected the Kohavi-John relevancy concepts with BNs and Markov Boundaries as follows: In faithful distributions (see chapter "Foundations and Properties of AI/ML Systems," "Foundations of Causal ML") there is a BN representing the distribution and mapping the dependencies and independencies so that:

We will further elaborate on the nature of these problem types by explaining subtypes 3 and 10 in the next two figures. These explanations along with the material of chapter "Foundations and Properties of AI/ML Systems" and especially the

1. The strongly relevant features to T are the members of the MB(T).
2. Weakly relevant features are variables, not in MB(T), that have a path to T.
3. Irrelevant features are not in MB(T) and do not have a path to T.

Thus in faithful distributions: the Markov boundary MB(T) = solution to the standard feature selection problem.

Local causally augmented feature selection problem and Causal Markov Boundary. In faithful distributions with causal sufficiency (see chapter "Foundations of Causal ML") there is a causal BN that is consistent with the data generating process and can be inferred from data in which: strongly relevant features = members of MB(T) and comprise the solution to the local causally augmented feature selection problem of finding:

1. The direct causes of T
2. The direct effects of T
3. The direct causes of direct effects of T

Thus in faithful distributions with causal sufficiency the causal graphical MB(T) set is connected with the probabilistic MB(T). Inducing the probabilistic MB(T) then:

1. Solves the standard predictive feature selection problem, and
2. Solves the *local causally augmented feature selection problem.*

Equivalency-augmented feature selection problem and Markov Boundary Equivalence Class. In faithful distributions the MB(T) exists and is unique [39]. However, in non-faithful distributions where target information equivalences exist ("TIE distributions") we may have more than one MB(T) [40]. The number of Markov Boundaries can be exponential to the number of variables and in empirical tests Statnikov and Aliferis extracted tens of thousands of Markov boundaries before terminating the experiments [40].

In TIE distributions:

1. The Kohavi John definitions of relevancy break down since there are no Kohavi-John strongly relevant features any more, only weakly relevant and irrelevant ones. This is because if S1, S2 are both in the MB equivalence class {MBi(T)} then: *independent (S1, T | S2) and independent (S2, T | S1).*
2. *The 1-to-1 causal and probabilistic relationship of the probabilistic and graphical MB(T) breaks down.* A variable can be a member in some MBi(T) without having a direct causal or causal spouse relationship with T.
3. The standard predictive feature selection problem is solved by the smallest member in the equivalence class of MBi(T).
4. The *Equivalency-augmented feature selection problem* is to find the equivalence class of all probabilistic MBi(T).

These feature selection problem types can be further subdivided as shown in Fig. 14. The problem types depicted were chosen on the grounds that (a) applied ML papers often aim to solve them, and (b) proofs of feature selection soundness often use these subtypes as the goal. They are organized from simpler/lower complexity or hardness in the bottom, increasing while moving to the top.

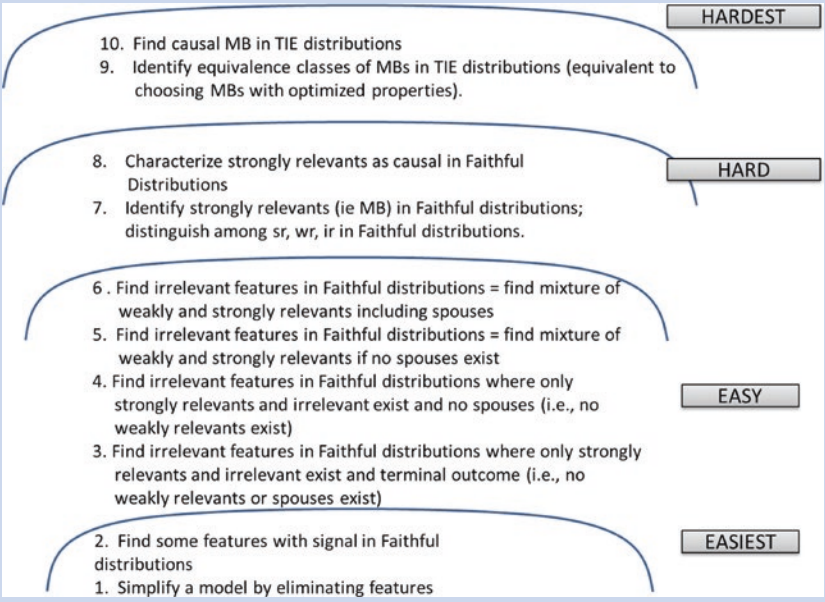


Fig. 14 A taxonomy of progressively harder feature selection problems. *From bottom to top of the figure, ten FS problem types of increasing complexity are depicted. Problems 1–5 are addressed with simple association criteria and can be tackled with regularized algorithms. Problems 6–7 correspond to the Standard Feature Selection Problem and require specialized algorithms. Problem 8 corresponds to the Causally-extended Standard Feature Selection Problem and requires specialized algorithms. Problems 9–10 correspond to the Causally-extended Standard Feature Selection Problem with Equivalence Classes and requires specialized algorithms.*

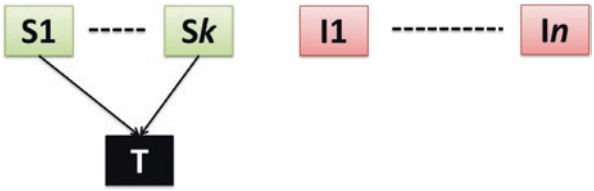


Fig. 15 Demonstration of FS Problem type 3 in the feature selection complexity taxonomy. The response variable is depicted in black. Green variables are the ones we seek to retain and red the ones to be discarded. Variables starting with *S* depict *strongly relevant* variables (i.e., cannot be discarded without loss of information—see chapter "Foundations and Properties of AI/ML Systems"). Variables starting with *I* depict *irrelevant* variables (i.e., can be discarded without loss of information—see chapter "Foundations and Properties of AI/ML Systems"). In domains with the data structure depicted and Faithful distributions, this problem is easily solvable by selecting all and only variables with non-zero marginal (i.e. univariate) association with the response. Regularized methods typically give zero weights to such irrelevant variables under these conditions

definitions of standard, causal and equivalency class problems, should make obvious their relevance to many real-life tasks (Figs. 15 and 16).

Modern Markov Boundary Algorithms—Faithful Distributions

We do not mention MB algorithms with only historical significance. For a review see [34]. Modern MB algorithms with guaranteed correctness, sample efficiency, and excellent empirical performance are instantiations of a broad family called GLL and include several HITON variants (HITON MB, HITON PC, interleaved or not, with symmetry correction or not, with additional wrapping or not, etc.), and MMMB and MMPC algorithms. They can be instantiated for recovery of full Markov boundaries or direct causal edges only. The IAMB family also exhibits sound and computationally efficient behavior in real data but is not as sample efficient.

These algorithms have been extensively tested and compared to all major feature selectors including wrappers, UAF, SVM-RFE, Lasso, LARS, LARS-EN, etc. See the benchmarks in [34, 41] for experiments covering in total > 120 algortithms, >270 dataset/tasks and multiple loss functions and data types. Consistent with the theory of feature selection the Markov Boundary algorithms in these benchmarks return the smallest feature sets and lead to classifiers with maximum predictivity. They also achieve near-perfect empirical discrimination among strongly relevant, weakly relevant and irrelevant features and are causally consistent.

Modern Markov Boundary Algorithms—TIE Distributions

Statnikov et al. [40, 42] invented the TIE* and iTIE* algorithhm families that can extract the full equivalence class of Markov Boundaries. The algorithms are correct and efficient. In typical usage they utilize GLL as subroutines . In [40, 42] extensive experiments are reported with comparisons to dozens of real and simulated datasets over multiple domains including comparisons with all availialbel comparators for feature selection equivalence class discovery. According to these benchmarks, MB

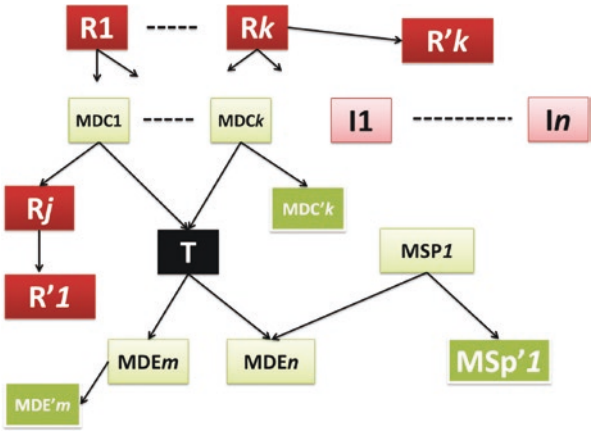


Fig. 16 Demonstration of feature selection Problem type 10 in the FS complexity taxonomy. The response variable is depicted in black. Plain green variables are the ones we seek to retain and differentiate. Red ones are to be differentiated (deep vs pale red) and discarded from predictive modeling. Variables starting with M depict Markov Boundary variables (cannot be eliminated without loss of predictive accuracy). An example MB is {MDC1, …, MDCk, MDEm, …, MDEn, MSP1}. Variables with names starting with MDC are members of the MB and direct causes of T. Variables with names starting with MDE are members of the MB and direct effects of T. Variables with names starting with MSP are members of the MB and direct causes of direct effects of T (i.e., "spouses" of T). Variables starting with I depict irrelevant variables (i.e., have no information about T and can be discarded without loss of information). Variables starting with R depict redundant variables (i.e., have information about T but should be discarded without loss of information if most compact models are needed). Variables with same name and the corresponding "prime" ones without prime names are equivalent in information content with respect to the response T (we only consider for simplicity contect-free equivalence [40]). A variable or variable set can have a large number of equivalent variables/sets (not depicted here for simplicity). By substituting equivalent variables, we obtain equivalent Markov Boundaries. For example MB {MDC1, …, MDCk, MDEm, …, MDEn, MSP1} is equivalent to MB {MDC1, …, MDC'k, MDEm, …, MDEn, MSP1}, and to {MDC1, …, MDCk, MDEm, …, MDEn, MSP'1}, and {MDC1, …, MDCk, MDEm, …, MDE'n, MSP1}, and so on. An exponentially large number of equivalent MBs can exist in a distribution. The feature selection/causal FS problem depicted requires highly specialized algorithms and cannot be solved with simple regularization or variable filtering

equivalence classes are common, their sizes varies across domains, and TIE* algorithms recover them with great accuracy.

We next provide a summary table (Table 3.) with properties of widely used feature selection approaches comparing their ability to tackle the feature selection problems 1–10.

We close this section with the method label of feature selection methods.

Table 3 *Comparative capabilities of current FS strategies and algorithms including simple univariate filtering, regularized regression*, SVM-RFE and Markov Boundary methods across the FS complexity categories of Fig. 10."+" = can solve, "-"cannot solve

	UAF	Regularized Regression Variants (e.g., Lasso, LARS, LARS-EN)	SVM-RFE	Markov boundary induction in faithful distributions (e.g., GLL, IAMB)	Markov boundary induction in tie distributions (e.g., TIE*)	Markov boundary induction and active experimentation in tie distributions (e.g., ODLP)
FS problem type 1	N/A (model-dependent)	+	+	N/A (model-dependent)	N/A (model-dependent)	N/A (model-dependent)
FS problem types 2–5	+	+	+	+	+	+
FS problem type 6	-	-	-	+	+	+
FS problem type 7	-	-	-	+	+	+
FS problem type 8	-	-	-	+	+	+
FS problem type 9	-	-	-	–	+	+
FS problem type 10	-	-	-	–	+	+

Method label: feature selection (FS)	
Main Use	• Finding a small set of variables that has all information about the response (FS)
Context of use	• Can be used to reduce the number of inputs to a classifier or regressor model so that: – Over fitting is avoided – Learning is faster – Deployment of a model is easier, faster, cheaper – The ML models are more understandable • Causal FS methods also reveal local causal structure around the response variable • Some learning algorithms have embedded FS (for example Decision Tree learning and Random Forests). In such cases adding formal FS algorithms often further enhances their performance
Secondary use	• Data simplification, compression and visualization • Clustering and subgrouping based on FS transforms of the data

Method label: feature selection (FS)	
Pitfalls	**Pitfall 3.5.1.2.** FS methods that are not designed for causal discovery cannot be interpreted causally and any estimates of causal effects will be biased **Pitfall 3.5.1.3.** FS itself can be over fitted to the data if model selection protocols used are not well-designed (see chapter on overfitting) **Pitfall 3.5.1.4.** FS like any other component of analysis needs be tailored to the data and problem. Using a default FS everywhere may lead to suboptimal results **Pittfall 3.5.1.5.** If a classifier has embedded FS, DR, or regularization does not mean that it cannot benefit from FS
Principle of operation	Highly-dependent on the specific FS method • Markov boundary FS is based on Bayesian Network theory and is additionally concordant with Kohavi-John FS theory in faithful distributions. In non-faithful distributions MB FS has strong advantages over Kohavi-John FS • RFE-SVM is based on fitting SVM models and performing wrapping over models with progressively smaller feature sets chosen based on the SVM weights • Univariate association filtering (UAF) rank-orders variables according to association with the response and chooses the top *k variables* • Wrapping is heuristic search over the space of all possible subsets, each one evaluated for a specific classifier and loss function of interest • Stepwise regression procedures originated in statistics and examine a series of regression models by iteratively including and discarding variables according to inclusion and exclusion criteria while conducting tests of statistical significance of model improvement at each step
Theoretical properties and empirical evidence	• Markov boundary FS accurately solves the standard FS problem by finding the smallest subset of variables that has all the information in the data about the response. Worst case computational complexity for inferring the MB is exponential to the number of variables but real-life complexity of best-of-class algorithms on common data is very efficient. In faithful distributions with causal sufficiency the Markov boundary solves a causal version of the standard FS problem: It finds the direct causes, directs effects and direct causes of direct effects of the response. Equivalence class MB induction recovers the whole set of MBs in the data. Excellent empirical performance in most domains • RFE-SVM is not guaranteed to find the optimal FS solution and is not causally valid. Computational complexity is low order polynomial. It is very robust to small sample size and has very good performance in many domains • Univariate filter selection (UAF) is not guaranteed to give the smallest set of variables with all information about the response. The top-ranked UAF variables do not need to be causally related to the response. Computational complexity is very small and sample efficiency is high • Wrapping is learner-specific, computationally intensive and tends to overfit. Not suitable for causal discovery, typically • Stepwise regression procedures are relatively fast but do not guarantee correct results and tend to overfit

Method label: feature selection (FS)	
Best practices	**Best practice 3.5.1.1.** Markov boundary procedures are first choice for FS when modest sample size (or more) is available and regardless of how high is the dimensionality. They are particularly appropriate when causal interpretation of findings is desired and when we wish to have consistent and coherent predictive and valid causal models. Also when we wish to find equivalence classes of optimal feature sets or optimal classifiers **Best practice 3.5.1.2.** SVM-RFE is a first choice in very small sample size and high dimensional/small sample when causal conclusions are not sought **Best practice 3.5.1.3.** UAF is common in genomics. Contrary to common over-interpretation by some researchers, the top ranked variables are not strongly suggestive of biologically/mechanistically/causally important or even valid factors. UAF has a place however when sample sizes are extremely small **Best practice 3.5.1.4.** Generic wrapping and stepwise procedures should be (and are increasingly) retired from practice
References	[36–42]

Dimensionality Reduction

As we mentioned earlier, the main objective of dimensionality reduction is to transform a high-dimensional space into a lower-dimensional representation. While feature selection achieves a lower dimensional space by keeping a subset of the original features without modifying the actual features, dimensionality reduction combines several of the original features into new features.

Dimensionality reduction techniques can be categorized as supervised vs unsupervised and linear vs nonlinear.

1. *Supervised versus unsupervised*: supervised techniques can use supervising information (such as outcome) to guide the dimensionality reduction. For example, linear discriminant analysis uses the class label to help project a multi-dimensional feature space into a lower-dimensional representation that maximally distinguishes among the classes. Unsupervised dimensionality reduction does not use outcome information. In this section, we focus on unsupervised dimensionality reduction; high-dimensional classification or regression are handled in other parts of this chapter.
2. *Linear vs nonlinear*. The transformation that reduces a high-dimensional space into a the lower-dimensional representation can be linear or non-linear. New features created by linear dimensionality reduction techniques are linear combinations of the original features, while non-linear dimensionality reduction uses nonlinear combinations. For example, autoencoders are arbitrarily complex non-linear transformations (they may increase or reduce the dimensionality), while classic Principal Component Analysis PCA is linear. Nonlinear dimensionality reduction is also known as manifold learning ([43], Chapter 20). Unsupervised dimensionality reduction has a vast literature; here, we focus on two classical approaches. For other popular dimensionality reduction techniques, the reader is referred to [44].

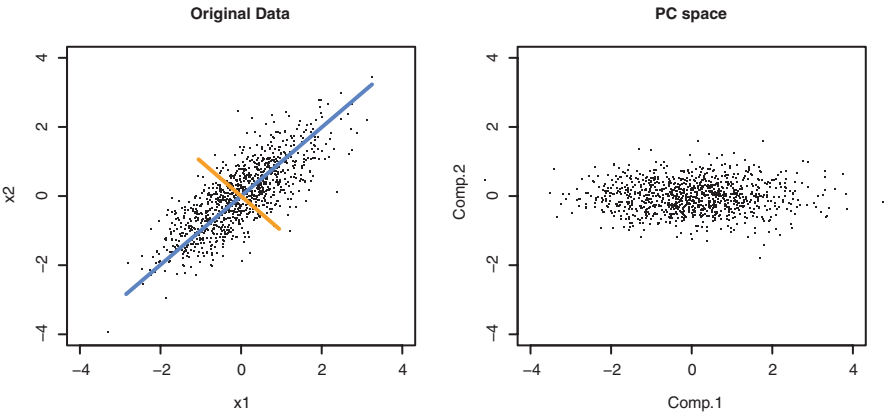


Fig. 17 Illustration of PCA. The left pane depicts a two-dimensional synthetic dataset. The blue and orange lines are the axes of the transformed space. The right pane depicts the same data set in the principal component space. The horizontal axis is the first and the vertical axis is the second principal component. The variance of the data is much higher along the first principal component than along the second

Principal Component Analysis (PCA)

Given a data matrix X, with columns as variables and rows as observations, find a matrix $U = [u_1, u_2, \ldots, u_p]$, such that (i) the u_i 's are orthogonal to each other and (ii) each subsequent **principal component** (or component for short) u_i, captures a maximal portion of the remaining variance.

Figure 17 depicts an illustration of PCA. The left pane shows the original Gaussian data. The variance of the data along both dimensions is approximately equal. PCA transforms this space into a new representation, the principal component (PC) space. Data in the PC space is depicted in the right pane. The horizontal axis corresponds to the first PC and the vertical axis to the second. As we can see, the variance (and thus information content) of the data along the first PC is much larger than along the 2nd. If we had to create a lower dimensional (i.e. one-dimensional) representation of the original data, we could choose the first PC as this new dimension, as it would capture much more variability than any of the original variables. In fact, among all linear combinations of the original variables, the first PC captures the highest amount of variance (under the constraint the total variances of the original and transformed space must equal).

Properties of PCA

1. The components computed by PCA are linear combinations of the original features. Thus PCA is a linear dimensionality reduction method.
2. Each vector u_i is an eigenvector of X^TX for centered X.
3. The i^{th} PC has variance λ_i, where λ_i is the eigenvalue corresponding to the i^{th} eigenvector.
4. $\sum \lambda_i$ is the total variance.

Exploratory Factor Analysis

The motivation behind factor analysis is that a (relatively) small number of unobservable **factors** can explain the observed variables. For example, "intelligence" is a quantity that is directly unobservable, thus it is measured through a battery of tests that is believed to be related to intelligence. In this example, the test results are the observations and intelligence is the latent factor.

Given an $m \times n$ observation matrix X consisting of n observations (columns) and m features (rows), we wish to explain these observations by p factors, then a factor model is of form

$$X - M = \Lambda F + \varepsilon$$

where M is the mean matrix containing the row means of X in its rows, Λ is the $(m \times p)$ **loadings** matrix, F is the $(p \times n)$ **factor** matrix, and ε is the error matrix $(m \times n)$ with mean 0 and finite variance.

Assumptions. We assume that

1. F and ε are independent
2. The factors in F are independent of each other
3. F is centered.

PCA can be viewed as a special case of factor analysis where Λ is orthogonal.

Method label: dimensionality reduction (DR)	
Main Use	• Computes a lower dimensional representation of a high-dimensional features space
Context of use	• Lower dimensional mapping can help with visualization • Can be used to reduce the number of inputs to a classifier or regressor model so that: – Overfitting is avoided – Learning is faster • May reveal structure properties of the domain • Some learning algorithms have embedded DR (for example deep learning and other ANNs) • Factor analysis: Estimate the values of unobserved factors through multiple observed variables
Secondary use	• Data simplification, compression and visualization • Clustering and subgrouping based on DR transforms of the data
Pitfalls	**Pitfall 3.5.2.1.** Some nonlinear methods can be very computation intensive **Pitfall 3.5.2.2.** DR does not reduce the number of inputs that need to be measured in order to deploy a model. Many expensive, dangerous (to measure) and unnecessary inputs discarded by FS will be needed by DR to be measured for model deployment **Pitfall 3.5.2.3.** DR methods that are not designed for causal discovery cannot be interpreted causally and any estimates of causal effects will be biased **Pitfall 3.5.2.4.** DR itself can be overfitted to the data if model selection protocols using it are not well-designed (see chapter "Evaluation") **Pitfall 3.5.2.5.** DR like any other component of analysis needs be tailored to the data and problem. Using a default DR everywhere may lead to suboptimal results **Pittfall 3.5.2.6.** If a classifier has embedded DR, that does not mean that it cannot benefit from FS or regularization

Method label: dimensionality reduction (DR)	
Principle of operation	• Greatly differs by the method • Generally, variables in the transformed representation are required to have some sort of independence of each other, and they collectively capture maximal amount of information across the full distribution • Embedded DR constructs lower-dimensional transforms of the original data in a way that is consistent with the inductive bias of the embedding learner
Theoretical properties and empirical evidence	PCA: • The number of PCs does not exceed the sample size • PCs are independent of one another • PCs cannot be interpreted as causal factors and loadings cannot be interpreted as causal effect sizes EFA: • It is a probabilistic model • PCA is a special case of EFA when loadings vectors are orthogonal • Causal interpretation of hidden factor effects on measured variables under strong assumptions
Best practices	**Best practice 3.5.2.1.** When eliminating expensive, dangerous and unnecessary inputs by predictor models is beneficial, then use FS instead of DR **Best practice 3.5.2.2.** For prediction of specific outcomes, FS targeting these outcomes should be the methods of choice **Best practice 3.5.2.3.** Using a top-2 PC data transform is a staple of data visualization for exploratory purposes **Best practice 3.5.2.4.** Both PCA and EFA should not be over interpreted causally, predictively or otherwise **Best practice 3.5.2.5.** PCA for classification can be overfitted, so it needs to be treated like any other data operation by the model selection and error estimation protocol
References	• For an overview, see chapter 20 in Murphy KP. Probabilistic Machine Learning: An Introduction. MIT Press, 2022 • Hinton, G.E. and Roweis, S., 2002. Stochastic neighbor embedding. *Advances in neural information processing systems*, *15*

Time-to-Event Outcomes

Survival data, (aka **time-to-event data**), describes the distribution of time until an event occurs. This **event** can be the failure of a device, incidence of a disease, a recurrence of a disease, an adverse event, or death. **Time** is the number of days, weeks, months, years, etc. from the beginning of follow-up until the event. Alternatively, it can also be calendar time such as the subject's age at the time of the event. We tend to think of events as negative, such as death (after all the field of survival analysis is named after studying survival time, the time to death), but it can also be a positive event, such as discharge from hospital. In the following, we use the terms "survival" and "time-to-event" interchangeably as long as context clarifies the use, and we also use the terms "event", "failure" and "death" interchangeably, unless this causes confusion.

Analytic tasks involving a time-to-event outcome are analogous to most other outcome distributions. The main tasks are (1) estimating the time-to-event (or the survival probability distribution S(t)); (2) testing whether two time-to-event distributions are statistically different; and (3) assessing whether one or more covariates (e.g. exposures) significantly affect the survival distribution.

The need for survival analysis. At first glance, time-to-event could be viewed as a continuous quantity and be modeled as one of the many known non-negative distributions, however, this approach breaks down for the following reasons. First, some subjects never experience the event of interest within the practical time frame of the study. Discarding these patients (with unknown time-to-event) leads to loss of information, because we know that these patients did not experience an event until the end of the study. In other words, time-to-event is not missing completely, it has been bounded. Second, some subjects are lost to follow-up before the study ends. Again, discarding such patients because their time-to-event is missing, discards useful information (i.e., that they had not experienced an event until the time they were lost to follow-up). Both of these situations are referred to as *right censoring* (see terminology section below). Third, in a study where the outcome is not death, many enrollees may have already experienced the event before enrollment. If this is allowed, cases with time-to-event = 0 can have high probability. Moreover, parametric distributions handle the general properties of time-to-event modeling poorly. As an example, fourth, outliers (extreme survivors) are common, and they have potential to become an *influential point* for some distributions. Also, fifth, many parametric distributions have parameters that mathematically relate to their moments (mean, variance). Censored data can make the estimation of moments on which model parameters depend, difficult, thus compromising the model.

Pitfall 3.6.1
In most practical settings, it is a significant pitfall to model time-to-event/survival using ordinary predictive modeling classification or regression.

Best Practice 3.6.1
When modeling time-to-event outcomes, specialized methods, such as the ones described in this section, should be used, at minimum as comparators with conventional techniques.

Terminology

Let T be a random variable with T_i denoting the time at which an event happened to subject i. Let $f(t)$ denote the density of T and let $F(t)$ denote the cumulative density of T. The cumulative density is referred to as the **failure** function and is defined as

$$F(t) = \Pr(T \le t) = \int_0^t f(\tau) d\tau.$$

The **survival** (or survivor) function is the complement of the failure function and is defined as the probability that a subject survives beyond a particular time *t*

$$S(t) = \Pr(T > t) = \int_t^{\infty} f(\tau) d\tau = 1 - F(t).$$

Properties of the survival function. The survival function is monotonic, non-increasing, equals 1 at time 0 and decreases to 0 as time approaches infinity. [45].

Often, instead of the survival function, we model the instantaneous "probability" of an event. The **hazard** function is the instantaneous "probability" per unit time that an event occurs exactly at time *t* given that the patient has survived at least until time *t*,

$$h(t) = \lim_{\Delta T \to 0} \Pr\left(T \le t \le T + \Delta T \middle| T > t\right)$$

Properties of the hazard function. The hazard function can be thought of as the "velocity" of the failure function or the rate of change in the failure function. Since the survival function is non-increasing, the failure function is non-decreasing and *h(t)* is non-negative. The hazard is not a true probability, it is a rate [45].

The **cumulative hazard** is

$$H(t) = \int_0^t h(\tau) d\tau.$$

The hazard and survival functions are linked to each other through the following relationship [46]. By taking the derivative of ln *S(t)*, we get

$$\frac{d \ln S(t)}{dt} = \frac{dS(t)/dt}{S(t)} = -\frac{f(t)}{S(t)} = -h(t),$$

which leads to

$$S(t) = \exp(-H(t))$$

Figure 18. shows the survival (left) and the hazard (right) functions for the diabetes dataset in [47]. The horizontal axis corresponds to the follow-up time (in years). For visualization purposes we show points (in grey color) on the actual hazard "curve". There is one point every follow-up day. The hazard estimates can change frequently in any direction as long as they remain non-negative. To further improve interpretability, a smoothed version of the hazard curve is also presented in black. The survival curve is a non-increasing step function starting at 1 at time 0 and ending at 0 at time infinity. It appears smooth in this figure because of the high resolution (daily) and large sample size, but it is nonetheless a step function. Note that the survival

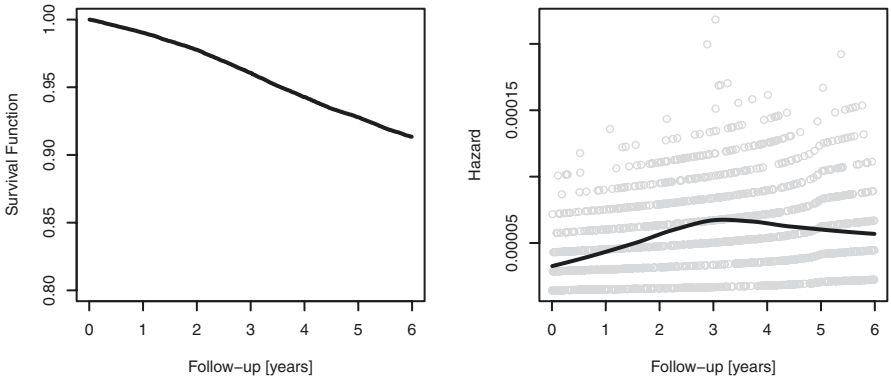


Fig. 18 Illustration of the Survival and Hazard functions. The left panel shows the survival function, while the right panel shows the smoothed hazard function for the diabetes data set in [47]

function relates to the *lack of* event (probability of *not having* an event), while the hazard function relates to experiencing an event (the rate of *having* an event).

Censoring

When a patient is lost to follow-up and is no longer observable, the time-to-event beyond the time of the patient dropping out cannot be observed. This is not a typical missing data problem as it first appears, because we have partial observations: the event did not occur while the subject was under observation. This partial observability is called **censoring**.

Left censoring happens, when the event takes place before the subject enters observation. We know that the event has already occurred at time 0, but we do not know when. **Right censoring** happens when the event takes place after the subject is no longer observed. We know that the event did not take place during the observation period but we do not know when/whether it occurred afterwards. Common reasons for right censoring are that the study ended, the subject is lost to follow-up or the subject withdrew from the study. Finally, **interval censoring** brackets the time of the event between two time points. We know that the event did not take place before the first time point and that it already occurred by the second time point.

Let C denote the time to censoring with density $g()$ and cumulative density $G()$. With $\tilde{T}$ denoting the true time-to-event, the subject's **follow-up time** T is $T = \min\left(C, \tilde{T}\right)$. Let δ denote the event type: $\delta = 1$ if an event took place ($\tilde{T} \leq C$); and $\delta = 0$ if the subject got censored ($\tilde{T} > C$).

Censoring is **random**, if $\tilde{T}_i$ is independent of C_i given X_i, where X_i is the covariate vector of observation i. Random censoring assumes that subjects who are censored at time t are similar in terms of their survival experience to the subjects remaining in the study. **Independent censoring** is a related concept. When a study has subgroups of interest, **independent censoring** is satisfied if censoring is random in all subgroups. **Uninformative censoring** happens when the distribution of C_i and $\tilde{T}_i$ do not share parameters [46, 48].

Competing risks arises when we have multiple outcomes of interest and the occurrence of one outcome prevents us from observing another outcome. As an example, consider heart disease and mortality as two outcomes of interest. If a patient dies (from a cause other than heart disease) we can no longer observe the patient's time to heart disease. In this case, we may have complete observation of the time-to-death, but we only have partial information about the time to heart disease: we only know that it is greater than the time-to-death.

Inference About Survival

In this section, we discuss methods to summarize the time-to-event distribution of a population. First, the time-to-event distribution can be summarized into a statistic (a single number) much in the same way as the mean or median summarizes aspects of a typical distribution. The fundamental difference is censoring: some subjects may not experience an event and thus their exact time-to-event is unknown. Next, we describe the time-to-event distribution as a function of time. We show methods to estimate the survival function and equivalently the cumulative hazard function. Finally, we present methods of constructing confidence intervals around the survival and cumulative hazard functions.

Summary Statistics of Survival

A concise way of describing the survival distribution is by presenting summary statistics. Often used summary statistics of common statistical distributions include the mean, the standard deviation, and the median. However in survival analysis, in the presence of censoring, it is desirable to account for the follow-up times when we compute summary statistics. Below, in Table 4, we describe some of the commonly used survival statistics [45].

Estimating the Survival Function

We present two estimators of the survival function: the Kaplan-Meier and the Nelson-Aalen estimator. They yield very similar results.

Table 4 Common statistics to summarize survival time distributions

Statistic	Definition	Remark
Average survival time	$\bar{T} = 1/N \sum_i T_i$.	Ignores censoring
Average hazard rate	$\bar{h} = \frac{\sum_i \delta_i}{\sum_i T_i}$	Uses hazard instead of survival to account for censoring
Median survival time	Survival time *t*, where $S(t) = 0.5$	Lessens the impact of outliers
k-year survival rate	Percentage of patients surviving k-years after their diagnosis [49]	Common choices for k include 5, 7, 10

The Kaplan-Meier estimator is more commonly used for estimating survival itself, and this is the preferred method for exploring and visualizing time-to-event data.

The Nelson-Aalen estimator, on the other hand, estimates the cumulative hazard function, and is mostly utilized by other methods, such as the Cox Proportional Hazards model.

Kaplan-Meier (Product Limit) Estimator

Let the index j iterate over the distinct time points t_j when an event took place. Let us assume that there are J such time points. The product limit formula is

$$\begin{aligned}\hat{S}\left(t_j\right) &= P\left(T > t_j\right) = P\left(T > t_j \middle| t > t_{j-1}\right) P\left(T > t_{j-1}\right) \\ &= \left[1 - P\left(T = t_j \middle| T > t_{j-1}\right)\right] P\left(T > t_{j-1}\right) \\ &= \left(1 - h_j\right) S\left(t_{j-1}\right),\end{aligned}$$

where h_j is the hazard at time t_j. Expanding this formula yields the Kaplan-Meier estimate

$$\hat{S}\left(t_j\right) = \prod_j \left(1 - \hat{h}_j\right) = \prod_j \left(1 - \frac{d_j}{n_j}\right),$$

where d_j is the number of events and n_j is the number of patients at risk at time t_j.

Nelson-Aalen Estimator

The Nelson-Aalen estimator estimates the cumulative hazard as

$$\hat{H}\left(t\right) = \sum_{j:t_j \le t} \frac{d_j}{n_j}.$$

The relationship between the cumulative hazard and the survival function can be used to estimate survival, yielding the **Breslow formula**

$$\hat{S}\left(t_j\right) = \exp\left(-\hat{H}\left(t\right)\right) = \prod_{j:t_j \le t} \exp\left(-\frac{d_j}{n_j}\right)$$

Comparison of the Kaplan-Meier and the Breslow (Nelson-Aalen) estimators (Fig. 19). Since $exp\ (-h_j) \sim 1 - h_j$ for small h_j, the Kaplan-Meier and the Breslow estimates are very similar and asymptotically equal. The Breslow estimate has uniformly lower variance but is upwards biased [46]. When ties are present in the data, the Kaplan-Meier estimate is more accurate. **Fleming and Harrington** proposed a

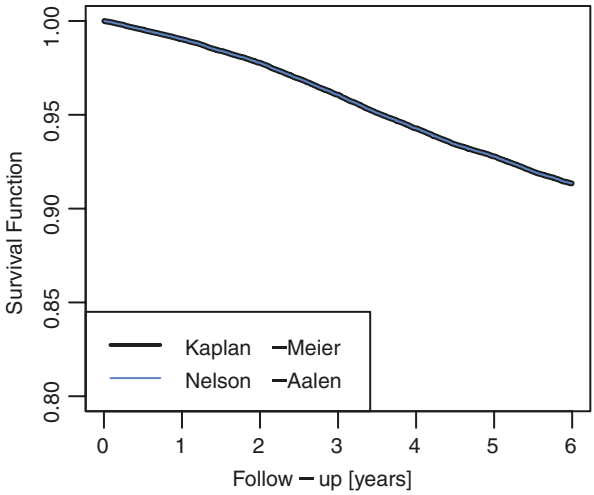


Fig. 19 The Kaplan-Meier and the Nelson-Aalen survival curves for the diabetes data set [47]. The two curves are so close that they are virtually indistinguishable

modification to the Breslow estimate by introducing a small jitter to break the ties in the follow-up times.

Confidence Intervals for the Survival Curves

Whenever conducting a survival analysis it is imperative to present confidence intervals (CIs). Statistical packages routinely offer such estimates. However when survival analysis is conducted with less conventional time-to-event modeling methods, often packages that implement these methods offer no facilities for CI estimation. We thus present here the fundamentals of estimating CIs for survival curves and hazard curves.

There are two fundamentally different approaches to constructing the confidence intervals and for each approach there are numerous variants. For brevity, in this section, we focus on one common method for directly estimating the confidence interval of the survival function. The interested reader is referred to Appendix 1 for the other methods.

Greenwood's formula. We consider constructing the confidence interval in survival space (as opposed to log survival or hazard space). The variance of the log survival function can be estimated using Greenwood's formula

$$\mathrm{Var}\left(\log \hat{S}(t)\right) = \sum_{j:t_j \le t} \frac{d_j}{n_j\left(n_j - d_j\right)},$$

where d_j and n_j are defined previously as the number of events at t_j and the number of patients at risk at time t_j, respectively. The *delta method* can be used to derive the

variance of the (non-log) survival function, which yields the **plain-scale** confidence interval

$$\hat{S}(t) \pm z \sqrt{\hat{S}(t)^2 \sum_{j:t_j \le t} \frac{d_j}{n_j (n_j - d_j)}},$$

where z is the normal quantile corresponding to the confidence level [46].

Method label: Kaplan-Meier (KM) estimator of survival curves	
Main Use	• Estimate survival curves • Visualize the survival curves
Context of use	• Non-parametric modeling • Predict survival probability at time t • The data does not meet the assumptions of more sophisticated (e.g., cox regression) survival modeling
Secondary use	• Checking the proportional hazards assumptions
Pitfalls	**Pitfall 3.6.1.1.** Estimating the effect of covariates is difficult. A separate curve is computed for each covariate combination. Does not scale to more than a very small number of covariates
Principle of operation	• Non-parametric estimator
Theoretical properties and empirical evidence	• In biomedicine it is practically expected and used in every publication involving survival
Best practices	**Best practice 3.6.1.1.** Plotting the KM curve can reveal data problems. Consider the complementary log-log plot of the KM curve
References	Recommended textbooks include [45, 46, 48]

Comparing Survival Curves

Comparing the estimated survival curves from two or more populations. Two survival curves are considered statistically equivalent when the data supports the hypothesis that these two curves are identical and any apparent difference between them is due merely to random variations in the samples that were used to estimate the curves.

In this section, we focus on the **log rank test.** Extensions of the log rank test are described in Appendix 1. Consider a group variable, which divides the population into G groups. At each unique event time, $j = 1,...,J$, the association between grouping and survival can be assessed. The null hypothesis is that the hazard at time t_j is the same across all groups for all j. The alternative hypothesis is that the hazard differs between the groups at at least one j.

Let n_{gj} denote the number of subjects at risk in group g at time t_j and let d_{gj} denote the number of failures in group g at time t_j. For simplicity, we concentrate on the

two-sample test, where $G = 2$. The expected number of failures in group 1 at time t_j is

$$e_{1j} = \frac{n_{1j}}{n_{1j} + n_{2j}}\left(d_{1j} + d_{2j}\right)$$

The observed number of failures across time in group g is $O_g = \sum_j d_{gj}$ and the expected number of failures is $E_g = \sum_j e_{gj}$. The **log-rank statistic** becomes

$$Z = \frac{\left(O_g - E_g\right)^2}{\mathrm{Var}\left(O_g - E_g\right)},$$

and the variance can be estimated from the hypergeometric distribution. Z follows a X^2 distribution with 1 degree of freedom and can be used as test of curve equivalence [48].

Cox Proportional Hazards Regression

Two important uses of regression models is to assess the effect of covariates on the hazard and to make predictions. Regression models we consider fall into two categories: semi-parametric and parametric models. **Semi-parametric** models, the Cox proportional hazards regression in particular, models the hazard as a product of a non-parametric **baseline hazard** function (which is a function of time) and (the time-invariant) **multiplicative effect** of the covariates. The covariates thus have a proportional (multiplicative) effect on the baseline hazard. **Fully parametric** models make a distributional assumption about the cumulative hazard (as a function of time) and model the parameter of this distribution as a linear additive function of the covariates. In this section, we focus on the Cox proportional hazard model (aka Cox model, Cox PH); fully parametric models will be discussed in the section "Parametric Survival Models".

The proportional hazards assumption. Fig. 20 illustrates the proportional hazards assumption using the diabetes example from [47]. The left panel shows the cumulative hazard of diabetes as a function of years of follow-up time. The orange curve in the plot corresponds to patients with impaired fasting glucose (IFG) and the blue line corresponds to patients with healthy glucose. At all time points, the ratio of cumulative hazard along the orange line versus the blue line is constant, 6.37. In other words, having IFG (versus not having IFG) confers a proportional, 6.37-fold, increase of diabetes risk upon the patients, and it remains constant across time. To translate this into the terminology of Cox models, the **baseline hazard** corresponds to patients without IFG (the corresponding covariate $x = 0$) and they have a time-dependent risk of diabetes depicted by the blue curve. Patients with IFG ($x = 1$), experience a risk (hazard) that is proportionally (6.37 times) higher across the entire timeline (orange curve).

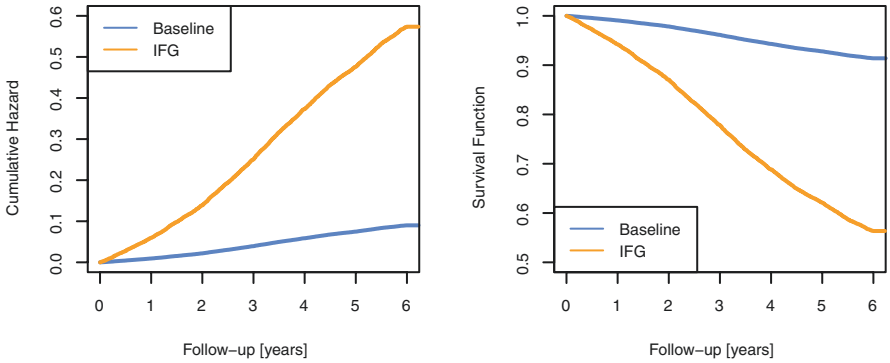


Fig. 20 Proportional Hazards Assumption. The left panel shows the cumulative hazard of patients with normal glucose (in blue) and impaired fasting glucose (IFG) (in orange) as a function of follow-up time (in years). The ratio of the underlying hazards of the orange line to the blue line is constant: the hazard along the orange line versus the blue line has the same proportion. The right panel transforms the cumulative hazard into survival probability

The Cox model. Let X be the covariate matrix, and let X_i denote the covariate vector for subject i. The hazard at time t is modeled as

$$h_i(t) = h_0(t)\exp(X_i\beta) \tag{1}$$

where $h_i(t)$ is the hazard of the ith subject at time t, $h_0(t)$ is the baseline hazard (common across all subjects) at time t, and β are regression coefficients. The cumulative hazards can be expressed as

$$H_i(t) = H_0(t)\exp(X_i\beta) = \exp(X_i\beta)\int_0^t h_0(\tau)\,d\tau$$

showing that the covariates increase (or decrease) the cumulative hazard proportionally relative to the baseline cumulative hazard. For additional details about the model (e.g. the partial likelihood function), see Appendix 1.

Assumptions.

1. The proportional hazards assumption: the covariates have a proportional (multiplicative) effect on the hazard relative to the baseline hazard.

   Consider two subjects, i and j, with covariate vectors X_i and X_j, respectively. The **hazard ratio** of these two subjects is

$$\frac{H_i(t)}{H_j(t)} = \frac{H_0(t)\exp(X_i\beta)}{H_0(t)\exp(X_j\beta)} = \frac{\exp(X_i\beta)}{\exp(X_j\beta)}$$

   and is constant with respect to time (the $\dfrac{(\exp(X_i\beta))}{(\exp(X_j\beta))}$ ratio does not depend on time). The name proportional hazards reflects the fact that the hazards of two patients are proportional to each other.

Continuing with the diabetes example, if patient *i* has IFG ($X_i = 1$) and patient *j* does not ($X_j = 0$), with $\beta = 1.85$, the hazard ratio is exp.(1.85) = 6.37. Therefore, the ratio of the hazards between the orange and the blue curves in Fig. 20 is 6.37.

2. Independence. Observations with an event are independent of each other. Only observations with an event are multiplied in the partial likelihood.
3. The effect of the covariates is linear and additive on the log-log survival.

Testing the Significance of the Covariates

Generally, in regression, we have two ways to test the significance of a coefficient. The first method is the **likelihood ratio test** and the second one is the **Wald test**. Although the proportional hazards regression maximizes a partial likelihood (as opposed to a full likelihood) as it leaves the baseline hazard unspecified, this does not affect the likelihood ratio test and both methods remain applicable.

Estimating the Baseline Hazard

Fitting a Cox proportional hazards model does not require the estimation of the baseline hazard. After the model has been fitted, the baseline hazardfunction is estimated using a variant of the Nelson-Aalen estimator that incorporates effects of covariates

$$\hat{H}_0(t) = \sum_{j:t_j \le t} \frac{\delta_j}{\sum_k R_k(t_j)\exp(X_k\beta)}.$$

where $R_k(t)$ indicates whether subject *k* is in the risk set at time t_j. Notice, that when $\beta = 0$, this reduces to the Nelson-Aalen estimatorfrom the "Terminology" section.

The variance of the baseline hazard is also based on the Nelson-Aalen estimator $\mathrm{Var}\left(\hat{H}_0(t)\right) = \sum_{j:t_j \le t} \frac{d_j}{\left(\sum_k R_k(t_j)\exp(X_k\beta)\right)^2}$.

Making Predictions

For an individual *i*, hazard can be estimated as

$$\hat{H}_i(t) = \hat{H}_0(t)\exp(X_i\beta)$$

and the corresponding survival can be computed using the Breslow estimator (see section "Estimating the Survival Function".)

$$\hat{S}_i(t) = \exp\left(-\hat{H}_i(t)\right)$$

Testing the Proportional Hazards Assumption

There are three methods for testing the proportional hazards assumption: (1) visual inspection, (2) formal statistical testing with time-dependent covariates, and (3) Schoenfeld residuals. We describe the first two methods and refer the interested reader to Appendix 1 for a more thorough discussion of the Schoenfeld residuals.

Visual Inspection

The first method is visual inspection of the log-log survival plot. Since under the proportional hazards assumption,

$$\hat{S}_i(t) = \exp\left(-\hat{H}_0(t)\exp\left(X_i\beta\right)\right),$$

its log-log transform is

$$\log\left(-\log\hat{S}_i(t)\right) = \left(\log\hat{H}_0(t)\right) + X_i\beta.$$

The log-log transform of two survival curves, corresponding to two different values of X_i, (say) x_1 and x_2, only differ in the $X_i\beta$ term, which is not a function of time t, thus the two curves should be parallel with a distance of $(x_2 - x_1)\beta$ between them.

To check the validity of the proportional hazards assumption, we plot the log-log transform of the Kaplan-Meier survival curves for two different values of X_i and expect these curves to be parallel.

A benefit of visual inspection is that we can see where (at what t) the violation of the proportional hazards assumptions happens and we may also see patterns that suggest the functional forms to correct the violation. *However, the decision whether the proportional hazards assumption is violated is subjective, no formal test is applied and hence no test statistic or p-value is obtained to guide the decision as to whether the proportional hazards assumption is violated.*

Figure 21 shows the complementary log-log plot of the diabetes data set. The two curves correspond to two levels of the covariate glucose status: the blue line shows

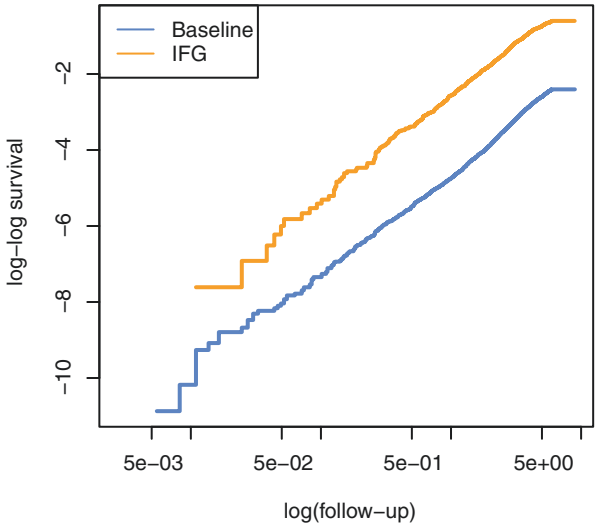


Fig. 21 Log-log survival plot of the diabetes dataset. The blue line corresponds to patients with healthy glucose levels, and the orange line to patients with impaired glucose levels. The log-log plots for the two levels of glucose status (normal versus impaired glucose) are parallel, suggesting that the proportional hazards assumption is acceptable

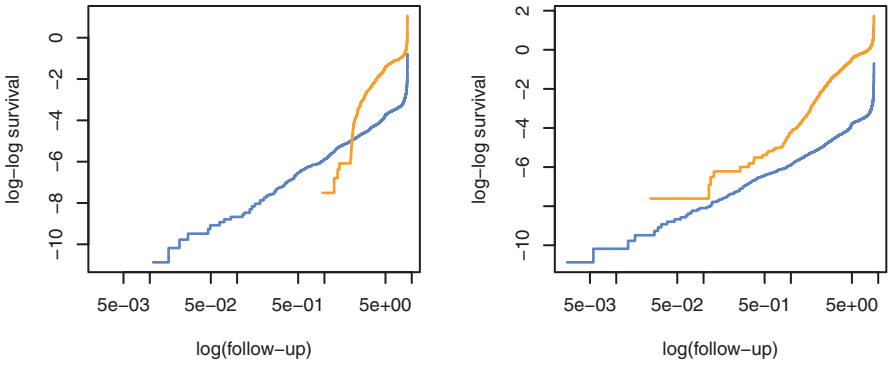


Fig. 22 *Violations of the proportional hazardassumption. The blue line is the baseline hazard*, while the orange line corresponds to some treatment. The left panel shows a violation where the treatment effect "switches over": while it is beneficial initially, it becomes harmful after some time. The right panel shows a violation, where the treatment line is a function of time. The curve suggests a function form (quadratic)

patients without impaired fasting glucose (IFG) while the orange line shows patients with IFG. Since the two curves are parallel, the proportional hazards assumption appears to hold for glucose status.

Figure 22 shows two synthetic examples where the proportional hazards assumption is violated. In both examples, the blue line represents the baseline hazardand the orange line corresponds to some exposure. In the left panel, the effect of the exposure changes from beneficial to harmful at about 2 years. In the right panel, the effect of the exposure (orange line) is quadratically related to (log) time.

Time-Dependent Covariates

The second method is based on time-dependent covariates. Under the proportional hazards assumption, adding regression terms involving interactions between the covariates and functions of time should not improve the fit. To check the validity of the proportional hazards assumption, we fit models of the form

$$h(t) = h_0(t)\exp\left(X\beta + \left(X \times g(t)\right)\gamma\right),$$

where $g(t)$ are vectors of function of time, $X \times g(t)$ are covariate-time interactions and γ is the coefficient vector of the covariate-time interaction terms. Under the proportional hazards assumption, we expect $\gamma = 0$.

A benefit of this method is that a statistical test is performed, a p-value is obtained, and thus the decision is objective. A weakness is the need for choosing an appropriate function $g(t)$. Different choices of $g()$ can lead to different conclusions. Common choices include the identity: $g(t) = t$; the log transform of time: $g(t) = log\ t$; and the heaviside function, where $g(t) = 1$ if t exceeds a threshold τ and $g(t) = 0$ otherwise.

In practice different functions of t are tested. The complementary log-log plot can provide hints as to the functional form of the violation.

Addressing the Violations of the Proportional Hazards Assumption

The consequences of violating the proportional hazards assumption are usually not dire. Violations do not usually affect the predictions, they mostly affect the error estimates.

Workarounds for the violations exist, however, they end up answering a question that is different from the original research question.

When the data set is large, violations are almost unavoidable. Thus depending on the extent of the violation and purpose of the study, we may opt to ignore the violation.

Suppose a test reports a proportional hazards violation, we start by verifying that the non-proportionality is substantial. Not all non-proportionalities are substantial. Statistically significant non-proportionality can arise from large sample sizes, where even small deviations from proportionality can become significant; or violation can arise also from influential points. The former can be ignored, the latter can be removed. To assess whether a non-proportionality is substantial, the key method is visualization. Not only can visualization show whether the non-proportionality is substantial, but it can also suggest a functional form to correct it.

For example, a formal test reports violations for the diabetes data set. However, inspecting the complementary log-log plot (Fig. 21) shows no violation of concern; the statistically significant violation is simply a result of the large sample size (54,700 patients) and is inconsequential to the analysis results.

Once we verified that the violation is substantial and decided to address it, we have several options.

1. The first option is **stratified Cox models**. If the covariate with the non-proportionality is a factor with relatively few levels, it can be used as a stratification factor in a stratified Cox model. The non-proportional effect now becomes part of the baseline hazard. If the covariate is a quantitative (continuous valued) variable, stratified Cox models can still be constructed, but the variable needs to be categorized (into a few categories) before it can be used as a stratification factor.
2. If the non-proportionality is present in a relatively short timeframe and not in the entire timeline, the **timeline can be partitioned into segments** in which the proportional hazards assumption holds and separate Cox models can be constructed in each time segment.
3. Finally, if the non-proportionality was detected through methods (2) or (3—see Appendix 1), using time or a **transformation of time**, *g(t),* adding an interaction term with the appropriate time-transformation can resolve the non-proportionality.

Method label: Cox proportional hazards regression	
Main Use	• Regression models for time-to-event outcomes
Context of use	• Right-censored data • Interest is the effect of covariates and making predictions • Same interpretability as classical regression models for other outcome types
Secondary use	N/A
Pitfalls	**Pitfall 3.6.3.1.** The key assumption is the proportional hazards assumption. Often, violation of the proportional hazards assumption is a non-issue, occasionally it can lead to problems **Pitfall 3.6.3.2.** The models assume linearity and additivity. Not appropriate if these assumptions are violated **Pitfall 3.6.3.3.** High dimensionality is a problem for the unregularized model
Principle of operation	• It is a semi-parametric regression model • The effect of covariates is a proportional (multiplicative) increase/decrease relative to a time-dependent baseline hazard • Coefficient estimates are obtained from maximizing a partial likelihood
Theoretical properties and empirical evidence	• Although a partial likelihood is maximized, the favorable properties of maximum likelihood estimation are preserved: Estimates are consistent, efficient and asymptotically normally distributed • Partial likelihood is convex and thus easy to solve
Best practices	**Best practice 3.6.3.1.** First-choice model for time-to-event data **Best practice 3.6.3.2.** Consider, additionally, whether the problem can be solved as a classification problem, or using survival modeling versions of ML predictive models **Best practice 3.6.3.3.** In the presence of substantial violations, different models, including extensions of the cox PH, may be more appropriate **Best practice 3.6.3.4.** Consider the Markov boundary feature selector for survival analysis that results from using cox proportional hazards models as conditional independence testing within the Markov boundary algorithm **Best practice 3.6.3.5.** For high-dimensional data, consider regularized cox proportional hazards models. Also consider the cox Markov boundary method described above **Best practice 3.6.3.6.** If age is included in the model and is nonlinear, consider an age-scale Cox PH model
References	Kleinbaum DG, Klein M. survival Analysis. A self-learning text. 2020, springer Therneau T, Grambsch P. modeling survival data. Extending the cox model. 2000, springer

Extensions of the Cox Proportional Hazards Regression

Several extensions to the Cox PH model have been proposed. In this section, we review some of them.

Stratified Cox Model

Stratified Cox models allow the population to be divided into different non-overlapping groups, called "strata". Each stratum has its own baseline hazard and each group may also have its own coefficient vector. The standard form of a stratified Cox models is

$$h_i(t) = h_{0k}(t)\exp(X_i\beta)$$

which assumes a common covariate effect across all strata that is proportional to the stratum-specific baseline hazard, $h_{0k}(t)$ for the kth stratum. The coefficients represent an "average" hazard ratio across the population (regardless of strata). This is the most flexible way of incorporating effects that violate the proportional hazards assumption, but stratified cox models offer no direct way of assessing the significance of the stratifying factor. An alternative form of the stratified Cox models considers the possibility of some covariates in a stratum (or some of the strata) having an effect that differs from its effect in other strata. Such effects are incorporated as interaction effects between the covariate and the stratum. If all covariates have interactions with the strata, then the resulting Cox model is the same as fitting separate Cox models for each stratum. Naturally, having to estimate separate baseline hazards and interaction terms requires sufficient sample size.

Recurring Events and Counting Process Cox Model

So far, time-to-event data was described by the triplet $\{T_i, \delta_i, X_i\}$, where T_i denotes the time to event, δ_i the event type (event or censoring), and X_i is the covariate vector. Alternatively, each subject's timeline can be divided into multiple segments and each segment can be described by a quartet $\{start_i, end_i, \delta_i, X_i\}$, where start_i and end_i are the two end points of the time segment, δ_i denotes whether an event occured in the time segment, and X_i is the covariate vector. This format is called the **counting process format**. Many applications of the counting process format exist, here we highlight a few.

The first application is the change of the time scale. The term time scale refers to the way time is measured. The triplet format measures time on the *study scale*, and, specifically, time 0 is when subjects entered the study. The counting process format allows for different time scales. For example, time can be measured as patients' age, where $start_i$ is the age when they entered the study and end_i is the age when they experienced an event. We discuss different time scales later in more detail.

Another commonly used application of the counting process format is **time-dependent covariate Cox models**. Time-dependent covariate Cox models allow for modeling under the assumption that the covariates can change over time. The time scale is divided into multiple segments and each segment can have its own covariate vector. As long as the subjects experience at most one event, the time-dependent covariate Cox model does not cause any complications, even though each subject can contribute multiple observations (rows). This stands in contrast to longitudinal data analysis (section "Longitudinal Data Analysis"), where observations from the same subject are correlated and this causes estimation issues. The key assumption to avoid such estimation problems is that the subjects have at most one event.

A third application of the counting process format is when subjects can experience multiple events. The timeline can be divided into multiple segments when subjects experience an event: resulting in a separate timeline for the first, second, etc. event. Now, each subject can enter the partial likelihood function multiple times. Several remedies exist. First, we can consider only the first event of all patients. Second, we can use longitudinal data analysis techniques. Analogues of both GEEs and mixed effect models exist for time to event outcomes. A third, commonly used option is to initially fit a model ignoring the correlation due to the possibly multiple observations per subject (with event) and then re-computing the error estimates, taking the correlation into account. Chapter 8.2.2 of [19, 46] describes three popular variations of this option in detail.

Age-Scale Models

The term **time scale** refers to the way time is measured for a time-to-event outcome. Typically, time is measured from a particular event, e.g. enrollment into the study, to the end of study. This is the **study time scale**. An alternative is **calendar scale**, where time is measured based on a calendar, e.g. the age of the participant.

Changing the time scale has two important effects. First, the risk sets are different. At first sight it may appear that age scale can be easily converted into a study-scale by $T_i = end_i - start_i$, however, the risk sets are different. Consider two patients. The first one enters the study at the age of 40 and suffers a heart attack (event of interest) at the age of 51. The second one enters the study at 55 and suffers a heart attack 5 years later at the age of 50. On the study-time scale, we have two events, one at 5 and one at 11 years. At the time of the first event, at year 5, we have a risk set of two patients. In contrast, on the age scale, we have two events, one at 51 and one at 60. At both events, the risk set contains only one patient. Since the risk sets are different, the survival estimates (or equivalently the hazard estimates) are different, as well. These two time-scales yield different results and admit different interpretations.

The second effect of age-scale relates to how age is entered into the model. One option is to use study-scale and add a covariate that represents age; and the other option is to use age-scale. In case of using age scale, age is modeled completely non-parametrically; the baseline hazard is a function of age. As such, the statistical significance of the age effect is difficult to assess. Conversely, when age is added as a covariate, the usual assumptions (linear, additive effect) apply and the baseline hazard is based on time in the study. Whether we use age-scale or study-time scale can also be determined based on whether the model assumptions about age as a covariate are reasonable.

Parametric Survival Models

The Cox proportional hazards model estimates the effects of the covariates first and then estimates the baseline hazard in a non-parametric manner. Non-parametric estimation typically requires more samples than parametric estimation.

Parametric models that model the time-dependent hazard (or equivalently, the survivor) curves in a fully parametric manner, can be more sample efficient if their assumptions are met.

In this section, we model the time-to-event variable T using parametric distributions. Consider X, a covariate matrix, β the regression coefficients and W is the error term. Rather than modeling T directly, we model its natural logarithm as

$$\log T = \mu + X\beta + \sigma W$$

In this model, μ is called a **location parameter**, σ is called the **scale parameter** and W is the error term. Similarly to linear regression, in parametric survival models, the coefficients have a linear effect on the location parameter of the distribution of $\log T$.

Principle of operation. Recall from section "Predictive Modeling Tasks", that in OLS regression with covariates X, outcome y, and error term ε, the model can be written as $y = X\beta + \varepsilon$. The error term is assumed to follow a normal (Gaussian) distribution, with location parameter (mean) $\mu = 0$ and scale parameter (standard deviation) σ. The covariates linearly affect the location parameter and the outcome thus have the same distribution as the noise, i.e. Gaussian, but with location parameter $\mu = X\beta$ and scale parameter σ (which remained unchanged).

Parametric survival models work analogously. The error term W is assumed to have a particular distribution with location and scale parameters μ and σ, respectively. The outcome $\log T$ then follows the same distribution as W, with location parameter $\mu + X\beta$. The model assumes that the covariates effect the location parameter linearly. The various parametric survival models differ in their choice of the distribution of W. We refer the reader to Appendix 1, which discusses several such distributions and the corresponding parametric survival model.

Property [Accelerated Failure Time (AFT)]. The covariates shift the location μ, which accelerates or decelerates the passing of time. This class of models is referred to as **accelerated failure time (AFT)** models. Let $S_0(t)$ denote the survival time distribution when all covariates are 0. The survival time distribution for a subject with covariates X is

$$\begin{aligned} S(t) &= \Pr(T > t) = \Pr(\log T > \log t) \\ &= \Pr(\mu + X\beta + \sigma W > \log t) \\ &= \Pr(\mu + \sigma W > \log t - X\beta) \\ &= \Pr\left(\exp\left(\mu + \sigma W\right) > t \exp\left(-X\beta\right)\right) \\ &= S_0\left(t \exp\left(X\beta\right)\right) \end{aligned}$$

The covariates, depending on the sign of $X\beta$, accelerate or decelerate the passing of time by a factor of $exp(-X\beta)$.

Figure 23 shows an AFT model fitted to the diabetes dataset. The outcome is diabetes-free survival, the horizontal axis is follow-up years. The orange line

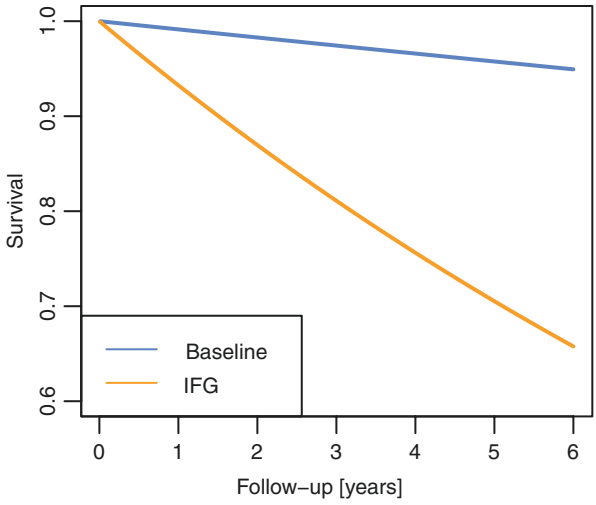


Fig. 23 Illustration of an accelerated failure time model on the diabetes data set

represents patients with impaired fasting glucose (IFG) and the blue represents patients with normal fasting glucose. Patients with normal fasting glucose have higher diabetes-free survival probability. If we draw a horizontal line at a particular (diabetes-free) survival probability, and compute the ratio of the time it takes to get to that probability along the blue line versus the orange line, we would find that this ratio is constant, exp.(−2.08) = 0.12 in this example. In other words, the time it takes for the diabetes-free survival to drop to a probability P is much shorter (takes 0.12 times as long) for patients with IFG than without.

Method label: accelerated failure time (AFT) models	
Main Use	• Regression models for time-to-event outcomes
Context of use	• Right-censored data • Interest is the effect of covariates and making predictions • Same interpretability as regression models for other outcome types
Secondary use	
Non-recommended Uses and Pitfalls	**Pitfall 3.6.4.1.** The key assumption is the accelerated failure time (AFT) assumption. Not appropriate if this assumption is violated **Pitfall 3.6.4.2.** The models assume linearity and additivity (location shift). Not appropriate if these assumptions are violated **Pitfall 3.6.4.3.** High dimensionality is a problem
Principle of operation	• Fully parametric model that specifies the full likelihood • The error term is assumed to have a location-scale distribution. This ensures that the log survival time has the same distribution. Covariates change the location parameter, accelerating/decelerating the passing of time

Method label: accelerated failure time (AFT) models	
Theoretical Properties and empirical evidence	• Parameter estimates are obtained using maximum likelihood estimation. They are consistent, unbiased, efficient and asymptotically normally distributed • AFT is a family of distribution with different properties Exponential survival model—Constant hazard assumption Weibull survival model—AFT and PH Log-logistic survival model—AFT and proportional odds assumption
Best practices	**Best practice 3.6.4.1.** Use AFT if the assumptions are met **Best practice 3.6.4.2.** Use cox PH if only the PH assumption is met
References	KleinJP, Moeschberger ML. SURVIVAL ANALYSIS techniques for censored and truncated data. 2003, springer Kleinbaum DG, Klein M. survival Analysis. A self-learning text. 2020, springer

Parametric Survival Models Versus Cox PH Models

If the model assumptions of the parametric models are met, the parametric models are more sample efficient. If the assumptions are not met or if we are in doubt, the semi-parametric model is more robust to model misspecification and only requires the proportional hazards assumption.

Appendix 1 describes method to check the appropriateness of various parametric survival models. In this section, we show one example, comparing the fit from a Weibull model (a particular type of parametric survival model) with Cox PH model.

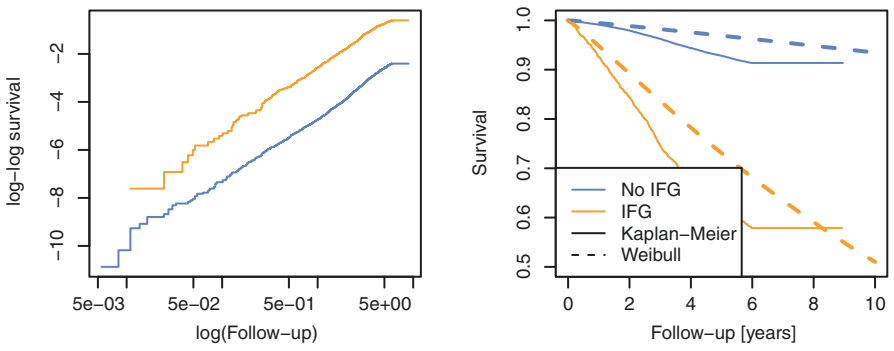


Fig. 24 Weibull survival model on the diabetes data set. The right panel shows the complementary log-log survival curve. The orange line corresponds to patients with IFG and the blue line without. The right panel shows the survival curves. The solid lines are estimated using the Kaplan-Meier estimator, while the dashed lines are computed from a Weibull model (see Appendix 1 for details). Orange corresponds to patients with IFG, while blue corresponds to patient with healthy fasting glucose

The left panel in Fig. 24 shows the complementary log-log plot of the diabetes data set. We continue to use impaired fasting glucose (IFG) as the sole covariate and the two survival curves were computed using the Kaplan-Meier estimator. The two lines corresponding to the two values of this covariate, IFG in orange and non-IFG in blue, are reasonably straight and parallel for the first 6 years. As shown in Appendix 1, the curves being parallel indicates that the proportional hazards assumption holds. If the curves are straight the AFT assumption holds. Beyond 6 years, the curves turn and become horizontal. They remain parallel but they no longer continue to have a constant slope. The turn signals a violation of the AFT assumption, however, they remain parallel, indicating that the PH assumption is still met. This appears to be a small violation, however, a large portion of the population have a follow-up time in excess of 6 years.

The right panel in Fig. 24 shows the Weibull fit (in dashed lines) and the Kaplan-Meier survival curve (in solid line) for the IFG patients (orange) and non-IFG patients (blue). We can see that the lack of events beyond 6 years caused a substantial bias in the Weibull estimates. We expected this bias based on the violation of the AFT assumption. Since the PH assumption is still met, a Cox model would be a better fit for this data.

Non-Linear Survival Models

The regression models in the previous sections all assume that the covariates have a linear (additive and proportional) relationship with the log hazard or log survival time. To overcome this limitation, the original features X can be transformed through a non-linear non-additive transformation to serve as the input to the partial or full likelihood function of the above models. Deep-learning based survival models and the Gradient Boosting Machine (GBM) for time-to-event outcome have taken this approach. The $X\beta$ term in the Cox partial likelihood is replaced by a non-linear non-additive function $f(X)$. This function is an ANN for deep learning and a GBM for Cox GBM.

A Random Survival Forest (RSF) consists of a collection of B trees. This collection does not directly maximize a likelihood function like the previously discussed methods, so RSF works slightly differently. In RSF, each of the B trees models the cumulative hazard of a patient using the Nelson-Aalen estimator. The cumulative hazard estimates from the B trees are then averaged to obtain an overall prediction for the cumulative hazard [50].

One key in time-to-event modeling is censoring. The partial likelihood automatically takes censoring into account, but the full likelihood may not. Deep learning models based on the full likelihood, assuming a Weibull distributed survival time, have been proposed. An alternative to the partial likelihood in the presence of censoring is the censoring unbiased loss (CUL), which is a general method for bias-correcting the unobservable loss. Censoring unbiased deep learning (CUDL) follows this strategy [51, 52].

High-dimensional data. Similar to non-survival regression models, high dimensionality, when the number of predictor variables is large relative to the number of observations, poses a challenge. In non-survival regression, regularizing the likelihood function was one of the solutions. Analogous solutions by regularizing the partial likelihood function of the survival models has been proposed in the form of an elastic-net style Cox model.

Survival models for longitudinal data. When we have longitudinal data, the covariates and the outcome can change over time. We have already discussed extensions to the Cox model that allow for changing predictors (time-dependent covariates) and recurring events. In the general regression setting, longitudinal data is handled through marginal models or through mixed effect models, because the observations become correlated. We have also discussed that in the Cox model, as long as we only have one event per patient, marginal or mixed effect models are not required [46].

Apart from providing the correct error estimates in the longitudinal setting, mixed effect models are also used for separating subject-specific and population effects. Frailty models are the time-to-event outcome analogues of the mixed effect regression models and allow for separating subject-specific effects and population effects.

Longitudinal Data Analysis

Longitudinal data is generated when measurements are taken for the same subjects on multiple occasions. For example, EHR data of patients is longitudinal as the same measurements, e.g. vitals, are taken at multiple encounters. Longitudinal data stands in contrast with single cross-sectional data, where measurements are taken (or aggregated) at a single particular time point. It also contrasts with **time series** data, where measurements are taken for a single subject (or for few subjects) for a long period of time and inference is conducted within the subject.

Using longitudinal data offers several advantages. (1) It can provide more information about each subject than data from a single cross-section since we observe the subject over a time span. (2) It also allows for a crossover study design, where a patient can be a control patient for himself: When a subject experiences an exposure during the study period, he/she is a "control" subject before the exposure and is an "exposed" patient after the exposure. (3) it also allows for separating aging effects from intervention effects. Finally, (4) it allows for separating subject-specific effects from population effects [5, 53].

Figure 25 shows an illustrative synthetic data set. Five subjects are followed over 10 time periods and a measurement is taken in each time period. The left panel shows a plot of the data set. The horizontal axis represents time, and the vertical axis is the measurement. We can see an overall upward trend: as time increases the measured values increase. We fitted a linear regression model to the entire data, which is shown as the bold black line. This model is a **population-level** model and it confirms this increasing trend. We also fitted a regression line, shown as dashed gray

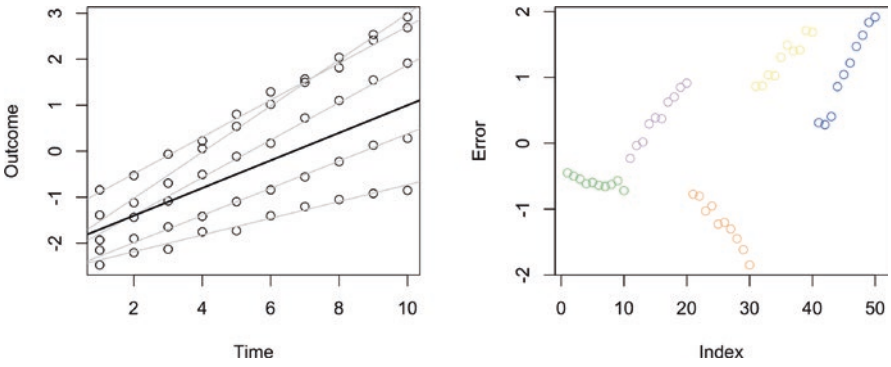


Fig. 25 Longitudinal Data Illustration. Five subjects are followed over 10 time periods and a measurement is taken in each time period. The left panel shows a plot of the data set. The horizontal axis represents time and the vertical axis is the measurement. The bold black line depicts an overall trend (population trend) and the 5 dashed lines represent the (individual) trends of the five subjects. The right panel shows the error relative to the population trend. The horizontal axis is an index, grouped by subject. Different colors represent different subjects

lines, to each individual subject. These are called **individual-level** lines. We can see that most (all five in this sample) subjects also exhibit an increasing trend, but their initial points (y-intercepts) vary, and their slopes also vary. Some methods allow for modeling individual effects such as the per-subject intercept and per-subject slope.

These advantages of longitudinal data analysis, however, come at a price. The multiple observations of the same patients are correlated with each other, which violates the i.i.d. (independent, identically distributed) assumption that most analytic methods make.

The right panel in Fig. 25 shows the error relative to the population-level regression model (the bold black line in the left panel). The horizontal axis is simply an observation index and the vertical axis is the (signed) error (residual). Observations from the five subjects are grouped together along the horizontal axis in increasing order of time: index 1–10 corresponds to the 10 time points of the first subjects, etc. Different subjects are depicted in different colors. We can see that the errors of each subject (errors depicted in the same color) tend to form clusters. Within a subject, once we know the error of one observation, errors of the other observations will typically not differ as much as errors from a different subject. This means that *errors of the same subject are correlated with each other*. There is also a trend within each subject: as time increases, the errors tend to increase or decrease. This is due to the differences in the growth rates of the different subjects (the differences across the slopes of the gray lines in the left panel).

When we assume that the errors in the right panel are generated from 50 independent observations, we would estimate the variance of the outcome to be about 1 (ranging between -2 and 2). Once we account for the fact that the observations came from 5 different subjects, the spread of the error becomes the range covered by the same color, and the variance becomes approximately 0.57; and after accounting for the differences in individual growth rates, the error variance drops to

(approx.) 0.1. Such reduction in the noise variance leads to much improved estimates and is very beneficial for detecting significant effects from exposures.

The data is **balanced** when measurements for all subjects are taken at the same time points.

When the data is balanced, coefficient estimates, whether they are computed using methods for longitudinal data or for cross-sectional data, will be similar albeit with substantially different errors. If the purpose of the analysis is prediction for previously unseen subjects, no individual effect estimates will be available, thus the results obtained from the regular regression models will be very similar to those obtained from the longitudinal models.

Conversely, when the design is not balanced, methods specifically designed for longitudinal data should be used. Also, when the significance of the coefficients needs to be estimated, or estimating errors is important, or individual (within-subject) effects are of interest, or if predictions are to be made for previously seen patients (whose individual effect sizes are already estimated), methods specifically designed for longitudinal data should be used (regardless of whether the design is balanced or not).

As we mentioned earlier, the key drawback of using longitudinal data is the correlation among the observations of the same subject. All methods in this section address this correlation. Moreover, linear mixed models (LMMs) can additionally model within-subject variability, while generalized estimating equations (GEE) offer improved coefficient estimates at lower computation cost relative to LMMs. Both of these techniques are described in later sections.

Terminology and Notation

The sampling unit of the analysis is a subject or a patient and we index the sampling units $i = 1, \ldots, N$. The analytic units are observations. Each patient can have multiple observations, indexed by $j = 1, \ldots, n_i$, taken at n_i different occasions (time points). The time of these occasions are denoted by t_{ij}, the time of the jth occasion for the ith patient.

The design is **balanced**, if all subjects share the same time points.

Let y_{ij} denote the response variable (of patient i at occasion j) and let X be covariates. The covariates for subject i can be time-invariant (constant across time) or they can vary across time (a situation referred to as **time-varying covariates**). The vector of time-invariant covariates for subject i is denoted by X_i and the vector of time-varying covariates from subject i at occasion j is denoted by X_{ij}.

Random effects are effect estimates that are computed for observation units that are thought of as a random sample from a population. In contrast, **fixed effects** are effect estimates computed for specific observation units. The within-subject effects are random effects, because the corresponding units, namely the subjects, are thought of as a (hopefully) representative random sample (the discovery cohort) from a population of patients. We could have conducted our study with a different random sample from the same population and we would expect similar results. Conversely, the time effects are fixed effects, because we wish to know the effect of a specific time period j on the outcome. The time points are not a representative random sample from a population of time points, they represent periods of exposure to the intervention. If we conducted our study using different time periods, say 2 months exposures as opposed to 2 days, we would certainly expect to get different results.

The questions we ask about longitudinal data are similar to and are a superset of the questions we ask about cross-sectional data. These questions include:

1. Are two sets of observations (y_{i1}, y_{i2}, …, y_{in} and y_{k1}, y_{k2}, …, y_{kn}), one for patient i and the other one for patient k, different?
2. Are observations at different time points j and k different ($y_{.j} =^{?} y_{.k}$)? Or more broadly, describe the changes in observations over time.
3. Making predictions. We may wish to predict the value of the observation at a particular time point for a subject we have observed before; or we may want to predict the value of an observation for a subject that we have not seen before.
4. Estimate the effect of exposures.
5. Estimate subject-specific effects.

ANOVA and MANOVA for Repeated Measures

Before the advent of more advanced and flexible analysis methods, repeated ANOVA and MANOVA were the first-choice methods for analyzing repeated measures data. In this chapter, we focus on the more advanced methods (which subsume ANOVA and MANOVA), and detailed discussion of ANOVA and MANOVA are presented in Appendix 2. Given their historic importance and hence presence in the health sciences literature, we still provide method labels for them below.

Method Label: Repeated Measures ANOVA	
Main Use	• ANOVA for repeated measures data
Context of use	• Single-sample or multiple-sample ANOVA • Assumes the data to be in the PP (person-period) format • Assessing the significance of time effects and treatment effects
Secondary use	
Pitfalls	**Pitfall 3.7.2.1.** Repeated measures ANOVA is not a predictive model **Pitfall 3.7.2.2.** Repeated measures ANOVA assumes compound symmetry; not appropriate when this assumption is violated
Principle of operation	• Operates on the same principle as most ANOVA methods • See Appendix 2 for detailed models

Method Label: Repeated Measures ANOVA	
Theoretical properties and empirical evidence	• Requires balanced design • Assumes the compound symmetry • Performs statistical tests of time effect and treatment effects • Contrasts can be used to perform specific tests (e.g. difference between two treatment levels)
Best practices	**Best practice 3.7.2.1.** Also consider the random intercept LMM. The LMM is more flexible and contains the ANOVA specification as a special case
References	• Hedeker D, Gibbons RD. Longitudinal Data Analsyis. Wiley, 2006. Chapter 2

Method label: repeated measures MANOVA	
Main Use	• MANOVA for repeated measures data
Context of use	• Single-sample or multiple-sample MANOVA • Assumes the data to be in the PL (person-level) format • Assessing the significance of time effects and treatment effects
Secondary use	
Pitfalls	**Pitfall 3.7.2.3.** Repeated measures MANOVA is not a predictive model **Pitfall 3.7.2.3.** Repeated measures MANOVA in its original form, does not allow for missing observations
Principle of operation	• Operates on the same principle as most ANOVA /MANOVA methods • See Appendix 2 for detailed models
Theoretical properties and Empirical evidence	• Requires balanced design • In contrast to ANOVA, it does not make the compound symmetry assumption, but it does not allow missing values • Performs statistical tests of time effect and treatment effects • Contrasts can be used to perform specific tests (e.g. difference between two treatment levels)
Best practices	**Best practice 3.7.2.2.** Also consider LMMs
References	• Hedeker D, Gibbons RD. Longitudinal Data Analsyis. Wiley, 2006. Chapter 3

Linear Mixed Effect Models

The key difference between methods developed for longitudinal data and for cross-sectional data lies in their ability to take within-subject correlations into account. Linear Mixed Effect Models (LMM), the subject of the present section, aim to partition the variance-covariance matrix into within-subject and between-subject variances.

> If differentiating and estimating within-subject versus between-subject variance is of interest, then Linear Mixed Effect Models should be used.

Model Specification and Principle of Operation. Regular regression models model the outcome as a combination of deterministic "fixed" effects and a random noise

$$y_i = \beta_0 + X_i\beta + \varepsilon_i,$$

where β_0 is an intercept, β is a vector of coefficients for the fixed effects imparted by the covariates X_i and ε is a normally distributed noise term with mean 0 and variance σ^2.

Mixed effects regression models, similarly to regular regression models, allow for fixed effects, but they further partition the "noise" into different anticipated random effects. Different types of LMM models differ in the random effects they anticipate, which in turn, confers different structures on the variance-covariance matrix.

Let the subscript i correspond to the subject and j to the (index of) the occasion when the subject was observed. Let X_{ij} denote the covariate vector and y_{ij} the response of subject i at occasion j. The time point of this occasion is t_{ij}.

Mixed effect models are often expressed in the hierarchical format. The **first-level model** is on the *level of the population*

$$y_{ij} = \beta_{0i} + X_{ij}\beta + t_{ij}\beta_{ti} + \varepsilon_{ij}$$

and the **second-level** (*subject-level*) models define the models for the (subject-specific) intercept β_{0i} and (time) trend β_{ti} for subject i. Mixed effect models are a family of models that chiefly differ in the way β_{0i} and β_{ti} are defined.

Assumptions. Different definitions lead to different variance-covariance matrices based on different assumptions, however, all mixed effect models share some common assumptions.

1. As in all linear models, the fixed effects, X_{ij}, are assumed to have a linear (additive and proportional) relationship with y_{ij}. This assumption can be relaxed by including a priori known interactions and nonlinearities.
2. Time enters the mixed effect models explicitly (t_{ij}). This allows for observation times to vary across subjects. In many models, time has a linear additive effect on the response, however, models with curvilinear relationships will be discussed later.
3. The structure of the variance-covariance matrix is specified through a random intercept and/or trend. This allows for the dimension of the variance-covariance matrix to vary across patients, which in turn, allows for a differing number of observations across subjects. The second and third properties combined make mixed effect models appropriate for the analysis of longitudinal data that is not of repeated measures design (observation times vary) or for repeated measures design with missing observations.
4. Models in this chapter assume an outcome with Gaussian distribution, but mixed effect models have been extended to the exponential family outcomes through a linkage function that linearizes these outcomes. These models, Generalized Mixed Effect Models, are the mixed-effect analogues of GLMs.

In the following sections, we describe specific mixed effect models, their assumptions, relationships between covariates, time and outcome they can represent, and the variance-covariance matrix forms these assumptions yield.

Random Intercept Models

Random intercept models are mixed effect models with a subject-specific random intercept effect but only with a population average trend effect. The second level models are thus

$$\beta_{0i} = \beta_0 + \upsilon_i$$

$$\beta_{ti} = \beta_t$$

The subject-specific intercept β_{0i} is decomposed into a population average effect β_0 and a subject-specific random effect υ_i. The time effect β_{ti} is simply the population average trend (slope) β_t (without a subject-specific random effect). Thus, the random intercept model decomposes the "noise" into a subject-specific random effect υ_i and the actual noise at the jth occasion ε_{ij}.

It is further assumed that

$$\upsilon_i \sim N\left(0, \sigma_\upsilon^2\right)$$

$$\varepsilon_i \sim N\left(0, \sigma_e^2\right)$$

This yields a block-diagonal variance-covariance matrix. Each block corresponds to a subject and is of the form

$$\Sigma_i = \begin{bmatrix} \sigma_v^2 + \sigma_e^2 & \sigma_v^2 & \sigma_v^2 & \cdots & \sigma_v^2 \\ \sigma_v^2 & \sigma_v^2 + \sigma_e^2 & \sigma_v^2 & \cdots & \sigma_v^2 \\ \sigma_v^2 & \sigma_v^2 & \sigma_v^2 + \sigma_e^2 & \cdots & \sigma_v^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_v^2 & \sigma_v^2 & \sigma_v^2 & \cdots & \sigma_v^2 + \sigma_e^2 \end{bmatrix}$$

This form of variance-covariance matrix is referred to as **compound symmetry**. It assumes that the covariance between observations of the same subject are constant over time. This is often unrealistic: observations closer to each other in time are typically more correlated than observations further away in time.

Random Growth Models

Random growth models, in addition to the subject-specific random intercept, also have a random slope for time. This allows (i) for changes (slopes) to vary across subjects and (ii) for time to enter the variance-covariance matrix. The second-level model is

$$\beta_{0i} = \beta_0 + \upsilon_{0i}$$

$$\beta_{ti} = \beta_t + \upsilon_{ti}$$

Similarly to the way the intercept was decomposed into a subject-specific effect υ_{0i} and a population-level effect β_0 in the random intercept model, in the random growth model the time effect is also decomposed into a subject-specific effect υ_{ti} and a population-level time effect β_t. It is assumed that

$$\upsilon_{0i} \sim N\left(0,\sigma^2_{\upsilon_0}\right), \upsilon_{ti} \sim N\left(0,\sigma^2_{\upsilon_t}\right)$$

$$\varepsilon_i \sim N\left(0,\sigma^2_e\right)$$

With subjects i and k being independent, the variance-covariance matrix is block-diagonal, with each block representing a patient and taking a form of

$$\Sigma_i = \sigma^2_e I + T_i \Sigma_\upsilon T_i^T$$

where

$$T_i^T = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ t_1 & t_2 & \cdots & t_{n_i} \end{bmatrix}$$

and

$$\Sigma_i = \begin{bmatrix} \sigma^2_{\upsilon_0} & \sigma_{\upsilon_0\upsilon_t} \\ \sigma_{\upsilon_0\upsilon_t} & \sigma^2_{\upsilon_t} \end{bmatrix}.$$

With time entering the covariance matrix, the covariance among the observations of the same patient can change over time.

Polynomial Growth Model

To model non-linear time effects, the level-1 model can be extended with polynomials of time.

Specifically, in vector notation, it becomes

$$y_i = \beta_{0i} + X_i\beta + T_i\upsilon_i + \varepsilon_i.$$

where T_i contains polynomial of t_i. To be able to model a quadratic time effect, T_i would be

$$T_i = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & \vdots & \vdots \\ 1 & t_{n_i} & t_{n_i}^2 \end{bmatrix}.$$

Comparison of the Various Model Assumptions

Figure 26 illustrates the difference among the three model types. Four synthetic data sets were generated using four different assumptions. In all four data sets, five subjects were observed at 10 time points. The four data sets are plotted in the four panels. For all four panels, the horizontal axis is the index j of the observations, grouped by subject. Since the key issue in longitudinal data is partitioning the errors (based on these four assumptions), the vertical axis corresponds to the error relative to a population-level model.

The first assumption is the *random intercept*. This causes errors to cluster by subject. The mean of the error in each subject is the subject's random intercept β_{oi}. No other structure can be observed: the scale of the errors remains the same over time.

The second assumption corresponds to the *growth model*. In addition to clustering due to the random intercept, the plot also shows that the errors consistently increase over time, at a rate that differs across patients. This growth rate is the random slope v_{it}. Observations of the same subject closer together in time have more similar errors (and thus observations) than observations of the same subject further apart in time. This is a violation of the compound symmetry structure, but the random growth model can handle this situation correctly.

The third assumption is *quadratic time, random intercept*. The data has both linear and quadratic population-level time effect but only a random intercept. We

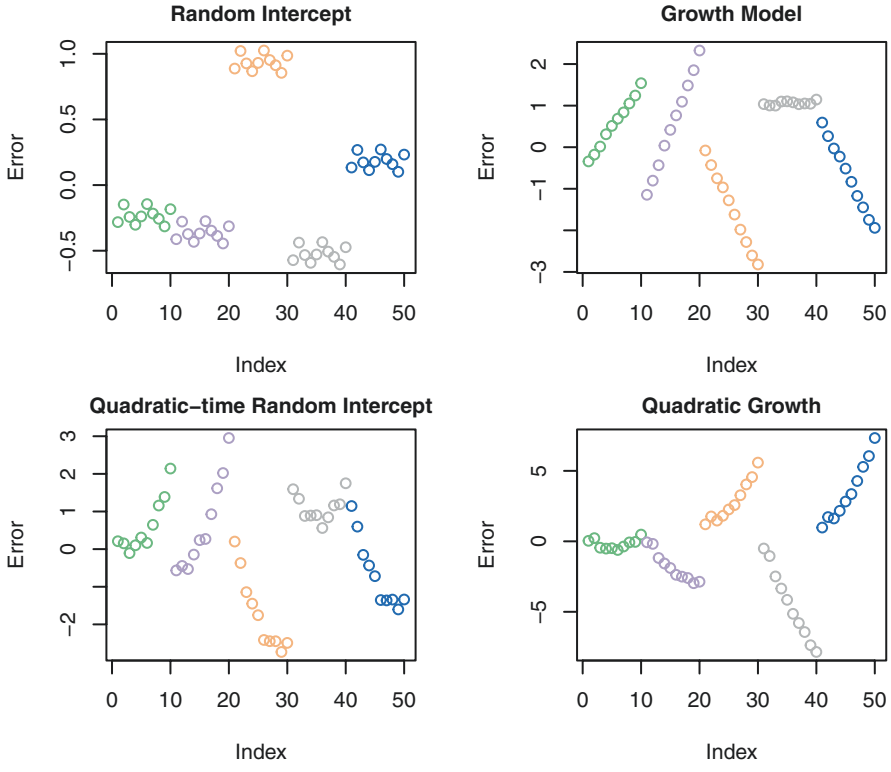


Fig. 26 Comparison of the various model assumptions

only removed the linear time effect, thus the errors (residuals) form a per-subject parabola, indicative of a quadratic effect. The parabolas have similar shape across patients (although different parts of the same parabola are visible), which suggests that this is a (quadratic) population-level effect, but the parabolas have different foci along the y axis, suggesting a subject-level random intercept.

Finally, the *quadratic growth model* has both population-level as well as a subject-level quadratic time effect. The quadratic structure is apparent in the parabolic shapes of the within-subject errors, however, the shape of the parabolas change across the patients, suggesting a subject-level effect. Because of the strong population-level quadratic time-effect, it is difficult to see whether the subject-level time effect is only linear or quadratic. The parabolas are located at different positions along the vertical axis, which indicates a subject-level random intercept.

Generalized Linear Mixed Effect Models (GLMM)

Generalized Linear Mixed Effect Models relate to LMMs the same way as Generalized Linear Models (GLMs) relate to linear regression models. GLMMs allow for a link function to link the expectation of the outcome with the linear predictor. Similarly to GLMs, GLMMs can thus be used to solve regression problems with non-Gaussian dependent variables, such as classification problems (logistic outcome), counting problems (Poisson outcome), etc.

Method label: linear mixed effect models (LMM)	
Main Use	• Regression models for longitudinal data
Context of use	• Longitudinal data with balanced or unbalanced design • Separates subject-level effects from population-level effects • Predictive modeling with within-subject predictions • Accurate error estimates are required or the interest is the statistical significance of covariates
Secondary use	• Generalized LMM has been developed for non-Gaussian response variables
Pitfalls	**Pitfall 3.7.3.2.** GEEs can be computationally more efficient and may produce better predictive models. Use LMM when the goal is to identify subject-level effects
Principle of operation	• Partitions the error into subject-level and population-level components • Random intercept model: Assumes a subject-specific intercept • Random growth model: Assumes a subject-specific intercept and time-trend • Polynomial growth model: Assumes a subject-specific curvilinear time effect
Theoretical properties and empirical evidence	• See the text for the detailed assumptions • ML estimator. Coefficient estimates are consistent, asymptotically normal
Best practices	**Best practice 3.7.3.1.** Use LMM when the goal is to identify subject-level effects **Best practice 3.7.3.2.** If the main purpose is estimating the effect size of covariates or making predictions for previously unseen subjects, GEE can be more computationally effective
References	• Hedeker D, Gibbons RD. Longitudinal Data Analsyis. Wiley, 2006. Chapter 4

Generalized Estimating Equations

As discussed earlier, the key statistical challenge with longitudinal data is the correlation among the observations of the same subject. This challenge is addressed by assuming a variance-covariance matrix for the error when the regression parameters are estimated. In the previous section ("Linear Mixed Effect Models"), we described a method for constructing such a matrix by separating the error variation into a set of subject-specific and a set of population-level effects. These effects define the form of the variance-covariance matrix. An alternative strategy is to assume a functional form for the variance-covariance matrix. This second strategy is the subject of the current section.

In this approach, the parameters that define the variance-covariance matrix are treated as nuisance parameters and the main interest is the coefficients of the covariates, including time. The variance-covariance parameters are marginalized (integrated out) and hence this type of models are referred to as **marginal models**.

Model Specification. The specification of the generalized estimating equations models proceeds similarly to that of generalized linear models (see section "Foundational Methods"). Given a covariate matrix X, the following components are defined.

1. Linear predictor: $\eta_{ij} = X_{ij}\beta$;
2. Linkage function that links the mean of the linear predictor to the expectation of the outcome $g(E(Y_{ij})) = \mu_{ij}$;
3. A variance function relating the mean of the outcome to its variance: $\mathrm{Var}(y_{ij}) = \phi V(\mu_{ij})$;
4. A working variance-covariance matrix parameterized by a: $R(a)$

The first three components are shared with the generalized linear models; GEEs add the fourth component.

Several variance-covariance matrix forms are implemented by statistical software packages and the most common matrices are described below.

1. **Identity**: $R(a) = I$. This assumes that the observations of a subject are independent of each other and thus it reduces a GEE to a regular GLM.
2. **Exchangeable**: $R(a) = \rho$. Observations of the same subject have constant covariance ρ, which does not depend on time. This matrix form is the same as the compound symmetry in the random intercept models.
3. **Autoregressive**: $R(a) = \rho^{|j-j'|}$. With j and j' denoting two time steps, the covariance among observations of the same subject depends on time. If $\rho < 1$, then the further away the observations are in time, the smaller the covariance.
4. **Unstructured**. Each element of the matrix is estimated from data.

Among the four matrices, we have already seen the identity and the exchangeable structures and the unstructured matrix is straightforward to imagine. The autoregressive matrix will take the following form

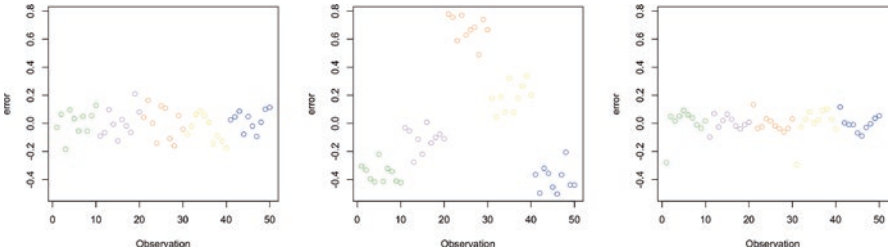


Fig. 27 Illustration of the error distributions corresponding to the Independent, Exchangeable and autocorrelated variance/covariance structures

$$R(\rho)=\begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 & \cdots \\ \rho & 1 & \rho & \rho^2 & \cdots \\ \rho^2 & \rho & 1 & \rho & \cdots \\ \vdots & \vdots & \vdots & 1 & \ddots \end{bmatrix}.$$

When $|\rho| < 1$, increasing powers of ρ become smaller, thus the more distant two observations are in time, the smaller their covariance.

Figure 27 shows three types of error distributions. For 5 subjects, 10 observations were generated using independent error (left panel), exchangeable error (middle panel) and autocorrelated error (right panel). The 5 subjects are shown in different colors and their 10 observations are ordered by time along the horizontal axis. The noise has standard normal distribution with $\sigma = .1$ in all three cases. The error in the left panel is noise and all errors, regardless of which subject they came from, are independent: knowing the error of an observation for a patient does not provide any information about the error of another observation of the same patient or about any observation of any other subject. In the middle panel, the error has a noise component and a random intercept component. Errors are correlated within each subject and subjects are independent of each other. We have seen this correlation structure earlier. Finally, in the right panel, we have autocorrelated errors. Two errors of the same subject are more similar to each other the closer they are to each other in time

Method label: generalized estimating equations (GEE)	
Main Use	• Regression models for longitudinal data • A linkage function can be specified
Context of use	• Longitudinal data with unbalanced design • Most used when the focus is on coefficient estimates and making predictions for previously unseen patients

Method label: generalized estimating equations (GEE)	
Secondary use	
Pitfalls	**Pitfall 3.7.4.1.** No individualized effects are estimated. Consider the LMM if separation of the individual effects from the population effect is desired
Principle of operation	• Uses estimating equations • It is a marginal model. Assumes a parametric form for the working variance/covariance matrix and marginalizes it out
Theoretical properties and empirical evidence	• Uses M estimation. Specification of the likelihood is not required • Solving estimating equations is very computationally efficient • Even if the structure of the variance/covariance matrix is misspecified, it yields good results
Best practices	**Best practice 3.7.4.1**. Use GEE when predictions for previously unseen subjects is needed **Best practice 3.7.4.2.** Use LMM when subject-specific effects are of interest **Best practice 3.7.4.3.** GEE can be more computationally efficient than LMM
References	• Hardin, J.W. and Hilbe, J.M., 2002. *Generalized estimating equations*. Chapman and hall/CRC • Hedeker D, Gibbons RD. Longitudinal Data Analsyis. Wiley, 2006. Chapter 3

Brief Summary of Other Techniques of Interest

Network science. The field of network science [54] offers a completely different approach to conventional predictive modeling and causal discovery methods. Network Science leverages the remarkable consistency in the properties of a broad array of systems that are adaptive and robust. Systems that exhibit these measurable properties are called *Complex Adaptive Systems* (CAS) [55]. The application of Network Science to problems of health and disease is called *Network Medicine* [56] and its main idea follows: a disease represents a pathologic biological process that emerges, and is sustained over time, because it is embedded in a transformed biologic system that acquires adaptive properties. Accordingly, if such an adaptive system related to a given disease is identified, the capacity to determine its areas of vulnerability may reveal promising targets or new approaches for treatment. A typical network science analysis proceeds by building network representations of complex systems and then calculating a number of metrics on the network model. Such metrics include: Network Diameter, Characteristic Path Length, Shortest Path Distribution, Degree Distribution, and Clustering Coefficient. The specific structure and properties of

the network model help the analyst identify drug or other intervention targets and other important system properties.

Active learning. The field of Active Learning studies methods for the iterative collection of data and corresponding refinement of models until an accurate enough model is built or other termination criteria are met. Active Learning methods address both predictive modeling and causal discovery tasks [57–61].

Outlier detection. Outlier (or novelty) detection methods seek to find extreme or otherwise atypical observations among data. "Super utilizer" patients is a prototypical example of outliers that has great importance for healthcare. Numerous methods have been invented for outlier detection over the years in many fields including statistics, engineering, computer science, applied mathematics etc. and they are based on multivariate statistics, density estimation, "1-class" SVMs, clustering, and other approaches [15, 62].

Genetic Algorithms (GAs). Genetic Algorithms are heuristic search procedures in the space of models that the analyst wishes to consider. For example, the analyst may use GAs to find a good linear regression, a good SVM, a good Decision Tree or other model of choice. The search resembles the process of genetic evolution and can be shown to advance rapidly to better models [21]. On the other hand, GAs are computationally very expensive and prone to get trapped in local optima (i.e., solutions that cannot be improved in the next reachable steps in any direction in the model search space, although a better solution does exist several steps away). GAs also are used when the analyst does not have a good insight about the process that generates the data, or about which method may perform well for the task at hand. When such insight exists, it is typically better to use methods that have known properties that guarantee high performance for the desired analysis [63, 64].

Visualization. Visualization methods rely on the capability of the human visual apparatus to decode complex patterns when these patterns have been represented in convenient visual forms. Another use of visualization serves explanatory purposes; that is, presenting and explaining results that were obtained via computational means. Interactive data visualizations, where users are allowed to manipulate their views of the data to obtain more information, have been found to be rapid and efficacious in identifying early infection and rejection events in lung transplant patients [65]. Data visualization can also be useful in displaying health care data, such as that coded with the Omaha System; and intraoperative anesthesia data, such as maintenance of blood pressure [66]. Evaluation techniques have been developed to gauge visualization effectiveness in clinical care [67]. Significant challenges exist, however, in implementing visualization more widely in electronic health records, many of them resulting from the highly multivariable nature of case-oriented medical data which can lead to misleading results. In biological research, heatmaps, clustering, PCA-based visualization and lately t-SNE are very widely used [68].

Recommended Level of Understanding for Professional Data Scientists

The information provided in chapters "Foundations and Properties of AI/ML Systems," "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science," and "Foundations of Causal ML" describing fundamental techniques and their properties aims to provide on one hand a **big picture description** of methods, and a **concise summary of their relative and absolute strengths and weaknesses and types of outputs (e.g., models) produced by each method**.

We recommend that the reader commits to memory the methods information in the above chapters to the extent feasible, and especially for the application domain(s) of interest to them. This will help them evaluate, choose and appropriately apply the right methods, a skill set that eventually, with time and practice will become second nature.

The professional data scientist however should have a much deeper level of understanding that in addition to the information here includes knowing the key algorithms of each method family and possess the ability[1] for each algorithm to:

(a) Describe it in pseudocode from memory;
(b) code it in a programming language of choice;
(c) trace the algorithm on paper for small but representative example problems;
(d) describe the algorithm's function to an expert, a novice, or a lay person at the appropriate level of nuance/simplification;
(e) recite its key theoretical properties;
(f) prove the properties or at least outline the essence of the proofs; and
(g) interpret the algorithms' output.

These skills are typically developed with a combination of formal training, and hands-on experience. The many technical references provided throughout this volume provide a core knowledge base for the technically-oriented reader.

[1] "Possessing the ability" should not be interpreted that the professional data scientists should code all the programs personally, but rather know how it should be done correctly so that they can manage programmers, or evaluate third party codes.

Classroom Assignments and Discussion Topics Chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"

1. What kind of ML tasks are implied by the following questions? (There could be more than one correct answers.)

   (a) What is a particular patient's risk of type-2 diabetes mellitus (T2DM) in 7 years?
   (b) How many years will it take for a particular patient to develop T2DM?
   (c) What is the likely next diseases a particular patient with T2DM will develop?
   (d) What diseases do patients with T2DM typically develop?
   (e) In patients with T2DM, what other diseases are commonly present?
   (f) What is the average age at which patients develop T2DM?
   (g) At what age is a particular patient going to develop T2DM?
   (h) What is the expected cost of medications (per annum) for an average T2DM patient?
   (i) What is the expected cost of medications (per annum) for a particular patient (given other diseases the patient may suffer from)?

2. What kind of modeling tasks are described by the following questions? Also, name the outcome type. There can be more than one solution; give as many answers as you can.

   (a) Predicting the length-of-stay for hospitalized patients.
   (b) Predicting whether the length-of-stay of hospitalized patients will exceed 9 days.
   (c) Predicting the risk of developing diabetes (in patients who are not known to have diabetes currently).

      - We wish to know the probability that the patient develops diabetes within 7 years from now.
      - We wish to know the probability that the patient develops diabetes on any day between now and 7 years from now.
      - We wish to know how many days (from now) it will take for a patient to develop diabetes.

   (d) Predicting the type of cancer (e.g. small-cell, non-small-cell, large-cell, squamous cell).
   (e) Given a sequence of diseases a patient has already developed, predict the most likely next disease.
   (f) What kind of diseases do hospitalized patients with a length-of-stay in excess of 4 days suffer from?
   (g) What are the most common reasons (e.g. admitting diagnoses) for receiving opioids?
   (h) Identify distinct patient groups, based on their lab results, in a cohort of pre-operative patients.

3. What are the most appropriate modelling methods in the following scenarios? Assume you are tasked with building a diabetes risk prediction model that estimates the probability that a patient develops diabetes in 7 years given the patients' health records.

   (a) Suppose you have 20 different predictor variables, which are reasonably informative and uncorrelated, have 100 patients in your training sample, and each patient has one observation vector (for all 20 predictor variables).
   (b) Suppose you have the same 20 predictor variables as in (a), but now you have 10,000 patients, each contributing one observation vector.
   (c) Suppose you have 200 predictor variables that form highly correlated blocks. Each variable is important and has its own unique effect despite the high correlation. Further, you only have 200 patients, one observation vector per patient.
   (d) Suppose you have 2000 predictor variables, most of which are irrelevant to the task at hand. Unfortunately, you do not know a priori which variables are irrelevant. You have 200 patients, one observation vector per patient.
   (e) Suppose you have 20 different predictor variables, which are reasonably informative and uncorrelated, 1000 patients, and you have 10 observation vectors per patient. These 10 observations were collected at equally spaced time intervals for all patients.
   (f) Suppose you have 20 different predictor variables, which are reasonably informative and uncorrelated, 1000 patients, and you have 10 observation vectors per patient. These 10 observations were collected at exactly the same time for all patients.
   (g) Suppose you have 20 different predictor variables, which are reasonably informative and uncorrelated, 1000 patients, and you have 10 observation vectors per patient. These 10 observations were collected at different times for each patient and the collection time is known.
   (h) Suppose you have 20 different predictor variables, which are reasonably informative and uncorrelated, 1000 patients, and you have varying number of observations per patient.

4. You are tasked with building a survival (time-to-event) model that estimates patients' risk of developing diabetes at any time within the next 7 years. What kind of model would you use in the same scenarios as in question 3?

5. You have decided to build classifiers for a classification task. You are given the predictor variables, the outcome and a training data set. What models would be appropriate under the following scenarios?

   (a) The predictor variables are not highly correlated, all have approximately linear effects, and your training data set contains 10 observations per predictor variable.
   (b) The predictor variables are not highly correlated, all have approximately linear effects, and you have several million observations per predictor variable. Which algorithms are most likely to run into computational problems?

(c) The predictor variables are not highly correlated, but they may have unknown non-linear effects. You have sufficient amount of data, but not to the extent where you would expect computational issues.
(d) The predictor variables are not highly correlated, but they may have unknown non-linear effects. You have only 1 observation per predictor variable.
(e) The predictor variables are correlated and may have unknown non-linear effects. You have sufficient amount of data for any algorithm.
(f) The predictor variables are not highly correlated, all have approximately linear effects, and the clinical expects are asking for an "interpretable model". Select a model type and explain why (or how) it is "interpretable".
(g) The predictor variables are correlated and may have unknown non-linear effects. You need to build an "interpretable" model. The predictive performance "is not the primary concern".
(h) The clinical experts are asking for a model that is highly interpretable and has the best possible predictive performance. What do you tell the experts?

6. A data representation is a collection of features (variables) obtained by transforming the original variables. For example, a variable set obtained through dimensionality reduction is a (lower-dimensional) data representation. In this question, your goal is to create a data representation, appropriate under the following conditions.

(a) Predictor variables are reasonably linear, have no interactions, and sufficient observations exist. We want the resulting variables to be orthogonal to each other.
(b) Predictor variables are reasonably linear, have no interactions, and sufficient observations exist. We want the resulting variables to be "independent" of each other in some sense. Select a method and explain what "independent" means for that method.
(c) Predictor variables are reasonably linear, have no interactions, and we wish to have a low-dimensional representation to visualize the data.
(d) Predictor variables may have nonlinear effects, interactions and we wish to build a survival model or a classifier (we do not know yet) using the new representation.

References

1. Stanton, J.M., 2001. Galton, Pearson, and the peas: a brief history of linear regression for statistics instructors. J Stat Educ, 9(3).
2. Taboga M. "Gauss Markov theorem", Lectures on probability theory and mathematical statistics. Kindle Direct Publishing; 2021. Online appendix. https://www.statlect.com/fundamentals-of-statistics/Gauss-Markov-theorem.
3. Hilbe JM. Generalized linear models. Encyclopedia of mathematics. http://encyclopediaofmath.org/index.php?title=Generalized_linear_models&oldid=38890.
4. GLM N, McCullagh P, Nelder JA. Generalized linear models. CRC Press; 1989.
5. Stroup WW. Generalized linear mixed models. Modern Concepts: Methods and Applications. CRC Press; 2013.

6. Agresti A. Categorical data anlaysis. 2nd ed. Chapter 7.2. Wiley Interscience; 2002.
7. Zhang, Wei. Shift-invariant pattern recognition neural network and its optical architecture. Proceedings of Annual Conference of the Japan Society of Applied Physics.1988.
8. Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation. Tech. rep. ICS 8504. San Diego, California: Institute for Cognitive Science, University of California; 1985.
9. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.
10. Attention is all you need. NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017 P. 6000–6010.
11. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. IEEE Trans Neural Netw. 2009 Jan;20(1):61–80. https://doi.org/10.1109/TNN.2008.2005605.
12. Zhang M, Li J. A commentary of GPT-3 in MIT technology review 2021. Fundament Res. 2021;1(6):831–3.
13. Jia X, Willard J, Karpatne A, Read JS, Zwart JA, Steinbach M, Kumar V. Physics-guided machine learning for scientific discovery: an application in simulating lake temperature profiles. ACM/IMS Transactions on Data Science. 2021;2(3):1–26.
14. Vapnik V. The nature of statistical learning theory. Springer Science & Business Media; 2013.
15. Statnikov A, Aliferis CF, Hardin DP, Guyon I. A gentle introduction to support vector machines. In: Biomedicine: theory and methods, vol. 1. World Scientific; 2011.
16. Statnikov,A, Aliferis, CF, Hardin DP, Guyon I. A gentle introduction to support vector machines. In: Biomedicine: case studies and benchmarks (Vol. 2). World Scientific. 2012.
17. Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Mach Learn. 1997;29:103–30.
18. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory. 1967;13(1):21–7.
19. Hart PE, Stork DG, Duda RO. Pattern classification. Hoboken: Wiley; 2000.
20. Tan PN, Steinbach M, Kumar V. Introduction to data mining. Pearson Education; 2018.
21. Mitchell, T.M., 1997. Machine learning (Vol. 1, 9). New York: McGraw.
22. Dupuy A, Simon RM. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. J Natl Cancer Inst. 2007;99(2):147–57.
23. Wolpert DH. Stacked generalization. Neural Netw. 1992;5(2):241–59. https://doi.org/10.1016/s0893-6080(05)80023-1.
24. Breiman L. Stacked regressions. Mach Learn. 1996;24:49–64. https://doi.org/10.1007/BF00117832.
25. Couronné R, Probst P, Boulesteix AL. Random forest versus logistic regression: a large-scale benchmark experiment. BMC Bioinformatics. 2018;19:270. https://doi.org/10.1186/s12859-018-2264-5.
26. Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction, vol. 2. New York: springer; 2009. p. 1–758.
27. Zou H. The adaptive lasso and its oracle properties. J Am Stat Assoc. 2006;101(476):1418–29.
28. Yuan M, Lin Y. Model selection and estimation in regression with grouped variables. J R Stat Soc Ser B. 2007;68(1):49–67.
29. Simon N, Friedman J, Hastie T, Tibshirani R. A sparse-group lasso. J Comput Graphical Stat. 2013;22(2)
30. Friedman J, Hastie T, Tibshirani R. Sparse inverse covariance estimation with the graphical lasso. Biostatistics. 2008;9(3):432–41.
31. You J, Yu B, Maybank SJ, Tao D. Knowledge distillation: a survey. 2021. https://arxiv.org/abs/2006.05525.
32. Zheng X, Aragam B, Ravikumar P, Xing EP. DAGs with no tears: continuous optimization for structure learning. 2018.
33. Goodfellow I, Bengio Y, Courville A. Deep learning. MIT Press; 2016.

34. Aliferis CF, Statnikov A, Tsamardinos I, Mani S, Koutsoukos XD. Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation. J Mach Learn Res. 2010;11(1):171–234.
35. Aliferis, CF, Statnikov, A, Tsamardinos, I, Mani, S and Koutsoukos, XD, 2010. Local causal and Markov blanket induction for causal discovery and feature selection for classification part II: analysis and extensions. Journal of Machine Learning Research, 11(1).
36. Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res. 2003;3(Mar):1157–82.
37. Kohavi R, John GH. Wrappers for feature subset selection. Artif Intell. 1997;97(1–2):273–324.
38. Harrell FE. Regression modeling strategies: with applications to linear models, logistic regression, and survival analysis, vol. 608. New York: springer; 2001.
39. Pearl J. Causality. Cambridge university press; 2009.
40. Statnikov A, Lemeir J, Aliferis CF. Algorithms for discovery of multiple Markov boundaries. J Mach Learn Res. 2013;14(1):499–566.
41. Aphinyanaphongs Y, Tsamardinos I, Statnikov A, Hardin D, Aliferis CF. Text categorization models for high-quality article retrieval in internal medicine. J Am Med Inform Assoc. 2005;12(2):207–16.
42. Statnikov A, Aliferis CF. Analysis and computational dissection of molecular signature multiplicity. PLoS Comput Biol. 2010;6(5):e1000790.
43. Murphy KP. Manifold learning. In: Probabilistic machine learning: an introduction, chapter 20. MIT Press; 2022.
44. Murphy KP. Probabilistic machine learning: an introduction. MIT Press; 2022.
45. Kleinbaum DG, Klein M. Survival Analysis. A self-learning text. Springer; 2020.
46. Therneau T, Grambsch P. Modeling Survival Data: extending the Cox Model. Springer; 2000.
47. Castro MR, Simon G, Cha SS, Yawn BP, Melton LJ, Caraballo PJ. Statin use, diabetes incidence and overall mortality in normoglycemic and impaired fasting glucose patients. J Gen Intern Med. 2016;31:502–8.
48. KleinJP MML. Survival Analysis techniques for censored and truncated data. Springer; 2003.
49. National Cancer Institute. Five-year survival rate. https://www.cancer.gov/publications/dictionaries/cancer-terms/def/five-year-survival-rate
50. Ishwaran, H., Kogalur, U.B., Blackstone, E.H. and Lauer, M.S., 2008. Random survival forests.
51. Wang P, Li Y, Reddy CK. Machine learning for survival analysis: a survey. ACM Comput Surv. 2019;51(6):1–36.
52. Buckley J, James I. Linear regression with censored data. Biometrika. 1979;66:429–36.
53. Hedeker D, Gibbons RD. Longitudinal Data Analsyis. Wiley; 2006.
54. Barabási AL. Network science. Philos Trans R Soc A Math Phys Eng Sci. 2013;371(1987):20120375.
55. Holland JH. Complex adaptive systems. Daedalus. 1992;121(1):17–30.
56. Barabási AL, Gulbahce N, Loscalzo J. Network medicine: a network-based approach to human disease. Nat Rev Genet. 2011;12(1):56–68.
57. Tong S, Koller D. Support vector machine active learning with applications to text classification. J Mach Learn Res. 2001;2(Nov):45–66.
58. Meganck S, Leray P, Manderick B. *modeling decisions for artificial intelligence* 58–69. Springer; 2006.
59. Settles, B., 2009. Active learning literature survey.
60. Ren P, Xiao Y, Chang X, Huang PY, Li Z, Gupta BB, Chen X, Wang X. A survey of deep active learning. ACM computing surveys (CSUR). 2021;54(9):1–40.
61. Olsson, F., 2009. A literature survey of active machine learning in the context of natural language processing.
62. Zimek A, Schubert E, Kriegel HP. A survey on unsupervised outlier detection in high-dimensional numerical data. Stat Anal Data Min: ASA Data Sci J. 2012;5(5):363–87.
63. Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl. 2021;80:8091–126.
64. Srinivas M, Patnaik LM. Genetic algorithms: A survey. Computer. 1994;27(6):17–26.

65. Pieczkiewicz DS, Finkelstein SM, Hertz MI. Design and evaluation of a web-based interactive visualization system for lung transplant home monitoring data. AMIA Annu Symp Proc. 2007;2007:598–602.
66. Lee S, Kim E, Monsen KA. Public health nurse perceptions of Omaha system data visualization. Int J Med Inform. 2015;84(10):826–34. https://doi.org/10.1016/j.ijmedinf.2015.06.010.
67. Pieczkiewicz DS, Finkelstein SM. Evaluating the decision accuracy and speed of clinical data visualizations. J Am Med Inform Assoc. 2010;17(2):178–81. https://doi.org/10.1136/jamia.2009.001651.
68. Van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res. 2008;9(11):2579–605.
69. Hardin JW, Hilbe JM. Generalized estimating equations. Chapman and hall/CRC; 2002.

Foundations of Causal ML

Erich Kummerfeld, Bryan Andrews, and Sisi Ma

Abstract

The present chapter covers the important dimension of causality in ML both in terms of causal structure discovery and causal inference. The vast majority of biomedical ML focuses on predictive modeling and does not address causal methods, their requirements and properties. Yet these are essential for determining and assisting patient-level or healthcare-level interventions toward improving a set of outcomes of interest. Moreover causal ML techniques can be instrumental for health science discovery.

Introduction

Previous chapters have discussed methods for using ML to predict outcomes. We will start by illustrating the concepts of causal ML techniques using a hypothetical vignette. Imagine a scenario where we have used predictive methods that estimate that a particular patient, Amy Anonymous, who has been diagnosed with alcohol use disorder (AUD) but is currently abstaining, has a high probability of relapsing. The next step would naturally be to perform interventions with the goal of preventing the relapse. Can ML help identify the best interventions for preventing Amy's relapse? Causal ML methods can help solve such problems, specifically addressing questions like:

1. How much would Amy's chance of relapsing be reduced if Amy receives a specific therapy?

E. Kummerfeld (✉) · B. Andrews · S. Ma
Institute for Health Informatics, University of Minnesota, Minneapolis, MN, USA
e-mail: erichk@umn.edu

G. J. Simon, C. Aliferis (eds.), *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, Health Informatics,
https://doi.org/10.1007/978-3-031-39355-6_4

2. What factors other than therapy, if any, might also help prevent relapse?
3. What additional complications might ensue if Amy receives a specific therapy?

These are all fundamentally causal questions because they refer to actions that change the usual function of the modeled system, whereas predictive modeling applies only when we model the system's (the human organism or a healthcare system) behavior in its natural state (without any interventions). Using data to answer them requires causal ML. Using data to answer the first of these questions is a **causal inference** problem, [1] while using data to answer the second and third questions are **causal structure discovery** problems [2].

Causal inference is the problem of quantifying the effect of specific interventions on specific outcomes.

Causal structure discovery is the problem of identifying the causal relationships among a set of variables.

Like regression and classification, these are very broad problems. Numerous solutions within AI/ML and outside these disciplines have been developed for each. Both of these problems can further be complicated by the presence of unmeasured (aka "hidden" or "latent") variables. Specialized algorithms exist to address such settings [1–4]. For pedagogical simplicity we focus in the present chapter mostly on situations where all relevant factors have been measured.

We will next review the core concepts of causal modeling. As we saw in chapter "Foundations and Properties of AI/ML Systems" any ML method can be conceptualized as search in the space of models appropriate for the problem domain. Causal ML therefore deals with the space of causal models.

Causal Models Versus Predictive Models

Predictive knowledge is associative, capturing co-variation between two or more phenomena. In contrast, causal knowledge is etiological, and captures whether and to what degree the manipulation of one phenomenon results in changes in another. For example, to reduce Amy's risk for relapse, one needs to manipulate, (aka intervene on, or treat), its causes. In contrast, while being in a rehab facility is strongly associated with experiencing the symptoms of a relapse, preventing Amy from entering the rehab will not prevent or treat a relapse, since presence in the rehab facility is not a cause of relapse but an effect. These forms of knowledge are distinct, and distinct methods are required to model them.

Causal models must therefore be able to (a) represent cause-effect relationships; (b) answer questions of the type "what will be the effects of manipulating factor X" and "which factors should one manipulate in order to affect X"; (c) generate data from the model for simulation purposes. In addition, causal models can answer also the usual predictive queries, e.g.: if we observe X what is the probability of Y?"

The **fundamental distinction between predictive and causal ML models** is that predictive models inform us about the **unperturbed distribution** over a set of variables (i.e., patterns that occur "normally", without any intervention on the factors); whereas causal models inform on what modified patterns and distributions one will obtain when interfering and altering the underlying process that generates the data.

Pitfall 4.1
Popular and successful predictive ML methods are not designed and equipped to satisfy the essential requirements of causal modeling.

While causal ML methods are capable of being used for prediction under no interventions as well under interventions, predictive ML methods have several practical advantages over causal ML ones when used for prediction under no intervention. Therefore:

Best Practice 4.1
For predictive tasks (i.e., without interventions contemplated) use of Predictive ML should be first priority. For causal tasks (i.e., with interventions contemplated) use of Causal ML should be first priority.

Also as discussed in chapter "Foundations and Properties of AI/ML Systems", in some cases we need to construct **flexible predictive models** that can accept queries designating any subset of variables as evidence and other subsets as outcomes of interest, while leaving the remaining variables unspecified. We saw in chapter "Foundations and Properties of AI/ML Systems" how BNs can attain this goal without building a new model for every new query (as the majority of predictive modeling algorithms do; see chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"). The same is true when a mixture of observational and manipulation evidence variables are part of the query (i.e., **flexible causal/predictive modeling**). Causal Probabilistic Graphs (e.g., Causal BNs) can handle such flexible reasoning.

Properties of Well-Constructed Causal Models[1]

First and Foremost Causal Models Must Be Able to Represent the Cause-and-Effect Relationships Among Variables

For causal discovery, this is necessary for answering questions about which variables cause which other variables. For causal inference, this is necessary for removing bias due to confounding. For example, consider a study (Fig. 1) on diet and alcohol abuse, that finds that people who eat fast food regularly are more likely to abuse alcohol. This does not mean that preventing a fast food diet will decrease the chance of Amy's alcohol abuse relapse. Other factors, for example motivation and stress may lead a person to eat more fast food and may also lead a person to abuse alcohol (thus the relationship between fast food and alcohol abuse is confounded by motivation and stress in the Fig. 1 hypothetical example). Causal models can represent complex systems of relationships involving up to hundreds of thousands of variables with clear distinction between confounded vs causal associations.

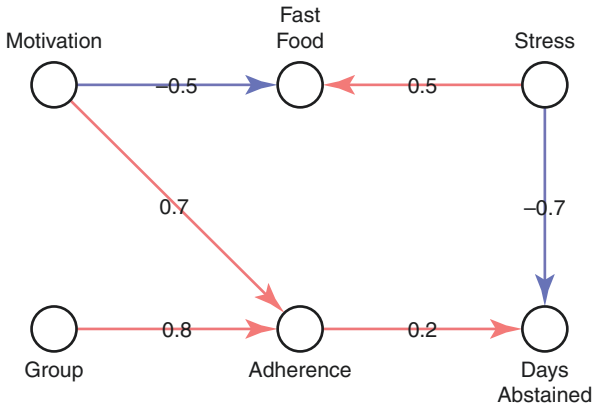


Fig. 1. *A highly simplified example of a clinical trial for a hypothetical drug for treating alcohol use disorder (AUD). Group (Treatment) indicates whether participants receive a placebo or the tested treatment, Adherence indicates how often the participant takes the assigned treatment pills, Motivation indicates a measure of the participant's motivation to have a healthy lifestyle, Fast Food indicates the amount of fast food the participant eats, Stress indicates the stress levels that the participant experiences, and Days Abstained indicates primary outcome, of number of days the person is able to go without relapsing into problematic alcohol use behaviors.* Note: causal ML does not require experimentation in the form of randomized clinical trials (RCTs) or otherwise to infer causal relationships, although it can as in this example be combined with such data to uncover more fine grain information about the causal process than a simple average treatment effect that a RCT provides.

[1] So that the present chapter is self-contained some of the BN definitions and properties of chapter "Foundations and Properties of AI/ML Systems" are re-visited in "Properties of Causal Models" and "Structural Equation Models (SEMs)". Readers can skip previously encountered material.

Second, Causal Models are Markovian

That is, **from the perspective of what are the effects of causal manipulations**, the distribution of every variable can be entirely determined by its immediate causes [1]. If Variable T has direct causes A and B and we manipulate A and B this will have the maximum possible effect on T. No other simultaneous manipulation of variables in the system will have an effect on T once we manipulate A and B. Stated differently, T is "shielded" causally by all variables once we manipulate its direct causes.

From an information transfer (and thus predictive) perspective, however, every variable is independent of every non-descendant variable given its immediate causes (a condition known as the Markov Condition [1]). In other words, non-direct causes of T will have no information about what happened to T after we manipulated A and B. Downstream effects of T *and spouses of T* (i.e., direct causes of the direct effects of T) *will have additional information* about what happened to T after we manipulated A and B.

It is widely accepted that causation itself is Markovian (in the macroscopic world, whereas exceptions happen in the quantum world [2], luckily with no relevance to health science and care). This is reflected by common practice in medical science, epidemiology, biology etc., where causation is studied in two fundamental ways:

(a) Causal chains of the type $A \rightarrow B \rightarrow C$
(b) Causal chains of the type $A \leftarrow C \rightarrow B$

In the chain (a) common health science and health care intuitions are that manipulating A will affect B and thru B will affect C; A, B and C all correlated with one another; once we fix B then manipulation of A will stop affecting C; the correlation of A and C will vanish if we observe or fix B; and the correlation of A and B and of B and C will not ever vanish regardless of observing any other variable.

In the chain (b) common health science and health care intuitions are that manipulating A will not affect B and manipulating B will not affect A; A, B and C all correlated with one another; once we fix C or observe C then the correlation of A and B will vanish; and the correlation of A with C and of C with B will not ever vanish regardless of observing or fixing any other variable.

Third, Causal Models are Modular [1]

Being Markovian leads to the models being decomposable into smaller parts, where we can study these local regions and their function without reference to remote areas in the causal process.

Fourth, Causal Models Are Manipulatable and Generative

Manipulatable refers to capturing in the model the changes that a physical manipulation makes in the actual part of the world that the model represents. A way to represent a physical world manipulation in the model is to assign the corresponding value to the manipulated variable and eliminate all causal edges to it (i.e., because physical manipulations neutralize all other possible causes).

A generative model is one that encodes the full distribution of variables at hand so that every probabilistic calculation can be made with the model. This is sharp distinction with discriminative predictive models that only seek to encode a small fragment of the full distribution, typically the conditional probability of a response given a fixed set of inputs (and in many cases less than that, for example decision surfaces that encode even less information but manage to predict the response to arbitrary accuracy).

In the next sections, we review the predominant class of causal models.

Causal Probabilistic Graphical Models (CPGMs) Based on Directed Acyclic Graphs (DAGs)

A CPGM comprises (1) a DAG; (2) a joint probability distribution (JPD) over variable set $\boldsymbol{V}$ such that each variable corresponds to a node in the DAG; and (3) a restriction of how the JPD relates to the DAG, known as the Causal Markov Condition (CMC).

1. A directed graph is a tuple $<\boldsymbol{V},\boldsymbol{E}>$, where $\boldsymbol{V}$ is a set of nodes representing variables 1-to-1, and $\boldsymbol{E}$ is a set of directed edges, or arrows, each one of which connects a pair of members of $\boldsymbol{V}$. A *path* is any set of adjacent edges. A directed path is a path where all edges are pointing in the same direction. A directed graph is a DAG if it has no cycles in it, that is, if there is no directed path that contains the same node more than once. Fig. 1 is an example of a DAG.
2. The JPD over $\boldsymbol{V}$ is any proper probability distribution (i.e., every possible joint instantiation of variables has a probability associated with it and the sum of those is 1).
3. The CMC states that every variable V is independent of all variables that are non-descendants (i.e., not downstream effects) of V given its direct causes.

Causal ML Models Have Well-Defined Formal Causal Semantics

They typically take the form:

1. Parent set (direct causes) $\boldsymbol{Pa}(\boldsymbol{V}\text{i})$ of variable V_i, $\boldsymbol{Pa}(\boldsymbol{V}\text{i}) = \{Pa(V_i)_1, Pa(V_i)_2, \ldots\}$ is defined over all V_i; and
2. Probability $\Pr(V_i = j | Pa(V_i)_1 = k, Pa(V_i)_2 = l, \ldots)$ is defined over all variables V_i for all value combinations of V_i and its direct causes.

1. Describes the direct causal relationships among all variables
2. Describes the conditional probability distribution of every variable given its direct causes.

If V_i has Parent $Pa(V_i)_j$ that means that in a randomized experiment where one would manipulate $Pa(V_i)_j$, changes would be observed in the distributions of V_i relative to the distribution when $Pa(V_i)_j$ is not manipulated and these changes are entailed by the conditional probability distribution of the variable given its parents.

Unique and Full Joint Distribution Specification in GCPMs

If the fundamental property of the Causal Markov Condition (CMC) holds, then (1) and (2) (parents set, and conditional probabilities of each variable given its parents, respectively) together specify a full and unique joint distribution over variables set ***V.***

Joint Distribution of GCPMs Can Be Factorized Based on Local Causes

This joint distribution is factorized as a product of the conditional probability distribution of every V_i given its direct causes ***Pa***(***V***i)***:***

$$\Pr\left(\{V_1, V_2, \ldots, V_n\}\right) = \prod_i \Pr\left(V_i \left| Pa(V_i)_1 \right|, Pa(V_i)_2 \left| , \ldots \right|, Pa(V_i)_{m_i}\right), \tag{3}$$

where V_i has m_i parents.

Inspection of the Causal Graph of a GPCM Informs About Conditional Independencies in the Data

By inspection of the causal graph (and application of an interpretive rule following from the CMC, called *d-separation*) we can infer a set of conditional independencies in the data, *without statistically analyzing the data*.

If furthermore, *all dependencies and independencies* in the data are the ones following from the CMC, then we speak of **Faithful distributions** encoded by such GCPMs.

A CPGM encoding a faithful distribution entails that all dependencies and independencies in the JPD can be inferred from the DAG. Therefore the DAG becomes a map (so-called i-map) of dependencies and independencies in the data JPD. Conversely by inferring dependencies and indecencies in the data we can construct the CPGM's DAG and parameterize the CPDs of every variable given its parents effectively recovering the causal process that generates the data. This is the **fundamental principle of operation of causal ML methods.**

Additional Notes

Two nodes are **adjacent** if they are connected by an edge. Two edges are **adjacent** if they share a node.

Parents, children, ancestors, and descendants: In a directed graph, if variables X, Y share an edge $X \rightarrow Y$ then X is called the parent of Y, and Y is called the child of X. In Fig. 1, Motivation is a parent of Adherence, and Adherence is a child of Motivation. If there is a directed path from X to Y then X is an ancestor of Y and Y is a descendant of X.

Degree: The degree of a node is the number of edges connected to it. In a directed graph, this can be further divided into in-degree and out-degree, corresponding to the number of parents (connected edges oriented towards the node) and children (connected edges oriented away from the node) that the node has. For example, in Fig. 1, Group has degree 1, in-degree 0, and out-degree 1. In the same figure, Adherence has degree 3, in-degree 2 and out-degree 1. A node with degree 0 is said to be "disconnected".

Collider: A collider is a variable receiving incoming edges from two variables. For example in: $X \rightarrow Y \leftarrow Z$, Y is the collider. A collider is either "shielded" or "unshielded" iff the corresponding parents of the collider are connected by an edge or not, respectively. Unshielded colliders give form to the so-called "v-structure". In Fig. 1, Fast Food is a collider of stress, and motivation.

Trek: A trek is a path that contains no colliders. In Fig. 1, the path from Motivation to Days Abstinent through Adherence is a trek; also the path connecting Fast Food and Days Abstinent through Stress is also a trek. However, the path connecting Motivation and Group through Adherence is not a trek, since it contains the collider Adherence.

D-Separation

1. Two variables X, Y connected by a path are d-separated (aka the path is "blocked") given a set of variables **S**, if and only if on this path, there is (1) a non-collider variable contained in **S**, or (2) a collider such that neither it nor any of its descendants are contained in **S**.
2. Two variables, X and Y, connected by several paths are d-separated given a set of variables **S**, if and only if for all paths connecting X to Y, they are d-separated by **S**.
3. Two disjoint variable sets **X** and **Y** are d-separated by variable set **S** iff every pair $<X_i, Y_j>$ are d-separated by **S,** where X_i and Y_j are members of **X, Y** respectively.

Structural Equation Models (SEMs)

SEMs are causal inference models that can be used after causal relationships have been discovered or when they are known a priori. Left panel of Fig. 2 shows an example causal graph with three variables X, Y, and Z. The three SEM equations show the quantitative functions for each variable given its parents.

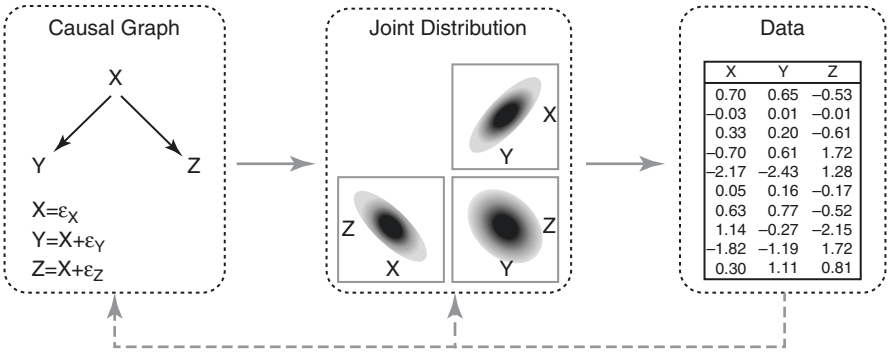


Fig. 2 Correspondence of causal models (left), distribution properties (middle) and data sample (right)

Note that these equations are termed structural equations to emphasize the causal/generative nature of the relationship. SEMs can be continuous, discrete, or mixed, thus extending the definitions of discrete causal models we have seen so far.

The general form of a structural equation for modeling variable x is $x = f(Parent(x), \varepsilon)$, where $Parent(x)$ represents the parents of x, and ε is a noise term representing the unexplained variance in x. The variables on the right hand side of the causal structural equations are causes of the variables on the left hand side. This information mirrors the causal graph, where directed edges (−>) represent direct causal relationships. The quantitative aspect of the causal relationship (e.g. how much change in Y is expected if X is manipulated from 0 to 1) is represented by function *f*). For example, in the structural equation for Y in Fig. 2, Y is a linear function of X with additive noise ε_Y.

In the same example the expected effect of changing X from 0 to 1 via manipulation of X, on Y is 1 (measured in units of Y), which can be computed by $E(Y|do(X = 1))–E(Y|do(X = 0)) = E(1 + e\_Y)–E(0 + e\_Y) = 1$. The *do(.)* notation in the equation represents a manipulation involving the assignment of a value that is enacted regardless of other factors in the model that appear as parents of the manipulated variable. To further clarify the causal vs. associational relationship, consider variables Y and Z. The expected effect of changing Z from 0 to 1 on Y is 0, which can be computed by $E(Y|do(Z = 1))–E(Y|do(Z = 0)) = E(X + e\_Y)–E(X + e\_Y) = 0$. Y is not affected by manipulating Z, since Z is not a cause of Y. This is also obvious from the causal graph, since there is no directed path (sequence of variables connected by directed edges pointing to the same direction) leading from Z to Y. However, the values of Y are associated with values of Z since they are both caused by X.

In summary, when Z is manipulated from value 0 to value 1, Y does not change, even though Y and Z are correlated in observational data without manipulations. The correct causal model explains the observed statistical association between Y and Z as confounded by X and indicates that if we wish to change (e.g., treat) Y, we should manipulate X rather than Z.

The above could be inferred using a CPGM with equivalent results (albeit using probabilities and probabilistic inference rather than structural equations and expectations).

The fundamental difference however is that for a SEM model to be useful we need to first infer the causal relationships via an external mechanism. By contrast many algorithms exist that infer a CPGM from data automatically.

Pitfall 4.2
Using SEMs to estimate causal effects with the wrong causal structure.

Going back to Fig. 1 assume that this model represents the causal process correctly and further that motivation and stress were not measured in this RCT. Also assume that a data analyst is interested in how diet affects relapse rates in patients with AUD, and has access to the trial data, which includes how much the participants eat fast food. The analyst regresses Days Abstinent on Fast Food, and finds that eating more fast food is negatively associated with days abstinent. With the goal of ensuring that this association is not related to the RCT design itself, the analyst also regresses Days Abstinent against both Fast Food and Group, and again finds that there is a negative association between fast food consumption and days abstinent. Excitedly, the analyst goes forth to publicize the findings, and recommends that perhaps limiting the consumption of fast food will reduce relapse rates for AUD patients. Based on these findings, other researchers carry out an RCT to test this theory, but find that (as expected by the true causal model) restricting fast food consumption has no effect on relapse.

The analyst has fallen into a common pitfall when they performed regression analyses without any knowledge of the structure of the underlying causal process, and inappropriately interpreted confounded associations as causal. Even if the analyst reports that an association was found, and does not make claims of causal effects, such associations carry a promise of possible valuable causal relationships.

The same problem is encountered when applying any predictive ML method. It is a grave error to apply such non causal methods when causal results are sought.

Consider the following variant of the above scenario where Group (treatment) is not manipulated but observed. Furthermore consider that the analyst, in an effort to eliminate confounding bias, models Adherence as a confounder (on the grounds that it correlates with both group and days abstained). In such a scenario the estimated causal effect of Group will be falsely zero because the plausible confounder, in reality, is on the causal path between group and the outcome.

Pitfall 4.3
Using regression to estimate causal effects without knowing the true causal structure (and making assumptions about which are the true measured confounders).

Causal Effect Estimation[2]

Further elaborating on causal inference, often we wish to estimate the quantitative causal influence that a variable C has on variable T for a manipulation that causes a 1 unit change of C. The key to obtaining unbiased causal effect estimates from observational data is to partition the total (bivariate, i.e., marginal) statistical association between pairs of variables into the components of that total association due to causal vs. confounded relationships across all connecting paths. In principle, this is relatively straightforward if we know the true causal structure governing the variables we observe. For example, consider estimating the causal effect of C on T using data generated from the causal system depicted in Fig. 3.

In the true causal graph, there are two paths contributing to the overall observed association between C and T. path 1: $C \rightarrow T$, and path 2: $C \leftarrow A \rightarrow D \rightarrow T$[3]. The first path is a causal path, and when one changes the value of C, the value of T will change as a result through this path. The second path is a confounding path, since the change in C cannot causally propagate through this path to affect T. In other

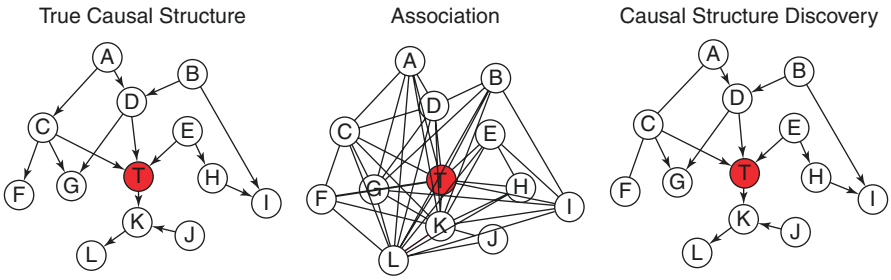


Fig. 3 Illustration of the local causal structure around a variable T

[2] In this section we make extensive use of d-separation to infer whether variables are dependent or independent given other variables (also assuming a faithful distribution encoded by the CPGM). Readers not proficient in application of d-separation in faithful DAGs are advised to simply use the provided dependence/independence statements.

[3] There are other connecting paths between C and T, but they do not contribute to the observed bivariate association between C and T because for the information to travel across them, we need condition on collider nodes. For example C → G ← D → T does not contribute to the association between C and T, unless G is observed, since G is a collider (i.e. both edges involving G have arrow head pointing to G).

words, when we compute the total marginal (aka univariate) association between C and T, the resulting value reflects the combined contributions from the causal effect $C \rightarrow T$ and the confounding $C \leftarrow A \rightarrow D \rightarrow T$.

In order to estimate the causal effect of C on T, we need to eliminate the component of association due to the confounding path. This can be easily achieved by estimating the relationship between T and C, using A as a covariate in a regression model, or more generally controlling for the associational effect of A. Assuming for example linear functions with Gaussian noise, the causal effect of C on T can be estimated by fitting the linear regression $T = \beta_c * C + \beta_A * A + \varepsilon$.

The estimated coefficient for C then will be the unbiased causal effect estimate of the causal influence of C on T. The reason that this regression model can result in unbiased estimates of causal effects is because adding A to the regression model (i.e. conditioning on A) blocks the confounding path $C \leftarrow A \rightarrow D \rightarrow T$ and therefore removes the association component between C and T due to the confounding (path 2).

Conditioning on the wrong variables will likely result in biased effect estimation. For example, conditioning on G by fitting the regression model $T = \beta_c * C + \beta_g * G + \varepsilon$ results in biased effect estimation, since conditioning on G, which is a collider on the $C \rightarrow G \leftarrow D \rightarrow T$ opens the path and introduces additional association between C and T due to this non-causal path (plus the confounding thru A is not controlled).

In practice there are numerous choices for variables to condition on, especially in high dimensional data and in domains with poor prior knowledge. For example, as an alternative to A, conditioning on D by fitting the regression $T = \beta_c * C + \beta_d * D + \varepsilon$ will also result in unbiased effect estimation, since it also blocks the $C \leftarrow A \rightarrow D \rightarrow T$ path and does not open any additional paths (the $C \leftarrow A \rightarrow D \leftarrow B \rightarrow I \leftarrow H \leftarrow E \rightarrow T$ path is still blocked when conditioning on D, due to the presence of I as a collider).

As illustrated above, the principle for achieving unbiased causal effect estimation from observation data is to ensure that only the true causal paths are open (between the pair of variables under consideration) given the variables controlled by conditioning.

Best Practice 4.2
In order to estimate unbiased causal effects, control variables that are sufficient to block all confounding paths. These variables can be identified by causal structure ML algorithms.

Best Practice 4.3
Often there is a choice of multiple alternative variable sets that block confounding paths. An applicable choice is to control/condition on the set Pa(A) in order to block all confounding paths connecting A and T. However this sufficient confounding blocking variable set is not the minimal one and it is recommended to use the minimal blocking variable set in order to maximize statistical power and minimize uncertainty in the estimation of the causal effect.

Pitfall 4.4
Discovering the correct variables to condition on can be hard or even impossible in the presence of hidden variables. Discovering the minimal blocking variable set may be computationally hard or intractable when the causal structure is large and complex.

It is also possible that the causal effect for a specific variable cannot be estimated from observational data for some causal data generating functions. In these cases experiments are needed. For those cases that causal effect estimation is feasible from experimental data, Pearl's Do-Calculous procedure will return the right set of conditioning variables. Do-calculus specifies a systematic procedure to determine if a causal effect is estimable and sequences of operations to compute the causal effect when possible [1].

Also in some distributions, discovery of a causal edge may require conditioning on all variables (which is statistically not feasible).

The Do-Calculus is critically different from conventional methods for causal structural learning and causal inference, e.g. structural equation modeling [5], path analysis [6], matching [7], propensity scoring [8]. The conventional methods of the structural equation family are generally hypothesis-driven and only examine a small fraction of possible causal structures governing the data, which make them likely to miss the true causal structure and result in biased estimates of causal effects. Moreover, even without any hidden variables present, the number of possible models is astronomical for even a few dozen variables, making specification of a good model like winning the lottery. Causal structure discovery algorithms together with Do-calculus circumvent these difficulties in many practical settings.

Causal Structure Discovery ML Algorithms

Given that the definition of causes involves manipulation, experimentation is by default one way to discover causal knowledge. However, in domains where experiments are unethical, technically not feasible, or resource prohibitive, or when we want to construct system-level causal models with numerous variables and all their interactions, it is inefficient, impractical and occasionally impossible to derive causation strictly with experimentation. When combining experimentation with observational causal algortihms (and simultaneous measurement of all variables) however, in the worst case, N-1 experiments are needed to derive all causal relationships in a non-cyclic causal system with N variables, given that one variable can be randomized per experiment [9].

Instead, investigating the statistical relationships among variables in observational data and using the result to guide experimentation can be more efficient. In fact, in many scientific domains, in order to discover causal factors, investigators often first examine observational data for variables that are associated with their outcome of interest and then conduct experiments on a subset of the associated

factors to determine the causes. This common practice reflects the attempt to use observational data to improve the efficiency of experimental causal discovery. However, as we will illustrate, association is a poor heuristic for causation. In some cases, it provides very little guidance to which experiments need to be performed.

The right panel of Fig. 3 illustrates the local causal structure around a variable T of interest. Based on this causal structure, association will be identified among most variables, indicated by lines connecting most pairs of variables shown in the middle panel (this is referred to as the *correlation network* in some literature). Specifically, other than variable J, all measured variables would be univariately associated with T. Therefore, statistical association is not a good strategy for identifying causes of T. Furthermore, the strength of the association is also not a good indicator of causality, since given the true causal structure, there exist distributions where non-causal variables (e.g. G, K, L) can have stronger associations with T than that of the causes of T. Chapter "Foundations and Properties of AI/ML Systems" provides additional theoretical results about causal ML.

Different from association based methods, causal structure discovery methods are designed to discover causal relationships given observational data up to statistical equivalency [2]. Domain knowledge and knowledge regarding the data collection process can be readily incorporated to facilitate the discovery process [1, 2]. Different algorithms for causal structure discovery leverage different statistical relationships. For example, **constraint-based algorithms** infer causal relationships using conditional independence relationships, whereas **score-based algorithms** search for the causal structure that maximizes likelihood-based scores given data. Further, algorithms such as the IC* [1] and FCI^3 [2] can identify hidden confounding variables, which is very helpful when we are not certain if we have measured all possible entities that participate in the causal process.

When the Causal Markov Condition and Faithfulness Condition hold, statistical associations that are non-causal can be identified by examining statistical properties such as conditional independence. For example, the association between A and T is deemed not directly causal, since A and T are independent given variables C and D (denoted as $A \perp T \mid C, D$ where $\perp$ denotes conditional independence, and $\mid$ denotes conditioning. Conditional dependence is denoted with $\not\perp$. Also, determination of the direction of causal relation can be resolved due to the collider relations (i.e., in "Y structures" such as T$\rightarrow$ C $\leftarrow$ A, where C is a variable known as an "unshielded collider"). Importantly, the presence of hidden variables can also be identified by examining conditional independence. It is worth noting that even in systems with a moderate number of variables (e.g. > a few dozen), it is computationally impossible and sample inefficient, to examine all conditional independence relationships, since the number of all possible conditional independence tests grows exponentially with the number of variables. Therefore, modern causal discovery ML algorithms implement efficient search strategies to ensure that the discovered causal structure is correct and the procedure is scalable to millions of variables in many real-life settings (see Chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems" for detailed description of development and validation of such methods). Table 1. below lists several classic causal structure discovery

Table 1 Summary of classic causal discovery algorithms

Method	Assumptions	Statistics	Search strategy
PC [2, 10]	CMC, faithfulness, causal sufficiency, acyclic	Conditional Independence	Start from full graph, eliminate indicated by CI tests
FCI [2, 11]	CMC, faithfulness, acyclic	Conditional Independence	Same as PC in the first phase. Potential hidden variables are identified by examining conditional independences in a second phase
GES [12]	CMC, faithfulness, causal sufficiency, acyclic	Likelihood of model equivalence class given data	Start from empty graph, greedy edge addition and greedy edge elimination operating on the model equivalence class
GFCI [13]	CMC, faithfulness, acyclic	Likelihood of model given data for skeleton discovery and causal orientation, conditional independence for hidden variable identification.	Same as GES in the first phase. Potential hidden variables identified by examining conditional independence in the second phase
MMHC [14]	CMC, faithfulness, causal sufficiency, acyclic	Conditional Independence; likelihood of model given data	Identify local graph of each variable by conditional independence tests, connect local graphs to global graph, greedy edge addition, elimination, and reversal according to likelihood
LGL-GLL [15, 16]	CMC, faithfulness, causal sufficiency, acyclic	Conditional Independence	Flexible algorithm families. Generalized local learning (GLL) identifies local graph of each variable by conditional independence tests, local to global learning (LGL) constructs the global graph by stitching the local neighborhood of variables together. Maybe combined with additional algorithms, e.g., post-process results with equivalence class and hidden variable algorithms
LiNGAM [17]	CMC, faithfulness, causal sufficiency, acyclic, non-Gaussianity	ICA; Wald test; chi-square test; difference test	A combination of ICA, causal-order permutations, and edge pruning tests

algorithms, their assumptions, the statistical relationships they examine, and the search strategies they employ.

It is worth noting that models that rely solely on conditional independence cannot fully resolve the orientation of all edges in the causal system in general. Since the same conditional independence relationships can correspond to multiple causal

structures (i.e., belonging to the same so-called Markov Equivalence Class). For example, the conditional independence relationships corresponding to the following causal three structures $X \rightarrow Y \rightarrow Z$, $X \leftarrow Y \rightarrow Z$, and $X \leftarrow Y \leftarrow Z$, are identical (i.e. X, Y, and Z are pairwise dependent, but $X \perp Z \mid Y$, $X \not\perp Y \mid Z$, $Y \not\perp Z \mid X$), i.e. they are Markov equivalent. Markov equivalent causal structures may still be distinguishable by statistical properties other than conditional independence. Methods that aim to tackle this problem are generally referred to as pairwise edge orientation methods, since they explore the statistical asymmetry between pairs of variables to determine causal direction. These methods, originally pioneered by D. Janzing et al. [42] typically require non-linear generating function and/or non-Gaussian noise terms to break the symmetry in causal direction [18].

Trace of PC on Alcohol Abstinence Example

The PC algorithm is an early algorithm in the field that is no longer used in practice (because of low efficiency and high error) but is useful pedagogically to explain how causal discovery may take place. PC begins by forming a completely connected graph of undirected edges, representing no conditional independence anywhere. The algorithm proceeds by iteratively testing for conditional independence between pairs of (currently) adjacent variables conditioned on sets of increasing size. To begin with, unconditional independence is tested, followed by conditional independence with conditioning sets of cardinality one, then of cardinality two, then of cardinality three, and so forth. The members of these conditioning sets are pulled from the variables adjacent to either member of the current pair being tested. After completing all such conditional independence tests (which thins out the graph), the PC algorithm orients the undirected edges by referencing which conditioning sets were used to separate the independent pairs. For each unshielded triple, if the mediating node was not in the stored conditioning set (aka "sepset"), then it orients the triple as a collider. Lastly, a final set of orientation rules are applied that take advantage of the acyclicity constraint [2].

Let us revisit the AUD example and step through a run of the PC algorithm. First, a completely connected graph of undirected edges is formed in Fig. 4. This represents that we have not yet seen any conditional independences. Next, unconditional independence relations are tested (i.e., conditional sets of size zero). In this case, five independencies are found which results in the removal of five edges as shown in Fig. 5. Next conditional independence relations with conditioning sets of cardinality one are tested. In this case, three conditional independencies are found which results in the removal of three more edges as shown in Fig. 6. Next conditional independence relations with conditioning sets of cardinality two are tested. In this case, one more conditional independence is found which results in the removal or one more edge in Fig. 7. The PC algorithm will go on testing conditional independences (up to the point that the conditioning sets lead to under-powered test), but it will not find any more. Accordingly, the conditional independence phase of the algorithm will result in the undirected graph in Fig. 8.

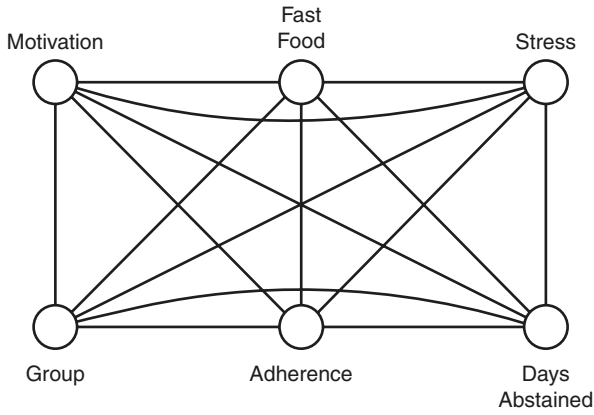


Fig. 4 PC trace on the AUD example: forming a completely connected graph

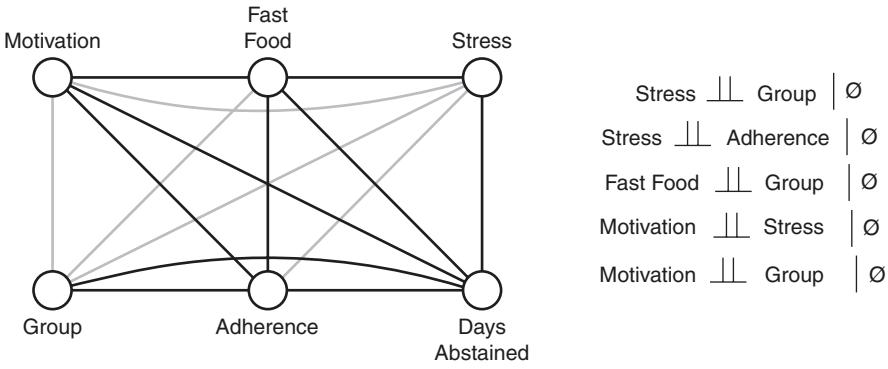


Fig. 5 PC on the AUD example after testing unconditional independence

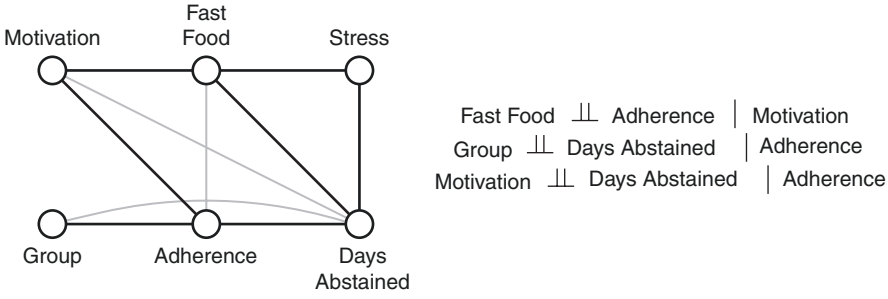


Fig. 6 PC on the AUD example after testing conditional independence with conditioning sets of cardinality one

The edges of the undirected graph in Fig. 8 are oriented by referencing which conditioning sets were used to separate the independent pairs. For each unshielded triplet, if the mediating node was not in the conditioning set, then PC orients the

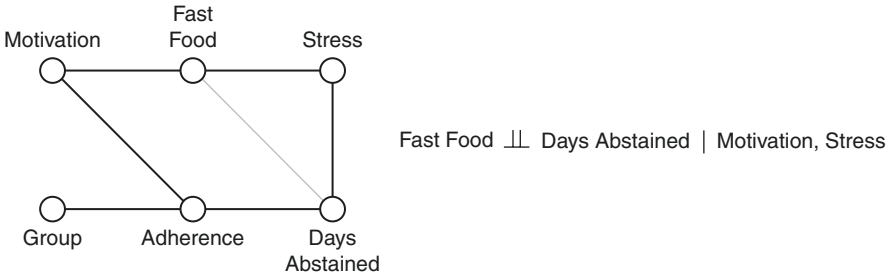
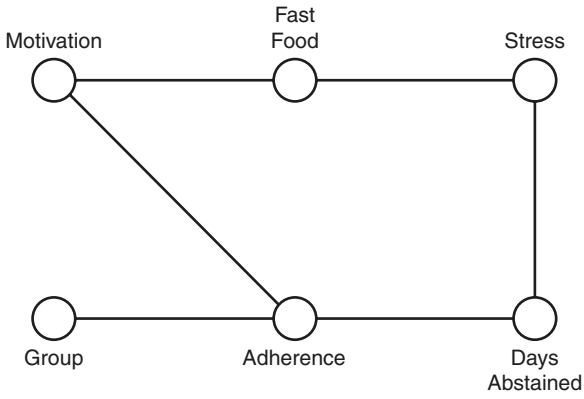


Fig. 7 PC on the AUD example after testing conditional independence with conditioning sets of cardinality two

Fig. 8 PC on the AUD example after completing the conditional independence phase



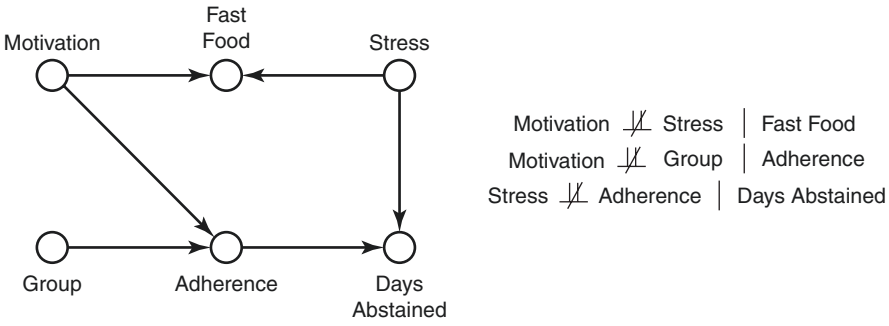


Fig. 9 PC on the AUD example after orienting unshielded colliders

edges to make the middle node in the triplet a collider. In the AUD example, the endpoints of the triplets <Motivation, Fast Food, Stress>, <Motivation, Adherence, Group>, and < Stress, Adherence, Days Abstained> were not separated by Fast Food, Adherence, and Days Abstained, respectively. Accordingly, PC orients these triplets as involving colliders in Fig. 9. At this point, the final set of orientation rules

would be normally applied, however, the graph has already been fully oriented. Note that in general, there are many cases where the final model produced by PC will have some undirected edges remaining, corresponding to a set of possible graphs that are not fully resolved.

Latent Variables

As depicted in Fig. 10 (left), suppose Motivation and Stress are latent (i.e., unmeasured, aka "hidden"). In this case, if we run PC, the resulting graph would be the one shown in Fig. 10 (right). This graph correctly identifies the causal path from Group to Days Abstained and can be used to correct estimation the effect of Group to adherence. However the graph is misleading in that it suggest that the effect of Adherence to Days Abstained can be estimated by conditioning on Fast Food, when in reality conditioning on Fast Food opens a confounding path and leads to inaccurate causal effect estimate. Such examples showcase the need to use algorithms that reveal latent variables and their confounding on measured ones.

We can think about latent variables as variables that have been marginalized out of a larger, complete, but not fully observed, set of variables. In this paradigm, we assume that the causal model over the complete set of variables is a DAG. Thus, under this assumption, the "margin" of a DAG is a natural choice to represent the model's structure over the observed set of variables.

Intuitively, the "margin" of a DAG should be a graph whose restriction of the model manifests as conditional independence in the marginal probability distribution. More precisely, the conditional independence statements implied by the marginalized graph should be the subset of conditional independence statements implied by the DAG over the remaining variables after marginalization.

Unfortunately, DAGs are not closed under marginalization in this sense. That is, there are DAGs with margins that are not consistent with any DAG. For example, in Fig. 10 (left), Group is independent of Fast Food and independent of Days Abstained

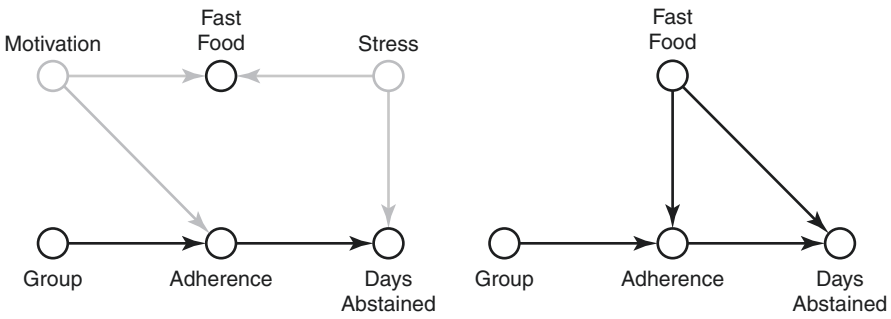


Fig. 10 A DAG with latents and the corresponding PC output

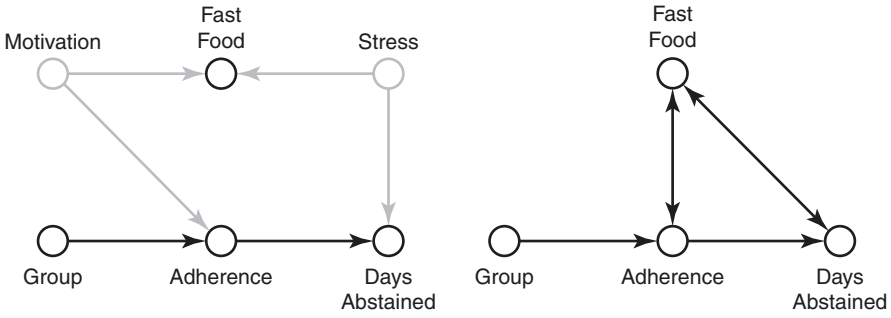


Fig. 11 A DAG with latents and the corresponding ADMG

conditioned on Adherence, however, no DAG represents these exact relations. Accordingly, a richer family of graphs is required to represent margins of DAGs.

Acyclic Directed Mixed Graphs

Acyclic directed mixed graphs (ADMGs) characterize margins of DAGs (Richardson and Spirtes 2002, Richardson 2003). These graphs additionally include bi-directed edges; see Fig. 11 (right) for an illustration. Intuitively, an ADMG can be constructed from a DAG with latent variables by a simple latent projection.

1. For each unshielded non-collider with a latent mediating variable,
   (a) If the triple is directed, add a similarly directed edge between the endpoints,
   (b) Otherwise, add a bi-directed edge.
2. Remove all latent variables.

Importantly, ancestral relationships are invariant under latent projection. For example, in Fig. 11, Group and Adherence are ancestors of Days Abstained while Fast Food is not. These (non-)ancestral relationships are preserved during the marginalization process. Accordingly, it is sufficient to learn an ADMG over the measured subset of variables to infer the presence (or absence) of causal relationships between the measured variables. To this end, the FCI and GFCI algorithms learn Markov equivalence classes of ADMGs [2, 11, 13].

General Practical Approach to Causal ML

In this section, we describe a protocol for conducting causal analysis that involves the following 6 steps:

Best Practice 4.4
A Protocol for health science causal ML
1. Define the goal of the analysis.
2. Preprocess the data.
3. Conduct causal structure discovery.
4. Conduct causal effect estimation.
5. Assess the quality and reliability of the results.
6. Implementation and enhancement of results.

We will next walk the reader through the six step process, pointing also out common pitfalls and how to overcome them.

Evaluate whether the Goals of Modeling Require Causal Analysis

Problem solving that is causal in nature is best addressed by causal modeling. In health-related domains, causal questions generally involve the mechanism and especially the treatment of diseases or interventions on the healthcare system, or discovery of biological causality etc. Some example causal questions in our simple vignette are: (1) what are the *causes* of alcohol abuse? (2) How much improvement in abstinence days can be expected if some motivation enhancer improves it by one standard deviation? (3) What is the best *treatment* for a 35 year-old male with a high stress job that suffers from alcohol abuse?

Questions regarding risk prediction, (e.g. what is the probability of failing 30 days abstinence for a 35 year-old male with a high stress job?), can also be answered with causal discovery analysis. After all, if we have a accurate causal model, we can instantiate the causal model with relevant observational information to deduce a risk prediction. However, the current generation of predictive modeling methods have advantages in answering risk prediction questions not involving manipulations compared to the current generation of causal discovery methods. The main reasons for this are: the current generation of predictive models are discriminant and not generative, can represent more complex mathematical relationships, have built in regularization to avoid overfitting, and most importantly are fitted by analytical protocols such as nested cross validation to maximize their predictive performance via model selection and unbiased performance estimation (whereas causal models seek first and foremost causal validity and have no access to causal error estimators that are available to predictive models). Therefore, we recommend using predictive modeling methods (see chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science") for prediction related tasks.

We also note that Markov Boundary feature selectors **naturally bridge the predictive and causal domains by revealing local causal edges and retaining the maximum predictive information**. They are not however full-fledged causal discovery procedures and need be combined with other causal algorithms for complete causal discovery (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science").

It is also worth noting that some questions appear to require predictive modeling on the surface, but they actually require casual knowledge to answer correctly. For example, the question "what is the risk of relapse within 30 days for a 35 year-old who suffers from alcohol abuse, if he were treated with naltrexone?" This question can be addressed correctly with predictive modeling if we have observed similar patients taking the medication and others not in a randomized assignment. If, however, observed treatments were not randomized, then we would need causal modeling to answer correctly.

Check If the Data Is Suitable for Causal Discovery Analysis and Preprocess Data Appropriately

Causal discovery analysis can be applied to a wide range of data. To ensure that the discovered causal relationships correspond to the goal of the project and have biological and clinical relevance, special attention needs to be paid to the data design, the data collection process, and the data elements being collected. We point out several common situations where data preprocessing might be needed.

Deterministic relationships: Existing causal discovery algorithms may produce erroneous results when deterministic relationships are present among variables. Examples of deterministic relationships are: item scores for a depression inventory vs. the total score for the same depression inventory; height and weight vs. BMI. How to incorporate this kind of information into causal discovery analysis is an area of active research, but at present our recommendation is to eliminate deterministic relationships by using a subset of the variables that are involved in the deterministic relationships.

Specificity of Measurements: Some measurements/variables can contain information from multiple related variables. This is common in mental health data. For example, the depression inventory also often measures anxiety symptoms, and vice versa. As a result, when causal discovery analysis is conducted on these types of variables, a high amount of interconnectivity is found. High connectivity in a causal graph is not in general problematic. But in this case, it is an artifact of the lack of specificity in the measurements and the findings are hard to interpret causally. One way to improve the measurement specificity is through feature engineering, i.e. instead of directly using the original variables with low specificity, we can construct new variables (e.g. separate out depression specific items from the depression inventory and anxiety inventory, and construct a variable that represents depression more specifically). This can be done by either using prior knowledge, or using data driven methods such as factor analysis [19].

Using EHR Data: Different from observational research and clinical trial data, EHR data are collected at irregular time intervals as part of the patients' clinical care. Using the EHR data for any modeling generally requires the researcher to come up with a study design, construct a specific patient cohort, extract relevant EHR data from it, and preprocess the data according to the goal of the study. These steps should follow the same principles for designing observational studies while considering specific properties of the EHR data [20–23]. For data preprocessing, one needs to consider the nature of various EHR data elements and the nature of diseases. For example, a diagnosis code might appear in the patients' record multiple times. This may be due to multiple episodes of acute disease, or may be due to chronic disease and differentiating between them may be important. Similarly, the timestamps in the EHR reflect the documentation time of an event and often do not coincide with the onset time of the measured event. Further, missing measurements in the EHR data is almost definitely not missing at random, since the care providers decide if a measurement would be taken based on the patients' symptoms. Therefore, using imputation algorithms that assume missing at random would cause errors in the analysis. Some of these challenges in the EHR data can be handled with preprocessing, but others might require adaptations to generic causal discovery algorithms [24, 25].

Causal Structure Discovery

Prior knowledge can be readily incorporated into many causal structure discovery algorithms and can greatly facilitate the structure discovery, especially for edge orientation. Prior knowledge can come from multiple sources. One source is the knowledge of the data collection process. For example, in datasets with longitudinal design, information is collected at multiple time points. This timing information can assist edge orientation since events that happen later cannot be the cause of events that happened earlier. It is worth noting that one needs to distinguish between the time that an event happens vs. the timing that the event was measured or documented. For example, we might have measured a patient's Single-nucleotide polymorphism (SNP) and their depression score at the same clinical encounter, but to study the causes of depression, we would assign the SNP to an earlier time point than the depression score. Further, one needs to consider if the variable contains information over a time span. For example, HbA1C contains information regarding glucose over the past 2 or 3 months, it can be assigned an earlier time point compared to variables that reflect instantaneous information such as the blood oxygen level if these variables were measured at the same time.

Prior knowledge can also come from experimental design. For example, in randomized trials, due to the randomization, the participant's pretreatment measurements should not cause the treatment assignment. This prior knowledge can be encoded by prohibiting edges that emerge from the pretreatment variables to the treatment variable.

A third source of prior knowledge is existing domain knowledge. One can enforce the presence or absence of edges according to existing domain knowledge, but this needs to be done with caution. Since the condition under which the domain knowledge was obtained can be different from the current dataset and its data generating process.

Since incorporating prior knowledge can have a significant effect on the resulting causal structure, we recommend only incorporating the most reliable prior knowledge as input to the causal discovery algorithm. We also encourage performing the causal structure discovery analysis with and without all or a subset of the prior knowledge as sensitivity analysis to investigate the added value of incorporating prior knowledge.

Choice of Algorithms At a high level, the performance of the causal structure discovery largely depends on: (1) if the algorithm used is theoretically guaranteed to produce the correct results under its assumptions, (2) how far does the data deviate from the assumptions of the algorithms, and (3) statistical power. One should always choose algorithms that have solid theoretical properties and clearly defined assumptions (see for example the list of algorithms and assumptions in Table 1). Choice of algorithms can also be informed by benchmark studies applied to data with similar characteristics [15, 26, 27].

One important assumption is causal sufficiency, i.e. the absence of latent common causes. *Latent common cause*, also referred to as *latent variables* are likely present for health data, therefore, it is recommended to apply algorithms that can detect latent variables such as the FCI or GFCI directly to the data for causal discovery, or use FCI and GFCI [2] as a second step for latent variable discovery following the skeleton discovery by another more scalable algorithm.

Another common violation of assumptions of most causal discovery algorithms *is target information equivalence* [28]. In health data, overlapping or redundant information often exists, such as co-occurring symptoms in multiple organs and systems, concurrent abnormal lab values from different lab tests, and simultaneously disturbed molecular pathways. This information redundancy can result in target information equivalence, where multiple variable sets contain statistically equivalent information regarding a target variable of interest. Due to the statistical equivalency, the causal role of these variables cannot be determined from observational data alone. Applying algorithms that are not designed and equipped to handle target information equivalence will likely result in missing important causal information. We recommend to always investigate presence and consequences of target information equivalence when appropriate algorithms are available (e.g., TIE*) [28–31].

Statistical power also influences the choice of algorithm. For smaller sample sizes with large numbers of variables, local causal discovery algorithms have advantages over global causal discovery algorithms due to their sample efficiency [15, 16]. It is also worth noting that the algorithms' parameter setting also impacts statistical power and therefore the algorithms' performance. The choice of the underlying statistical test (for constraint based algorithms) or scores (for score based algorithms) need to correspond to the distribution of data to maximize the statistical power. For example, for constraint based algorithms, Fisher's test is recommended

for multivariate Gaussian data, and the G^2 test is recommended for discrete data. Special distributions and data designs (e.g., time-to-event longitudinal designs) might require specially-designed statistical tests. Further, only a subset of algorithms can scale to a very high number of variables without compromising correctness thus being applicable to high dimensional data such as omics data [15].

Causal Effect Estimation

Causal effect estimation should follow the causal structure discovery and be based on the discovered causal structure. Causal effects can be estimated using the Do-calculus principles with SEM as stated in section "Structural Equation Models (SEMs)" above. Note that the SEM model for effect estimation is not restricted to linear regressions. Other mathematical models can be adopted to accommodate non-Gaussian distributions and complex relationships as needed.

It is worth noting that traditional causal effect estimation methods, such as **propensity score-based methods and matching** generally are either based on hypothesized causal graphs, untestable assumptions for correctness (i.e., "strong ignorability") or do not have an explicitly defined causal graph associated with the effect estimation. Given the large space of possible graphs over the set of observed variables, it is highly unlikely that the hypothesized graph would correspond to the true causal structure, therefore these methods can and often lead to biased effect estimation. On the other hand, the lack of an explicitly defined causal graph makes it difficult to interpret the result practically and state the properties of the result of effect estimations [1].

Quality Check and Interpretation of Results

There are several ways to evaluate the discovered causal graphs and estimated causal effects. We recommend several analyses that are suitable for most causal analysis, but problem-specific evaluation should be designed and conducted when appropriate.

Stability of causal discovery. The stability of the causal discovery procedure due to sampling variability can be assessed by bootstrap analysis [32]. In bootstrapping, the same causal discovery procedure (causal structure discovery and causal effect estimation) that was performed on the entire sample can be repeatedly applied to different bootstrap samples. The percentage of time an edge is discovered across all bootstrap samples represents the stability of the causal structure discovery. Edges with percentages closer to 0 or 1 indicate higher stability, representing they are consistently absent or present across all bootstrap samples (i.e., they are more robust to sampling variation). The empirical distribution of the estimated causal effect for manipulating one variable on another variable over all bootstrap samples represents the stability of both the causal structure discovery and the effect estimation. An empirical distribution with smaller variance indicates better stability. Poor stability can indicate issues related to

the distribution of the data. For example, it is possible that the identification of a particular edge based on the entire dataset was driven by outliers and such edges would have low stability in bootstrap samples. When poor stability is observed, we recommend careful inspection of the empirical distribution of the data, as well as target information equivalence analysis (that may be driving the instablity).

Fit of the causal model to the data. After the causal structure discovery and parameterization of the causal model (i.e. estimating all parameters for functions X_i = f(Parent(x))), one can assess the fit of the causal model to the data. In general, the fit can be assessed with scores like the BIC. If the model parameters are estimated with SEM softwares (e.g. OpenMx, Mplus, Lavaan), one can also examine common metrics for goodness of fit from the SEM literature [33–35].

Generalizability. To test the generalizability of causal discovery results, one can identify a separate dataset that contains the same or similar measurements as the primary dataset, conduct causal analysis and compare the results. For example, comparing the causal discovery results on EHR data collected from different hospital systems to assess the generalizability of causal mechanisms over different patient populations [24]; comparing the causal discovery results in a veteran population with PTSD vs. civilian population with traumatic brain injury tests the generalizability of causal mechanism over different disease populations [36].

It is worth noting that the goal for testing generalizability is not to require that the causal mechanism underlying two datasets must be the same, but rather to assess whether the causal mechanisms are indeed the same in different populations. The discrepancies among causal mechanisms discovered from multiple datasets do not indicate that the discovered causal mechanisms are incorrect. It merely indicates that the discovered causal mechanisms are different (because of population or external factor differences). The differences can be due to a variety of technical factors, such as sample sizes, sampling bias in one or more of the datasets, differences in data collection protocol, and differences in measurements. They can also be due to genuine differences in the causal mechanisms of the two populations. Nevertheless, assessing causal mechanisms in multiple datasets that bear similarity is beneficial, since it helps identify stable causal relationships across different datasets and highlights different causal pathways.

Quality of the local causal neighborhood. The predictive performance for a variable of interest can be used to assess the quality of the local causal structure around the variable. This is related to the concept of Markov boundary (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"). Recall that the Markov boundary of a variable of interest is the smallest variable set that contains the maximum amount of (predictive) information about the variable [1]. Under the faithfulness condition, and with no latents, the Markov boundary consists of the direct causes, direct effects and direct causes of the direct effects of the variable of interest [37]. Therefore, one way to assess if our causal structure discovered captures the Markov boundary of the response variable, we can compare the predictive performance of the discovered Markov boundary to the best predictive model (see chapters "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models,"

"Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI" for more details) we can build for this variable given this dataset. If the predictive performance of the discovered Markov boundary is statistically indistinguishable from that of the best model, we can be assured that the Markov boundary variables contain the true local causes and effects (subject to any confounding due to latents). It is worth noting that, when target information equivalency is present for the response variable of interest (which constitutes a faithfulness violation), there are many variable sets (Multiple Markov Boundaries) that are predictively equivalent and contain the maximum amount of information regarding the variable of interest, and are minimal. In this case, observing statistically indistinguishable predictive performance of one Markov boundary vs. the best model does not guarantee the causal role of that Markov boundary. However one of the multiple Markov boundaries contains exactly the direct causes, direct effects, and direct causes of direct effects of the variable of interest (always subject to latent confounding). Finally, a variable that appears in all members of the Markov boundary equivalence class is guaranteed to be causal subject to confounding. An example of applying this method is [31].

Experimental Validation. Another way to partially assess the validity of the discovered causal relationships and effect sizes is experimental validation. For example, one can select a variable to manipulate, observe its effects on another variable, and compare the experimental result with the effect estimated by the causal model. This form of validation is in general costly and possibly not feasible. Experimental results can also come from prior studies, such as RCTs. An example of this is [38].

Implementation and Enhancement of Results

Causal Discovery Guided Experimentation. In many domains of medicine, it is common practice to observe correlational relationships, hypothesize that they are causal, and then test this hypothesis with experimentation. This procedure is not efficient since numerous correlational relationships are due to confounding and are not causal. A more efficient approach is to use causal structure discovery algorithms to eliminate the majority of correlational relationships that are non-causal and resolve any false positives by experimentation. Hybrid causal ML/experimental algorithms exist that are designed to minimize the number of experiments needed for discovery [39].

Consideration for Clinical Translation. One of the main goals for causal discovery applied to biomedical data is to discover novel treatments that can benefit patients. Molecular and other targets that causally affect patient outcomes are potential treatments. Key considerations for which targets to select for treatment depend on their effectiveness, robustness and safety. Causal modeling can help us evaluate these aspects. For example, a causal factor with large effect size and small variability indicates that a corresponding drug treatment would work well and have consistent performance over the patient population. Also in such cases, this potential treatment could be prescribed to patients regardless of their characteristics. If a

putative treatment's effect has large variability, however, this indicates that the response to the treatment could benefit precision medicine administration [40, 41]). Further, with the help of causal analysis, one can also evaluate not only the effect of the treatment on outcomes but also side effects, and select therapeutic targets that maximize patient benefit and minimize side effects.

In summary, the goals for causal ML in health is to discover knowledge that are (1) biologically and clinically relevant, (2) correct and generalizable, and (3) can be translated into clinical application and incorporated into the clinical workflow to benefit patients. A multidisciplinary team consisting of clinical and biological domain experts, health data scientists specialized in causal discovery, clinical informaticists and implementation scientists working closely together is well suited to achieve these goals.

Key Concepts Discussed in Chapter "Foundations of Causal ML"

Causal Inference, Causal Structure Discovery.
Graph, Directed Acyclic Graph (DAG).
Causal Model, Causal Markov Condition, Causal Probabilistic Graphical Model (CPGM), Properties of CPGMs.
Distinction between predictive and causal ML models.
d-separation and Faithfulness.
Structural Equation Models (SEMs).
Causal Effect Estimation and Do-Calculus.
Causal Structure Discovery algorithms.
Acyclic Directed Mixed Graphs (ADMG).
Protocol for health science causal ML.

Pitfalls Discussed in Chapter "Foundations of Causal ML"

Pitfall 4.1. Popular and successful predictive ML methods are not designed and equipped to satisfy the essential requirements of causal modeling.

Pitfall 4.2.: Using SEMs to estimate causal effects with the wrong causal structure.

Pitfall 4.3. Using regression to estimate causal effects without knowing the true causal structure (and making assumptions about which are the true measured confounders).

Pitfall 4.4. Discovering the correct variables to condition on can be hard or even impossible in the presence of hidden variables. Discovering the minimal blocking variable set may be computationally hard in intractable in when the causal structure is large and complex.

Best Practices Discussed in Chapter "Foundations of Causal ML"

Best Practice 4.1. For predictive tasks (i.e., without interventions contemplated) use of Predictive ML should be first priority. For causal tasks (i.e., with interventions contemplated) use of Causal ML should be first priority.

Best Practice 4.2. In order to estimate unbiased causal effects, control variables that are sufficient to block all confounding paths. These variables can be identified by causal structure ML algorithms.

Best Practice 4.3. Often there is a choice of multiple alternative variable sets that block confounding paths. An applicable choice is to control/condition on the set Pa(A) in order to block all confounding paths connecting A and T. However this sufficient confounding blocking variable set is not the minimal one and it is recommended to use the minimal blocking variable set in order to maximize statistical power and minimize uncertainty in the estimation of the causal effect.

Best Practice 4.4. A Protocol for health science causal ML.

1. Define the goal of the analysis.
2. Preprocess the data.
3. Conduct causal structure discovery.
4. Conduct causal effect estimation.
5. Assess the quality and reliability of the results.
6. Implementation and enhancement of results.

Classroom Assignments and Discussion Topics for Chapter "Foundations of Causal ML"

1. Give:
   (a) 2 examples of causal discovery problems and 2 examples of predictive problems in healthcare management.
   (b) 2 examples of causal discovery problems and 2 examples of predictive problems in clinical care.
   (c) 2 examples of causal discovery problems and 2 examples of predictive problems in health sciences research.
   (d) Discuss how predictive modeling that does not take into account causality may lead to flawed decisions in each of the above causal example applications.
2. Someone presented to you a model for predicting the probability of cancer relapse with high predictive performance, derived from observational data using a convolutional neural network. Can you deduce potential relapse prevention strategies from this model?

3. Write a computer program in your favorite programming language to generate data based on the causal graph specified in Fig. 1.
   (a) Based on the data you generated and the causal structure in Fig. 1, estimate the causal effect for each cause-effect pair in Fig. 1 (hint: you should obtain coefficients similar to what is specified in Fig. 1).
   (b) Apply the PC algorithm or the FGES algorithm to the data you generated to discover the causal structure and compare it to the true causal structure specified in the figure.

References

1. Pearl J. Causality. Cambridge University Press; 2009.
2. Spirtes P, Glymour CN, Scheines R. Causation, prediction, and search. MIT Press; 2000.
3. Kummerfeld E, Ramsey J. Causal clustering for 1-factor measurement models. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16. New York, NY, USA: ACM; 2016. p. 1655–64.
4. Kummerfeld E, Ramsey J, Yang R, Spirtes P, Scheines R. Causal clustering for 2-factor measurement models. In: Calders T, Esposito F, Hullermeier E, Meo R, editors. Machine learning and knowledge discovery in databases, volume 8725 of Lecture notes in computer science. Berlin, Heidelberg: Springer; 2014. p. 34–49.
5. Kaplan D. Structural equation modeling: foundations and extensions, vol. 10. Sage Publications; 2008.
6. Wright S. Correlation and causation. J Agric Res. 1921;20:557–85.
7. Rubin DB. Matching to remove bias in observational studies. Biometrics. 1973;29:159–83.
8. Rosenbaum PR, Rubin DB. The central role of the propensity score in observational studies for causal effects. Biometrika. 1983;70:41–55.
9. Eberhardt, F., Glymour, C. & Scheines, R. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. arXiv preprint, arXiv:1207.1389 (2012).
10. Colombo D, Maathuis MH. Order-independent constraint-based causal structure learning. J Mach Learn Res. 2014;15:3741–82.
11. Zhang J. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. Artif Intell. 2008;172:1873–96.
12. Chickering DM. Optimal structure identification with greedy search. J Mach Learn Res. 2002;3:507–54.
13. Ogarrio JM, Spirtes P, Ramsey J. A hybrid causal search algorithm for latent variable models. In: Conference on probabilistic graphical models. PMLR; 2016. p. 368–79.
14. Tsamardinos I, Brown LE, Aliferis CF. The max-min hill-climbing Bayesian network structure learning algorithm. Mach Learn. 2006;65:31–78.
15. Aliferis CF, Statnikov A, Tsamardinos I, Mani S, Koutsoukos XD. Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation. J Mach Learn Res. 2010;11(7):171–234.
16. Aliferis CF, Statnikov A, Tsamardinos I, Mani S, Koutsoukos XD. Local causal and Markov blanket induction for causal discovery and feature selection for classification part II: algorithms and empirical evaluation. J Mach Learn Res. 2010;11(8):235–84.
17. Shimizu S. LiNGAM: non-Gaussian methods for estimating causal structures. Behaviormetrika. 2014;41(1):65–98.

18. Statnikov A, Henaff M, Lytkin NI, Aliferis CF. New methods for separating causes from effects in genomics data. BMC Genomics. 2012;13(8):1–16.
19. Rawls E, Kummerfeld E, Mueller BA, Ma S, Zilverstand A. The resting-state causal human connectome is characterized by hub connectivity of executive and attentional networks. NeuroImage. 2022;255:119211.
20. Rosenbaum PR, Rosenbaum PR, Briskman. Design of observational studies, vol. 10. New York: Springer; 2010.
21. Kramer MS. Clinical epidemiology and biostatistics: a primer for clinical investigators and decision-makers. Springer Science and Business Media; 2012.
22. Botsis T, Hartvigsen G, Chen F, Weng C. Secondary use of EHR: data quality issues and informatics opportunities. Summit Transl Bioinform. 2010;2010:1–5.
23. Weiskopf NG, Bakken S, Hripcsak G, Weng C. A data quality assessment guideline for electronic health record data reuse. Egems. 2017;5(1):14.
24. Shen X, Ma S, Vemuri P, Castro MR, Caraballo PJ, Simon GJ. A novel method for causal structure discovery from EHR data and its application to type-2 diabetes mellitus. Sci Rep. 2021;11(1):1–9.
25. Shen X, Ma S, Caraballo PJ, Vemuri P, Simon GJ. A novel method for handling missing not at random data in the electronic health records. In: 2022 IEEE 10th international conference on healthcare informatics (ICHI). IEEE; 2022. p. 21–6.
26. Sanchez-Romero R, et al. Estimating feedforward and feedback effective connections from fMRI time series: assessments of statistical methods. Netw Neurosci. 2019;3(2):274–306.
27. Ma S, Kemmeren P, Gresham D, Statnikov A. De-Novo learning of genome-scale regulatory networks in S. cerevisiae. PLOS ONE. 2014;9(9):e106479.
28. Statnikov A, Lemeir J, Aliferis CF. Algorithms for discovery of multiple Markov boundaries. J Mach Learn Res. 2013;14(1):499–566.
29. Karstoft KI, Galatzer-Levy IR, Statnikov A, Li Z, Shalev AY. Bridging a translational gap: using machine learning to improve the prediction of PTSD. BMC Psychiatry. 2015;15(1):1–7.
30. Statnikov A, Aliferis CF. Analysis and computational dissection of molecular signature multiplicity. PLoS Comput Biol. 2010;6(5):e1000790.
31. Kraus VB, Ma S, Tourani R, Fillenbaum GG, Burchett BM, Parker DC, Kraus WE, Connelly MA, Otvos JD, Cohen HJ, Orenduff MC. Causal analysis identifies small HDL particles and physical activity as key determinants of longevity of older adults. EBioMedicine. 2022;85:104292.
32. Kummerfeld E, Rix A. Simulations evaluating resampling methods for causal discovery: ensemble performance and calibration. In: 2019 IEEE international conference on bioinformatics and biomedicine (BIBM). IEEE; 2019. p. 2586–93.
33. Fan X, Thompson B, Wang L. Effects of sample size, estimation methods, and model specification on structural equation modeling fit indexes. Struct Equ Model Multidiscip J. 1999;6(1):56–83.
34. Bentler PM. On tests and indices for evaluating structural models. Personal Individ Differ. 2007;42(5):825–9.
35. Barrett P. Structural equation modelling: adjudging model fit. Personal Individ Differ. 2007;42(5):815–24.
36. Pierce B, Kirsh T, Ferguson AR, Neylan T, Ma S, Kummerfeld E, Cohen B, Nielson JL. Causal discovery replicates symptomatic and functional interrelations of posttraumatic stress across five patient populations. Front Psychiatry. 2023;13:1018111.
37. Guyon I, Aliferis C. Causal feature selection. In: Computational methods of feature selection. Chapman and Hall/CRC; 2007. p. 79–102.
38. Maathuis MH, et al. Predicting causal effects in large-scale systems from observational data. Nat Methods. 2010;7(4):247–8.

39. Statnikov A, Ma S, Henaff M, Lytkin N, Efstathiadis E, Peskin ER, Aliferis CF. Ultra-scalable and efficient methods for hybrid observational and experimental local causal pathway discovery. J Mach Learn Res. 2015;16(1):3219–67.
40. Gunlicks-Stoessel M, Klimes-Dougan B, VanZomeren A, Ma S. Developing a data-driven algorithm for guiding selection between cognitive behavioral therapy, fluoxetine, and combination treatment for adolescent depression. Transl Psychiatry. 2020;10(1):1–11.
41. Winterhoff B, Kommoss S, Heitz F, Konecny GE, Dowdy SC, Mullany SA, Park-Simon TW, Baumann K, Hilpert F, Brucker S, du Bois A. Developing a clinico-molecular test for individualized treatment of ovarian cancer: the interplay of precision medicine informatics with clinical and health economics dimensions. In: AMIA annual symposium proceedings, vol. 2018. American Medical Informatics Association; 2018. p. 1093.
42. Janzing D, Mooij J, Zhang K, Lemeire J, Zscheischler J, Daniušis P, Steudel B, Schölkopf B. Information-geometric approach to inferring causal directions. Artif Intell. 2012;182:1–31.

Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems

Constantin Aliferis and Gyorgy Simon

Abstract

The chapter outlines a comprehensive process, governing all steps from analysis and problem domain needs specification, to creation and validation of AI/ML methods that can address them. The stages are explained and grounded using existing methods examples. The process discussed equates to a generalizable Best Practice guideline applicable across all of AI/ML. An equally important use of this Best Practice is as a guide for understanding and evaluating any ML/AI technology under consideration for adoption for a particular problem domain.

Keywords

Method development and evaluation process · Properties of AI/ML methods (theoretical, empirical) · AI/ML stacks · Protocols · Engines · AI/ML method properties map

Establishing Properties of AI/ML Methods During New Method Development and Evaluating Properties when Choosing the Best Method for the Problem at Hand

In chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML" we reviewed major health AI/ML methods in common use in healthcare and the health sciences. We examined

C. Aliferis (✉) · G. Simon
Institute for Health Informatics, University of Minnesota, Minneapolis, MN, USA
e-mail: constantinaibestpractices@gmail.com

G. J. Simon, C. Aliferis (eds.), *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, Health Informatics,
https://doi.org/10.1007/978-3-031-39355-6_5

how they work and summarized important properties. We especially focused on the following **theoretical properties and corresponding conditions (if known) that guarantee these properties**:

1. Representation power: can the models produced by the method represent all relevant problem instances and their solutions?
2. Transparency: are the models produced by the method easy to understand (i.e., are they "transparent box") and can they be easily understood by human inspection (i.e., are they explainable and human interpretable)?
3. Soundness: When the methods output a solution to a problem instance, is this solution correct? If there is a degree of error (measured in some scale of loss, risk or other scale) how large is the error and its uncertainty?
4. Completeness: Does the method produce correct answers to all problem instances? If only for a fraction of the problem space, how large or important is the fraction?
5. Computational complexity for learning models: what is the exact or asymptotic computational complexity of running the method to produce models as a function of the input size (e.g., number of variables, or sample size)?
6. Computational complexity for using models: what is the exact or asymptotic computational complexity of running the models produced by the method to answer problems as a function of the input size (e.g., number of input variables)?
7. Other cost functions: for example, what is the cost to obtain and store input data and run analyses on a compute environment, either at model discovery or at model deployment time?
8. Space complexity for learning models: what is the exact or asymptotic storage complexity of running the method to produce models as a function of the input size (e.g., number of variables, or sample size)?
9. Space complexity for storing and using models: what is the exact or asymptotic storage complexity of running the models to answer problems as a function of the input size (e.g., number of variables, or sample size)?
10. Sample complexity, learning curves, power-sample requirements: how does the error of the produced models varies as function of sample size of discovery data? How much sample size do we need in order to build models with a specific degree of accuracy and acceptable statistical error uncertainty, and how much sample size is needed to establish statistically superiority to random or specific models and performance levels?
11. Probability and decision theoretic consistency: is the ML/AI method compatible with probability and utility theory?

We also discussed **empirical performance properties which we can differentiate for the purposes of the present chapter into**:

1. **Comparative and absolute empirical performance in simulation studies**: when we give the method discovery data produced by simulations, what is the empirical performance of the method?
2. **Comparative and absolute performance in real data with hard and soft gold standard known answers**: when we give the method discovery data from real world sampling, what is the empirical performance of the method?

Chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML" made it abundantly clear to the reader that the majority of widely-used formal methods have known properties along the above dimensions. Ad hoc and heuristic systems and methods do not, however, and thus are viewed in this volume as pre-scientific or early-stage and preliminary, thus carrying great risk of failure (at the present stage of scientific knowledge about them). It also became obvious that knowing the above properties is essential knowledge for determining which among several available methods and corresponding libraries, tools and commercial offerings are the ones that are best suited for a problem we wish to solve using AI/ML.

In general, theoretical properties are stronger than empirical ones in the sense that they describe very large parts of the problem-solving domain more efficiently and with greater clarity than empirical studies. Figure 1 shows the coverage and interpretation of **sufficient conditions**, vs. those provided by **sufficient and necessary conditions and vs. necessary conditions**. Sufficient and necessary criteria describe the whole problem space, whereas sufficient conditions and necessary conditions only part of it, but they are usually more difficult to establish, thus usually theoretical analysis for AI/Ml methods is provided in the form of sufficient conditions. It is not uncommon for the set of sufficient conditions to grow over time so that the total problem space is better understood.

Notice that sufficient conditions point to parts of the problem space where desired performance is achieved. So, if we establish that in some domain a set of sufficient conditions hold, we can expect desirable performance. In other words, *sufficient conditions are constructive*. Necessary conditions point to parts of the problem space where undesired performance will occur. They provide warnings against pitfalls, but do not tell us the complete picture of how to obtain desirable results (other than avoiding the pitfalls, i.e., safeguarding that necessary conditions are not violated).

Even if we do not violate necessary conditions, we are not guaranteed desired performance however (because additional sufficient conditions may not hold). Necessary and sufficient criteria map out precisely the totality of the problem space

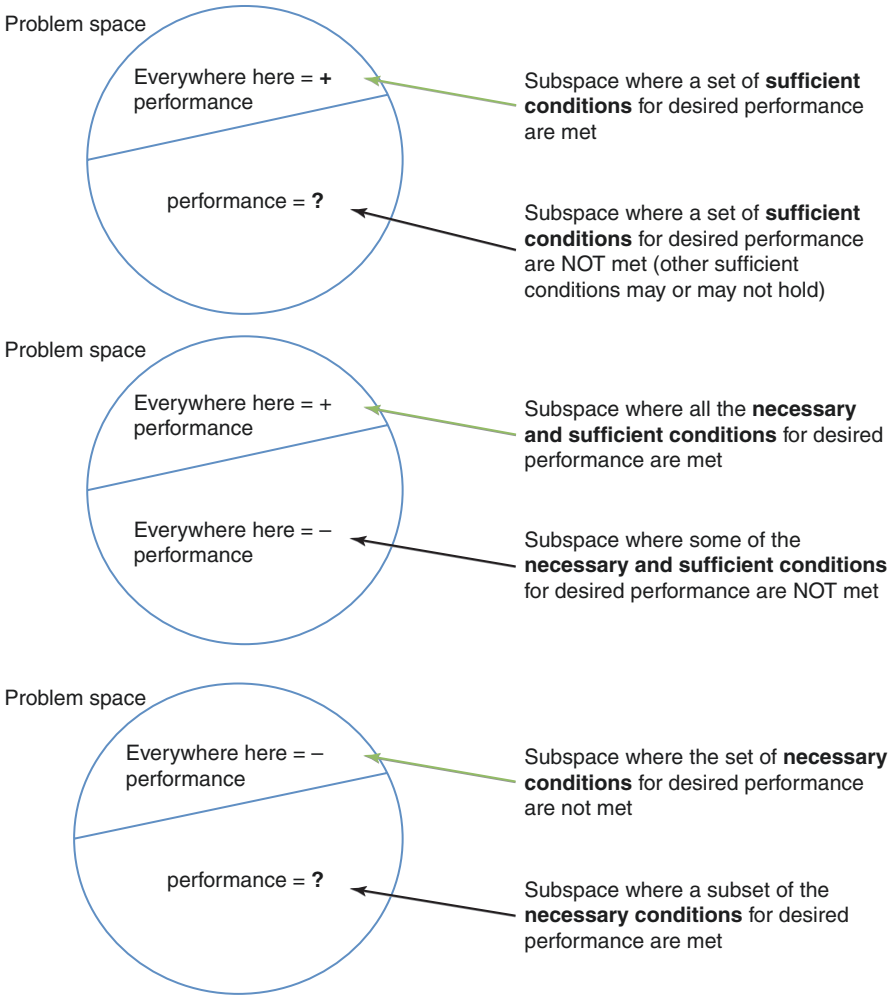


Fig. 1 Conditions for desirable performance of AI/ML methods. Sufficient vs. sufficient and necessary, vs. necessary conditions

in which we will obtain desirable performance and the space that we will not. Again, it is much harder in practice to derive necessary and sufficient conditions.

By comparison to the above, empirical studies provide *limited coverage* of the problem space as shown in Fig. 2. In the absence of theory it would take an astronomical number of empirical studies to cover the space that a single theorem can (assuming that the combinatorial space of factors involved is non-trivial).

One caveat of applying theoretical analysis is that we may not know with absolute certainty if sufficient, necessary, or sufficient and necessary conditions are met in some problem domain. It is very important for **theoretical conditions to be testable**. For example, we can easily test whether the data is multivariate normal and thus we can know whether in some domain we satisfy the relevant sufficient

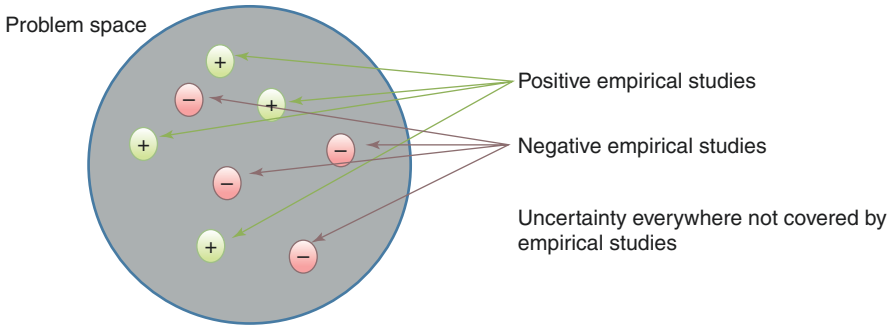


Fig. 2 Empirical studies. Empirical studies showing positive (green) and negative (red) performance results, can sample the problem space and collect evidence about the applicability of method (s) of interest (extrapolated, inductively, to similar populations). The vast majority of the space (grey) has unknown performance however since it has properties that may affect performance, and which properties are not covered by the empirical studies. Empirical studies are vastly less efficient than theoretical analysis

condition of OLS regression (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"). By comparison, within classical statistics frameworks we do not know in general, and cannot test practically, whether the strong ignorability sufficient condition of propensity scoring method holds (chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring problems, and the role of Best Practicess"). The existence of such a condition then just says that it is *theoretically possible* for the method to work in some real or hypothetical distribution (but we cannot tell whether it will work in any real life data of interest with certainty).

In some situations it is also possible to fix violations of conditions. For example, we can transform (some) non normal distributions so that they become normal. Sometimes, violating sufficient conditions for desired performance is mistaken to imply guranteed undesired perfromance. This is not the case, always, since not meeting a set of suffiecient conditions may still be followed by good performance (i.e., if other sufficient conditions exist and are met, aka "mitigation factors" [1]).

There is a clear relevance of pitfalls and guidelines for best practices, to the above concepts. These must describe sufficient, necessary or sufficient and necessary conditions for desired or undesired performance, and these conditions must be testable or identifiable.

In Fig. 3 we demonstrate the importance of **combining theoretical analysis with empirical testing**. If theory predicts that in some identifiable part of the problem space - described in the example of the figure by the subspace where sufficient conditions hold - we expect there desirable behavior. But we may still be unsure about whether these conditions were tested accurately. Or if no testing of conditions was conducted, whether the theory applies to this particular problem space and data sampled from it, that we are facing. By 'sampling" this part of the problem space we can obtain evidence strengthening or refuting the appropriateness of the applicability of the methods in question.

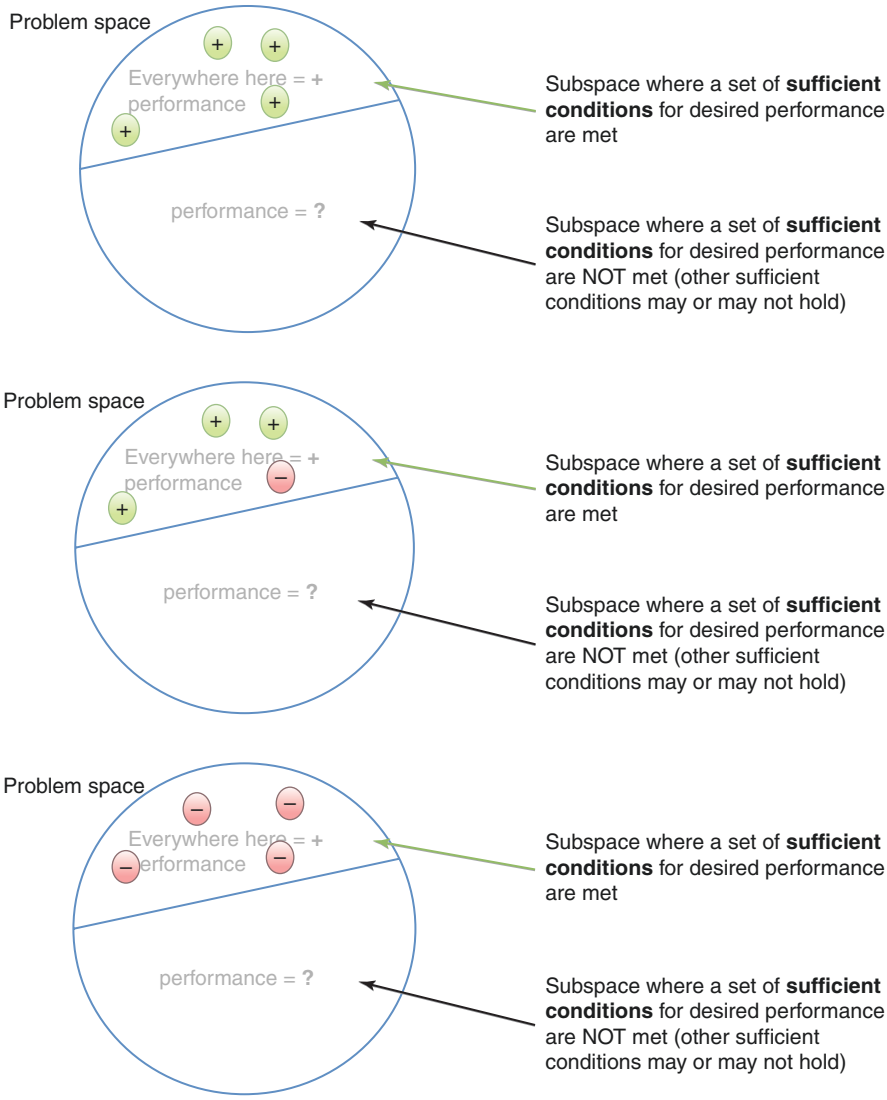


Fig. 3 Combining theoretical analysis' expectations with empirical studies (green = empirical studies showing desired performance, red = empirical studies with undesired performance). Top: empirical studies (verify that our problem space is well aligned with theoretical expectations of AI/ML method performance. Middle: some studies show misalignment with theoretical expectations. Possibly, criteria used to test whether assumptions hold may be inaccurate, or the empirical studies were flawed, or both. Bottom: many studies show misalignment of our domain with theoretical expectations. Either criteria used to test whether assumptions hold are wrong or assumptions were never tested (and the problem domain does not oblige)

To elaborate, if we obtain positive empirical verification of theoretical expectations, we can be sure that the theoretical roadmap is aligned with our practical problem solving setting. If, however we see empirical results that violate the theoretical expectations, this is evidence that we did not sufficiently test the

preconditions for method success, or that the means in our disposal for testing the preconditions of the theoretical properties are not accurate and we need better ways to test the suitability of our methods to the real-life problem.

Going back to Fig. 2, the reader may wonder: why don't we dispense with theory and treat the characterization of the problem space with regards to performance of any set of methods, as an empirical ML problem itself? For example, dispel with theory and base acceptance of, e.g., clinical AI models by doing Clinical Trials? With enough methods and enough trials/datasets over enough problem spaces we can circumvent the need to derive complex theoretical analyses. The answer is three-fold:

1. As indicated earlier, the number of empirical studies needed would be astronomical, as the number of studies needed grows combinatorially to the number of factors involved.
2. In the absence of theory we do not even known which factors affect performance and thus lack the knowledge necessary to design a set of empirical studies that can cover the space of interest. This is a *Theory Ladeness* problem (i.e., scientific conclusions are affected by the framework determining what to study, how to measure it etc. [2, 3].
3. This practice may jeopardize human subjects or waste valuable resources, and is thus unethical in such cases. Moreover, as the field of ML matured, it has become evident that even thousands of datasets used for a variety of benchmarks across domains cannot cover the full space of possibilities of evaluating a single method. At most we can divide and conquer the full ML problem space into application areas and conduct large benchmark studies there. However for this endeavor in order to be efficient and practical, the specific choices of focused areas and datasets must be constrained and guided by a robust theoretical understanding of the factors affecting successful modeling.

In summary, theory and empirical study work synergistically together and provide a concrete roadmap of which methods are suitable for what task, by establishing the properties of AI/ML methods. The same is true for individual models produced by such methods as detailed in the next chapter on developing and validating AI/ML models. Knowing AI/ML properties (theoretical and empirical) provides crucial information that allows one to make informed decisions about the **Performance, Safety and Cost-effectiveness requirements and potential** of corresponding AI/ML approaches, methods and systems (and the decision models produced by them). Successfully meeting requirements that ensure desirable AI/ML behavior can then lead to building trust in the AI/ML solution (as discussed in chapter "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: the Need for Best Practices Enabling Trust in AI and ML") which encompasses: *Scientific and Technical Trust; Institutional Trust; System-of-science Trust; Beneficiary Trust; Delivery Trust; Regulatory Trust; and Ethical Trust.* We can codify the above as 4 best practices:

Best Practice 5.1
Methods developers should strive to characterize the new methods according to the dimensions of theoretical and empirical properties.

Best Practice 5.2
Methods developers should carefully disclose the known and unknown properties of new methods at each stage of their development and provide full evidence for how these properties were established.

Best Practice 5.3
Methods adopters and evaluators (users, funding agencies, editorial boards etc.) should seek to obtain valid information according to the dimensions of theoretical and empirical properties for every method, tool, and system under consideration.

Best Practice 5.4
Methods adopters and evaluators should map the dimensions of theoretical and empirical properties for every method, tool, and system under consideration to the problem at hand and select methods based on best matching of method properties to problem needs.

Major pitfalls can and do ensue when the above best practices are not followed. In chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring problems, and the role of Best Practices" we will discuss many case studies of real-life dire consequences of failing to follow these best practices.

Pitfall 5.1
Developing methods with theoretical and empirical properties that are:
(a) Unknown, or
(b) Poorly characterized in disclosures and technical, scientific or commercial communications and publications, or
(c) Clearly stated (disclosed) properties but not proven, or
(d) Not matching the characteristics of the problem to be solved at the level of performance and trust needed opens the possibility for major errors, under-performance, poor safety, unacceptable cost-effectiveness, and lack of trust in AI.

Best Practice Workflow to Establish the Properties of any New or Pre-Existing Method, Tool or System

In the remainder of the present chapter we will outline a best practice workflow, shown in Figs. 4, and 5 that can be used to create new (Fig. 4) or establish the properties of any pre-existing (Fig. 5) method, tool or system, so that a rational, effective and efficient solution to the problem at hand can be identified as per the guidelines 5.1–5.4.

Throughout the description of the process details that follow, we will highlight how the steps apply to several well-known methods (which are detailed in chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML") and furthermore interleave a running in-depth example of putting these principles to practice in a real-life method development context (new method development for biomarker, signatures and biomedical pathway discovery).

Step 1. Rigorous problem definition (in precise mathematical terms, and with precise correspondence to health care or health science objectives)

It is always worthwhile to invest time to define the problem addressed by new methods we plan to develop or evaluate, in very precise terms. This enables an

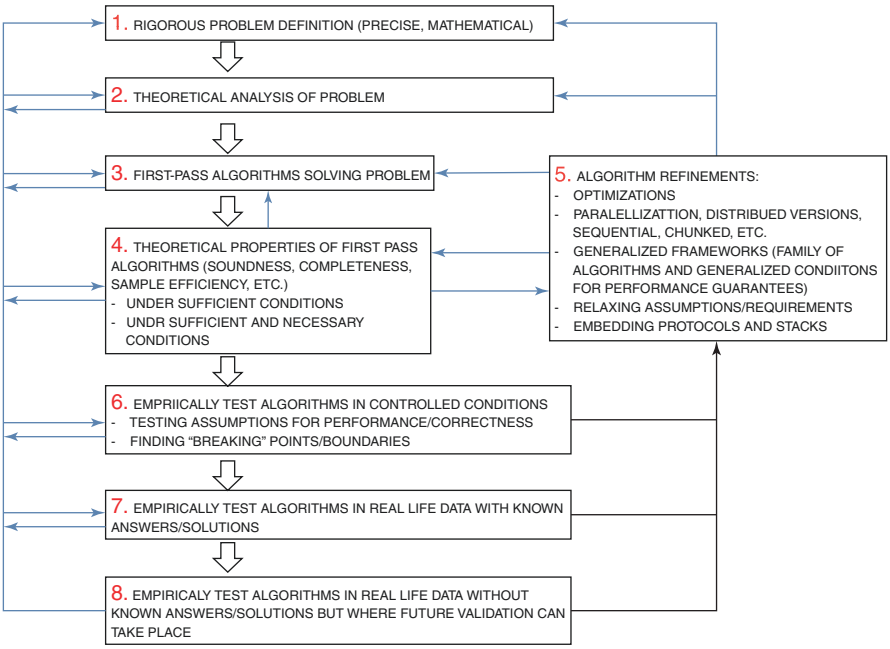


Fig. 4 Steps in developing and validating new AI/ML methods. Details in text. Notice the highly non-linear flow that describes the frequent need to go back and revise earlier steps if they do not lead to desired performance

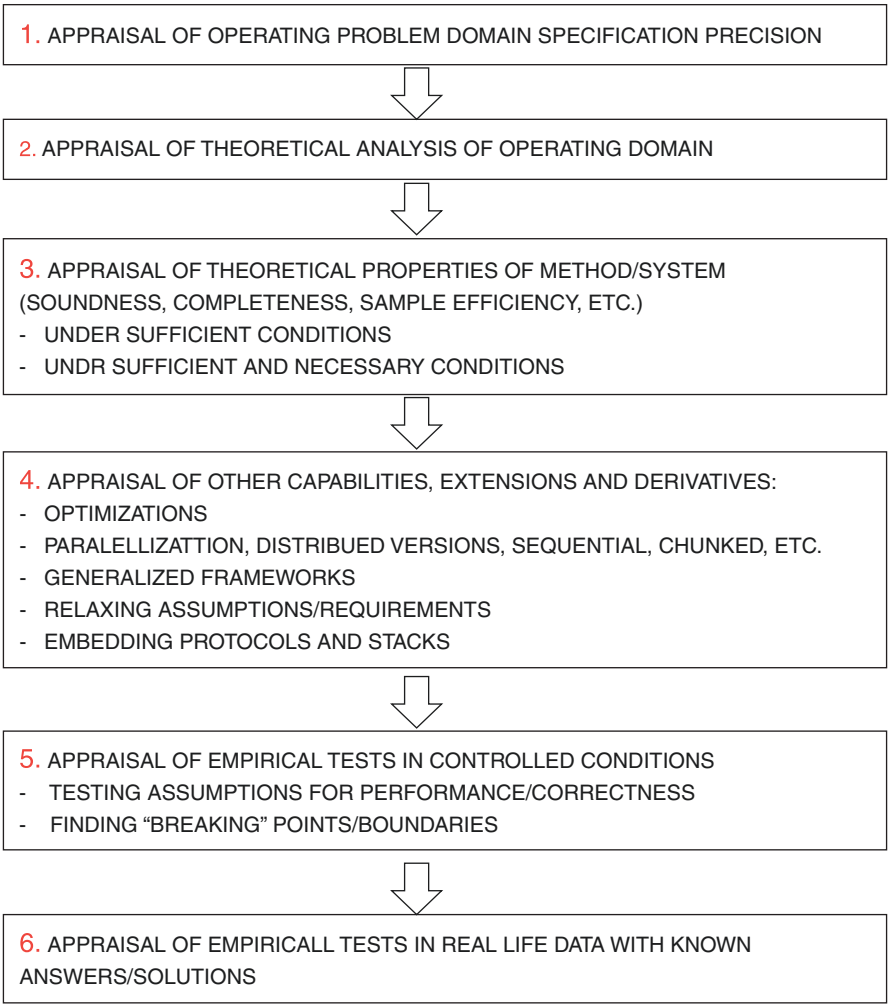


Fig. 5 Steps in evaluating existing AI/ML methods. Details in text. This process is linear since the evaluators do not concern themselves with fixing problems found in the appraisal

accurate and effective mapping of the data design and modeling onto the most appropriate AI/ML methods if they exist, or establishing the need for creating new methods (if methods do not exist or their properties are not up to par with the specifications of the problem solving effort at hand).

A mathematical-level description of the problem we wish to solve enables theoretical (mathematical, algorithmic, information theoretic, statistical, or other) appropriate analysis and proving that the problem can or cannot be solved, within specific performance requirements. This includes whether the problem can be solved in acceptably small time, storage, or sample size and input costs. Or, if it

cannot, whether, the development effort needs to focus on subclasses of the broad problem space that are solvable efficiently.

Unfortunately it is often the case that AI/ML is used by applying on data and expecting that "useful" patterns will be revealed, or "insights" will be generated without a precise description of what constitutes a desired solution.

Real-life relevance. Characteristic examples of this pitfall in healthcare AI/ML applications is seeking "useful patterns", "anomalies", "practice variation", "actionable insights", "subpopulations", or any number of other fuzzy goals that are hard to critically and conclusively evaluate in terms of meaningfully meeting the goals of the project. Another common pitfall is in the risk and predictive modeling domains where "accurate" algorithms (or models) are being sought without reference to the degree of accuracy expressed in some evaluation scale, that is required to advance the goals of the project at hand.

In the health sciences, a characteristic example of this pitfall is in bioinformatics analyses of high-dimensional omic molecular data where application of AI/ML methods is used to reveal "biological insights", "structure in data", "shape of the data", "clusters", "pathways", "signatures", "gene lists", often with very imprecise or inconsistent language about what exactly each one of these entities is meant to encompass, or about how discovering them will enable a specific scientific investigation, answer a concrete hypothesis, or generate results with practical clinical, scientific or technological value. Often such fuzzy goals and results are combined with ex post facto narratives by the authors of the study that overlay meaning and significance to the findings although they may inherently lack such meaning.

The ability of experts to create explanations in their field is an important component of their professional success [4]. A serious possibility exists that the domain expert investigators may be prone to creating (on the surface) convincing narratives around any set of results, even random ones, however. The term *Scientific Apophenia* describes the tendency to find evidence of order where none exists in scientific results [5]. The human mind is a powerful recognizer of patterns but is also subject to seeing patterns over random sequences and conversely, believing that non-random patterns are random [6].

Moreover some pattern may emerge out of random structures as a result of mathematical necessity as studied by Ramsey Theory [7]. For example, in any group of 6 people, by mathematical necessity, either at least three of them are (pairwise) mutual strangers or at least three of them are (pairwise) mutual acquaintances. The reader can see the relevance of this theorem to, e.g., interpreting bioinformatics analyses describing clusters among genes or other molecules. Such clusters even, if randomly generated, will contain structured patterns of known relationships and in turn will give credence to the (false) validity of previously unknown relationships.

The requirement for rigorous problem definitions prevents such problems early in the discovery process, at a stage where the very methods used for discovery are formulated and validated.

Another common pitfall of new method development related to defining the goals of methods is to define them in mathematical terms but without establishing

how the mathematical goal maps to the health science discovery goals, or clinical value, or how it addresses known limitations of existing methods. For example, the problem of topological clustering can be very precisely defined (and is a worthwhile and non-trivial mathematical theoretical endeavor), however it is not clear how solving this class of problems overcomes practical limitations of existing methods for concrete health problems.

A related pitfall to imprecise problem definitions is that of "re-inventing the wheel", that is creating an unnecessary new solution to a problem for which equally good or better solutions exist. In the history and practice of AI/ML re-invention of solutions to problems with established solutions is unfortunately rampant. For example, numerous pathway reverse engineering methods have been proposed in bioinformatics without leveraging or acknowledging the mathematically and algorithmically robust literature on causal discovery or improving upon their performance. Similarly the kernel regression method has been re-discovered several times in recent decades. In yet another example, numerous heuristic feature selection algorithms with "non-redundancy" properties have been proposed well after several sound, complete and efficient Markov Boundary algorithms (that solve this class of problem optimally) had been previously introduced, validated and successfully applied.

Pitfall 5.2
Evaluating the success of methods with poorly defined objectives, by employing ex post facto expert narratives as a proxy of "validity".

Pitfall 5.3
Defining the goals of methods in mathematical terms but without establishing how the mathematical goals map to the healthcare or health science discovery goals.

Pitfall 5.4
Reinventing the wheel: whereby a new method is introduced but it has been previously discovered yet ignored (willfully or not) by the "new" method developers.

Pitfall 5.5
Reinventing a method but make it worse to established methods (…"reinventing the wheel and making it square"!).

In-depth real-life example for step 1 (Rigorous problem definition): Development and evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data.

Investigators started by asking: how can we make this problem concrete? Biomarkers are a diverse group of conceptual entities which across the literature includes the following: (1) substitutes (proxy outcomes) for outcomes in clinical trials; (2) downstream effects of interventions that are indicative of toxicity and adverse events; (3) complex computable models (aka "signatures") that can be used to diagnose, prognosticate or precision-treat patients; Pathways, on the other hand, are causal subsystems in bigger healthcare or biological systems with defined functions and modularity. Pathways can be instrumental in revealing drug targets for new therapeutics. Biomarkers are involved in one or more drug target pathways [8–10].

One can see that the general concepts of biomarkers and related disease or drug pathways includes, therefore, both predictive and causal modeling which indicates that one would need to develop methods that seamlessly support both types of reasoning. In addition, such development is sensitive to the need for discovery and modeling with very high dimensional and often low sample size datasets.

One natural way to frame the predictive/prognostic/outcome surrogate biomarker discovery aspect is as a feature selection problem, and the causal aspect as a causal induction problem. Recall from chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML" that the feature selection problem in its simplest standard form can be stated as: *given a distribution P, with variable set V including a response variable ("target") T, and data sampled iid from P, find the smallest set of variables S in V such that S contains all predictive information about T in P.* The observational causal induction problem can be stated as: *given a distribution P, generated by a causal process C comprising a variable set V including a response variable ("target") T, and data sampled iid from P, without interventions on V, find the set of ordered relations (Vi,Vj) such that Vi directly causes Vj, whereby the causal semantics are as follows: Vi, temporally precedes Vj and a randomized experiment determining values of Vi yields changes in the distribution of Vj (compared to not manipulating Vi).*

These definitions directly point to three computational and mathematical frameworks for AI/ML modeling addressing the problem definitions: first, theory of relevancy; second, theory of causal graph induction; and third, theory of Bayesian Networks and in particular Markov Boundaries [11–14].

Step 2. Theoretical analysis of problem (complexity, problem space characteristics etc.)

The second step, after the method goals are precisely defined, is to conduct theoretical analysis that shows the characteristics *of the problem*, *across all possible AI/ML methods* that could be employed. This at first glance may seem to the non-expert exceedingly hard, or even impossible, since there is an infinity of possible AI/ML algorithms that one can devise. In practice, a precise and thoughtful problem definition in step 1 of the development process presented, often makes it feasible or even easy to derive properties by mapping the problem considered (i.e., by establishing its correspondence) to known problems which themselves have established properties.

For example, recall from chapter "Foundations and Properties of AI/ML Systems" one of the foundational achievements of computer science is the theoretical toolkit to prove that a problem class has a certain computational complexity. Similarly, the whole practice of Operations Research relies on having a catalog of prototypical problems with efficient solution algorithms such that practitioners can solve infinite problems just by mapping them to the smaller set of pre-established archetypal problem solutions [15].

If such mapping is not possible, the methods developers or evaluators can apply other established more granular and general-purpose techniques from the field of design and analysis of algorithms to understand the feasibility and hardness of the problem space [16].

Real-life example for step 2 (Theoretical analysis of problem). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

How one would go about characterizing the theoretical properties of the problem as formally defined in step1? Recall from chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science" that existing theory of relevancy includes the Kohavi-John framework (K-J) which differentiates between *strongly relevant, weakly relevant and irrelevant features*. Strongly relevant features in K-J theory are features that have unique information and can never be dropped by feature selection without loss of predictive signal. Weakly relevant features are predictive but lack unique signal, thus can be dropped by feature selection without loss of predictivity. Irrelevant features carry no signal so they are effectively noise (for the target T) and can be dismissed by feature selection. So there is a theory guiding the discovery of predictive biomarkers. What about the causal aspect?

As we saw, the connective tissue between causation, feature selection and causal discovery is given, locally around a target variable T, by the Markov Boundary (MB) of T. Specifically, a Markov Blanket of T is any set of variables that renders all other variables in the data independent of T, once we know the Markov Blanket variables. A Markov Boundary is a minimal

Markov Blanket which means that we cannot discard any member of the Markov Boundary without losing its Markov Blanket property. Moreover in a vast family of distributions (the majority of all possible distributions including all distributions modeled by classical statistics), called Faithful Distributions, and when no latent variables are present,: (a) there is always a single Markov Boundary and (b) the Markov Boundary comprises the direct causes, plus direct effects plus direct causes of the direct effects of T (so-called "spouses" of T). Thus, the Markov Boundary contains the local causal pathway around T (minus those spouses that lack causal edges to T). Moreover because BNs are probability and decision theoretically sound, MBs are also consistent with probability and decision theory. Finally, the Markov Boundary feature selection is connected with K-J relevancy since in Faithful distributions the strongly relevant features are the members of the Markov Boundary [11–14].

So far the problem space was well-characterized, looked feasible and the natural next question was what are algorithms that solve it?

Step 3. First-pass algorithms solving problem

The third stage of new AI/ML method development is a first attempt at identifying or (if none exists) creating an algorithmic method that solves the problem as previously defined and analyzed. If the problem has been precisely defined and its properties established in steps 1 and 2, then it is often easy to modify existing methods or put together a first algorithm that solves the problem. Typically this first-pass solution is not meant to be optimally efficient, as such optimization is attempted in subsequent steps. Existing method evaluation may also apply here if the existing method is an early-stage one.

Real-life example for step 3 (First-pass algorithms solving the problem). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, the theoretical specifications and analysis described above provide solid footing for moving to first-pass algorithm development. Algorithms that solve the example problem do exist (chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science") and have solid properties, however for illustrative and pedagogical purposes we will consider their development at the time that they did not exist and had to be invented.

Developing new such algorithms was a necessity in the early 00s because at the time there was no sound and scalable algorithms for discovery of Markov

Boundaries from data. In principle, one could use existing causal graph induction algorithms to discover the whole graph and then extract the Markov Boundaries from it. However the algorithms were not scalable beyond approx. 100 variables in practice, and it had been shown that learning the causal graph with Bayesian search and score algorithms was NP-Hard, whereas the Conditional Independence and constraint-based algorithm family was worst case exponential (chapter "Foundations of Causal ML"). Heuristic algorithms introduced by Kohler and Sahami in 1996 and by Cooper et al. in 1997 were informative first attempts but not sound. In the Kohler-Sahami case they were also poor empirical performers, whereas the Cooper et al. algorithm was better performer in small data but was not scalable in compute time or sample size. Another recent (at the time) algorithm by Margaritis and Thrun was sound but not scalable and also not sample efficient. With these necessities in mind Tsamardinos and Aliferis invented a novel sound and scalable algorithm IAMB, and variants, that could be instantiated in a variety of ways (e.g. by combining it with full-graph algorithms for intermediate results pruning or post-processing) [18–21].

Step 4. Theoretical properties of first pass algorithms: focus on representation power, soundness and completeness.

Once a first algorithm that *prima facie* solves the problem has been created, its key theoretical properties should ideally be established, typically under (a) sufficient or necessary conditions, or (b) sufficient and necessary conditions. If analysis shows that no such conditions exist, or that they are too narrow and unworkable in real life conditions, then revisiting step 3 is mandated and steps 3 and 4 are iterated until a method has been identified that guarantees soundness and completeness (or reasonable approximations thereof) under real-life realistic conditions.

Real-life example for step 4 (Theoretical properties of first pass algorithms: focus on representation power, soundness and completeness). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example and from the viewpoint of when the corresponding development steps were conducted, the IAMB algorithm family was theoretically sound and complete. In preliminary empirical analyses its inventors established that it is both sound and scalable. IAMB was however a *definitional* Markov Boundary algorithm meaning it would apply the definition of the Markov Boundary in each step of its operation. This entailed that it would conduct conditional independence tests with large sets of variables which increased its sample size needs to exponentially large to the number of conditioning variables. So, refinements and optimizations were needed which led to a second (more refined) family of algorithms (see below).

Step 5. Algorithm refinements, extensions and derivatives

Step 5 is a multi-faceted and critical step in the sense that its success will determine the practical utility of the new method. This is the step where various optimizations and adaptations to real life performance requirements will take place. This step has several sub-steps that may require many years' worth of incremental improvements. Hardly ever the full range of optimizations is accomplished close to the first time a method appears in the literature. It is not uncommon for such efforts to constitute a career-long research program of the method innovators and their teams, as well as independent researchers. The refinement sub-steps comprise:

Step 5.a. Performance Optimizations that cover achieving high computational efficiency for learning and for using models; optimizations for other costs (for example cost to obtain and store input data and run analyses on a compute environment, either at model discovery or at model deployment time); efficient space complexity for learning, storing and using models; efficient sample complexity and establishing learning curves and broad power-sample requirements.

Real-life example for step 5.a. (Performance Optimizations - Sample efficient algorithms and going beyond local pathways). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, the second-pass algorithms designed to overcome IAMB's sample size inefficiency described earlier was HITON and MMMB. These were *compositional* algorithms, that is, instead of applying the definition of Markov Boundary, they composed it edge-by-edge (exploiting the link between graphical and probabilistic Markov Boundaries and causality).

At the core of the causal discovery problem from observational data there is a foundational theorem (chapter "Foundations of Causal ML") that states that to establish a direct edge between variables Vi and Vj one must establish the conditional dependence of these variables given every subset of the remainder of the variables in the data [13]. This is clearly an exponential cost operation in both compute time and sample, which becomes super-astronomical in datasets with more than a few dozen variables. Thus smart algorithms are needed to exploit the sparsity of causal processes and identify quickly a single subset that shows independence so that the vast majority of tests can be omitted from computations.

With this asymmetry in mind, the designers of HITON and MMMB sought to apply informed search functions that would identify quickly a single subset needed to establish that variable Vi was not directly linked to T, for every Vi in the data. Whereas the algorithms are still worst-case exponential (because the problem is itself worst-case exponential regardless of algorithm), their exponentiality is directly linked to the connectivity of the underlying causal structure. For sparse or predominantly sparse causal data generating processes (as most biological networks are, for example), the majority of the causal

process can be identified fast if **the algorithms' time complexity is adaptive to the connectivity**. Both HITON and MMMB are locally adaptive to the hardness of the causal problem at hand. Additional variants return direct causes and effects only (HITON-PC, MMPC) and Markov Boundaries (HITON-MB, MMMB).

Because of the compositional nature of this second generation of algorithms these investigators and their collaborators were able to introduce algorithm variants that discover not just the Markov Boundary for T, but also *the local causal pathways (causal neighborhood) only* around T (i.e. without need to find spouses and remove them with post processing), also *local causal regions* of depth *k* (depth specified by the user), and *the full causal graph* by inducing all local causal edges around every variable in the data (algorithm MMHC and it generalized family LGL) [22–24].

Step 5.b. Parallelization, distributed/federated, sequential, chunked, versions [25–27]. These are derivative and enhanced forms of the main algorithmic solution with the following properties:

Parallel algorithms: can be run in parallel computing architectures and environments whereby the computational steps are divided among many processors. The parallelization can be *coarse, intermediate, or fine grain*, depending on how large and complicated are the unit computations divided among the parallel processors.

Distributed/federated algorithms: operate across federated and distributed databases which exist in diverse locations without the need to bring all data into a centralized database and computing environment.

Sequential algorithms operate in steps corresponding to incremental availability of input data, for example with increasing sample size over time, or with increasing sets of variables over time. At each processing step, a different set of results is obtained that over time increases in quality (i.e., converges to the right solution or approximation thereof).

Chunked algorithms address the situation where data is so large that overwhelms the memory limits of the computing environment. The data is divided then in chunks, each one of which fits in memory, and analysis proceeds across all chunks until all data is analyzed and final results obtained.

Real-life example for step 5.b. (Performance Optimizations - Parallelization, distributed/federated, sequential, chunked, versions). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, and motivated by the need to process datasets with vast numbers of variables not fitting in single-computer memories of the time, the investigator team was also forced to invent a chunked version of IAMB and HITON. Because of the nature of these algorithms it was immediately obvious that they can also be modified to obtain parallel, distributed, and sequential versions as well. Over the years, the investigator team conducted massive experiments in parallel compute clusters taking advantage of these algorithms. These algorithms also gave important insights on the feasibility and requirements for **sound federated** Markov Boundary and causal discovery. For example, it was established that for sound federated/distributed Markov Boundary discovery exactly two passes of local processing was required plus one global step with the results from the first two passes, and a subset of variables had to be shared among all nodes depending on intermediate results [1, 28].

Step 5.c. Relaxing assumptions/requirements. This step corresponds to efforts to relax the assumptions guaranteeing its properties and broaden the space of conditions under which the new or existing method will have the desired guaranteed properties.

Real-life relevance. For example, extending Bayesian classifiers from restricted distributions with Naive Bayes, to algorithms that can operate on all discrete distributions. Other examples include: extending from discrete to continuous decision trees, extending single tree models to ensembles of trees (e.g., Random Forests) that are more robust to sampling variance, extending artificial neural networks from linearly separable functions to non linearly-separable ones, extending SVM binary classification to multi-class classification, extending SVMs from noiseless data to noisy data, extending linear to non-linear SVMs, extending standard Cox Proportional Hazards regression to accommodate time-dependent covariate effects, extending causal discovery algorithms that require no hidden (aka latent) variables to ones that can operate in the presence of latents, etc. (see chapters “Foundations and Properties of AI/ML Systems”, “An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science”, and “Foundations of Causal ML” for details and more examples).

In-depth example for step 5.c. (Performance Optimizations: Relaxing assumptions/requirements allowing equivalence classes, latent variables, guided experimentation). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, the methods outlined so far depended on two fundamental assumptions: one is Faithful distributions and the other is causal sufficiency (i.e., no unmeasured confounders). To address the former, the development team introduced new algorithms addressing distributions with *information equivalences*, i.e., distributions where several variable groups can have the same information regarding the target response. Such situations are very common in omics data, complex survey data, clinical data, and many other types of biomedical data. The result of information equivalency is the existence of multiple (not just one) Markov Boundaries and ambiguity of the causal pathways.

Specifically, in such distributions there is an equivalence class containing multiple statistically indistinguishable MBs and local causal edges and the size of this class can be exponential to the number of variables. Statnikov et al. introduced a family of algorithms called TIE* which extract from data all MBs and local causal neighborhoods. TIE* algorithm family members are sound, complete and adaptable to the distribution at hand by choice of conditional independence tests and component subroutines (used for single Markov Boundary induction).

The second relaxation concerning latent variables was addressed by post-processing the main algorithms' results with algorithms that can detect presence of hidden variables (e.g., IC* and FCI, see chapter "Foundations of Causal ML"). Such algorithms could not be used for end-to-end analysis because they are not scalable and in most cases they are also error prone empirically.

Another important algorithmic extension addressing both equivalence classes and latent variables was the introduction of algorithms that resolve these ambiguities by limited algorithm-guided experimentation. Specifically the ODLP algorithm family combines using MB and local neighborhood algorithms plus equivalency class algorithms and guides an experimenter to conduct a series of experiments that resolve statistical ambiguity due to latents and equivalence class due to information equivalency. The ODLP algorithm attempts to minimize the number of experiments and has worst-case number of experiments that is at most the total number of variables in the equivalence class of the local pathway [29–31].

Step 5.d. Generalized frameworks (generalized family of algorithms and generalized conditions for performance guarantees).

This step involves extending the new method to a more general family of inter-related similar methods. It also involves establishing testable rules for instantiating the family into specific methods in that class, and testable rules for guaranteeing that the properties of the family will be shared by every method in that family (without further need to prove these properties of empirically test them).

It is not always obvious if such generalizations are possible, and if yes how to accomplish them. Thus it is not always pursued, especially in initial stages of developing a new method. But whenever it is possible, it confers a number of important benefits which we summarize here. Developing a generalization of a fundamental method explains in mathematically precise terms how the core method can be modified so that:

(a) It can address slightly different problem instance classes and situations;
(b) To allow for modifications that do not alter its foundational nature;
(c) It will enable other method developers to create variations without having to undergo the whole development process from scratch and at the same time inherit the main performance and other properties of the core method;
(d) It will help understand other methods and their properties by showing how they relate to the generalized core method;
(e) It will prevent confusion about apparently similar methods with different properties or apparently different methods with same properties; and
(f) It will protect scientific priority claims and commercial intellectual property, by establishing which methods are just variations or derivatives of the original core methods, especially when such variations and derivatives were anticipated by the generalized framework.

Real-life relevance. Examples of generalized families of inter-related algorithms are many and include: the Best-First-Search algorithm family (chapter "Foundations and Properties of AI/ML Systems"), the General Linear Model (GLM) family (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"), the penalty+loss regularized classifier family (chapter "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science"), the Generalized Local Learning (GLL) and Local to Global causal discovery (LGL) algorithm families (chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML"), and the TIE* family for equivalence class modeling (chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science").

In-depth example for step 5.d. (Performance Optimizations: Generalized frameworks). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, of developing principled and scalable biomarker and pathway discovery algorithms, it was realized within the team developing these algorithms, that infinite variations could be had that would preserve the soundness, completeness and other desired properties of the first algorithms introduced. To facilitate the study and further analysis and development of the families of the algorithms in a systematic way and minimizing confusion, Aliferis at al introduced a generalization of HITON-PC/HITON-MB and MMMP/MMMB, and of MMHC.

The former was termed GLL family and the latter LGL. Around the same time Statnikov and Aliferis introduced the generalized TIE family termed TIE*, and later generalized versions of ODLP and of parallel/distributed/sequential and chunked variants. Aliferis et al. introduced the notion of a 2-part **Generative Algorithm Framework** comprising:

1. *A general template* statement of the algorithm family.
2. *Admissibility rules* for instantiation of the template components.

If the admissibility rules are followed when instantiating components, then this guarantees that the instantiated algorithm have guaranteed properties *without the need for de novo theory or proofs*.

Figure 6 illustrates one instantiation of the GLL-PC generative template algorithm (shown in 6.a) by the admissibility rules of the Generative Algorithm Framework (shown in 6.b) to yield an infinity of algortihms with guaranteed properties such as the original MMPC (presented in 6.c). Contrary to the intricacy of the original MMPC, however, the generative framework describes the whole family of algorithms by specification of a few simple components.

Moreover, a Generative Algorithm Framework does not allow just the re-creation and compact representation of pre-existing algorithms, but as shown by these investigators, several new instantiations of the original generalized algorithms were demonstrated and they matched or exceed empirical performance of the original set of algorithms in validation data. One of the new algorithms is shown in part (6.d) of the figure. The new instantiations exhibited different traces of navigating the solution space toward the correct (same) output, demonstrating that they are not just a rehash of known algorithms [1, 29–32].

a

GLL-PC: High-level pseudocode and main components of Generalized Local Learning - Parents and Children. Returns PC(*T*)

1. U ← GLL-PC-nonsym(*T*) // first approximate *PC*(*T*) without symmetry check
2. For all $X \in U$
3. If $T \notin$ GLL-PC-nonsym(*X*) then $U \leftarrow U \setminus \{X\}$ // check for symmetry
4. Return *U* // true set of parents and children

GLL-PC-nonsym(*T*) *// return a set which is a subset of EPC(T) and a superset of PC(T)*

1. Initialization
   a. Initialize a set of candidates for the true *PC*(*T*) set: *TPC*(*T*) ← *S*, s.t. $S \subseteq V \setminus \{T\}$
   b. Initialize a priority queue of variables to be examined for inclusion in *TPC*(*T*): OPEN ← $V \setminus \{T \cup TPC(T)\}$
2. Apply inclusion heuristic function
   a. Prioritize variables in OPEN for inclusion in *TPC*(*T*);
   b. Throw away non-eligible variables from OPEN;
   c. Insert in *TPC*(*T*) the highest-priority variable(s) in OPEN and remove them from OPEN
3. Apply elimination strategy to remove variables from *TPC*(*T*)
4. Apply interleaving strategy by repeating steps #2 and #3 until a termination criterion is met
5. Return *TPC*(*T*)

Fig. 6 Example of Generative Algorithm Framework. GLL-PC shown here and generating two from an infinite number of members of this family of algorithms (pre-existing MMPC algorithm in part (**c**) and new algorithm semi-interleaved HITON-PC with symmetry correction in part **d**), with guaranteed properties, merely by instantiating a simple general template statement (part **a**), following a small set of admissibility rules (part **b**)

b

GLL-PC: Admissibility rules

1. The inclusion heuristic function should respect the following requirement:

   // *Admissibility rule #1*

   All variables $X \in PC(T)$ are eligible for inclusion in the candidate set $TPC(T)$ and each one is assigned a non-zero value by the ranking function. Variables with zero values are discarded and never considered again.

   Note that variables may be re-ranked after each update of the candidate set, or the original ranking may be used throughout the algorithm's operation.

2. The elimination strategy should satisfy the following requirement:

   // *Admissibility rule #2*

   All and only variables that become independent of the target variable T given any subset of the candidate set $TPC(T)$ are discarded and never considered again (whether they are inside or outside $TPC(T)$).

3. The interleaving strategy iterates inclusion and elimination any number of times provided that iterating stops when the following criterion is satisfied:

   // *Admissibility rule #3*

   At termination no variable outside the set $TPC(T)$ is eligible for inclusion and no variable in the candidate set can be removed at termination.

Fig. 6 (continued)

c

Algorithm 1 $\overline{MMPC}$ Algorithm

```
1:  Procedure MMPC (T,𝒟)
        Input: target variable T; data 𝒟
        Output: the parents and children of T in any Bayesian
        network faithfully representing the data distribution
        %Phase I: Forward
2:      CPC = ∅
3:      repeat
4:          ⟨F, assocF⟩ = MaxMinHeuristic(T; CPC)
5:          if assocF ≠ 0 then
6:              CPC = CPC ∪ F
7:          end if
8:      until CPC has not changed

        %Phase II: Backward
9:      for all X ∈ CPC do
10:         if ∃S ⊆ CPC, s.t. Ind (X; T | S) then
11:             CPC = CPC \ {X}
12:         end if
13:     end for

14:     return CPC
15: end procedure

16: Procedure MaxMinHeuristic(T,CPC)
        Input: target variable T; subset of variables CPC
        Output: the maximum over all variables of the minimum
        association with T relative to CPC, and the variable that
        achieves the maximum
17:     assocF = max_{X∈V} MinAssoc(X; T|CPC)
18:     F = arg max_{X∈V} MinAssoc(X;T|CPC)
19:     return ⟨F, assocF⟩
20: end procedure
```

Fig. 6 (continued)

d

Semi-Interleaved HITON-PC with symmetry correction

Derived from GLL-PC with following instantiation specifics:

Initialization

$TPC(T) \leftarrow \emptyset$

Inclusion heuristic function

a. Sort in descending order the variables X in OPEN according to their pairwise association with T, i.e., $Assoc(X, T|\emptyset)$.
b. Remove from OPEN variables with zero association with T, i.e., when $I(X, T|\emptyset)$
c. Insert at end of $TPC(T)$ the first variable in OPEN and remove it from OPEN

Elimination strategy

If OPEN=Ø
  For each $X \in TPC(T)$
    If $\exists \boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\boldsymbol{Z})$ remove X from $TPC(T)$
Else
  $X \leftarrow$ last variable added to $TPC(T)$ // in step 2 of GLL-PC-nonsym
  If $\exists \boldsymbol{Z} \subseteq TPC(T)\backslash\{X\}$, s.t. $I(X, T|\boldsymbol{Z})$ remove X from $TPC(T)$

Interleaving strategy

Repeat
  steps #2 and #3 of GLL-PC-nonsym
Until OPEN=Ø

Fig. 6 (continued)

Step 5.e. Nested algorithms, embedding protocols and stacks, interactions with data design.

Real-life relevance. Per traditional computer science and AI/ML practice, algorithms can be used as subroutines in higher complexity algorithms. For example, decision tree induction algorithms can be used inside Random Forests. Weak learning algorithms can be used as components of boosting algorithms. Algorithms of various kinds can be components of Stacked ML models, and so on.

Generative Algorithm Frameworks can also be used to create more complicated and hierarchically- nested algorithms families, creating **nested systems of generalized algorithms tackling an increasingly complex problem-solving AI/ML construct**. For example the GLL-PC generative family is nested in the GLL_MB family, which is nested in the TIE* family, which is nested in the ODLP* family. The nesting does not force use of the most complex level algorithm. To the contrary, the algorithms with smallest complexity that solve a problem are sufficient for that problem.

Algorithms and their implementations are the conceptual and scientific backbone and "engines" of real-life AI/ML. Complicated tools and systems, designed to solve health science and healthcare problems are almost always organized in complex data science "stacks".

AI/ML (or data science) Stack: a hierarchically-integrated *architecture* for AI/ML software system delivery comprising data input management at the lower level, going upward to model selection, to error estimation, to error management, to decision support delivery and end-user interfacing, to embedding and healthcare integration, to model monitoring and full model lifecycle support (see chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models" for details).

At the core of the ML stack is the *ML protocol* (see chapter "Foundations and Properties of AI/ML Systems" and "The development process and lifecycle of clinical grade and other safety and performance-sensitive AI/ML models" for details).

ML Protocol. A ML system *architecture* implements a ML method which can be understood as a combination of data design, algorithm, and model selection procedure that ideally will incorporate an error estimator procedure. The higher-level algorithm that combines the ML algorithms, data processing subroutines, model selection strategies and error estimators used, is the ML protocol.

AI/ML "Pipelines", Automodelers, and Platforms. These represent discrete *software implementation* entities embodying implementation of the chosen algorithms and protocol, plus all other layers of the full AI/ML stack and are designed to be used reproducibly, in either fully automatic mode ("automodeler") or semi-automatically, or as a component of a broader modeling system (a "pipeline"). Platforms refer to even more complex software systems with additional facilities for user experimentation, model sandboxing, training, model development, integration with other enterprise systems, etc.

Best Practice 5.5
The properties of a ML algorithm can be negatively or positively affected by the ML protocol to extreme degrees (see chapter "Lessons Learned from Historical Failures, Limitations and Successes of AI/ML In Healthcare and the Health Sciences. Enduring problems, and the Role of Best Practices" for several important case studies that show the immense and often underappreciated practical consequences). Similarly the data design can negatively or positively affect the ML protocol and its embedded algorithms to extreme degrees. Therefore, it is imperative to design AI/ML methods taking into account any positive or negative interactions of data design with the protocols and embedded algorithms employed.

In-depth example for step 5.e. (Performance Optimizations: Nested algorithms, embedding protocols and stacks, interactions with data design). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, of developing principled and scalable biomarker and pathway discovery algorithms, a core choice was made to design the algorithms and ML protocols with a focus on nested balanced cross validation as a "canonical" preferred model selection and error estimation protocol (see chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models" for more details). The anticipated data designs were primarily case-control or natural, cross-sectional or longitudinal, i.i.d. sample designs. As long as distributions were faithful or TIE, the algorithms were guaranteed to exhibit well-defined and desirable soundness, completeness, computational and sample efficiency properties. Deviations from these "canonical" designs were also tolerated but would need careful tailoring of the methods to data designs outside these "canonical" specifications.

Many of the produced algorithms were embedded in software with data ingestion/transform, model selection, error estimation and adaptive selection among the core methods and state of the art comparators. Examples include the GEMS auto modeler system for microarray analysis, the FAST-Aims auto modeler system for Mass Spectrometry analysis, and several psychiatry-oriented, as well as bioinformatics, clinical and translational predictive and causal modeling stacks and pipelines. The construction of the auto modelers was further guided by extensive benchmarking of these and comparator methods and cross-referencing to expert analyses in the literature on the same datasets. These empirical performance benchmark studies measured a level of performance that matched or exceeded that of faculty level experts in AI/ML with the added advantage of almost immediate analysis, at no cost other than an inexpensive laptop or desktop [33–36].

Figure 7 shows components of the GEMS auto modeler.

Step 5.f. Explanation, clarity, and transparency

This step addresses the needs for **transparency and clarity in the method's semantics, syntax, inference mechanisms and mode of operation**, and including its transparency, explainability, and ability for humans to inspect and understand the **models produced by the new AI/ML method** and of the operations that led to these models.

We frequently refer to AI/ML methods as being "black box, and "transparent", "explainable", or "open box".

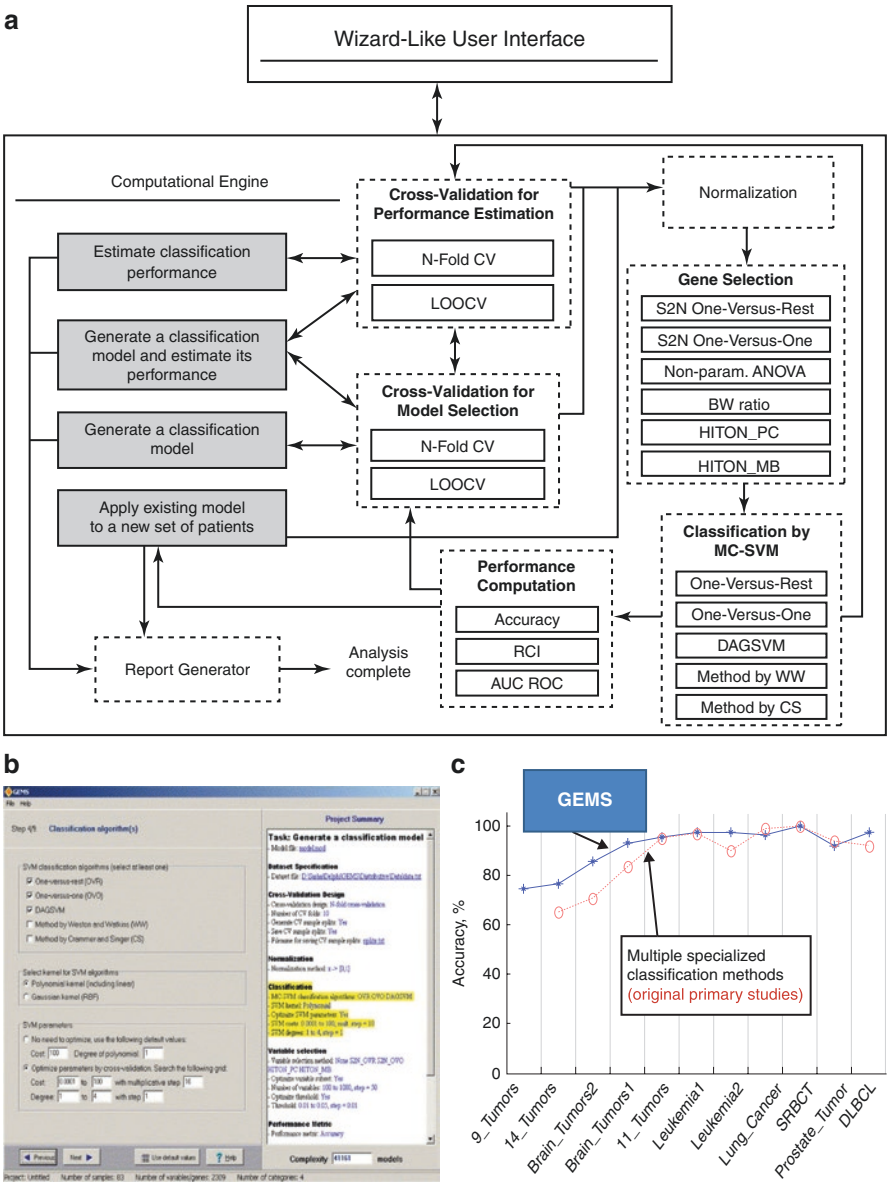


Fig. 7 Example of an auto modeler system. Components of the GEMS auto-modeler shown. (**a**) System architecture. (**b**) "Wizard"-like user interface (the user could enter values for specific parameters, load analysis templates, or run in fully automatic mode). (**c**) Empirical results of GEMS analyzing datasets from the relevant literature and comparison with performance of original studies conducted by experts. (**d**) Algorithms benchmarked to inform the construction of the system [37]. (**e**) Algorithms chosen to be included in the system based on results of the benchmarks

d

Classification algorithms
- K-Nearest Neighbors
- Backpropagation Neural Networks
- Probabilistic Neural Networks
- Multi-Class SVM: one-versus-rest
- Multi-Class SVM: one-versus-one
- Multi-Class SVM: DAGSVM
- Multi-Class SVM by Weston and Watkins
- Multi-Class SVM by Crammer and Singer
- Weighted Voting: one-versus-rest
- Weighted Voting: one-versus-one
- Decision Trees: CART

Ensemble classification algorithms

Based on output of Multi-Class SVM methods
- Majority voting
- Decision Trees: CART
- Multi-Class SVM: DAGSVM
- Multi-Class SVM: one-versus-rest
- Multi-Class SVM: one-versus-one

Based on output of all classifiers
- Majority voting
- Decision Trees: CART

Computational experimental design
- Leave-one-out cross-validation for performance estimation (outer loop) and 10-fold cross-validation for model selection (inner loop)
- 10-fold cross-validation for performance estimation (outer loop) and 9-fold cross-validation for model selection (inner loop)

Gene selection methods
- Signal-to-noise ratio in one-versus-rest fashion
- Signal-to-noise ratio in one-versus-one fashion
- Kruskal-Wallis nonparametric one-way ANOVA
- Ratio of genes between-categories to within-category sum of squares

Performance metrics
- Accuracy
- Relative classifier information (entropy-based performance metric)

Statistical comparison among classifiers
- Custom randomized permutation procedure

e

Classification algorithms
- Multi-Class SVM: one-versus-rest
- Multi-Class SVM: one-versus-one
- Multi-Class SVM: DAGSVM
- Multi-Class SVM by Weston and Watkins
- Multi-Class SVM by Crammer and Singer

Gene selection methods
- Signal-to-noise ratio in one-versus-rest fashion
- Signal-to-noise ratio in one-versus-one fashion
- Kruskal—Wallis nonparametric one-way ANOVA
- Ration of genes between-categores to within-category sum of squares
- HITON_PC
- HITON_MB

Normalization techniques
- For every gene x [a, b]
- For every gene x [x – mean (x)]/std. (x)
- For every gene x x/std. (x)
- For every gene x x/mean (x)
- For every gene x x/median (x)
- For every gene x $x/||x||$
- For every gene x x – mean (x)
- For every gene x x – median (x)
- For every gene x $|x|$
- For every gene x $x + |x|$
- For every gene x log(x)

Computational experimental design
- Leave-one-out cross-validation for performance estimation (outer loop) and *N*-fold cross-validation for model selection (inner loop)
- *N*-fold cross-validation for performance estimation (outer loop) and (N–1)-fold cross-validation (inner loop)
- Leave-one-out cross-validation for model selection
- *N*-fold cross-validation for model selection

Performance metrics
- Accuracy
- Relatives classifier information (entropy-based performance metric)
- **Area under ROC curve (AUC)**

Fig. 7 (continued)

For the purposes of the present book we will define:

Black box AI/ML methods and/or models: are methods or models for which the user (and possibly even the developer) know only the inputs and outputs but not the internal operation; or, alternatively, the internal operation is accessible but it is not readily fully interpretable by humans.

Transparent (aka explainable, open, or white, or clear box) are methods or models for which the user and the developer know not only the inputs and outputs but also the internal operation which is readily fully interpretable by humans.

The transparency of the new method and its models are critical for (a) debugging the method and models (see chapter "Characterizing, Diagnosing and Managing the Risk of Error of ML and AI Models in Clinical and Organizational Application"), and (b) managing risks associated with its use and establishing trust in the AI/ML method and its outputs (see chapters "Artificial Intelligence (AI) and Machine Learning (ML) for Healthcare and Health Sciences: the Need for Best Practices Enabling Trust in AI and ML" and "Characterizing, Diagnosing and Managing the Risk of Error of ML and AI Models in Clinical and Organizational Application"), as well as "Regulatory Aspects and Ethical Legal Societal Implications (ELSI)").

When AI/ML methods are transparent we can also explain and justify their results including on a case-by-case, input-output basis. However there is a somewhat subtle but important distinction between explanation by translation and other forms of justification which is germane to the purposes of best practices in AI/ML.

Justification of a method or a model (and their outputs) is any argument that supports the validity of the method, the models produced by it, and the outputs produced by the models.

Explanation of a method or model (and their outputs) by functional translation is a justification of the method or model's logic by fully equivalent translation in humanly-understandable language.

Where:

Human understandable language includes natural language but also other formalisms readily understood by humans such as decision diagrams, decision trees, propositional and first order logic, etc.

A common pitfall in AI/ML is providing peripheral/oblique and thus inadequate justifications of the model and its decisions which may be persuasive in some settings, but do not "open" the black box in the sense of creating a human-understandable and mathematically equivalent model to the black box model. For example, consider a hypothetical similarity-based "explanation" module of a neural network AI/ML model using exemplars. The module attempts to justify model decisions on a case C, by presenting a small number of cases similar to C, with gold standard labels same as the model's prediction for C. Because the neural network does not make decisions based on similarity to exemplars, this whole justification exercise amounts to a "sleight of hand". It can also be argued that local simple (e.g. linear) approximations to a very complex decision functions underlying the black box models, attempt to justify the decisions of the model are generally meaningful and trustworthy by examining a simplified version of the local individual decisions of the neural net but not explaining the global complex inductive logic of the model and its generalization.

We also highlight the distinction between "open source" and "closed source" software implementing AI/ML methods and models.

Open source software is software whose source code is at minimum open for inspection. Depending on the specific licensing terms of open source software, it may or not come with other rights granted, such as the right to modify the source code and release such modifications under same or different licensing terms.

Closed source (or proprietary) software is software with restrictions on code inspection, use, modifications/derivatives creation, sharing derivatives, or sharing the software.

A pitfall in AI/ML is conflating "open source" for open box" and "closed source" for "black box" software.

In reality an open source implementation of a method or model does not entail "open box" status, and a "closed-source" software does not entail "black box status. Artificial neural Network models are notoriously "black box" ones, yet this does not change if we have access to the code implementing them, or the right to modify and re-distribute implementation code. Conversely, a closed source implementation of, for example the ID3 decision tree algorithm, can be transparent in terms of both the algorithm used (which is well understood and openly accessible in the literature, and may also be accessible by licensed users), and the models it produces (i.e., decision trees which are intuitive and readily understood by humans). The latter case requires that the disclosures of the algorithms used by the software are (a) complete and (b) accurate.

Pitfall 5.6
Providing persuasive but peripheral/oblique justifications that lack fidelity to the AI/ML method, the models produced by them and their decisions.

Pitfall 5.7
Confusing "open source" for "transparent" and "closed source" for black box".

In this section we covered only introductory concepts about explainability, since at the method development stage it is typically quite clear whether the method and its models are interpretable. In chapter "The development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models" we will delve into the details of explainable AI (XAI) [38], interpretable ML [39] and some of the key techniques and nuances of explaining back box models.

In-depth example for step 5.f. (Performance Optimizations: Explanation, clarity, and transparency). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, of developing principled and scalable biomarker and pathway discovery algorithms, a core choice was made to design the algorithms and ML protocols within a causal graphical modeling framework. Causal graphs are very intuitive representations of causality, and Markov Boundaries have an intuitive causal and probabilistic meaning and lead to very compact and transparent models as well. Depending on the classifier used (e.g., decision trees, conditional probability tables/heatpmaps, rules, regression) results can be readily reviewed by human non-experts. In addition, the team members devised a more sophisticated method whereby if the MB was used to create black box models (because they were optimally predictive) or for any black box model for that matter, data would be sampled from the black box model and then a meta learning step involving MB induction and learning decision tree models over the MB that were equivalent to the black box, but perfectly transparent to humans. This explanation method was used to understand the black box reasoning of human physicians in the diagnosis of melanoma by Sboner et al. [40, 41]. More details are presented in chapter "The Development Process and Lifecycle of Clinical Grade and Other Safety and Performance-Sensitive AI/ML Models" in the context of general explanation of black box models.

Step 6. Empirically test algorithms in controlled conditions

The next stage in the development of new AI/ML methods is testing them in controlled conditions whereby they are given data that comply with the sufficient or necessary and sufficient conditions for their theoretical performance properties to hold. In these empirical tests, developers also vary parameters such as sample size, variable measurement noise, strength of signal for the functions to be learnt, dimensionality of the data as well as various parameters relevant to the specific characteristics of the learning methods (as dictated by their properties established in previous steps) The methods are also tested with data where assumptions are violated to varying degree and the effect on performance characteristics is studied.

There are four important types of controlled condition data testing:

(a) **Simulated data**: where developers or other evaluators first define a mathematical or computational model representing the data generating function to be learned. Then they sample from this function via simulation, give the data to the AI/ML method and study its performance characteristics.

(b) **Label-reshuffled data**: these are real data where evaluators randomly re-assign the response variable's values (aka "labels") across the dataset (Fig. 8). This has

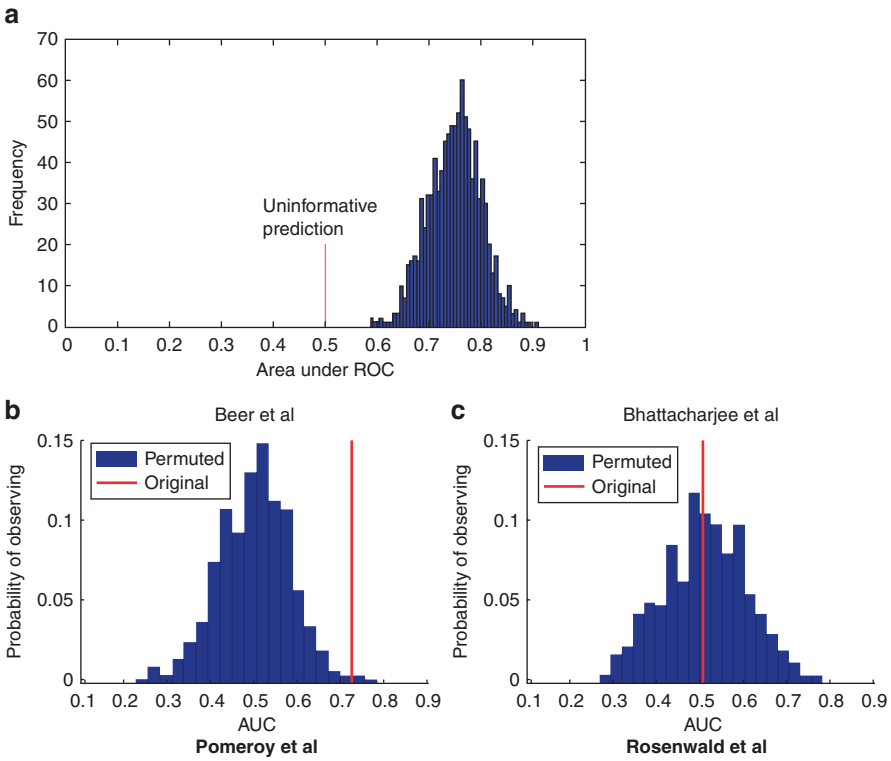


Fig. 8 Label reshuffling and its uses. In the top panel (**a**), the distribution of estimated predictivity for a hypothetical new ML method operating in a dataset that has been label-reshuffled (measured as area under the ROC curve—AUC ROC) is shown in blue. As can be seen, the distribution is not centered at the 0.5 point (the uninformative or no-signal point) in the AUC ROC. This indicates that the new ML method significantly overestimates performance [42]. In the bottom left panel (**b**) from a real data set modeling it can be seen that (1) the modeling protocol does not overestimate the model performance (because the distribution is centered on AUC ROC 0.5) and (2) the actual model obtained (red line) has performance that is statistically significantly different than that of the null hypothesis (i.e., no signal, represented again by the 0.5 point of the AUC ROC distribution of label-reshuffled datasets). By contrast, the analysis of the data depicted in the bottom right panel (**c**), leads to a model devoid of signal. Again the protocol used is unbiased with respect to error estimation [43]

the effect of maintaining the real joint distribution of inputs, as well as the real marginal distribution of response variable, but decoupling the inputs from outputs, so that over multiple such label-reshuffled datasets, on average there is no signal to be learned. This type of simulation is essential for testing ML methods' error estimation procedures. It is also a valuable tool for testing a learnt model against the hull hypothesis of no predictive signal in the data [42, 43]. See also chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI" on model over confidence for more details.

(c) **Re-simulated data**. *Resimulation* attempts to create controlled data simulations that are ideally hi-fidelity approximations of real data distributions of interest. Resimulation works as follows [32, 44]:

- We start with one or more real life datasets D^{real} where all variables (including response variables) have known values.
- Then we use a learning algorithm A to learn one or more models Mi capturing all or at least some desired characteristics of the real-life distribution from which the real data was sampled from.
- Then we sample from the encoded distribution in Mi using simulation ("resimulation" in his context) creating resimulated data D^{resim}.
- Then we test the properties of D^{resim} against the real data D^{real}. A variety of distribution similarity metrics can be used as well as custom predictive modeling, and custom tests of properties can be used for establishing that the resimulated data is a hi-fidelity representation of the real data.
- If adjustments are needed, we can iterate between the second and fourth steps until sufficiently accurate resimulated data is obtained.
- Once we have the hi-fidelity resimulated data we can now feed it in the new method, while varying parameters and obtain performance metrics just like in the case of simple simulation.

A conceptual nuance about resimulation is that if the resimulated data is perfectly indistinguishable from real data *in all modeling aspects of interest to us* (not just modeling the joint distribution), that implies that algorithm A is an optimal algorithm for discovery for the data generating function of the real distribution. Algorithm A would then represent a correct discovery algorithm procedure, rendering further method development effort potentially unnecessary (barring efficiency considerations). In common practice, we rarely have a perfect algorithm A at hand when performing resimulations. Instead we use algorithms that capture simplified and controllable aspects of the real data and therefore the performance of our new method is (loosely) an upper bound on the performance with real data. The rationale being that, if a new method cannot learn the process that creates the simplified version of real data D^{real}, i.e., D^{resim}, then the new method will most likely not be able to learn much more complicated real life data generating functions. If a perfect algortihm A exists in terms of quality of output, then the new algorithms may be more efficient ones.

(d) **Statistical tests for distributional or other conditions for method correctness.** These typically encompass statistical tests that do not test the algorithms but the data per se. For example, if an algorithm is devised to create regression models under multivariate normality of the data, we can test the real data for conformance to this assumption. This type of test is an important supplement to simulation studies and may also serve as a preferred alternative if no credible simulation or resimulation can be designed, and the algorithm's theory of correctness is well established.

In-depth example for step 6 (Empirically test algorithms in controlled conditions). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, of developing principled and scalable biomarker and pathway discovery algorithms, all of the algorithms discussed in section "Over- and Under-Interpreting Results", were tested extensively with simulated and resimulated data. These experiments revealed a number of important properties including: (1) verifying the MB theoretical expectation of maximum compactness and maximum predicitivity; (2) verified causal consistency; (3) showed high stability; (4) showed large added value over comparators and over random selection; (5) showed the role of various hyperparameters; (6) established the natural false discovery rate control in the GLL algorithm family that protecting against false positives due to conducting massive numbers of conditional independence tests; (8) showed the role of inclusion heuristics for computational efficiency; (9) demonstrated how perilous is to use non causal comparator methods to infer causality (something that GLL algorithms accomplish very well); and (10) showed the relatively large insensitivity of the algorithms to hyperparameter choice [1, 32].

Step 7. Empirically test algorithms in real life data with known answers/ solutions

This testing involves real data where known gold standard answers have been established by prior research. This is a very informative testing stage because a myriad of factors may exist in real data that have not been anticipated in theoretical analysis or in simulations. Conversely, it is also possible for real data to exhibit higher simplicity than what was anticipated by methods' developers which if true, typically leads to relaxing some of the related assumptions of the new method.

We re-iterate for emphasis that there are two reasons why we do not jump directly to step 7 (i.e., omitting empirical testing under controlled conditions):

First, real data does not afford *full control* of all relevant parameters that may influence the new method's performance. For example, we cannot arbitrarily control sample size, or signal strength, or dimensionality, or % of unmeasured variables, connectivity of the causal data generating process, etc. in real data since these factors have fixed and in many cases unknown characteristics in the available real datasets.

Second, real data with known high quality answers may be very limited and we do not wish to overfit the new method development to a small number of validation datasets (as will invariably happen, see chapter "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI" on overfitting).

Empirically testing new and existing methods with real data with known answers typically follows three designs:

1. **Centralized benchmark design.** A small group of expert data scientists organize and execute a series of tests of a new or existing method using several datasets and several state of the art alternative methods. Ideally all reasonable alternatives and baseline comparators including the best known methods for this type of problem are included and are executed according to the best known specifications (e.g., the specifications provided by their inventors). The datasets used are sufficient to cover a wide range of data typically encountered in the application domain. A multitude of factors are varied and their effects studied. Simulated and resimulated data may also be included a priori or ante hoc (e.g., to shed light on behaviors in real data). Example such studies are [1, 30–32, 37, 45–48, 51, 54–56, 59, 60].
2. **Distributed benchmark design.** This design is a variation of the centralized benchmark and typically occurs in the context of a scientific or industry consortium or coalition. A central team of experts organizes a benchmark similar to design 1 however analyses are conducted by several groups within the consortium. Each group may employ different methods, protocols, etc., and this natural variation is studied by the organizers who analyze results across participating teams. For an example of this design see [49].
3. **Public challenge design.** This design is a variation of the distributed benchmark and typically occurs in the context of a *participatory science* framework. A central team of experts or a challenge organization organizes the challenge typically with one or few datasets and a fixed data design. Analyses are conducted by volunteers across the globe. Each group may employ different methods, protocols, etc., and this natural variation is studied by the organizers who analyze results across participating teams. For an example of this design see [44].

Table 1 summarizes the characteristics of each design and points to strengths and weakness.

ML challenges serve two fundamental purposes. One is the *education* of scientists from different fields about the problems the challenge addresses and giving them data, and a platform to experiment. The second purpose is to explore which

Table 1 Empirical testing of AI/ML with real data and know answers: Three alternative designs (green: positive characteristics, red: weaknesses)

Characteristic	Challenges (ideally-conducted)	Centralized Benchmark Studies (ideally-conducted)	Distributed Benchmark Studies (ideally-conducted)
1. Led by	Academic groups or commercial companies	Academic groups (typically) or commercial companies (rarely)	Academic groups (typically) or commercial companies (rarely)
2. Conducted by	Many participants with varying (often not high) qualifications	Small teams of expert data scientists	Small teams of expert data scientists + Participant members of consortium
3. Participatory and open science	Strong	Limited	Limited
4. Educational intent	Yes	Secondary Focus	Secondary Focus
5. Empirical performance intent	Yes	Yes	Yes
6. Scope and representativeness of data	Very small/biased	High/unbiased	medium/ modestly biased
7. Updated over time by same or different groups	Almost never	Often	Almost never
8. Explore effects of data design	No	Yes	Yes
9. Explore effects of sample size	Almost never	Yes - controlled	Yes – as part of normal variation
10. Explore effects of model selection	Yes	Yes - controlled	Yes – as part of normal variation
11. Explore effects of error estimation	No	Yes - controlled	Possibly
12. Explore effects of algorithm choice	Yes	Yes - controlled	Yes – as part of normal variation
13. Suboptimal execution of algorithms and model selection	Common	Rare	Part of studied variation
14. Cost	Externalized to competitors	Internal to benchmark team	Distributed among consortium members

algorithmic methods are better at a particular point in time for a particular problem. Well-designed challenges can generate valuable information and enhance interdisciplinary engagement. Poorly-designed challenges (which in our estimation are the majority, currently) can be very misleading designs with respect to evaluating AI/ML methods. We elaborate in the following pitfall:

Pitfall 5.8
Issues and pitfalls of ML challenges. In many if not most cases, challenges suffer from **fixing the data design and the error estimation** thus *removing from consideration two out of the three determinants of ML success (i.e., data design, ML model selection and error estimation protocol, algorithm).*

Challenges also routinely restrict the design of modeling by pre-selecting variables, and over-simplifying the statement of problems, sometimes to meaningless extremes.

Challenges also often suffer from *incomplete or highly biased representation in the competitor pool*. Typically participants in challenges are either students or interested scientists who have competencies in areas unrelated to AI/ML.

Another limitation is that *not all appropriate algorithms are entered in a challenge and when they enter, they are not necessarily executed according to optimal specifications*.

Finally, challenges typically involve a *very small number of datasets that do not represent a large domain*. Such representative coverage typically requires dozens of datasets or more *in the same comparison*.

Despite these limitations, a select number of challenges that are designed to a high degree of quality, when interpreted carefully and with the appropriate qualifications, can provide valuable empirical scientific information. For examples of well-designed, high quality, and carefully interpreted challenges the reader may refer to the challenges conducted by the ChaLearn organization [50].

Best Practice 5.6
The preferred design for validating AI/Ml methods with real life data with known answers is the centralized benchmark design. Distributed benchmark designs, whenever feasible, add value by exploring natural variation in how methods are applied by experts. Finally competitions have several intrinsic limitations and have to be interpreted carefully.

In-depth example for step 7 (Empirically test algorithms in real life data with known answers/solutions). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, CONTINUED.

In our running example, of developing principled and scalable biomarker and pathway discovery algorithms all of the algorithms discussed in section "Over- and Under-Interpreting Results" were also tested extensively with real data where true answers were known or could be established via predictive verification. Statnikov et al. showed the superlative performance for biological local pathway discovery. Aphinyanaphongs et al. showed the performance in text categorization comparing with all major acedemic and commercial comparators of the time. Statnikov and Aliferis showed the massive gene expression signature equivalence classes in cancer microarray data. Alekseyenko, Statnikov and Aliferis evaluated in GWAS prediction signatures and causal loci discovery. Ma, Statnikov and Aliferis showed minimization of experiments in biological experimental data. Other benchmarks addressed additional microbiomic, cancer, and other types of data [1, 18, 22–24, 29–34, 37, 40, 41, 45–47, 52, 53].

Step 8. Empirically test algorithms in real life data without known answers/solutions but where future validation can take place

In the final step of the new method development and validation process, the AI/ML method and the models produced by it are tested in real data where the correct answers can be obtained but only prospectively: If the models are predictive we obtain true values and compare them to the predicted values. In causal modeling we conduct randomized controlled experiments and compare the effects of interventions against the algorithmic estimates for such effects. Other forms of validation designs are also possible depending on the nature of the AI/ML methods' goals and intended outputs.

We caution the reader that **only after successful completion of ALL prior steps** in the new or existing AI/ML method development or appraisal, is applying the method in real-life problems with any risk, warranted.

In-depth example for step 8 (Empirically test algorithms in real life data with unknown answers/solutions). Development or evaluation of scalable discovery methods of biomarkers and molecular pathways from high dimensional biomedical data, FINAL.

In our running example, the methods were deployed in several real-life projects related to basic and translational science discovery, experimental therapeutics, and healthcare improvements. Application areas included: (1) predicting risk for sepsis in the neonatal ICU; (2) diagnosing stroke from

stroke-like syndromes using proteomic markers; (3) modeling the decision making and determining melanoma guideline non-compliance of dermatologists; (4) determining which patients with ovarian cancer will benefit from frontline Tx with bevacizumab; (5) understanding mechanisms and predicting outcomes in children with PTSD, (6) creating models that predict accurately citations of articles in deep horizons; (7) creating models that characterize the nature of citations; (8) creating models to classify articles for methodological rigor and content; (9) models that scan the WWW for dangerous medical advice; (10) models for diagnosis of psoriasis using microbiomic signatures from the skin; (11) multi-omic clinical phenotype predictions; (12) discovery of new targets for osteoarthritis; (13) detection of subclinical viral infection using gene signatures from serum; (14) analysis and modeling of longevity using clinical and molecular markers and (15) modeling of the mediating pathways between exercise and diet and cardiometabolic outcomes for drug target discovery [35, 36, 40, 51, 54–56, 59, 60–65].

In conclusion, the above interrelated case studies give an "insider's view" on, and showcase the feasibility and benefits of a complete rigorous development and validation approach for new methods using the example of local causal graph and MB algorithms and their extensions.

Similar in-depth rigorous development efforts have characterized the history of Bayesian Networks, SVMs, Boosting, Causal inference algorithms, and other methods (see chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML"). However we do point out that unfortunately in many cases, widely-adopted methods and tools both in the academic and commercial realms lack many of the steps outlined here and therefore they have to be used with caution especially in high-stakes (high risk, high cost) application domains. The next section gives a highly condensed overview of properties of major AI/ML methods.

Best Practice 5.7
Develop and validate ML/AI methods using the following stages/steps:

- Step 1. Rigorous problem definition (in precise mathematical terms and establishing how the mathematical goals map to the healthcare or health science discovery goal).
- Step 2. Theoretical analysis of problem (complexity, problem space characteristics etc.).
- Step 3. First-pass algorithms solving the problem.

- Step 4. Theoretical properties of first pass algorithms: focus on representation power, soundness and completeness, transparency.
- Step 5. Algorithm refinements and optimizations.
- Step 6. Empirically test algorithms in controlled conditions .
- Step 7. Empirically test algorithms in real life data with known answers/solutions.
- Step 8. Empirically test algorithms in real life data without known answers/solutions but where future validation can take place.

Best Practice 5.8
Avoid evaluating methods by employing expert narratives showing "validity".

Best Practice 5.9
Do not reinvent the wheel. Verify that a new method does not solve a problem previously solved by a better performing method.

Best Practice 5.10
Create open box methods to the full extent possible. Do not pursue weak justifications that fail to translate the models to accurate human readable representations.

Best Practice 5.11
Do not confuse "open source" for "transparent" and "closed source" for black box".

A Concise Overview of Properties of Major AI/ML Methods

Table 2 gives a high-level, very concise view on properties of key families of AI/ML methods. A few observations are in order:

1. Heuristic systems by definition lack well-defined, well-understood or confirmed properties. We include them in the table only to remind the reader that they are seriously handicapped in that regard, and should not be used in high-stakes applications (until a better understanding of their risk and benefits is achieved).

Table 2 Map of properties of major AI/ML methods

	THEORETICAL PROPERTIES									EMPIRICAL PERFORMANCE		
Method	Representational Power	Semantic Clarity	Transparency	Soundness	Completeness	Computational Complexity	Space Complexity	Sample size	Probability Theory Compliance	Empirical Accuracy **(in common use)**	Empirical Accuracy **(in recommended use)**	Method misapplication in practice
Heuristic systems (i.e., pre-scientific AI/ML)	?	?	?	?	?	?	?	D	?	D	D	Common
Logics	+↑	+↑	+↑	+↑	+o	+↓	+↑	N/A	D	+↑	+↑	Uncommon
Rule Based Systems & Logic-derived	+o	+↑	+↑	+↑	+o	+↓	+↑	N/A	+↓	+o	+o	Uncommon
MEU Decision Theory	+o	+↑	+↑	+↑	+↑	+o	+↑	N/A	+↑	+↑	+↑	Uncommon
Search	+↑	+↑	+↑	D	D	D	D	N/A	D	+↑	+↑	Uncommon
OLS	+o	+↑	+↑	+↑	+↑	+↑	+↑	+↑	+o	+o	+↑	
GLM	+o	+↑	+↑	+↑	+↑	D	D	+↑	+↑	+↑	+↑	Uncommon
ANN/DL	+↑	+↑	+↓	+o	+↓	+↓	+o	+↓	+↑	+o	+↑	Common
SVM	+↑	+↑	D	+↑	+↑	+↑	+↑	+↑	+o	+↑	+↑	Uncommon
NB	+↓	+↑	D	+↑	+↑	+↑	+↑	+↑	+↑	+↓	+↓	Common
Bayes Nets	+↑	+↑	+↑	+↑	+↑	D	+↑	+o	+↑	+↑	+↑	Uncommon
KNN	+↑	+↓	+↑	+o	+↑	+↓	+↓	+↑	+o	+o	+o	Common
DT	+↑	+↑	+↑	+o	+o	+↑	+o	+o	+↑	+o	+↑	Uncommon
Clustering	+↑	D	+↑	+↓	+↓	+o	[o↓]	+↓	+↓	+↓	+o	Common
Stacking	D	D	+↓	D	+↑	[o↓]	[o↓]	+o	+[↑↓]	+↑	+↑	Uncommon
RF	+↑	+↑	+↓	+↑	+o	+o	+o	+o	+↑	+o	+↑	Common
Boosting	+↑	D	+↓	+↑	+o	+o	+o	+o	+↑	+↑	+↑	Uncommon
Penalized Regression	+o	+↑	+↑	+ [↑o]	+↑	+o	+↑	+o	+[↑↓]	+↑	+↑	Uncommon
UAF	+↓	+↑	+↑	+↓	c	+↑	+↑	+↑	N/A	+↓	+o	Common
SV-RFE	+↓	D	+↓	+↓	+↓	+↑	+↑	+↑	+↓	+↑	+↑	Uncommon
Markov Boundary Feature Selection	+↑	+↑	+↑	+↑	+↑	D	+↑	D	+↑	+↑	+↑	Uncommon
Embedded Feature Selection	+↓	+o	+↑	+↓	+↓	+↑	+↑	+↑	D	+↓	+↑	Common
Wrapper Feature Selection	D	D	D	+↓	+↓	D	D	D	D	+↓	+↓	Common
Kaplan Meier	+↑	+↑	+↑	+↑	+↑	↑	+↓	+o	+↑	+↑	+↑	Uncommon
Cox PH	+o	+↑	+↑	+↑	+↑	+↑	+↓	+o	+↑	+↑	+↑	Uncommon
AFT	+↓	+↑	+↑	+↑	+↑	+↑	+↑	↑	+↑	D	D	Uncommon
ANOVA	+↓	+↑	+↑	+↑	+↑	+↑	+↑	↑	+↑	+o	+↑	Common
(G)LMM	+o	+↑	+↑	+↑	+↑	+o	+↑	+o	+[↑ o]	+↑	+↑	Uncommon
GEE	+o	+↑	+↑	+↑	+↑	+↑	+↑	+o	+[↑ o]	+↑	+↑	Uncommon
SEM (linear)	+o	+↑	+↑	+↑	+↑	+↑	+↑	↑	+[↑ o]	+o	+↑	Uncommon
PC	+↑	+↑	+↑	+↑	+↑	+↓	+↑	+↑	+↑	+↓	+↓	Common
FCI	+↑	+↑	+↑	+↑	+o	+↓	+↑	+↑		?	?	?
GFCI	+↑	+↑	+↑	?	+o	?	+↑	+↑	+↑	?	?	?
GES	+↑	+↑	+↑	+↑	+↑	+o	+↑	+↑	+↑	+o	+o	Uncommon
MMHC/LGL	+o	+↑	+↑	+o	+o	+↑	+↑	+↑	+↑	+↑	+↑	Uncommon

(continued)

Table 2 (continued)

Symbol Notation

+ = Properties are known.? = Properties are unknown
↑ = strong; o = medium; ↓ = weak;
D depends on choice of algorithm, or problem;
[.] depicts range of values.
Color Notation
Green = strong; orange = medium; red = weak or property is unknown; gray = depends on choice of algorithm, or problem; white = not applicable

2. Most methods that are widely used today have well understood properties, or at most a few gaps in the understanding of their properties.
3. There is no perfect method across all properties and problem types: every method has weak spots.
4. Some methods are compromised by being commonly used in problems that they should not (i.e., "user error"). In these cases there is always a better method for that problem category.
5. Statistical machine learning methods have stronger/better studied properties in general.
6. It is possible for some methods to have sub-optimal theoretical properties but exhibit excellent empirical performance in some problem categories and/or selected datasets.

Explanation of terms:

Representation	What kind of relationships can the method represent? In case of modeling methods, are the model assumptions restrictive? ↑ (strong): Any relationship (e.g. Universal Function Approximator) o (limited): Constrained functional form ↓ (weak): Restrictive assumptions apply D (Depends): Inherits property from an component method, has other kind of dependence (e.g. on an underlying distribution), or the properties varies across members of a broader category of methods
Semantic clarity	Is the method (or model) semantics clear? E.g. in case of a predictive model, is the meaning of the model components clear? ↑ (strong): Yes o (moderate): Some parts are not clearly defined ↓ (weak): Not having clear meaning
Transparency	Does the method semantics relate to real-world entities in a clear manner? Is it easily understood by humans? ↑ (strong): The relationship between real-world entities and method components can be explained. Humans easily understand method/models. ↓ (weak): Method is difficult to interpret, additional algorithms are needed to interpret o (moderate): Between ↑ and ↓

Soundness	*When the model assumptions are met*, are the results correct? (E.g. for a predictive model, is the model error optimal?) ↑ (strong): Yes, guaranteed (e.g. convex problem, Optimal Bayes Classifier, etc); o (medium): Trapped in local optima, only approximates target function to some acceptable degree; ↓ (weak): Output may considerably deviate from correct answer
Completeness	Will the method output correct answers for all problem instances? ↑ (strong): Yes, will produce correct answers for full problem space; o (medium): considerable but not full coverage of problem space; ↓ (weak): only small portions of problem space correctly answered, or very significant regions are omitted.
Compute	Computational complexity. For predictive models, it includes the complexity of both model construction and prediction. ↑ (strong): Very fast. E.g., for executing predictive models, linear in number of variables, and with a small hyperparameter space o (medium): May build multiple models. ↓ (weak): Requires extensive computing (e.g. immense hyper-parameter space, exponential in the number of variables, etc.) D (depends): Typical and worst-case are very different; can take advantage of properties of the problem (e.g. graph connectivity)
Space	Storage complexity required for the computation, or for storing the model (if there is a model) ↑ (strong): Approx. number of variables o (medium): linear in the number of variables ↓ (weak): super-linear in the number of variables or proportional to the data set size (if data set size exceeds order of number of variables)
Sample size	Sample size required to train the model (to an acceptable performance on problems that constitute the preferred use of this method) ↑ (strong): small sample size is sufficient (e.g. linear in the number of variables, in the number of effective parameters, support vectors, etc.) o (medium): moderate (e.g., low order polynomial) sample sizes required to number of effective parameters ↓ (weak): large sample sizes are required (e.g., super- or higher order-polynomial to number of effective parameters)
Probabilistic consistency	(1) Model can return probabilities (or output can be converted to probabilities) and (2) probabilistic output is calibrated or can be calibrated ↑ (strong): designed to be probabilistically consistent o (medium): output can be converted into probabilities ↓ (weak): not meant to produce probabilities and there is no easy way to convert predictions into consistent probabilities

Empirical accuracy in common use	In its common use, what is the method's empirical performance in terms of result accuracy (e.g. accuracy or AUC for classification models)? ↑ (strong): One of the strongest among methods that are best-of-class for this problem. (E.g. DL for imaging; Cox PH for time-to-event, etc.) o (medium): Performs noticeably worse than the best, but still outperforms several methods. ↓ (weak): Performs substantially worse than most applicable methods.
Empirical accuracy in recommended use	In its preferred use, what is the method's empirical performance in terms of result accuracy (e.g. accuracy or AUC for classification models)? ↑ (strong): one of best-in-class methods for this problem. (E.g. DL for images; Cox PH for time-to-event, etc.) o (medium): performs noticeable worse than the best-in-class, but still useful in some cases. ↓ (weak): Performs substantially worse than best and has no mitigating uses.
Method misapplication in practice	Common: The method is commonly used for tasks that is not best-in-class Uncommon: The method is seldomly used for tasks that is not best-in-class

We advise the reader to study this table and cross-reference with the description of methods and guidance for their use in chapters "Foundations and Properties of AI/ML Systems", "An Appraisal and Operating Characteristics of Major ML Methods Applicable in Healthcare and Health Science", and "Foundations of Causal ML".

A Worksheet for Use when Evaluating or Developing AI/ML Methods

It is highly recommended as new or existing methods are being evaluated to use a chart such as the one shown in Table 3. There are three purposes in this endeavor: (a) Remind the developer or evaluator about the necessary dimensions of validation/appraisal. (b) Maintain a record of progress as the various stages of evaluation are advancing. (c) Enforce due diligence both in an absolute sense but also in comparison to applicable alternatives. (d) Enforce honesty/reduce developer bias in assessing the added value of the evaluated method over established incumbents. (e) More objectively assess marketing claims about the strengths of commercial products.

Table 3 Worksheet for evaluating new and existing AI/ML methods

	Theoretical properties									Empirical performance		
Method	Representational Power	Semantic clarity	Transparency	Soundness	Completeness	Computational Complexity	Space Complexity	Sample size	Probability Theory Compliance	Empirical Accuracy(**in common use**)	Empirical Accuracy (**in recommended use**)	Method misapplication in practice
Method to be evaluated	OBTAIN BY THEORETICAL ANALYSIS PROPERTIES AND INSERT APPRAISAL IN EACH COLUMN									OBTAIN BY EMPIRICAL EXPERIMENTATION AND INSERT APPRAISAL IN EACH COLUMN		
Best-in-class known method #1 for that problem domain	SELECT FROM THE LITERATURE BEST KNOWN EXISTING METHOD FOR PROBLEM TYPE AND COPY ITS THEORETICAL PROPERTIES IN EACH COLUMN									COPY EMPIRICAL PROPERTIES IN EACH COLUMN		
Best-in-class known method #2 for that problem domain	SELECT FROM THE LITERATURE SECOND BEST KNOWN EXISTING METHOD FOR PROBLEM TYPE AND COPY ITS THEORETICAL PROPERTIES IN EACH COLUMN									COPY EMPIRICAL PROPERTIES IN EACH COLUMN		
Best-in-class known method #3 for that problem domain	SELECT FROM THE LITERATURE THIRD BEST KNOWN EXISTING METHOD FOR PROBLEM TYPE AND COPY ITS THEORETICAL PROPERTIES IN EACH COLUMN									COPY EMPIRICAL PROPERTIES IN EACH COLUMN		
Baseline comparator method for the problem domain	SELECT FROM THE LITERATURE BASELINE EXISTING METHOD FOR PROBLEM TYPE AND COPY ITS THEORETICAL PROPERTIES IN EACH COLUMN									COPY EMPIRICAL PROPERTIES IN EACH COLUMN		

Over-and Under-Interpreting Results

We conclude this chapter with a discussion of avoiding over-interpreting and under-interpreting AI/ML results.

A major principle for the scientifically valid use of AI/ML models is their proper interpretation for either driving healthcare decisions and improvements as well as for driving discovery in the health sciences. Two major and antithetical pitfalls are the over- and the under-interpretation of results given the data design (see chapter "Data Design") and the properties of the algorithms and protocols employed.

Pitfall 5.9
Interpreting results of a method's beyond what its known properties justify.

Pitfall 5.10
Interpreting results of a method's below what its known properties justify.

A few examples of the over-interpretation pitfall include**:**

(a) Interpreting weak predictive methods (and resulting models) as if they have much stronger accuracy (usually combined with omitting stating the weak aggregate signal of the method's models e.g., in the context of regression analyses of biomarkers).
(b) Assigning special biological or mechanistic significance to variables because they are stable in resampling or because they are ranked high according to univariate association with a response variable.
(c) Generally interpreting causally the findings of predictive methods (and resulting models).
(d) Failing to observe that some feature selection methods in omics data commonly do not, or marginally outperform random selection.
(e) Ignoring the possibility of hidden (aka unmeasured or latent) variables distorting the observed effects of measured variables.
(f) Ignoring effects of small sample size variation on results.
(g) Assuming (without proof) that case matching according to hypothesized confounders has controlled all confounding. Assuming in SEM modeling that the domain causal structure is known with certainty.
(h) Assuming that propensity scoring perfectly controls confounding.
(i) Treating coefficient values in regularized regression methods (Lasso family, SVMs, and other "penalty+loss" algorithms) as if they are equivalent to statistical conditioning (e.g., in classical regression).
(j) Assuming (without testing) that the assignment of subjects to treatment arms in trials from existing datasets is perfectly randomized and without bias. Etc.

A few examples of the under-interpretation pitfall include:

(a) Focusing on small individual variable effects without noticing that the aggregate signal over many variables is large (e.g., in GWAS studies).
(b) Focusing on small coefficient of variation of a model (i.e., total response variance explained) and failing to notice that some variables have strong individual effects (this is the reverse of the previous under-interpretation problem).
(c) Failing to pick up strong putative causal factors even when causal ML algorithms indicate their significance, because "correlation is not causation".
(d) Dismissing methods (and resulting models) because they are not statistically stable under resampling.

We will revisit these problems in subsequent chapters as they require a holistic understanding of data design and proper AI/ML algorithm design and execution. The best practice we will state at this point however is:

Best Practice 5.12
Interpret results of application of a method (and resulting models) at the level justified by its know properties.

Key Messages and Concepts Discussed in Chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems"

1. Establishing and knowing the properties of AI/ML methods enables informed assessments about the Performance requirements, the Safety requirements, and the Cost-effectiveness requirements of corresponding AI/ML solutions, and leads to building trust in the AI/ML solution.
2. A best practice workflow was presented, that can be used to establish the properties of any new or pre-existing method, tool or system, so that a rational, effective and efficient solution to the problem at hand can be identified.
3. The importance of rigorous problem definitions (in precise mathematical terms, and with precise correspondence to health care or health science objectives).
4. Re-inventing the wheel and why it is undesirable.
5. First-pass algorithms vs algorithm refinements and optimization.
6. Parallel algorithms, Distributed/Federated, Sequential, and Chunked algorithms.
7. Relaxing algorithmic assumptions/requirements.
8. Generalized algorithm frameworks and generalized conditions for performance guarantees.

9. What are AI/ML (or data science) Stacks.
10. "Pipelines", "Automodelers", and "Platforms".
11. Explanation, interpretability, and transparency: Black box AI/ML methods and/or models; Transparent (or open, or white, or clear) box.
12. Justification of a method or a model (and their outputs) vs high-fidelity explanation of a method or model (and their outputs) e.g., by functional equivalence.
13. Human-understandable models and formalisms.
14. Open source software vs Closed source (or proprietary) software.
15. The importance of testing algorithms in controlled conditions.
16. Simulated data; Label-reshuffled data; Re-simulated data, and their properties and use.
17. Real-life examples of using the new method development process to establish the properties of well known (new or pre-existing) methods, tools or systems.
18. Interpreting results of application of a method at the level justified by its known properties.

Pitfalls Discussed in Chapter "Principles of Rigorous Development and of Appraisal of ML and AI Methods and Systems"

Pitfall 5.1.: Developing methods with theoretical and empirical properties that are:
(a) Unknown, or
(b) Poorly characterized in disclosures and technical, scientific or commercial communications and publications, or
(c) Clearly stated (disclosed) but not proven, or
(d) Not matching the characteristics of the problem to be solved at the level of performance and trust needed.

Pitfall 5.2. Evaluating the success of methods with poorly defined objectives, by employing expert narratives showing "validity".

Pitfall 5.3. Defining the goals of methods in mathematical terms but without establishing how the mathematical goals map to the healthcare or health science discovery goal.

Pitfall 5.4. Reinventing the wheel: whereby a new method is introduced but it has been previously discovered yet ignored (willfully or not) by the "new" method developers.

Pitfall 5.5. Reinventing a method but make it worse to established methods (…"reinventing the wheel and making it square"!).