Applied Causal Inference

Uday Kamath, Kenneth Graham, Mitchell Naylor

2023-08-25

Table of contents

Preface

Welcome to the Applied Causal Inference book! We are excited you're here.

We intend to offer the web version of this book entirely free of charge. However, there are other ways to purchase other versions of the book or otherwise support the authors:

- E-book versions are available on LeanPub
  - Additionally, if you found the web version helpful and would like to support the authors, you can use the LeanPub price slider to do so
- Hard copies are available on Amazon (**add link**)

Who This Book is For

This book is designed to help anyone along the spectrum of experience with causal inference – nearly everyone from absolute beginners to experienced users of causality will be able to learn something about the world of causal inference and how it can be applied. If you use data to answer questions, then it's likely that you can benefit from the techniques described in this book. Whether you're in industry or academia, the causal inference toolkit will be valuable to have in your repertoire.

We wrote this book to make it easy to leverage these tools within the standard data science workflow. We assume no prior exposure to causal methods, but there are a couple of small prerequisites to help you get the most out of this book:

- Understanding of standard statistical and machine learning methods: correlation, linear regression, generalized linear models, tree-based models
- Familiarity with Python for case studies, illustrations, and implementations; experience using the standard PyData stack (Pandas, NumPy) will be useful

This book builds from these basic foundations to provide a practical understanding of how causal inference works and how to use it in applied settings.

Why This Book?

Recent advancements in causal inference have made it possible to gain profound insight about our world and the complex systems which operate in it. While industry professionals and academics in every domain ask questions of their data, traditional statistical methods often fall short of providing conclusive answers. This is where causality can help.

If you're here, it's likely that you are interested in learning about causal inference. You may have read some of the existing literature on causal inference and found it difficult to apply to the problems you face at work or in the lab. We've been there, and we wrote this book to serve as a resource bridging the gap between the theory and application of causal techniques. We are not aware of another book which goes from the foundations of causality to hands-on examples using common data science tooling. Specifically, this book contains:

- A comprehensive resource that builds up from elementary principles of causality to the more advanced techniques in causal inference
- An in-depth guide and explanation of causal estimation procedures and methods, coupled with an explanation of the process of causal discovery, illustrating how to discern the underlying causal structure from observational data
- An overview of how causal inference can be applied in various domains such as natural language processing, computer vision, and time series analysis
- A compendium filled with successful causal models, along with their fundamental mathematical theory, explained in a manner that's easy to comprehend
- A focus on important areas like fairness, explainability, and reinforcement learning within the scope of causality
- A practical approach that employs Python libraries like DoWhy and CausalML, demonstrating their use in real-world applications; the Python code is made available in Google Colab notebooks

We designed this book to be a resource for any researcher, analyst, or data scientist to go from basic understanding of causal inference to hands-on, applied usage. Simply put, we wrote this book because we wished a similar resource existed when we first began learning about causal inference. Causality is an exciting domain, and we hope you find this book helpful in your journey to incorporate causal tools into your day-to-day work.

What This Book Contains

This book is divided into chapters which build upon each other within 3 broad parts. The content of the chapters include necessary theoretical foundations, details for application and, where possible, practical case studies to illustrate the end-to-end process. We recommend readers begin with Part 1, which is the ground-up introduction to applied causal inference.

Parts 2 and 3 build upon the information introduced in Part 1, and can be explored according to the reader's preference.

Throughout the book, we use a green callout like the one below to denote, rephrase, or generally call attention to particularly important concepts.

> Tip
>
> Blocks like this one highlight especially helpful or foundational concepts referenced throughout the book and in causal inference literature more broadly!

Contents by Chapter

Part 1 begins by motivating why causality is a promising resource and laying the foundation of the necessary concepts from causal inference, culminating in an understanding of the potential outcomes framework. After this, we explore the full causal estimation process, providing the tools necessary to go from an initial question and a dataset to the creation and evaluation of causal estimates. We then provide an overview of causal discovery, which allows us to learn causal structures from observational data. The contents of Part 1 are as follows:

- **Chapter 1** first seeks to motivate the reader by answering the question "why should I use causality?" It does this by providing illustrative examples of Simpson's paradox and spurious correlation, and introducing how causal methods address these problems. The chapter then describes the paradigm shift between the traditional statistical or machine learning workflow to that of a causal inference setting. The chapter closes by giving a brief history of causal inference – from Judea Pearl's early work to modern applications in industry – and providing a list of existing resources for those wishing to learn more.
- **Chapter 2** introduces the potential outcome framework to explain concepts and theoretical foundations of causal effects. By exploring the key ideas and theories, this chapter aims to deepen readers' understanding of cause-and-effect relationships. A comprehensive case study is presented, utilizing a well-known causal study conducted with an educational television program. This practical example aims to give readers a firsthand understanding of the discussed concepts. The chapter concludes by addressing various challenges from the Gelman and Hill book, enabling readers to develop the necessary skills to contemplate, model, and apply the theory discussed throughout the chapter.
- **Chapter 3** delves further into the world of causal modeling, building upon the foundational knowledge established with the potential outcome framework and fundamental causal concepts. This chapter explores causal graphs as a practical approach for inferring causal relationships. We introduce causal graphs, discuss the high-level process of causal inference, examine various methods and techniques, and present a comprehensive case study with the Lalonde dataset for comparative analysis.

- **Chapter 4** probes into the challenges of constructing causal models in practice and highlights the emergence of causal discovery techniques based on observational data as an alternative. It introduces a range of techniques developed for this purpose and presents an array of causal discovery algorithms, explaining their relative strengths and limitations. The chapter concludes with a real-world case study, showcasing the practical utility of these algorithms in uncovering causal relationships.

We shift gears in Part 2 to discuss how causal inference is currently being used within other sub-domains of machine learning, including computer vision, natural language processing, and in time-dependent settings. These are the chapters in Part 2:

- **Chapter 5** focuses on how to apply methods of causal inference natural language processing, specifically for data that includes text. We consider how to compute causal effect sizes when the treatment and/or outcome is text, with or without the presence of confounding text. We include a case study analyzing film revenue data.
- **Chapter 6** details the intersection of causality and computer vision. It introduces the ever-present issues of spurious correlation and confounding in image data – problems well-suited for applications of causal methods. The chapter showcases research efforts to apply causal inference techniques in specific areas of computer vision, including image classification and visual question-answering. The chapter concludes with a case study of causal methods designed to improve robustness, using an adversarial transfer dataset.
- **Chapter 7** explores a recent method for time-dependent causal inference that is able to not only determine causation, but the temporal delay between the cause and effect variables. At present, this chapter does not describe time-dependent causal inference in the presence of confounding associations. We include a case study using an open-source bike sharing dataset.

In Part 3, we discuss some advanced topics within the field of causality:

- **Chapter 8** explores a special case of causal inference: assessing model fairness. The chapter provides a high-level introduction to the issues of algorithmic bias, describes existing non-causal approaches of measuring unfair bias, and presents an argument in favor of using causal model fairness techniques. It details causal approaches and additional considerations for the confounders that may exist in fairness settings. The chapter closes with a case study comparing causal and non-causal methods on the infamous COMPAS dataset.
- **Chapter 9** contains an overview of cutting-edge applications of causality in reinforcement learning, including techniques to improve world models in model-based RL, merge online and offline data, improve sample efficiency, and explain agent incentives. This chapter closes with a discussion of the challenges preventing large-scale adoption of causal RL technique.

About the Authors

Uday Kamath has over two decades of experience developing analytical products and has expertise in statistics, optimization, machine learning, deep learning, natural language processing, and evolutionary computing. With a Ph.D. in scalable machine learning, Uday has made significant contributions that have extended across numerous journals, conferences, books, and patents. Notable works by Uday include *Explainable Artificial Intelligence: An Introduction to Interpretable Machine Learning*, *Transformers for Machine Learning: A Deep Dive*, *Deep Learning for NLP and Speech Recognition*, *Mastering Java Machine Learning*, and *Machine Learning: End-to-End Guide for Java Developers*. He has held significant leadership positions throughout his career, including Chief Analytics Officer for Digital Reasoning, Advisor for Falkonry, and Chief Data Scientist for BAE Systems Applied Intelligence. Currently serving as the Chief Analytics Officer for Smarsh, his role encompasses spearheading data science and research in AI.

Kenneth Graham has two decades solving quantitative problems in multiple domains, including Monte Carlo simulation, NLP, anomaly detection, cyber security, and behavioral profiling. For the past ten years, he has focused on building scalable solutions in NLP for government and industry, including entity coreference resolution, text classification, active learning, automatic speech recognition, and temporal normalization. He currently works at AppFolio as a senior machine learning engineer and is a co-author of *Transformers for Machine Learning: A Deep Dive*. Dr. Graham has five patents for his work in natural language processing, seven research publications, and a Ph.D. in condensed matter physics.

Mitchell Naylor is an applied machine learning professional with seven years of experience, including roles within healthcare, technology, and insurance. He currently works as a lead data scientist at Azra AI, where he works primarily within clinical natural language processing, with applications in deep learning, transfer learning, robustness, fairness, and explainability. He has two research publications and contributions to two textbooks, and he holds a MS in Analytics from Georgia Institute of Technology.

Acknowledgments

The authors especially want to thank Grant Burroughs for producing the cover design. Additionally, the authors extend their heartfelt appreciation to the reviewers (in alphabetical order): Zack Burch, Krishna Choppella, Dr. John Liu, Dr. Joe Porter, Dr. Sangeeta Shukla, and Vedant Vajre. Their valuable feedback and insightful suggestions have played an instrumental role in enhancing the quality of this work.

> I would like to express my heartfelt gratitude to my wife, Pratibha, our nanny, Evelyn, our children, Brandy and Aaroh, as well as our friends and family, for their unwavering support and encouragement throughout this endeavor.
>
> – Uday Kamath

> I would like to express my deepest gratitude to my wife, Alyson, for her understanding and patient support throughout the entire process of writing this book. I also want to thank my friends and family for their enthusiastic support. Lastly, I want to thank my colleagues and co-authors for valuable discussions and inspiration.
>
> – Kenneth Graham

> I am incredibly grateful to all of my family, friends, and colleagues for their gracious support during this process. I especially want to thank my wife, Abby, for her constant encouragement and love – this would not have been possible without you.
>
> – Mitchell Naylor

1 Introduction to Causality

Greetings, and welcome to the first chapter of the Causal Machine Learning book! If you're here, it's likely that you are a data or machine learning professional who wants to incorporate causality into your toolkit. Alternatively, you might be skeptical about the usefulness of causal inference methods in applied settings. In either case, you're in the right place!

In this chapter, we will introduce some high-level concepts about causality and give some motivating examples to set the stage before our dive into causal inference. Specifically, we will answer the question *"why should I use causality?,"* provide a (brief) historical overview of causality, give a few current examples of causal inference applications in industry settings, and describe the mentality shift compared to a typical ML workflow. After that, the next few chapters will give you everything you need to get up and running in applied causal inference.

Background Knowledge

We assume that readers are familiar with standard statistical methods including correlation and logistic regression. Readers without this prior knowledge are encouraged to consult a resource such as Chapters 3 and 4 of *Introduction to Statistical Learning*[1] (James et al. 2013).

1.1 Why Should I Use Causality?

The amount of available data has increased dramatically over the last 20 years, and with it, there has been an exponential increase in the questions one might try to answer using data. Many of these questions are fundamentally causal in nature:

- Why did metric X change this month?
- Which of our customers should we target with Y campaign?
- What would happen if we instituted Z policy change?

Questions like these are commonplace in modern companies, and they all have something in common: they want to understand and quantify causal relationships using data. With large amounts of data and computing power available, it becomes easy to attempt to answer these questions using basic correlational analysis. Standard methods can sufficiently answer the

[1] Available at https://www.statlearning.com/

question of *what* happened, but when the question becomes *why* or *how* something occurred, correlation can mislead. Indeed, there are a couple of issues that arise in these settings which can be difficult to overcome without the right set of tools at your disposal. These challenges involve the ability to interpret associations within data as something meaningful, as well as a phenomenon where observed effects appear to reverse. We will see that causal inference provides an attractive solution to both of these problems.

Judging by trends in machine learning research, it becomes clear that interest in causal methods is growing beyond traditional settings in applied statistics as well. Figure 1.1 shows the increase in causal papers that appear at the popular ML research journal NeurIPS[2].

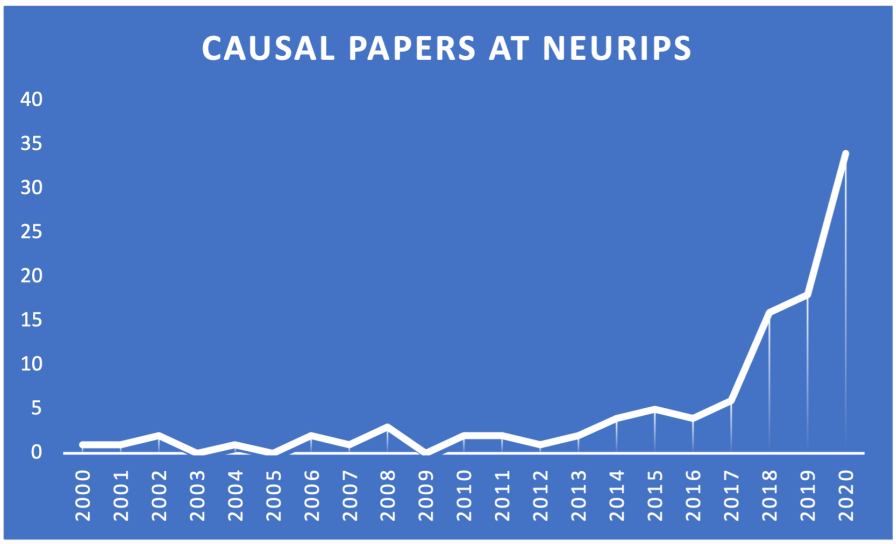


Figure 1.1: Causal papers featured at NeurIPS

Clearly there is growing momentum surrounding causal methods. In the following sections, we will dig deeper into the specific problems that causality can help address.

1.1.1 Challenge #1: *Correlation Does Not Imply Causation*

The phrase *"correlation does not imply causation"* has become somewhat of a cliché; it is a common response to any measured association, and it is often used in response to various analyses in public policy, medicine, economics, and any other scientific or quantitative field. However trite it may be, there is a degree of truth contained within this statement: we cannot just accept associations within data (however strong they may appear to be) as a meaningful relationship.

[2]Source: https://richardjoncarter.com/research/research-into-causal-ai-has-grown-significantly-in-the-last-3-years-if-neurips-is-anything-to-go-by/

One commonly used example of this is the strong correlation between ice cream sales and shark attacks. Figure 1.2 illustrates what this relationship might look like.[3] Judging by the data alone, one might conclude that eating ice cream causes shark attacks – a horrifying prospect for vacationers with a sweet tooth.

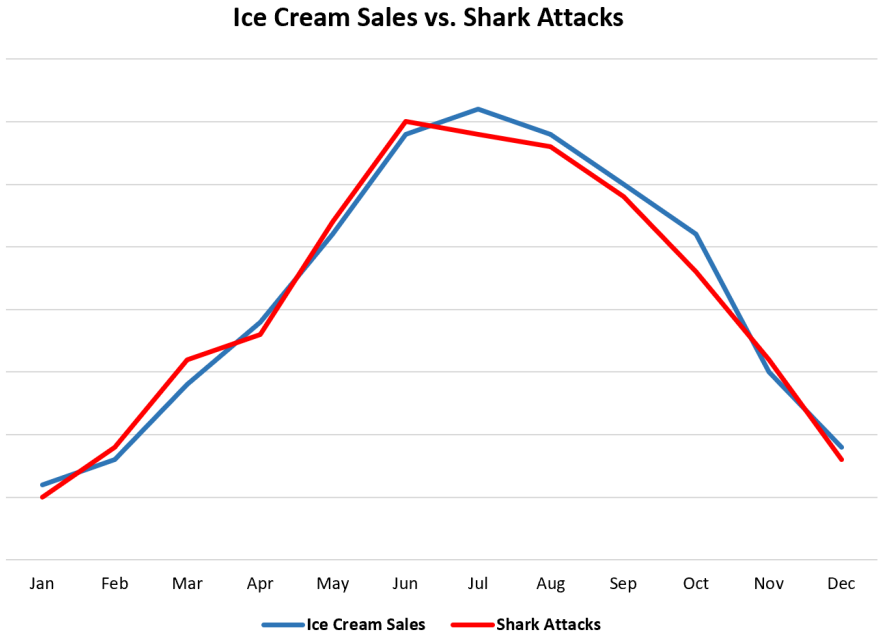


Figure 1.2: A plot of a hypothetical dataset of shark attacks and ice cream sales

Other associations between variables may not actually be grounded in any common causes or similarities in the data whatsoever. The name for such relationships is *spurious correlations*, and they can arise anywhere.

> **Tip**
>
> Spurious correlation describes an association between two variables that do not actually share a meaningful relationship.

For example, consider Figure 1.3, which illustrates the surprising relationship between the per-capita consumption of margarine in the US and the divorce rate in Maine[4]. There appears to be a very tight association between these annual time series, with a correlation coefficient of greater than 99%! Of course, no one actually believes that the amount of artificial butter produced has any impact on divorces, but blindly trusting the data could easily lead someone astray.

[3]Plot taken from https://www.statology.org/correlation-does-not-imply-causation-examples/
[4]This plot (along with other equally bizarre examples) can be found at https://www.spuriouscorrelations.com

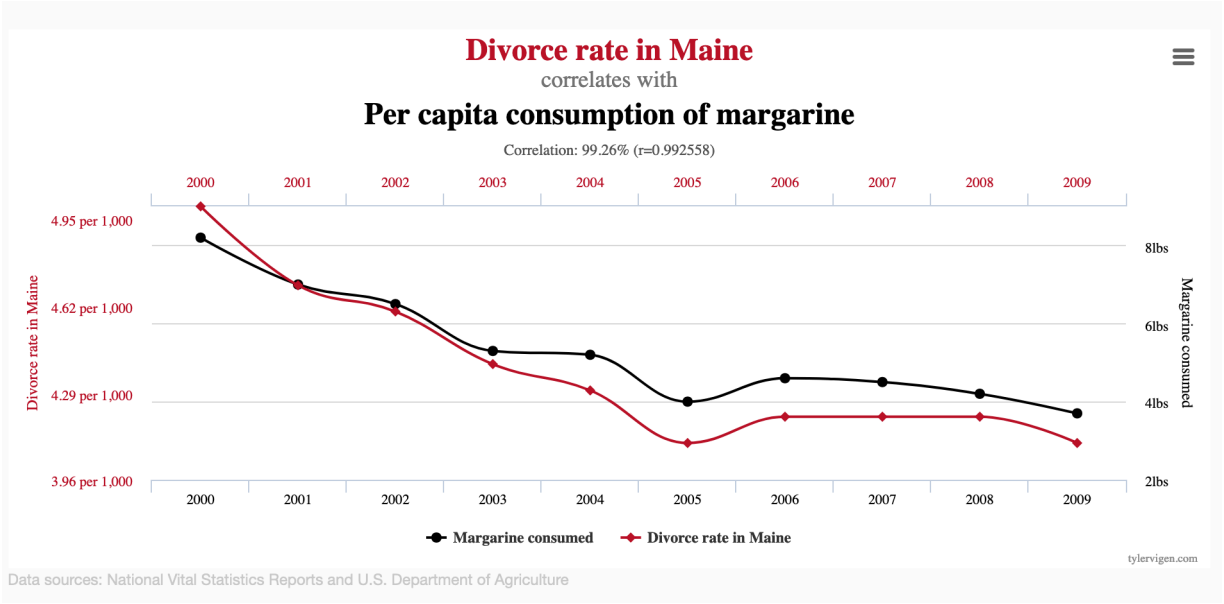


Figure 1.3: An example of a spurious correlation

These examples are somewhat extreme for the purpose of illustrating concepts, but similar challenges analysts encounter in the "real world" will undoubtedly be more nuanced and harder to identify. With the vast amounts of data and computing power available to companies, it is easy to see that these databases are ripe for mining correlations – even those that are entirely spurious in nature. Keen analysts will partner with subject matter experts to learn more about their domain, giving them more of an ability to discern whether a particular association makes sense. However, domain-specific insight does not remove the risk of mistaking a spurious correlation for something more, and confirmation bias is always lurking for even the most seasoned industry veteran or academic researcher.

1.1.2 Challenge #2: Simpson's Paradox

The other major challenge that arises in the context of data analysis is a phenomenon known as Simpson's Paradox, which describes a reversal in an apparent relationship between variables when controlling for another variable (Sprenger and Weinberger 2021).

> **Tip**
>
> Simpson's Paradox occurs when a relationship between two variables disappears or reverses when taking another factor into account.

To illustrate this, we'll use the standard dataset investigating gender bias in admissions at

UC Berkeley in 1973[5]. Figure 1.4 shows the high-level admission rate for male and female applicants, where there appears to be a significant bias against female students.

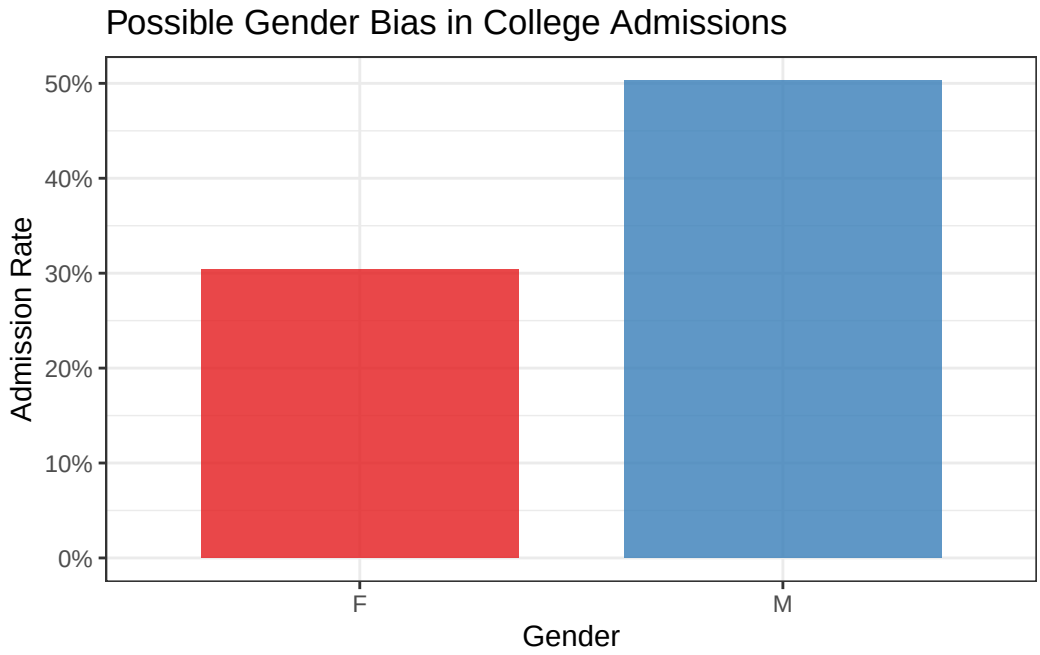


Figure 1.4: Plot of high-level admission rates by gender

As we can see, the overall admission rate of male applicants is *much* higher than for female applicants: a difference of roughly 30% versus 50%! This appears very concerning, however, things become less clear when looking at admission rate within each major separately. Figure 1.5 shows the male and female admission rates for each major. Surprisingly, the trend actually reverses for most of the majors in the dataset.

But how can this be? If female admission rates are equal or higher within most departments, shouldn't the overall female admission rate also be higher? This is a perfect example of Simpson's Paradox, and it has to do with unequal distribution of applicants between departments. Figure 1.6 shows one aspect contributing to this phenomenon: the departments with the highest acceptance rates are overwhelmingly male-dominated.

In fact, if we fit a logistic regression model predicting admission from the major and the applicant's gender, the coefficient for gender is quite near to zero and actually slightly favors female applicants: note the coefficient for `GenderM` in the following model output, denoting the impact of being male on admission odds.

[5]Accessed via https://discovery.cs.illinois.edu/dataset/berkeley/

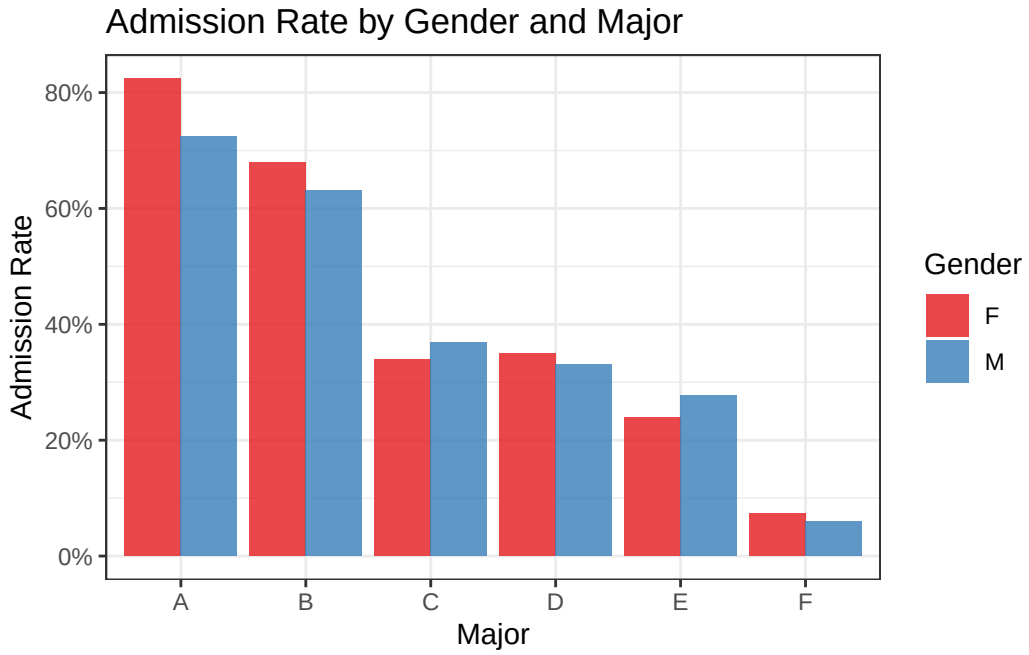


Figure 1.5: Reversal of trend when controlling for major

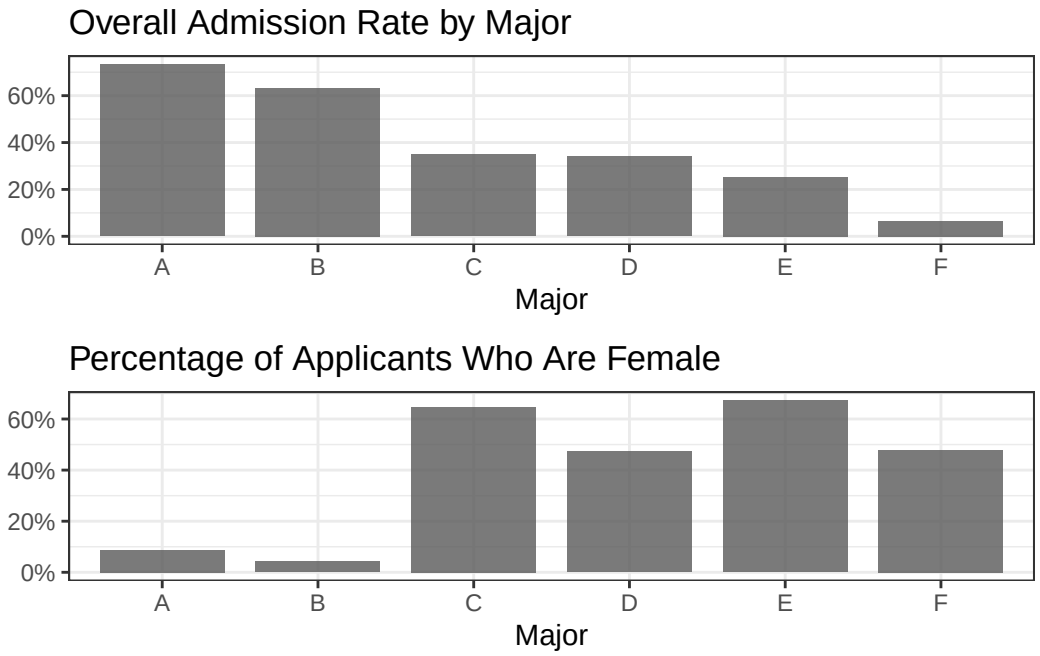


Figure 1.6: Admission rates and gender distribution by major

```
Call:
glm(formula = admission_int ~ Gender + Major, data = admissions)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.739957   0.018508  39.980  < 2e-16 ***
GenderM     -0.007018   0.015084  -0.465    0.642
MajorB      -0.100760   0.021877  -4.606 4.21e-06 ***
MajorC      -0.387799   0.020768 -18.673  < 2e-16 ***
MajorD      -0.396616   0.020668 -19.190  < 2e-16 ***
MajorE      -0.485950   0.023599 -20.592  < 2e-16 ***
MajorF      -0.670465   0.021310 -31.462  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.1903506)

    Null deviance: 1184.22  on 4838  degrees of freedom
Residual deviance:  919.77  on 4832  degrees of freedom
AIC: 5714.1

Number of Fisher Scoring iterations: 2
```

Ultimately, whether any gender bias is actually present in this dataset is not exactly clear, and other questions arise: are female students applying to more competitive departments, or are the male-dominated majors receiving significantly more funding than the others? We cannot know from this dataset, but it does illustrate the behavior of Simpson's Paradox.

Simpson's Paradox can appear in virtually any context, presenting much more nuanced problems which are more difficult to identify and manage than in the admissions example. When these reversals of trends occur "in the wild," it may not be obvious which variables should be included in an analysis. Given the large amount of data available for analysis, there may be a temptation to throw every available factor into a model (or even just the "usual suspects" from easily accessible databases) and accept whatever it returns. It can be tricky to understand what information is actually useful in answering a particular question, and traditional methods provide little guidance for how to proceed. When a particular coefficient is going to be used as the basis for an important decision, Simpson's Paradox can make it difficult to know which result to trust.

1.1.3 Causality to the Rescue

Causal inference provides a solution to both of these problems. We will dive deeper into the specific details in the next chapter, but we will now give a high-level introduction into how causal methods address these issues.

Causal inference is the name given to a formal process which allows us to measure causal effects from data: even messy observational data! This is made possible by a careful scientific approach which involves outlining assumptions and performing tests to determine whether such an estimate is possible in a given setting. Under the right conditions, this approach allows us to extend our findings from correlational to causal.

One foundational concept of causal inference is the need to define causal relationships that exist between variables: for example, by saying that *A causes B*, we are explicitly assuming that *B does not cause A*. We can enumerate all of the relevant relationships for a specific context, which in turn outlines all of the assumptions that are made about the environment. The process of defining these relationships within a directed acyclic graph (DAG) also provides a benefit of being able to know when variables are independent of each other. We will go into greater detail on *d*-separation and causal graph construction later, but what's important for now is that this practice of defining causal relationships provides a strong defense against Simpson's Paradox – we will only control for a particular variable when necessary, and we will do so in the manner specified by our assumptions and the causal question we are asking.

This practice also helps us avoid issues from spurious correlations: any relationship between two conditionally independent variables in a causal graph is assumed to be spurious, and will not affect the causal estimation.

1.2 A Shift in Perspective

One of the biggest changes when going from a standard analytical workflow to that of causal inference is the difference in mentality. For example, a typical machine learning task will focus on training a model whose predictions perform the best for a particular task, while statistical modeling uses tests of significance and goodness-of-fit to evaluate hypotheses. Causality is different, thanks to a framework which somewhat lines up with the scientific process. To summarize, the causal inference process requires us to:

- Formalize a question (e.g. *What if* X happened? *Why* did Y occur?)
- Enumerate assumptions about the environment and how variables relate to one another
- Collect relevant data
- Determine whether causal estimation is possible (meaning that the underlying assumptions hold and the causal quantities can be computed from the available data)
- Estimate the quantities of interest

In fact, actually producing the *causal estimand* (i.e. the quantified treatment effect) is a relatively small part of the causal inference process, and the math involved is often quite simple: in many cases, it's no more complex than logistic regression or calculating the expected values of a distribution. Much more of the effort is spent in the experimental design phase: ensuring you have properly framed the question, consulting with domain experts to validate the causal graph, and determining whether your desired result is actually identifiable from the data you have (and if not, identifying what new sources of data you would need to proceed).

This process culminates in a causal estimate which is based on the assumptions enumerated in the design phase and the data collected. If the causal model is an accurate reflection of the environment, the estimand can be interpreted as a causal effect. If it is not, more work is needed to make the assumptions match reality. This falsifiability – via objections to the causal graph – is a crucial feature of the causal inference process, as it allows for iterative refinement of the assumptions until it is possible to interpret the effects as something more than simple correlation. This is the beauty (and challenge) of causal inference!

1.3 Causal Inference: A Brief History

Before we dive into the details of causal inference methods, it might be helpful to have some historical context to understand how these techniques developed. In this final section of the first chapter, we will give a brief overview of causality: starting from the early days and finishing with examples of how modern companies are using causal inference in applied settings.

1.3.1 Beginnings

While related work can be traced back to the early 20th century, Donald Rubin's 1974 entry in the *Journal of Educational Psychology* entitled "Estimating Causal Effects of Treatments in Randomized and Non-Randomized Studies" introduced what would eventually be known as the *potential outcome framework*. In Rubin's words, "The basic conclusion is that randomization should be employed whenever possible but that the use of carefully controlled nonrandomized data to estimate causal effects is a reasonable and necessary procedure in many cases" (Rubin 1974). Rubin encourages practitioners working with observational data to think carefully about the other variables (aside from the treatment) that may impact the target of an analysis. As we saw earlier in this chapter, the choices we make about whether to control for a factor can significantly impact our results. Rubin's work represents a major step forward in the ability to interpret estimates from observational data as something more than simple correlation.

Starting in the late 1980s, Judea Pearl produced a large body of work solidifying causal inference as a scientific practice, culminating in his 2000 textbook *Causality* (Judea Pearl 2000). Pearl's early work focused on the use of directed acyclic graphs (DAGs) to represent causal structure, and he built upon this to develop a robust, thorough language and framework for

conducting causal inference. Specifically, Pearl introduced *do*-calculus, the language of interventional adjustments conducted on observational data – *do*-calculus was a groundbreaking development because it provides a set of formal definitions for how to perform adjustments in a way to achieve causal understanding.

Another significant contribution in (Judea Pearl 2000) is the ranking of causal inference problems into tiers, which later became known as the *Ladder of Causation* (also called the *Pearl Causal Hierarchy*). The Ladder of Causation places causal queries into the following "rungs:"

1. **Prediction**: how does knowledge of one variable change our expectation of another variable? For example, how does admission rate differ by gender?
2. **Intervention**: how does our target respond if we force the treatment to take a certain value? For example, if we randomly assigned male and female students to apply to each major, how does the admission disparity change?
3. **Counterfactual**: why did a particular outcome occur, and would it have been different under different circumstances? For example, would a particular female student have been accepted had she been male and applied to the same major?

Following the ladder illustration, as we go from one rung to the next we are asking increasingly causal questions. The first rung represents what is typically done with descriptive statistics or standard statistical modeling: for example, conditional expectations fall on the the first rung on the ladder of causation. These are methods that describe what actually happened within the natural world from which the data originated.

The second rung on the ladder of causation involves action on the part of the investigator. The questions on this rung are interventional in nature, meaning that a particular variable is fixed (either manually as in the case of a randomized controlled trial, or through an adjustment based on *do*-calculus). These questions seek to understand *what if*: for example, if I were to take a particular medication, would it make me feel better? Many of the objectives of predictive modeling are inherently asking similar questions: *what if* the predictors change in a certain way, based on the patterns identified in the training data?

The counterfactual questions on the third rung represent the groundbreaking notion of querying *why something occurred* or *what would have happened* in a so-called "dream world" made possible by the causal modeling process. Counterfactual reasoning opens a door to questions that are impossible to answer using only "second rung" tooling. These methods allow us to understand what caused particular outcomes, generate treatment effects at the individual level, and explore other hypothetical scenarios within our causal environment.

Pearl, his students, and his contemporaries have continued developing new methods within causal inference, and this work is beginning to grow in application throughout academic research and in industry.

The chapters that follow will introduce these techniques in greater detail and provide examples of how to use them. We will provide the relevant theory necessary to use and understand

these methods, but readers who are interested in going deeper into the theory of causality are encouraged to explore Pearl's work, including *Causality*.

1.3.2 Initial Adoption and Applications

Two areas of early application of causal inference came within the fields of economics and epidemiology. Each field shares the common goal of understanding the effects of policy changes, individual behavior, and other external factors; and each field had standard quantitative methods in place long before the formalization of causality as a scientific practice.

Prior to Pearl's work, economists had been using structural equation models (SEMs) to describe cause-and-effect relationships between variables since the 1950s (Reiss and Wolak 2007). These SEMs are similar in nature to the structural causal models (SCMs) introduced by Pearl; however, in *Causality*, Pearl contends that the language of SEMs does not allow for the distinction between causal questions and statistical questions (Judea Pearl 2000). Pearl's work on SCMs extends the structural model using the language of causal inference, the benefits of graphical modeling, and the ability to operate within nonlinear settings. In the years after *Causality*, econometricians began adopting causal inference methods within various contexts including policy evaluation (Heckman 2008).

Similarly, the field of epidemiology had its own methods of quantifying causal effects. These "classic" methods made important contributions to the quantitative discovery of disease causes, including providing evidence that smoking causes lung cancer. However, a crucial drawback of these methods include their lack of a "formal basis for evaluating causal hypotheses" (Glass et al. 2013). Pearl's methods provide this as a primary feature through the ladder of causation and *do*-calculus.

Naturally, debate still exists within both of these fields regarding whether causal inference via Pearl's methods should be adopted more widely. However, the potential outcomes framework is continuing to find applications within these domains.

1.3.3 Recent Applications in Industry

In recent years, causal inference has found several applications in industry. In this section, we will provide some examples of how modern companies are using causal methods in real-world settings. Don't worry if some of these terms are unfamiliar to you — we will provide definitions and examples of the underlying methods in the next few chapters. The goal of this section is to get you excited by showing how these techniques are having an impact in applied settings!

Personalized marketing is a common tactic used in industry: companies want to target their advertising efforts on the customers who are most likely to respond. One approach to this targeting problem is a method called *uplift modeling*, which uses counterfactuals to identify the potential effect of an action in order to target the individuals with the most opportunity.

Scientists at Uber developed a Python library called `CausalML` to package the common processes involved in uplift modeling (H. Chen et al. 2020). We'll talk more about uplift modeling in Chapter 3.

One company currently using causal inference in a significant way is Netflix: the video streaming giant has published various blog posts and whitepapers (such as (Wong 2020)) detailing their adoption and development of applied causal inference methods. One such blog post[6] outlines some of the specific uses of causal inference throughout their organization, from gaining understanding about how users respond to different features to developing a general framework for causal ranking within recommender systems.

Many published applications of causal inference in industry are focused within testing and experimentation: many technology companies run *lots* of tests to determine the effectiveness of new features, the appeal of new designs, or the impact of changes in processes. Running large-scale A/B tests can be expensive, and in some cases, it may not be feasible to directly test some of the "treatments" that need to be examined. On top of this, A/B tests are inefficient, and it can take a very long time to collect enough data to have trustworthy results. In the blog post, researchers at Netflix describe how they use causal inference in various ways to augment and improve their A/B testing regimes, including the use of double machine learning to match similar observations and extract incremental differences in outcomes.

1.4 Books and Resources

Before closing out the first chapter, we want to leave you with a list of helpful resources: namely books, courses, conferences, and codebases which pertain to causal inference. We use some of these resources and tools throughout the book; others will be helpful for readers with varying degrees of interest in specific applications or focuses in causality.

Books

No single resource will contain everything about causality – in this book, we make the choice to omit some theoretical detail in favor of practical application. In order to provide guidance to readers embarking on the journey of causal inference, we have compiled a list of other books and online courses that we believe would be beneficial. The list includes resources that cover causality at various levels of depth and complexity, providing readers with the opportunity to explore the topic in a manner that aligns with their needs and experience level.

[6] https://netflixtechblog.com/a-survey-of-causal-inference-applications-at-netflix-b62d25175e6f

Name	Description
Causal Inference in Statistics: A Primer	The book is highly recommended as a primary source of introductory knowledge on the subject for a wide range of individuals.
Elements of Causal Inference: Foundations and Learning Algorithms	The book is an excellent resource for readers who have a background in machine learning and understand the connection between causal inference and machine learning. It offers in-depth coverage of various topics, including causal discovery, structural learning, and structural causal models.
Data Analysis Using Regression and Multilevel/Hierarchical Models	The book features two important chapters devoted to the subject of causal inference and its connection to the widely-used machine learning technique of regression analysis. The chapters provide an introductory treatment of the subject and approach it in a practical, hands-on manner that is highly effective for practitioners.
Fundamentals of Causal Inference (With R)	This textbook is a good resource for researchers and students who seek to gain an understanding of the principles and techniques involved in causal inference with R background. The book is organized such that each chapter includes an R implementation of the introduced causal concepts and models, as well as accompanying datasets. Also, each chapter provides exercises designed to reinforce the understanding of the subject matter.

Name	Description
Observation and Experiment: An Introduction to Causal Inference	The author of this text employs a practical and accessible approach, requiring minimal mathematical and statistical knowledge, to elucidate crucial concepts and techniques. Drawing on examples from a diverse range of fields, including clinical medicine, economics, public health, epidemiology, clinical psychology, and psychiatry, the author's use of concrete examples and relatable case studies allows for a better understanding of complex ideas and abstract principles. Specifically, the book provides readers with a clear understanding of how to design and interpret randomized trials, as well as the distinctions between observational studies and randomized trials. The text also covers techniques for identifying, investigating, and evaluating bias in observational studies.
Causality: Models, Reasoning, and Inference	The book begins by covering fundamental concepts such as probabilities, graphs, and causal models before delving extensively into advanced causal techniques, including structural equation modeling. Subsequently, complex applications of these techniques in social sciences and economics are discussed at length. This book is particularly beneficial for individuals with prior knowledge of Structural Causal Models (SCMs) who seek to expand their expertise in this area. The book's comprehensive coverage of theory, particularly bounding effects and counterfactuals, is particularly noteworthy.

Name	Description
Explanation in Causal Inference: Methods for Mediation and Interaction	This book offers a thorough review of the latest advances in the causal inference literature pertaining to the essential areas of mediation, interaction, and spillover effects. Beginning with fundamental concepts and gradually progressing to more complex topics, the book covers recent developments in mediation and interaction research. The chapters extensively incorporate sensitivity analysis techniques to address violations of assumptions, a crucial aspect of the book's contents. These techniques are of value even to those employing conventional estimation methods.
Causal Inference: What If	The book is structured into three progressively challenging parts, beginning with causal inference without models, advancing to causal inference with models, and culminating in the analysis of causal inference from complex longitudinal data. The book strongly emphasizes the precise formulation of the causal question and highlights the need for subject-matter knowledge in observational causal inference. The book includes practical examples, as well as "Fine Points" and "Technical Points," to elaborate on key topics. The book is intended for researchers in a wide range of disciplines, including epidemiology, statistics, psychology, economics, sociology, political science, and computer science, and has been tested in multiple universities for graduate and advanced undergraduate courses in causal inference.

Name	Description
Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction	This book provides a thorough and comprehensive exploration of causal inference, employing the "potential outcomes" approach to establish a connection between theoretical underpinnings and real-world analyses that inform evidence-based decision-making across multiple fields, including medicine and public policy. The book presents invaluable guidance for designing research that focuses on causal relationships, and for interpreting research results with appropriate consideration. The authors draw on a broad range of examples using real data from their extensive research portfolios, which effectively elucidate important concepts and practical issues.

Tools

In addition to books, it is helpful to have established code bases to serve as a reference and simplify the application process. The following table includes the libraries we used in this book.

Name	Description
DoWhy	`DoWhy` is a Python library which makes it easy to estimate causal quantities given a dataset and a causal graph. `DoWhy` creates dedicated objects for causal models and estimates, allows the specification of a causal graph in graph markup language (GML), and works nicely with standard PyData tooling such as Pandas and NumPy. `DoWhy` is used for causal inference tasks throughout this book.

Name	Description
EconML	`EconML` is a library of Python implementations for many causal techniques, mostly centered around estimation of heterogeneous treatment effects. Its meta-learners and double ML techniques are used in the case studies for Chapters 3 (Estimation) and 5 (NLP)
CausalML	`CausalML` is a Python library designed to easily calculate treatment effects from observational data, including uplift modeling and many of the meta-learners introduced in Section 3.5
CausalLift	`CausalLift` is another Python library for uplift modeling
causal-learn	`causal-learn` is a library for causal discovery in Python. `causal-learn` implements many causal discovery algorithms and allows for extension and implementation of custom methods.
Bayesian Model-Building Interface (Bambi)	`Bambi` makes it very easy to fit Bayesian mixed-effects models in Python, which is helpful in causal modeling.

Conferences and Workshops

The following is a (non-exhaustive) list of conferences and recurring workshops which focus on causal inference.

Name	Description
American Causal Inference Conference	ACIC is organized annually by the Society for Causal Inference and its topics include theory and application of causal inference
Causal Learning and Reasoning	CLeaR is a relatively new conference whose topics range from fundamental theory of causation to cutting-edge application and adaptation in machine learning contexts
Miscellaneous	Workshops pertaining to causal inference appear often at conferences such as ICML, NeurIPS, and ICLR (to name a few)

1.5 Conclusions and Looking Ahead

In this chapter, we have illustrated two common challenges that arise when trying to interpret trends within data: Simpson's Paradox and spurious correlations are pervasive within every domain, and they can wreak havoc on any analysis. We have introduced how causal inference provides a scientific approach to dealing with these challenges, and we explored some of the questions that causal inference seeks to ask and answer. Finally, we gave some historical context along with some examples of how companies are using causal methods to deal with real-world issues.

Now it's time to jump into the actual methods! The next few chapters will build a solid foundation of understanding in causal inference concepts, introduce the math necessary to apply these techniques, and walk through the full process of how to actually approach a causal inference problem.

2 Causal Inference: Theory and Basic Concepts

This chapter offers a comprehensive overview of the fundamental concepts and theoretical foundations involved in explaining causal effects through the potential outcome framework. It provides insights into various challenges encountered in causal inference and presents a logical framework to address these challenges effectively. In this chapter, we present a comprehensive case study that utilizes a widely recognized causal study conducted with an educational television program. This case study serves as a practical example, aiming to offer readers a firsthand understanding of the concepts discussed in the chapter. The chapter concludes with solutions to numerous challenges from the Gelman and Hill book, intending to enable readers to develop the acumen required to contemplate, model, and apply the theory discussed in the chapter.

2.1 A Motivating Example

Let's consider a scenario where researchers are conducting a study to evaluate the effectiveness of an education program designed to improve students' math skills. The main question they want to answer is whether the education program itself is responsible for the observed changes in math performance. To understand this, let us explore the concepts of association and causality and how they relate to evaluating program effects.

Imagine a group of students participating in the study. Some students are selected to receive the education program, while others do not receive the education program. The researchers want to determine if the education program directly impacts students' math skills.

2.1.1 Question 1: Is there an Association?

The first step is to determine if there is an association between participating in the education program and improvements in math skills. Researchers compare the math performance of the students who received the program with those who did not. If the students who received the program consistently demonstrated higher average math scores than those who did not, it suggests an association between program participation and improved math skills.

2.1.2 Question 2: Is there Causation?

Establishing causality is the next step in understanding the true impact of the education program. While an association indicates a relationship between program participation and math performance, it does not prove causation. To determine causality, researchers need to consider other factors that may influence math skills and design the study to isolate the program's effects. These factors, sometimes related to students' prior knowledge or external support, could potentially influence the relationship between the program and math performance.

2.1.3 An Approach

Researchers can employ a randomized approach to address the issues and better assess the causal impact of the education program. They randomly assign students to different groups (a key assumption), ensuring they have similar characteristics. Doing so reduces the likelihood of other factors affecting the observed relationship between the program and math skills.

Through random assignment, any differences in math performance between the two groups can be more confidently attributed to the education program itself. Randomization helps mitigate the influence of other factors, leveling the playing field and allowing researchers to draw more accurate conclusions about the program's effects. While randomization does not guarantee the complete elimination of these other factors, it helps in reducing their potential impact.

By comparing the average math performance of the students who received the program with those who did not, researchers can assess the program's potential impact. Suppose the group of students who received the program consistently demonstrates higher average math scores than those who did not. In that case, it provides evidence that the education program may have a causal effect on improving math performance.

2.2 Potential Outcome Framework

This section will discuss the potential outcome framework, which represents a fundamental building block for causal inference. Although Splawa-Neyman originally introduced the potential outcome concept, it was D. Rubin's groundbreaking research on estimating causal effects in both randomized and non-randomized studies that established it as the *lingua franca* of causality-based frameworks (Rubin 1974; Splawa-Neyman, Dabrowska, and Speed 1990).

2.2.1 Basic Terminology

In order to clearly illustrate the concepts discussed, this section provides concise definitions for each term utilized within the potential outcome framework. The framework serves as a means

to connect the various elements, including treatment (referred to as policy, intervention, etc.), unit (encompassing individual, sample, etc.), and outcome (effect, function, response, etc.).

- **Variables**: Variables/attributes/features are the things associated with describing the state of the universe in the given dataset. In the above scenario, the available variables in the data include the students' group assignment (education program or no education program), math performance scores, and other potential variables such as demographics, prior academic achievement, and any additional relevant characteristics that may be collected as part of the study.
- **Unit**: Unit/Sample/Individual is the thing that one subjects to the treatment and on which the effect or the outcome is observed. It is defined as the atomic research object in the treatment effect study. For example, individual student participating in the study is the unit/sample.
- **Treatment**: Treatment/Policy/Intervention is the action or activity the unit is subjected to. We will denote this by T, where $T \in \{1, 2, ..N_T\}$ denotes $N_T + 1$ treatments possible. In most of the examples in the literature, T is considered to be binary ($T = 1$). The treatment in the example represents the education program itself, which is given to a specific group of students.
- **Treated group and Control group**: The group of units or samples exposed to any treatment is considered a treated group. When treatment values are binary, the group exposed to the treatment is the treatment group, and the other group not exposed is called the control group. In the example, The treated group consists of students who receive the education program and the control group consists of students who do not receive the education program.
- **Potential outcome**: When the unit is exposed to a treatment, the outcome is called the potential outcome. When an individual i is exposed to treatment $T = t$, then the potential outcome is given by $Y(T = t)$.Each student has two potential outcomes: $Y(T = 1)$ and $Y(T = 0)$. $Y(T = 1)$ represents the math score a student would achieve if they receive the education program, while $Y(T = 0)$ represents the math score they would achieve if they do not receive the program.

2.2.2 Counterfactuals

Counterfactuals represent the unobserved outcomes that would have occurred for each individual had they been assigned to a different treatment condition. In the context of evaluating the education program for students, counterfactuals help us imagine how a student would perform in math if they had been in a different group (given the program or not given the program).

For example, let us consider a student assigned to the treatment group and receiving the education program. The counterfactual for the student would be his/her math performance if the student had been assigned to the control group and not received the program. It allows

us to compare how the student performed with how he/she would have performed under a different condition.

By comparing the observed outcome (student's actual math performance after receiving the program) with the counterfactual outcome (the math performance student would have had without the program), researchers can estimate the causal effect of the education program on the student's math skills. This comparison helps determine whether the program positively impacted her math performance or if any observed improvement was merely coincidental.

Counterfactuals are important because they help us understand the specific contribution of the treatment (education program) by contrasting it with what would have happened if it had not been provided. They provide a basis for estimating the causal effect and enable researchers to conclude the effectiveness of the program.

2.2.3 Confounders

A confounding variable, also referred to as a confounding factor, confound, lurking variable, or confounder, is a variable that is positively or negatively associated with both the dependent variable (covariate) and an independent variable (outcome). Typically, the concept of a confounder is discussed in the context of a treatment study, wherein an external variable may influence the relationship between the treatment and the outcome of interest.

Examples of confounders in the students treated with the program scenario could be prior math skills or the socioeconomic background of the students. Confounding factors, such as prior math skills and socioeconomic background, can influence the relationship between the education program and students' math skills. Prior math skills may falsely make the program appear more effective if the treated group has higher initial math skills. Socioeconomic background can also independently impact math skills. To mitigate bias, it is essential to consider and control these confounders.

2.2.4 Indivdual Treatement Effect

Individual Treatment Effects (ITE) or Individual Causal Effect (ICE) refer to the specific causal effects of a treatment or intervention on each individual within a population. ITE captures the difference in outcomes that can be attributed to the treatment for a particular unit, comparing their actual outcome with the counterfactual outcome that would have occurred if they had received a different treatment or no treatment at all.

Observable or actual outcomes are different from the potential outcomes. For example, when a unit i is exposed to a treatment T with binary values (0 and 1), the causal effect can be written formally in terms of both the potential outcomes using a single equation, given by:

$$ITE = Y_i(T = 1) - Y_i(T = 0)$$

2.2.5 Randomized Control Trials/Tests (RCT)

Randomized control trials (RCTs) are experimental studies used in causal inference to evaluate the effectiveness of an intervention or treatment. They are designed to test the causal relationship between a treatment and an outcome by randomly assigning participants to treatment and control groups.

The process of conducting an RCT involves several steps. First, a sample of participants/units is recruited and randomly assigned to either the treatment or control groups. The treatment group receives the intervention or treatment being tested, while the control group receives either no treatment or a placebo. The outcome of interest is measured in both groups after a predetermined period of time.

> Tip
>
> By randomly assigning participants to the treatment and control groups, RCTs ensure that any differences between the two groups are due to chance and not to pre-existing differences in the characteristics of the participants. The process is crucial because it allows for the identification of causal effects, that is, the difference in the outcome between the treatment and control groups that is due to the treatment itself.

RCTs are considered the gold standard in causal inference, as they minimize the influence of confounding variables and allow for estimating causal effects.

In the student program example, randomly assigning a group of students to two groups: one group receives the education program, and the other group does not receive any additional intervention is the RCT process. This random assignment helps create two comparable groups in terms of their characteristics, reducing the likelihood of bias due to confounding factors.

2.3 The Fundamental Problem of Causal Inference

In the given scenario, counterfactuals pose an inherent challenge of observing multiple outcomes for the same individual. In the context of causal inference, the problem lies in the fact that we can only observe one outcome for each student, either with the educational program or without it.

This limitation creates a significant hurdle in establishing a causal relationship because we cannot directly observe what would have happened to a student had they been assigned to the alternative group.

Tip

The counterfactual is unobserved since only one outcome can be observed per unit for a treatment. Thus it is impossible to calculate the causal effect at the unit or individual level, as only one quantity is observed. This is widely regarded as the fundamental problem of causal inference (Rubin 1974).
As the counterfactuals remain unknown, the estimation of the individual treatment effect solely based on the available data, without the aid of additional tools or information, is not possible.

2.4 Overcoming the Fundamental Problem of Causal Inference

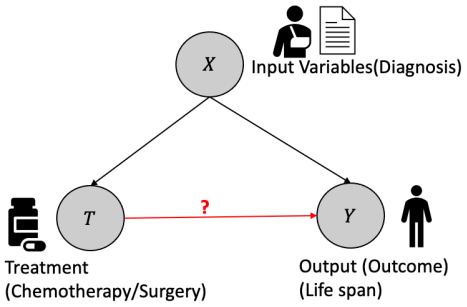


Figure 2.1: An Illustrative Example

In the context of causal inference, consider an example involving a cancer patient as shown in Figure 2.1. The patient's diagnosis determines the treatment plan, and both variables affect the outcome of survival time. The diagnosis is represented by the variable X, which could be health records, medical images, etc. The Treatment T, which is binary in this example (e.g., surgery or chemotherapy), but could be continuous in other cases, affects the survival time, which is a continuous variable represented by the outcome Y. It is assumed that the treatment is determined once based on the diagnosis and does not change. The treated outcome $Y_i(1)$ represents the outcome for surgery, while the control outcome $Y_i(0)$ represents the outcome for non-surgery (chemotherapy).

Causal effect question: **What is the causal effect of surgery versus chemotherapy on the survival time of cancer patients?**

Let us consider an example to demonstrate the situation with observational data. Let us consider four samples, each going through binary treatment $T = 0/1$ and resulting in outcome $Y = 0/1$.When the unit 1 undergoes the treatment $T = 0$, it results in outcome $Y(T = 0) = 0$, but we do not know the effect of $T = 1$ and hence $Y(T = 1) = ?$. Similarly, there is only one

observation for each subsequent unit, and the other is unknown, resulting in missing data for one of the outcomes in every case. Thus, the causal effect can be categorized as a missing data interpretation in statistical learning.

Unit(i)	Treatment T	Outcome Y	$Y(T=0)$	$Y(T=1)$	$Y(T=1)-Y(T=0)$
1	0	0	0	?	?
2	0	1	1	?	?
3	1	0	?	0	?
4	1	1	?	1	?

Thus, measuring the causal effect for each unit is impossible because of the missing data problem.

> **Question**: Is it possible to calculate the effect by taking the average of the Y(T=0) column and subtracting it from the average of the Y(T=1) column, disregarding the missing data (represented by the question marks)?
>
> Population measures such as the Average Treatment Effect (ATE) are used to address this challenge by taking the average of the outcomes column from the observational data.

ATE provides an estimate of the average causal effect of a treatment on the entire population, allowing us to assess the overall impact of the treatment intervention in the absence of counterfactuals and thus solving the fundamental problem of causal inference. In the next section, we will discuss some of the population measures that help us estimate the treatment effects.

2.5 Population Level Treatment Effects

In general, the **treatment effect** is a measure at the population level, treated group or subgroup level, or individual level that helps us quantify the relationship between the treatment T and the outcome Y based on the inputs X. Assume that a study is conducted with units that are randomly assigned to treatment and control, and the units represent a random sample from the target population. The employment of random sampling and random treatment allocation enables the estimation of the treatment's average causal effect on the population.

Some of the most well-known treatment effects using observations are:

- **Average Treatment Effect (ATE)** is the treatment effect measured at the population level as a difference between the treated and the control group and is given by:

$$ATE = \mathbb{E}[Y(T=1) - Y(T=0)]$$

Using the linearity of expectations

$$ATE = \mathbb{E}[Y(T=1)] - \mathbb{E}[Y(T=0)]$$

This expression can be written as the equivalent conditional expectations of Y given Treatment $T = 0/1$ as:

$$ATE = \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$$

Thus, from the causal effect calculation, we can remove all the unknowns and measure the average by treating the missing data with a value 0, as given below:

Unit(i)	Treatment$T$	Outcome Y	$Y(T=0)$	$Y(T=1)$
1	0	0	0	0
2	0	1	1	1
3	1	0	0	0
4	1	1	1	1

Thus, $ATE = \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0] = 0$

ATE calculation done here assumes $\mathbb{E}[Y(T=1)] - \mathbb{E}[Y(T=0)] = \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$. This assumption may not be true in all cases, resulting in the **associational difference** rather than **causal difference**, especially when some covariates or variables are confounding or the groups are not comparable. The next section will discuss the assumptions that help make the ATE the same as the associational difference.

- Average Treatment effect on the Treated group (ATT): The average treatment effect on the treated group can be defined as the expected difference between the outcomes $Y(T=1), Y(T=0)$ conditioned on the treated group $T = 1$ and is given as:

$$ATT = \mathbb{E}[Y(T=1)|T=1] - \mathbb{E}[Y(T=0)|T=1]$$

ATT focuses on the treated group and provides insights into the treatment's impact, specifically on those who underwent the intervention. ATT helps assess the effectiveness of the treatment for individuals who chose to or were selected to receive it.

- Conditional Average Treatment Effect (CATE): At the subgroup level, the average treatment effect conditioned on a specific subgroup, i.e. conditioned on a specific set of input covariates X, result in conditional average treatment effect (CATE) and is given by:

$$CATE = \mathbb{E}[Y(T=1)|X] - \mathbb{E}[Y(T=0)|X]$$

CATE is used when the treatment effect is expected to vary across different subgroups or individual characteristics within the population.

> **Tip**
>
> To summarize, ATE summarizes the treatment effect that applies to the entire target population. ATT is used when the interest lies in estimating the average causal effect of a treatment on those who received the treatment. CATE aims to understand the heterogeneous treatment effects and identify factors that moderate the impact of the treatment.

The choice of measure depends on the research question, the target population, and the level of heterogeneity in treatment effects of interest.

2.5.1 Assumptions in Causal Effect Computation

In this section, we will describe in detail some of the assumptions made to compute the causal effect and the implications of these assumptions.

2.5.1.1 Ignorability

> **Tip**
>
> This assumption states that the potential outcome for the treated $Y(T=1)$ and the control $Y(T=0)$ are independent of the treatment, i.e. $(Y(T=1), Y(T=0) \perp T)$.

Because of the ignorability assumption, we get the following equivalence in the calculation of the causal effect $\mathbb{E}[Y(T=1)] - \mathbb{E}[Y(T=0)] = \mathbb{E}[Y(T=1)|T=1] - \mathbb{E}[Y(T=0)|T=0]$.

> **Tip**
>
> The Ignorability assumption enables us to compute the ATE by ignoring the unknowns because $\mathbb{E}[Y(T=1)|T=1] - \mathbb{E}[Y(T=0)|T=0] = \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$

This assumption is also referred to as the exchangeability assumption. Ignorability and exchangeability are the same concepts, but they give different perspectives why the ATE is equal to the associational difference. Assume the treatment group $(T=1)$ and the control group $(T=0)$ both had the associated expectations $\mathbb{E}[Y|T=1] = y_1$ and $\mathbb{E}[Y|T=0] = y_0$ respectively. The exchangeability assumption states that the expectations remain unchanged if we

swap the sub-groups, i.e., the treatment and the control group, the expectations of the groups remain the same, i.e., $\mathbb{E}[Y|T=1] = y_1$ and $\mathbb{E}[Y|T=0] = y_0$ stays as before.

$$\mathbb{E}[Y(T=1)|T=1] \equiv \mathbb{E}[Y(T=1)|T=0] = \mathbb{E}[Y(T=1)]$$
$$\mathbb{E}[Y(T=0)|T=1] \equiv \mathbb{E}[Y(T=0)|T=0] = \mathbb{E}[Y(T=0)]$$

> **Tip**
>
> Ignorability/exchangeability gives the identifiability to the causal effect. A causal quantity is identifiable if it can be computed from the statistical quantity.

The equation below where the causal quantities, $\mathbb{E}[Y(T=1)]$ and $\mathbb{E}[Y(T=0)]$ are computed from the statistical quantities $\mathbb{E}[Y|T=1]$ $\mathbb{E}[Y|T=0]$, makes it identifiable.

$$\mathbb{E}[Y(T=1)] - \mathbb{E}[Y(T=0)] = \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$$

2.5.1.2 Conditional Exchangeability and Unconfoundedness

In analyzing causal effects, the exchangeability assumption is typically based on the assumption of unconfoundedness, where treatment assignment is independent of potential confounding factors. However, in practice, confounding factors are commonly present in the data, necessitating the estimation of average treatment effects while accounting for confounding.

Conditional exchangeability is defined where the potential outcome is independent of the treatment conditioned on the inputs, i.e. $(Y(T=0), Y(T=1) \perp\!\!\!\perp T)|X)$

$$\mathbb{E}[Y(T=1) - Y(T=0)|X] = \mathbb{E}[Y(T=1)|X] - \mathbb{E}[Y(T=0)|X]$$

(Linearity of expectations)

$$\mathbb{E}[Y(T=1) - Y(T=0)|X] = \mathbb{E}[Y(T=1)|T=1, X] - \mathbb{E}[Y(T=0)|T=0, X]$$

(conditional exchangeability)

Thus the causal estimates can be defined in terms of two statistical measures.

$$\mathbb{E}[Y(T=1) - Y(T=0)|X] = \mathbb{E}[Y|T=1, X] - \mathbb{E}[Y|T=0, X]$$

Conditional Exchangeability is observed in the data when we condition on X. Conditioning on X breaks the non-causal association between T and Y, effectively "blocking" the non-causal pathway at X.

Conditional exchangeability is a crucial assumption for conducting causal inference. With this assumption, we are able to identify the causal effect within specific levels of X, similar to how we did with the assumption of (unconditional) exchangeability.

To obtain the marginal effect that was previously obtained under the assumption of (unconditional) exchangeability, we can achieve this by marginalizing out variable X.

$$\mathbb{E}[Y(T=1) - Y(T=0)] = \mathbb{E}_X \mathbb{E}[Y(T=1) - Y(T=0)|X]$$

$$\mathbb{E}[Y(T=1) - Y(T=0)] = \mathbb{E}_X[\mathbb{E}[Y|T=1, X] - \mathbb{E}[Y|T=0, X]]$$

> **Tip**
>
> Mapping the causal quantity
>
> $$\mathbb{E}[Y(T=1) - Y(T=0)]$$
>
> to the statistical quanitity $\mathbb{E}_X[\mathbb{E}[Y|T=1, X] - \mathbb{E}[Y|T=0, X]]$, that can be derived directly from the observational data is called the adjustment formula.

The shift from exchangeability to conditional exchangeability in causal inference was motivated by the belief that it is a more realistic assumption. However, in practice, it can be challenging to determine with certainty if conditional exchangeability holds. There may exist unobserved confounding variables that are not included in the variables represented by X, thereby violating the assumption of conditional exchangeability. This is where randomized control trials (RCTs) come into play as they help address this issue by effectively randomizing the assignment of treatments, reducing the likelihood of unobserved confounding.

2.5.1.3 Positivity/Overlap

In a binary case, i.e., treatment with discrete value $T = 0$ *or* 1, if a subgroup has everyone in the treatment group ($T = 1$) or the control group($T = 0$), computing the average treatment effect will not be possible. Thus, some individuals or samples must be assigned to each group with a positive probability of making the computations possible. This is called the positivity assumption (overlap or common support).

For all values of input covariates ($X = x$) present in the group of interest $P(X = x) > 0$, then

$$0 < P(T = 1|X = x) < 1$$

and

$$0 < P(T = 0|X = x) < 1$$

Positivity is an important assumption, as it removes the zero probability cases in the computation of the treatment effect.

From the adjustment formula:

$$\mathbb{E}[Y(T=1) - Y(T=0)] = \mathbb{E}_X[\mathbb{E}[Y|T=1, X] - \mathbb{E}[Y|T=0, X]]$$

Assuming both X and Y are discrete, we can rewrite the adjustment formula as

$$\begin{aligned} ATE &= \mathbb{E}[Y(T=1) - Y(T=0)] \\ &= \sum_x P(X=x)(\sum_y yP(Y=y|T=1, X=x) - \sum_y yP(Y=y|T=0, X=x) \end{aligned}$$

based on the Bayes Rule, we can rewrite the above as:

$$\begin{aligned} &ATE = \mathbb{E}[Y(T=1) - Y(T=0)] \\ &= \sum_x P(X=x)\left(\sum_y y\frac{P(Y=y|T=1, X=x)}{P(T=1|X=x)P(X=x)} - \sum_y y\frac{P(Y=y|T=0, X=x)}{P(T=0|X=x)P(X=x)}\right) \end{aligned}$$

Without the positivity assumption, the denominator would 0 and result in not being able to compute the causal effect.

When we encounter a positivity violation, it implies that within a particular subgroup of the data, all individuals consistently receive either the treatment or the control. In such cases, estimating the causal effect of treatment versus control within that subgroup becomes unfeasible because we only observe either the treatment or the control. The alternative condition is never observed within that specific subgroup, making it impossible to estimate the causal effect in that particular context.

> **Tip**
>
> Put simply, our objective is to ensure that the distribution of covariates in the treatment group overlaps with the distribution of covariates in the control group. Hence positivity assumption is known as the overlap or common support.

2.5.1.4 No Interference

The no interference assumption ensures that treatment applied to one unit has no implications or effect on other units. This assumption is equivalent to

$$Y_i(T_1, \cdots, T_{i-1}, T_i, T_{i+1}, \cdots, T_n) = Y_i T_i$$

In our example of cancer and treatment, the chemotherapy treatment given to one individual patient does not impact the outcome of any other patient but him.

2.5.1.5 Consistency

The consistency assumption states that: When a unit is subjected to a particular treatment ($T = t$), the potential outcome does not change, i.e., $T = t \implies Y = Y(t)$. It also implies no different treatment versions with the same value, which can cause different outcomes.

If a particular person is given a chemotherapy treatment ($T = 1$), the outcome is that they recover ($Y = 1$). However, there cannot be another setting where there is a way chemotherapy can be administered ($T = 1$) to the same patient with all other conditions remaining the same, the outcomes are changed, i.e., they do not recover ($Y = 0$).

2.5.1.6 Stable Unit-Treatment Value Assumption (SUTVA)

In the existing literature, one often encounters the "stable unit-treatment value assumption" (SUTVA). It is imperative to postulate that the allocation of treatment to a particular individual (unit) within an experiment does not influence the result for another individual. This fundamental concept is commonly known as the "stable unit treatment value assumption" (SUTVA). Consequently, SUTVA combines the consistency and the absence of interference and is further based on deterministic potential outcomes.

Failure to uphold SUTVA would require defining a distinct potential outcome for the i-th unit, not only for every treatment administered to that unit, but for every possible combination of treatment allocations provided to all other units in the experiment.

2.6 Causality Inference: Logical Flow

A causal estimand represents the specific causal effect or quantity of interest that we want to estimate in a study. A **causal estimand** is any estimand that includes potential outcomes in its definition. It defines the causal relationship we seek to understand, such as the average treatment effect (ATE), the average treatment effect on the treated (ATT), or the conditional average treatment effect (CATE).

> Tip
>
> However, these causal estimands are often defined in terms of potential outcomes, which are not directly observable from the available data.

Therefore, the statistical estimand comes into play. A **statistical estimand** is a quantity derived from the available observational data and does not have a potential outcome.

Identification, as discussed in the last section, helps us move from a causal estimand $\mathbb{E}[Y(T = 1) - Y(T = 0)]$ to a statistical estimand $\mathbb{E}_X[\mathbb{E}[Y|T = 1, X] - \mathbb{E}[Y|T = 0, X]]$.

Once we have the statistical estimand, we can employ machine learning algorithms such as regression to compute the estimate from the statistical estimand. A machine learning estimator (e.g., Linear Regression) can be used to estimate $\mathbb{E}[Y|T=1,X]$ and $\mathbb{E}[Y|T=0,X]$ from the observational data.

Thus the entire logical flow from a target causal estimand to a corresponding estimate is shown in Figure 2.2.

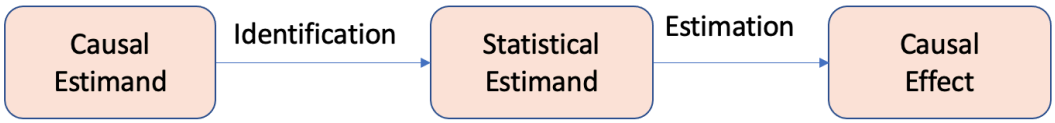


Figure 2.2: Causality Flowchart: From a Target Causal Estimand to an Estimate

Next, we illustrate how all of the assumptions discussed above and some general statistical assumptions are used to transform from the causal to the statistical estimands.

$$ATE = \mathbb{E}[Y(T=1) - Y(T=0)]$$

(Using the no interference assumption)

$$ATE = \mathbb{E}[Y(T=1)] - \mathbb{E}[Y(T=0)]$$

(Using the linearity of expectations)

$$= \mathbb{E}_X[\mathbb{E}[Y(T=1)|X] - \mathbb{E}[Y(T=0)|X]]$$

(law of iterated expectations)

$$= \mathbb{E}_X[\mathbb{E}[Y(T=1)|T=1,X] - \mathbb{E}[Y(T=0)|T=0,X]]$$

(unconfoundedness and positivity assumption)

$$= \mathbb{E}_X[\mathbb{E}[Y|T=1,X] - \mathbb{E}[Y|T=0,X]]$$

(consistency assumption)

This brings us to two important questions:

1. How do we perform identification and convert causal estimands into statistical estimands? What are the tools available for this process?
2. Once we have obtained statistical estimands, how do we proceed with estimation? What tools can be utilized for this purpose?

The next chapter will delve into Causal Graph modeling, which aids in providing a framework for practically performing the identification processes, i.e., conversion of causal estimands to statistical estimands and estimation methods. We will also discuss the algorithms and techniques for estimating the causal effect.

2.7 Case Study

The Electric Company study presents data from an educational experiment conducted in the 1970s on a collection of elementary school classes in two cities, Fresno and Youngstown. The primary objective of this experiment was to assess the impact of The Electric Company, a newly introduced educational television program, on students' reading ability.

For each grade and city, a small number of schools (10-20) were selected, and within each school, the two poorest reading classes were randomized into treated and control groups. The classes were randomly divided into treated and control groups, with exposure to the program serving as the treatment. Individual student data is not available for analysis, and as a result, the entire analysis was conducted at the classroom level.

To examine the causal impact of the program on the students, we will undertake experiments similar to those conducted by Gelman and Hill.

A version of this Python code is available in this Colab notebook.

2.7.1 Tools and libraries

We will use BAyesian Model-Building Interface **(Bambi)**, a Python library that works with the probabilistic programming frameworks **PyMC** and is designed to make it extremely easy to fit Bayesian mixed-effects. We will use it to model regression with various variables.

Python libraries such as **pandas** and **matplotlib** etc. are used for data processing and visualization.

2.7.2 Basic Regression Analysis

A general difference estimate is conducted, a regression analysis using a treatment indicator variable, under the assumption of a fully randomized experiment that generated the observational data and without the presence of any pre-test scores as variables. The obtained result shows a regression coefficient of **5.6** with a standard error of **2.5**. This initial estimate serves as the foundation, assuming that making modifications will enhance the estimate's accuracy.

```
# Fit the linear regression model
X = electric['treatment'].values.reshape(-1, 1)
y = electric['post_test'].values
reg = LinearRegression().fit(X, y)

# Print the coefficients and intercept of the linear regression model
print('Coefficients: ', reg.coef_)
print('Intercept: ', reg.intercept_)
```

```
Coefficients:  [5.65729167]
Intercept:  94.32083333333335
```

2.7.3 Regression Analysis Per Grade

The histogram plot of the post-test scores distributed across different grades, as presented in Figure 2.3, illustrates a considerable degree of variation among the test scores across various grades.

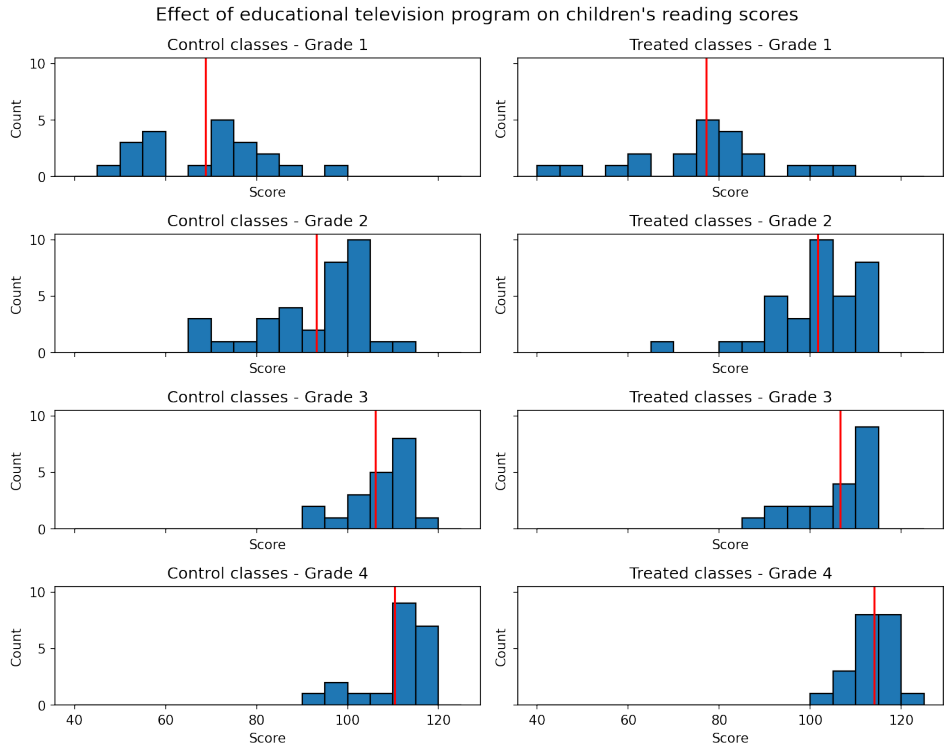


Figure 2.3: Scores Per Grade Distribution for Treated and Control

Considering the substantial heterogeneity observed in the test scores across different grades, it is logical to proceed with the next analytical step, which involves conducting individual regression analyses for each grade's dataset. This approach corresponds to fitting a model that accounts for the variability of treatment effects by grade, specifically the interaction between the treatment and grade indicators. It allows for the possibility of varying residual variance across different grades.

Figure 2.4 displays general effectiveness, particularly in the lower grades, albeit the large standard errors of estimation make it challenging to ascertain with certainty.

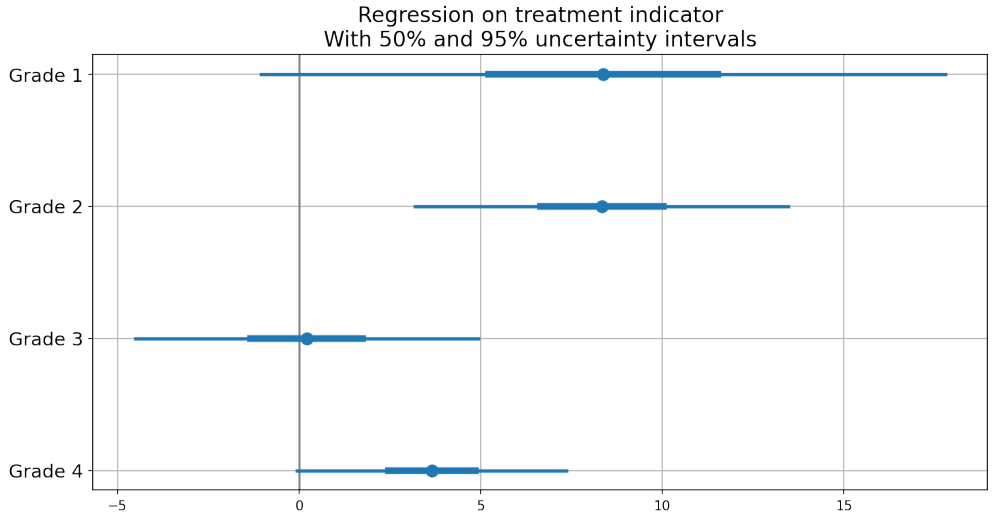


Figure 2.4: Regression on treatment indicator with 50 and 95 percent uncertainty intervals

2.7.4 Regression Analysis with Treatment and Pre-Test Scores

In an attempt to refine the treatment effect estimates, we incorporate the pre-test scores as an additional input variable alongside the treatment variable. Specifically, we fit separate models for each grade, with the treatment and pre-test variables included as input parameters. As the model does not account for any interaction, the treatment effect is presumed to remain constant across all pre-test score levels.

In Figure 2.5, the difference between the regression lines for the two groups depicts the estimated treatment effect as a function of the pre-treatment score, separately for each grade. It is observed that the regression lines for the treated groups lie slightly higher than those for the control groups.

In Figure 2.6, the regression analysis of post-test scores with treatment and pre-test scores is presented, with 50% and 95% uncertainty intervals illustrated through error plots and compared to one with just treatment. The inclusion of pre-test scores in the analysis provides

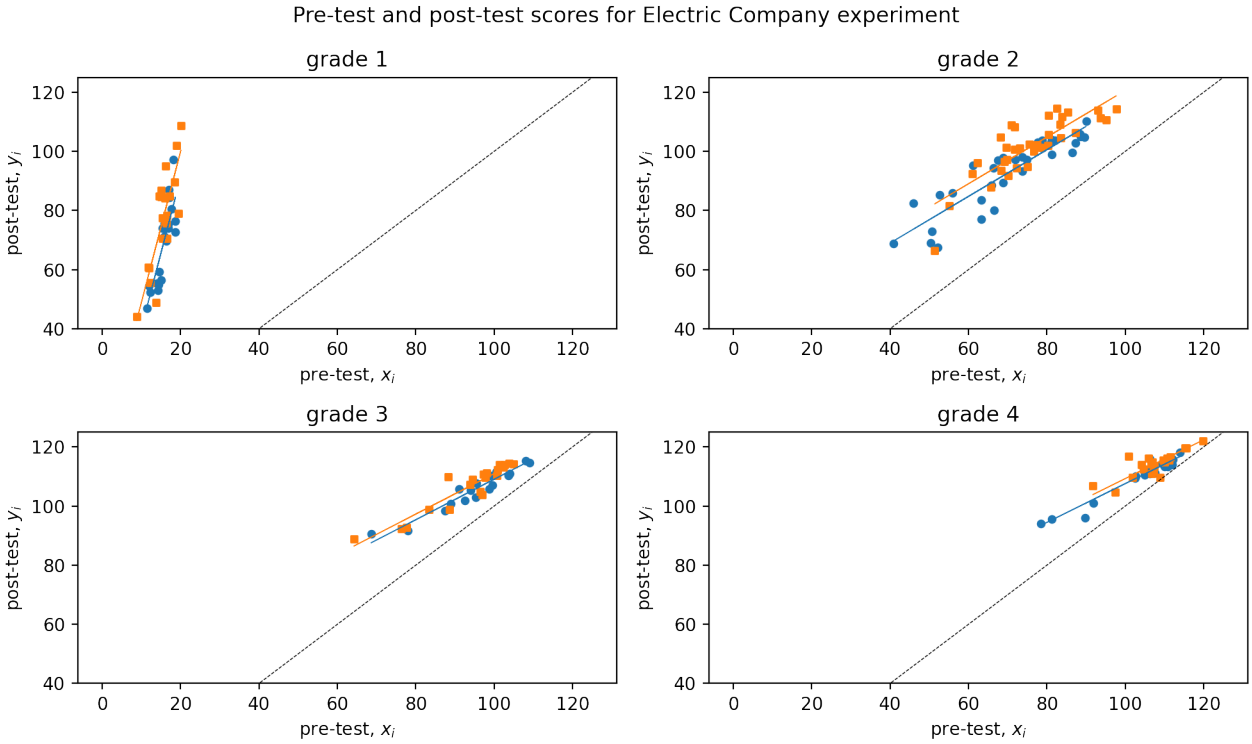


Figure 2.5: Regression with treatment and pretest scores per grade

greater clarity regarding the effectiveness of the treatment as compared to just the treatment. The results indicate that the treatment displays average effectiveness for each grade, with relatively more significant effects observed in the lower grades.

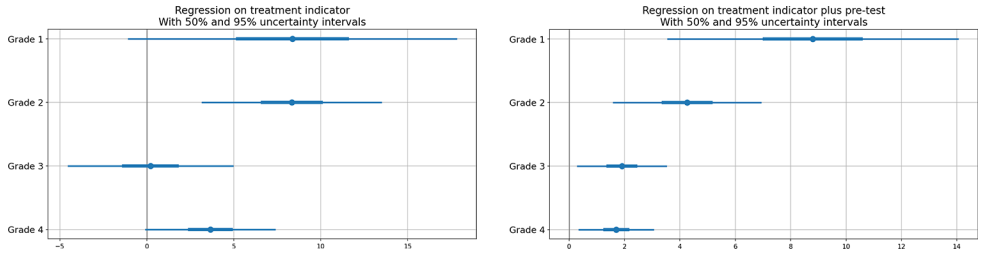


Figure 2.6: Coefficients of treatment on post scores with just treatment (left) and treatment plus pre-test scores (right), including 50 and 95 percent uncertainty intervals.

2.7.5 Interactions of Treatment Effect with Pre-Treatment Inputs

After incorporating the pre-test score as an input in the model, the subsequent analytical step enables its interaction with the treatment effect. This allows the treatment to influence both the intercept and the slope of the pre-test/post-test regression. Figure 2.7 shows regression with interaction element added.

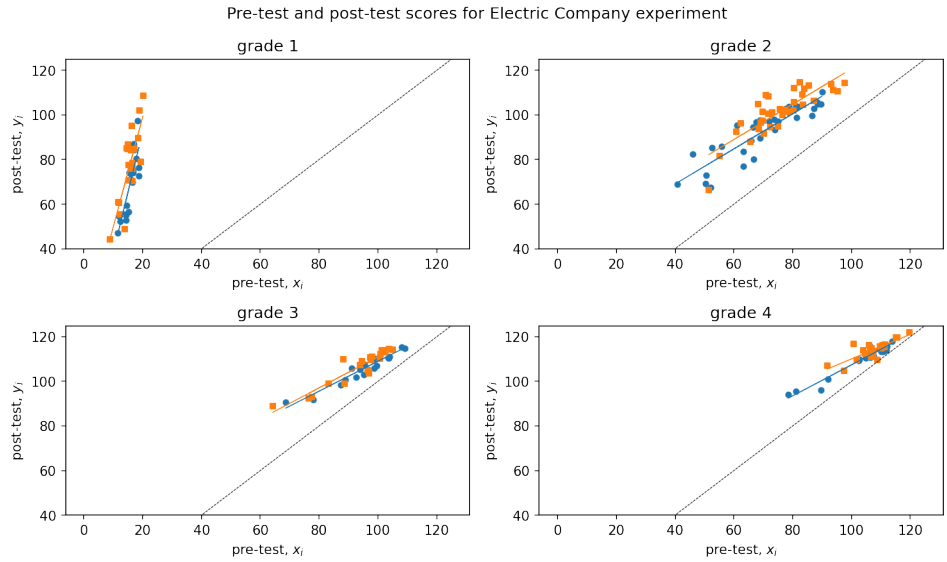


Figure 2.7: Regression with treatment and pre-test and their interactions on scores per grade

For grade 4, the estimated treatment effect is 3.7 with a standard error of 1.8,, when we add pre-test scores, the estimated treatment effect is 1.7 with a standard error of 0.7.

```
             coefficient  standard error
(Intercept)  110           1.3
treatment    3.7           1.8

             coefficient  standard error
(Intercept)  42           4.3
treatment    1.7          0.7
pre-test     0.7           0
```

Incorporating the interaction of treatment with pre-test introduces additional complexity. The estimated treatment effect, expressed as $17 - 0.15x$, cannot be readily interpreted without knowledge of the corresponding range of x. It is worth noting that pre-test scores have been observed to range between approximately 80 and 120 from the grade 4 plot as in Figure 2.7. Within this range, the estimated treatment effect varies between $17 - 0.15 \cdot 80 = 5$ for classes with pre-test scores of 80, and $17 - 0.15 \cdot 120 = -1$ for classes with pre-test scores of 120. The observed range illustrates the variation in estimated treatment effects with respect to pre-test scores, and does not represent the uncertainty associated with the estimation of treatment effects. To quantify the uncertainty, a graphical representation of the estimated treatment effects as a function of x can be generated, superimposing random simulation draws on the plot as shown in Figure 2.8.

The estimated treatment effect as a function of pre-test score is denoted by the dark line in Figure 2.8. This line represents the difference between the two regression lines in the grade 4 plot, as depicted in Figure 2.7. We can compute the treatment effect and is 1.8 with a standard error of 0.7 – similar to the result from the model adjusting for pre-test but with no interactions.

2.7.6 Effect of supplement,for each grade with pre-test.

The educational experiment detailed above includes an embedded observational study where, after treatments were assigned, the teacher for each Electric Company treatment group class decided to either replace or supplement the regular reading program with the Electric Company television show. This decision resulted in some classes watching the show instead of the regular reading program while others watched it in addition to it. The analysis of these observational data is simplified by treating the choice between the two treatment options, "replace" or "supplement", as randomly assigned based on pre-test scores, within the randomized treatment group.

A new variable, referred to as "supp", is generated in this study to differentiate between the replacement and supplement forms of treatment. The value of 0 is assigned to the replacement group, while the supplement group is assigned a value of 1. The control group is assigned a value of NA.

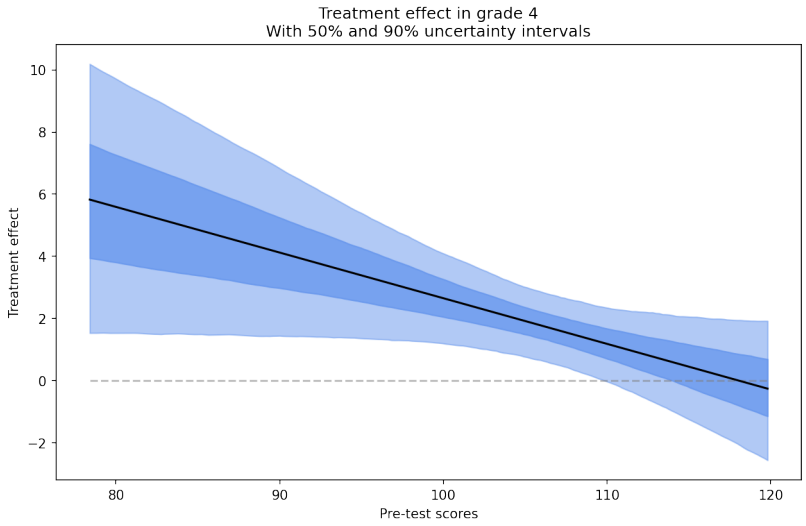


Figure 2.8: Treatment effect in grade with 50 and 90 perent uncertainty intervals for Grade 4 with interactions

Subsequently, the effect of supplement relative to replacement for every grade is evaluated through a regression of post-test on the indicator and pre-test.

The level of uncertainty is substantial, rendering the comparison inconclusive, barring grade 2 where the conclusion is less ambiguous as shown in Fig.@fig-withsupplement. Nevertheless, overall the results align with the reasonable supposition that supplementing is more efficacious than replacing in the lower grades.

2.8 Hands-on Excercises

This section aims to provide readers with a comprehensive understanding of the causal inference approach by solving various exercises from Gelman and Hill's Causal Inference chapter. The aim is to equip readers with the necessary skills to approach problems using causal reasoning and provide an opportunity to practice basic inference and estimation techniques.

2.8.1 Queston: Randomized Experiments in a Perfect World

(Adapted from Hill and Gelman, Chapter 9, Question 1)

Suppose you are interested in the effect of the presence of vending machines in schools on childhood obesity. What randomized experiment would you want to do (in a perfect world) to evaluate this question?

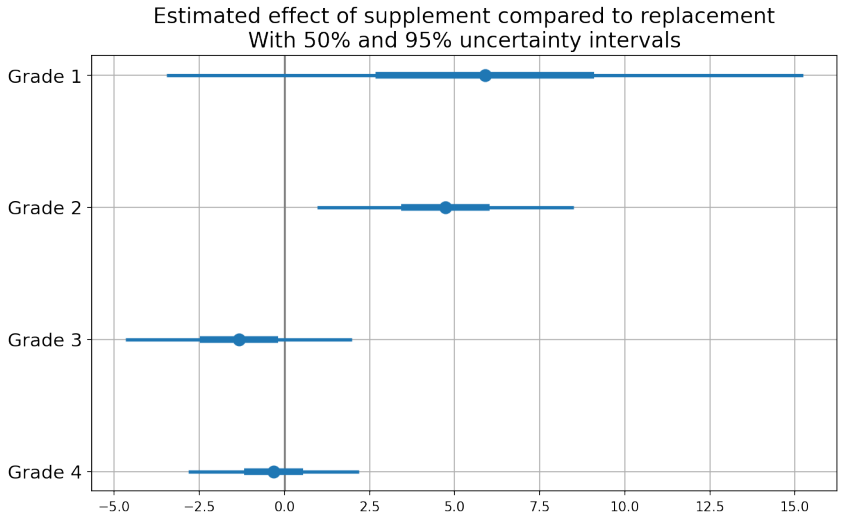


Figure 2.9: Estimated effect of supplement compared to replacement with 50 and 95 percent uncertainty intervals

A conceivable experimental design entails employing vending machines as the treatment and assessing the obesity prevalence as the outcome. A random assignment of schools to the treatment and control groups would be conducted, such that the vending machines would be installed in the treatment group while the control group would have no vending machines. Prior to the installation of the vending machines, the weight of children in all schools would be measured, and the same procedure would be repeated at the end of the school year.

2.8.2 Queston: Simulating Randomized Experiments in the Real-World

(Adapted from Hill and Gelman, Chapter 9, Question 2)

Suppose you are interested in the effect of smoking on lung cancer. What randomized experiment could you plausibly perform (in the real world) to evaluate this effect?

In real-world, one could identify an existing population that is already divided into smokers and non-smokers, and use statistical methods to adjust for potential confounding factors to estimate the causal effect of smoking on lung cancer. However, such observational studies have inherent limitations and the estimated effect may be subject to bias due to unobserved confounding factors.

2.8.3 Question: Real-world Causal Design Quesitons

(Adapted from Hill and Gelman, Chapter 9, Question 3)

Suppose you are a consultant for a researcher who is interested in investigating the effects of teacher quality on student test scores. Use the strategy of mapping this question to a randomized experiment to help define the question more clearly. Write a memo to the researcher asking for needed clarifications to this study proposal.

The clarifications for the study would include questions pertaining to:

Treatment and Control Groups: *Could you please clarify how you plan to assign students to the treatment and control groups? Will this be done randomly or through some other method?*

Sample Size and Selection: *Could you provide details about the population from which the sample will be selected? Will the sample size be large enough to ensure statistical validity?*

Teacher Quality: *How do you plan to measure teacher quality? Will this be based on standardized test scores, years of teaching experience, or some other metric?*

Test Scores: *What kind of test scores do you plan to use as an outcome measure? Will you use standardized tests or some other kind of test?*

Other Variables: *Are there any other variables that you plan to control for in the analysis? If so, could you provide more details on these variables and how they will be measured?*

2.8.4 Question:Understanding Effects of Treatment Using Causal Analysis

(Adapted from Hill and Gelman, Chapter 9, Question 4)

The table below describes a hypothetical experiment on 2400 persons. Each row of the table specifies a category of person, as defined by his or her pre-treatment predictor x, treatment indicator T, and potential outcomes y^0,y^1. (For simplicity, we assume unrealistically that all the people in this experiment fit into these eight categories.)

In this study, we define two treatment groups: $T = 0$ refers to subjects who do not receive aspirin, while $T = 1$ refers to those who do receive aspirin. Moreover, we consider the covariate x, which takes on the value 0 if the subject is male and 1 if the subject is female.

Category	#persons in category	x	T	y^0	y^1
1	300	0	0	4	6
2	300	1	0	4	6
3	500	0	1	4	6
4	500	1	1	4	6
5	200	0	0	10	12
6	200	1	0	10	12
7	200	0	1	10	12

Category	#persons in category	x	T	y^0	y^1
8	200	1	1	10	12

In making the table we are assuming omniscience, so that we know both y^0 and y^1 for all observations. But the (nonomniscient) investigator would only observe x, T, and y^T for each unit. (For example, a person in category 1 would have $x = 0$,$T = 0$,$y = 4$, and a person in category 3 would have $x = 0$,$T = 1$,$y = 6$.)

(a) What is the average treatment effect in this population of 2400 persons?

$\widehat{y}(0) = \frac{300\times4+300\times4+500\times4+200\times10+200\times10+200\times10+200\times10}{300+300+500+500+200+200+200+200} = 6$

$\widehat{y}(1) = \frac{300\times6+300\times6+500\times6+200\times12+200\times12+200\times12+200\times12}{300+300+500+500+200+200+200+200} = 8$

Thus, the average treatment effect $\widehat{y}(0) - \widehat{y}(1) = 6 - 8 = -2$

(b)Is it plausible to believe that these data came from a randomized experiment? Defend your answer.

In the event that the data was obtained through a randomized experiment, the expectation is that the observed covariates would be equally distributed between the experimental groups. In the present study, 500 *out of* 1000 *(*50*) subjects in the* $T = 0$ *group have* $x = 0$*, while* 700 *out of* 1400 *(*50*) subjects in the* $T = 1$ *group have* $x = 0$*. Hence, the data appears to have plausibly originated from a randomized experiment.*

(c) Another population quantity is the mean of Y for those who received the treatment minus the mean of Y for those who did not. What is the relation between this quantity and the average treatment effect?

The average for those that received $T = 0$ *can be given by:* $= \frac{300\times4+300\times4+200\times10+200\times10}{300+300+200+200} = 6.4$

The average for those that received $T = 1$ *can be given by:* $= \frac{500\times6+350\times6+200\times12+200\times12}{500+500+200+200} = 7.7$

The treatment effect using these averages is $7.7 - 6.4 = 1.3$

The relationship between this population quantity and the average treatment effect can be explained by the fact that this population quantity relies solely on the observed responses, whereas the average treatment effect takes into account both the observed and unobserved responses. As a result, the treatment effect obtained from this quantity may differ from that obtained from the average treatment effect.

(d)For these data, is it plausible to believe that treatment assignment is strongly ignorable given the covariate x? Defend your answer.

In case of ignorability, then $P(T|Y(T = 0) = y0, Y(T = 1) = y1, X = x) = P(T = 0|X = x)$

Let us consider couple of cases to check.

First, let us consider $Y(T=0)=10, Y(T=1)=12, X=0$, *then* $P(T|Y(T=0)=10, Y(T=1)=12, X=0) = \frac{200}{200+200} = 0.5$ $P(T=0|X=0) = \frac{300+200}{300+500+200+200} = \frac{5}{12}$

Thus, we can see that $P(T|Y(T=0)=y0, Y(T=1)=y1, X=x) \neq P(T=0|X=x)$.

Next, let us consider $Y(T=0)=4, Y(T=1)=4, X=0$, *then* $P(T|Y(T=0)=10, Y(T=1)=12, X=0) = \frac{300}{300+500} = \frac{3}{8}$ $P(T=0|X=0) = \frac{300+200}{300+500+200+200} = \frac{5}{12}$

Thus, we can see that $P(T|Y(T=0)=y0, Y(T=1)=y1, X=x) \neq P(T=0|X=x)$.

This suggests that the treatment assignment is non-ignorable, as given the covariate, it is not conditionally independent from the potential outcomes.

2.8.5 Question Causal Modeling and Assumptons

(Adapted from Hill and Gelman, Chapter 9, Question 6)

You are consulting for a researcher who has performed a randomized trial where the Treatment was a series of 26 weekly therapy sessions, the control was no therapy, and the outcome was self-report of emotional state one year later. However, most people in the treatment group did not attend every therapy session. In fact there was a good deal of variation in the number of therapy sessions actually attended. The researcher is concerned that her results represent "watered down" estimates because of this variation and suggests adding in another predictor to the model: number of therapy sessions attended. What would you advise her?

If there was a good deal of variation in the number of therapy sessions actually attended, including the number of therapy sessions as a predictor in the model could help account for this variation and help improve the precision of the estimated treatment effect.

However, there are a few potential issues to consider. First, including the number of therapy sessions attended as a predictor assumes a linear relationship between the number of sessions and the outcome, which may not be true. The relationship could be nonlinear or there could be threshold effects, where the number of sessions attended only has an effect above a certain threshold.

Second, including the number of therapy sessions as a predictor assumes that attending more therapy sessions is a good thing, which may not always be the case. If attending more sessions means that the treatment is more intense or difficult, it could actually have a negative effect on the outcome.

Finally, if the goal is to estimate the average treatment effect of receiving the therapy sessions, including the number of sessions attended as a predictor could result in biased estimates if attendance is correlated with unobserved variables that are also related to the outcome.

Therefore, before advising the researcher, it would be important to carefully consider these potential issues and assess whether including the number of therapy sessions attended as a predictor is appropriate in this particular context.

2.8.6 Question: Understanding the Causal Effect with Different Treatment Effects

Assume that linear regression is appropriate for the regression of an outcome, y, on treatment indicator, T, and a single confounding covariate, x. Sketch hypothetical data (plotting y versus x) and regression lines (for treatment and control group) that represent each of the following situations: (a) No treatment effect, (b) Constant treatment effect, (c) Treatment effect increasing with x.

(a) No treatment effect In this case, the regression lines for both the control and treatment groups will be parallel or overlapping with similar slope.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

np.random.seed(0)
x = np.random.normal(0, 1, 100)
y = 2 + 3*x + np.random.normal(0, 1, 100)
t = np.random.binomial(1, 0.5, 100)

plt.scatter(x[t==0], y[t==0], label='Control')
plt.scatter(x[t==1], y[t==1], label='Treatment')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')

# Fit regression lines
lr_t0 = LinearRegression()\
  .fit(x[t==0].reshape(-1, 1),
       y[t==0].reshape(-1, 1))

lr_t1 = LinearRegression()\
  .fit(x[t==1].reshape(-1, 1),
       y[t==1].reshape(-1, 1))

# Plot regression lines
plt.plot(x, lr_t0.predict(x.reshape(-1, 1)))
```

```
plt.plot(x, lr_t1.predict(x.reshape(-1, 1)))

plt.show()
```

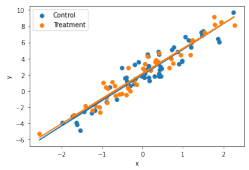


Figure 2.10: No Treatmentment Effect

(b)Constant treatment effect

In this case, the regression lines for both the control and treatment groups will be parallel to each other with different y-intercepts.

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
x = np.random.normal(0, 1, 100)
y = 2 + 3*x + 5*t + np.random.normal(0, 1, 100)
t = np.random.binomial(1, 0.5, 100)

plt.scatter(x[t==0], y[t==0], label='Control')
plt.scatter(x[t==1], y[t==1], label='Treatment')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')

# Fit regression lines
lr_t0 = LinearRegression()\
  .fit(x[t==0].reshape(-1, 1),
       y[t==0].reshape(-1, 1))

lr_t1 = LinearRegression()\
  .fit(x[t==1].reshape(-1, 1),
       y[t==1].reshape(-1, 1))

# Plot regression lines
plt.plot(x, lr_t0.predict(x.reshape(-1, 1)))
plt.plot(x, lr_t1.predict(x.reshape(-1, 1)))
```

```
plt.show()
```

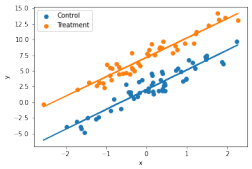


Figure 2.11: Constant Treatmentment Effect

(c) Treatment effect increasing with x:

In this case, the regression lines for the control and treatment groups will have different slopes and different y-intercepts.

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
x = np.random.normal(0, 1, 100)
y = 2 + 3*x + 5*x*t + np.random.normal(0, 1, 100)
t = np.random.binomial(1, 0.5, 100)

plt.scatter(x[t==0], y[t==0], label='Control')
plt.scatter(x[t==1], y[t==1], label='Treatment')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')

# Fit regression lines
lr_t0 = LinearRegression()\
  .fit(x[t==0].reshape(-1, 1),
       y[t==0].reshape(-1, 1))

lr_t1 = LinearRegression()\
  .fit(x[t==1].reshape(-1, 1),
       y[t==1].reshape(-1, 1))

# Plot regression lines
plt.plot(x, lr_t0.predict(x.reshape(-1, 1)))
plt.plot(x, lr_t1.predict(x.reshape(-1, 1)))
```

```
plt.show()
```

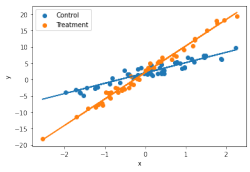


Figure 2.12: Increasing Treatmentment Effect

3 Causal Inference: A Practical Approach

Having laid the foundation with the potential outcome framework and fundamental causal concepts, we now delve into the world of causal modeling. In this chapter, we explore the power of causal graphs as a comprehensive approach for inferring causal relationships. Firstly, we introduce causal graphs, explaining the basics of graph modeling in the context of causal inference. Next, we offer an overview of the high-level process involved in causal inference using causal graphs. Subsequently, we discuss each stage of the causal inference process, providing a more detailed examination of the methods and techniques commonly employed. Finally, we present a comprehensive case study utilizing the Lalonde dataset. This study compares and contrasts the various techniques for causal inference discussed, accompanied by a thorough analysis.

3.1 Causal Inference: Logical Flow

In the last chapter, we introduced the causality flowchart for estimating the causal effect, as shown in Figure 3.1.

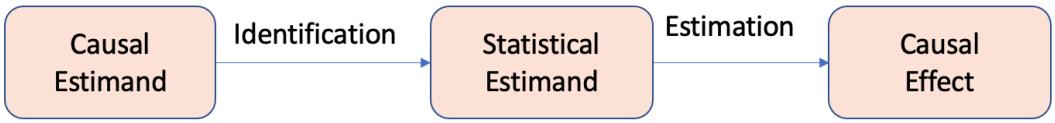


Figure 3.1: Causality flowchart: From a target causal estimand to an estimate

The two pertinent questions will be answered in this chapter, as shown in Figure 3.2.

1. How do we perform identification and convert causal estimands into statistical estimands? What are the tools available for this process?

   The identification and conversion of causal estimands to statistical estimands are achieved through causal modeling.

2. Once we have obtained statistical estimands, how do we proceed with estimation? What tools can be utilized for this purpose?

Estimation is carried out by leveraging the statistical estimands derived from the causal modeling process and utilizing observational data in conjunction with appropriate statistical techniques.

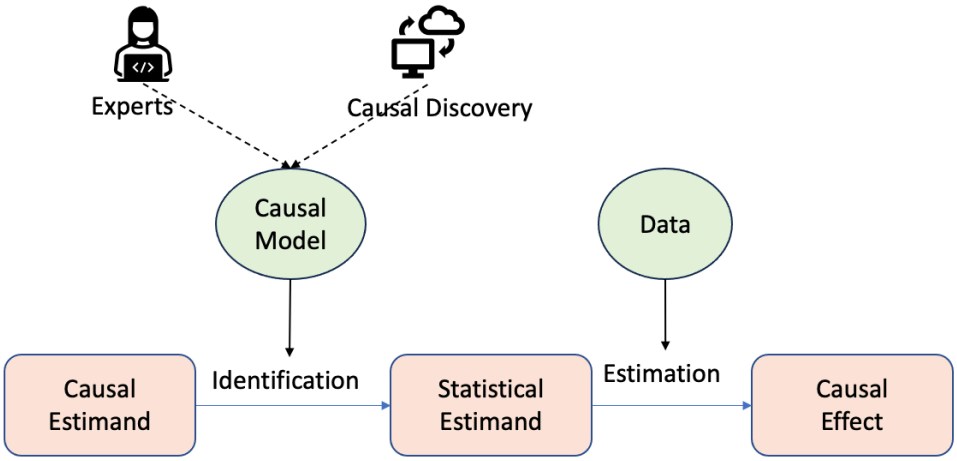


Figure 3.2: Causality Flowchart: How to go from a target causal estimand to an estimate using causal modeling

> **Tip**
>
> Causal models can be constructed either by experts with domain knowledge or through automated procedures known as causal discovery (discussed in Chapter 4). Building causal models requires domain knowledge, identifying relevant variables, and determining the relationships between them.

3.2 Causal Inference: Practical Flow

In the previous section, we examined the logical flowchart that illustrates transitioning from a target causal estimand to an estimate. Now, we will delve into the practical aspect of this process, providing a high-level overview of the steps involved in practice.

This section aims to bridge the gap between theory and application, offering insights into how the causal inference process unfolds in real-world scenarios.

Notably, unlike the machine learning process, the causal inference process involves distinct steps, as Dow et al. underscored in their work (Sharma and Kiciman 2020) and given by Figure 3.3.

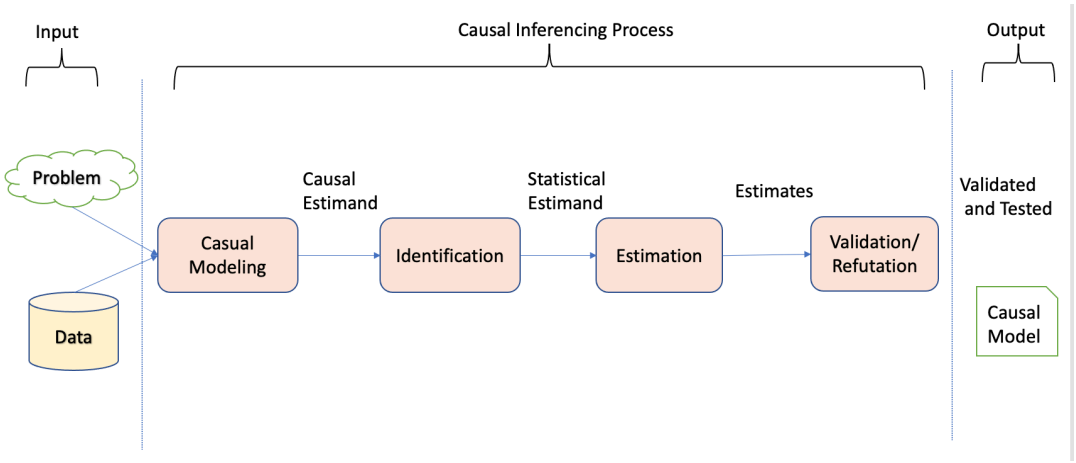


Figure 3.3: Causal inference process in practice

1. To initiate the causal modeling process, one of the methodologies employed is constructing a causal graph with appropriate structural assumptions. Causal graphs depict the causal structure by utilizing nodes to represent variables and edges to represent the causal relationships between them.
2. After establishing the causal assumptions within a causal model, the subsequent stage of causal analysis is identification. During this stage, the objective is to examine the causal model, encompassing the relationships between variables and the observed variables, to ascertain if there is sufficient information to address a particular causal inference question. Different techniques are utilized with the aim of converting the causal estimands into statistical estimands.
3. After confirming the estimability of the causal effect and transforming the causal estimands into statistical estimands, the third step revolves around estimating the effect using suitable statistical estimators. Different estimation techniques, such as regression models or propensity score matching, can be utilized to estimate the causal effect.
4. Finally, the fourth step involves validating and assessing the robustness of the obtained estimate through rigorous checks and sensitivity analyses. This step includes examining the sensitivity of the estimated effect to different model specifications, testing the robustness of the results against potential sources of bias or unobserved confounding, and assessing the generalizability of the findings.

3.3 Causal Modeling

The initial phase of any causal inference effort entails constructing causal models, graphical models in our case, which encode domain understanding and assumptions. A well-designed

causal model should capture most of the relationships between the outcome and variables and inter-relationships between the variables.

Causal graphs are a common way of representing the relationships between variables when modeling the joint distribution.

> **Tip**
>
> A cause in a directed acyclic graph (DAG), similar to Bayesian networks, is defined as if any changes are made to a node, a response, or corresponding modifications are observed in the connected node(s). In the context of graphical models, a causal graph is a type of Bayesian network in which each node's direct causes are represented by its parents.

For a comprehensive and in-depth exploration of causal graphs, we highly recommend referring to Judea Pearl's work (Judea Pearl 2009, 2000).

3.3.1 Assumptions in Causal Modeling

Figure 3.4 shows the entire flow of assumptions that help in causal modeling.

- **Local Markov Assumption**: Given the parents in the DAG, a node is independent of all its non-descendants.
- **Minimality assumption**: The adjacent nodes in the DAG are dependent and have to be considered in the factorization in addition to the local Markov assumption. This assumption removes independent assumptions when the edges are present in the DAG.
- **Causal Edge assumption**: In causal relationships, every parent is the direct cause of their children.

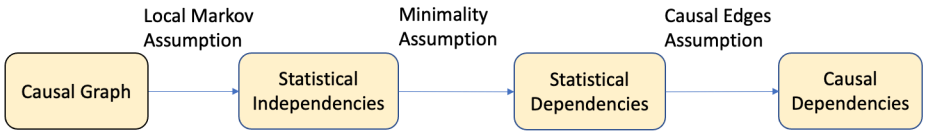


Figure 3.4: Causal Graph Assumptions

Thus for a given distribution represented by DAGs, the local Markov assumption helps to measure statistical independence, the minimality assumption helps to focus on statistical dependencies (at least between the adjacent nodes), and finally, layering the causal edge assumption gives us the causal dependencies.

Causal graphs possess an essential characteristic of being able to identify confounding variables. These variables are associated with the cause and the effect but are not causally connected. For instance, in the case of *smoking* and *lung cancer*, *age* may act as a confounding variable

related to both *smoking* and *lung cancer* but does not lie on the causal pathway between them. By including *age* as a node in the causal graph, researchers can control for its effects and isolate the causal relationship between *smoking* and *lung cancer*.

Causal graphs can identify the minimal set of variables, which are imperative in estimating the causal effect of one variable on another. This technique is known as the *identification* (discussed later), based on the notion that several paths may exist between two variables in a causal graph. However, only some are crucial in estimating the causal effect. By identifying the minimal set of variables needed, researchers can effectively and accurately estimate the causal effect with less bias and more efficiency.

3.3.2 Building Blocks in Causal Graphs

In graph theory, the term "flow of association" refers to the presence or absence of association between any two nodes in a given graph. This concept can also be expressed as the statistical dependence or independence between two nodes. In the following section, we shall delve into fundamental constituents such as the building block and terminologies essential to graphical representations of various causal associations between variables. Furthermore, we shall conduct preliminary analyses to investigate the variables' conditional independence or dependence within the building blocks.

3.3.2.1 Chains

A chain is a sequence of nodes such that each node is a parent of the next node in the sequence as shown in Figure 3.5.

There is dependence between X_1 and X_2 and also between X_2 and X_3 because of the causal edges assumption. X_1 and X_3 are dependent as there is a flow of association or statistical dependence from X_1 to X_3 through X_2. Similarly, if we condition on the X_2, X_1 and X_3 are independent of each other. In other words, X_2 will block the association flow between X_1 and X_3 by conditioning.

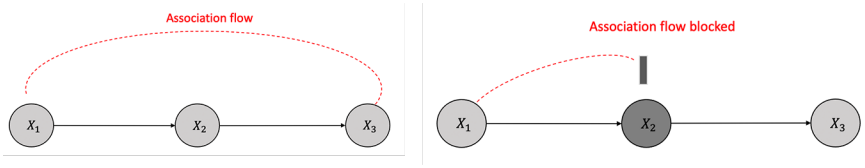


Figure 3.5: Association flow in chains without conditioning and with conditioning

3.3.2.2 Forks

In a fork, two variables have a single parent between them, as shown in Figure 3.6. Similar to chains, there is dependence between X_1 and X_2 and between X_2 and X_3 because of the causal edges assumption. X_1 and X_3 are dependent as there is a flow of association or statistical dependence from X_1 to X_3 through X_2. Also, if we condition on the X_2, X_1 and X_3 are independent. In other words, X_2 will block the association flow between X_1 and X_3 by conditioning.

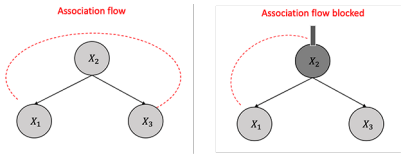


Figure 3.6: Association flow in forks without conditioning and with conditioning

3.3.2.3 Immoralities

Immoralities refer to a configuration in a directed acyclic graph where a single node relies on two parents who are not directly connected, as depicted in Figure 3.7. The node X_2 in this structure is called a *collider* and obstructs the flow of association between nodes X_1 and X_3 without conditioning, unlike in chains or forks. However, unlike chains or forks, by conditioning on the collider X_2, the flow of association or dependence persists between X_1 and X_3.

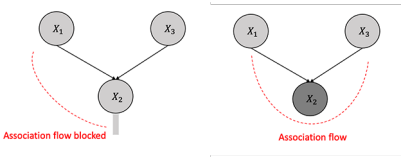


Figure 3.7: Association flow in immoralities without conditioning and with conditioning

3.3.2.4 Blocked Path

The concept of a blocked path is intimately tied to the flow of causal influence. A path between two nodes, X and Y, can be blocked or unblocked by a conditioning set (Z). The two scenarios that one comes across are:

1. If there exists a node W on the path from X to Y such that it is part of a chain structure ($X \rightarrow W \rightarrow Y$) or a fork structure ($X \leftarrow W \rightarrow Y$), and W is conditioned on ($W \in Z$).
2. There is a collider W on the path that is not conditioned on ($W \notin Z$), and none of its descendants are conditioned on, i.e. ($de(W) \notin Z$).

3.3.2.5 d-Separation

The concept of d-separation is tied to the concept of a blocked path in the graph. A path from a node(set) X to a node(set) Y is considered blocked given a set Z, if nodes block any node in X and Y in Z.

The concept of d-separation implies an important theorem that X and Y are conditionally independent given Z. On the contrary, if there exists at least one unblocked path from X to Y given Z, then X and Y are not d-separated by Z, suggesting that X and Y are not conditionally independent given Z. Mathematically,

$$X \perp_G Y|Z \implies X \perp_P Y|Z$$

3.3.3 Causal Graphs and Structural Interventions

In the context of Causal graphs, a **structural intervention** is represented as an exogenous variable I, a variable without any causes. It has two possible states (on/off), with a single arrow pointing to the variable it manipulates. When I is set to the off state, the passive observational distribution is obtained over the variables. In contrast, when I is set to the on state, all other arrows incident on the intervened variable are removed. The probability distribution over the intervened variable is determined solely by the intervention.

For example, consider a causal graph representing the relationship between smoking, age, and cancer, as shown in Figure 3.8. Suppose that *smoking* and *age* are the direct causes of *cancer*. A structural intervention can be performed on smoking, where I is introduced as an exogenous variable with two states: on and off. When I is off, the system behaves according to the passive observational distribution, where the effect of _age_confounds the effect of smoking on *cancer*. However, when I is on, all other arrows pointing to *cancer* from variables other than *smoking* are removed, and the probability distribution of *cancer* is a deterministic function of *smoking* only, allowing for the identification of the causal effect of *smoking* on *cancer*.

This "structural" property is critical to understanding the effect of interventions on causal graphs. Suppose there are multiple simultaneous structural interventions on variables in the graph. In that case, the manipulated distribution for each intervened variable is independent of every other manipulated distribution, and the edge-breaking process is applied separately to each variable. This process implies that all edges between variables subject to intervention are removed. After removing all edges from the original graph incident to variables that are the target of a structural intervention, the resulting graph is called the post-manipulation graph, which represents the manipulated distribution over the variables.

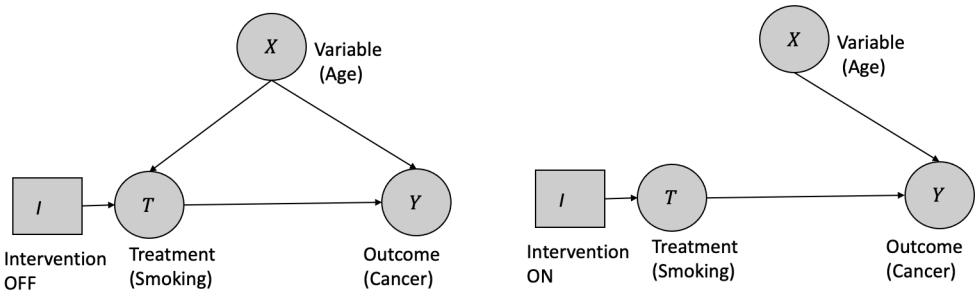


Figure 3.8: Structural Interventions

3.3.4 Observational Data and Interventional Data

Next, let us discuss the terminologies of observational and interventional data, which play a vital role in understanding the causal process in greater detail.

Observational data is collected simply by passively observing a system or population without any intervention or change in the process. In contrast, interventional data is collected by actively manipulating the system or population in some way, like in randomized control trials. Observational studies may be subject to various types of biases, such as confounding, selection bias, and measurement bias, making it difficult to distinguish between causal and non-causal relationships in the dataset.

Interventional data, on the other hand, is often considered to be the gold standard for establishing causal relationships between variables. This is because interventions allow actively manipulating the independent variable and observing the resulting changes in the dependent variable. Researchers can minimize the effects of confounding and other biases by randomly assigning participants to different treatment groups, like in randomized control trials.

The acquisition of observational data is generally less resource-intensive than interventional data, which can be expensive and impractical to obtain in specific scenarios. This raises the question of whether it is possible to derive interventional data from observational data.

3.3.5 The *do*-operator and Interventions

> **Tip**
>
> The *do*-operator and identification process are essential tools that facilitate us going from observational to interventional data. The *do*-operator helps distinguish interventional distributions from observational distributions, while identification helps determine which causal relationships can be inferred from the observed data. We will discuss the process in detail in the following section.

The *do*-operator is a symbolic notation used in causal inference to represent interventions. The *do*-operator is a notation to represent the population's intervention distribution. Given a treatment $(T = t)$, **intervention** corresponds to the **whole population** subjected to that treatment and given by $do(T = t)$. This is different from the **conditional** distribution $(P(Y|T = t))$, which represents a **subset** of the population $(T = t)$ rather than the whole population in the case of intervention as shown in the Figure 3.9. The interventional measures can be computed only through experiments, while the conditional measures can be computed directly from the observational data.

Also, the potential outcome $(P(Y(t) = y))$ in terms of interventional distribution using the *do*-operator is given by:

$$P(Y(t) = y) \triangleq P(Y = y|do(T = t)) \triangleq P(y|do(t))$$

Similarly, the average treatment effect (ATE) in the case of binary treatment using the *do*-operator is given by:

$$ATE = \mathbb{E}[Y|do(T = 1)] - \mathbb{E}[Y|do(T = 0)]$$

3.3.6 Modularity assumptions

A modular assumption is about the local impact of any intervention when applied to a causal graph. Modularity assumption is also known as invariance, autonomy, and independent mechanisms.

It states that if a node X_i is being intervened, then only the mechanism $P(x_i|pa_i)$ changes and the intervention has no impact on any other mechanism changes, i.e., $P(x_j|pa_j)$ where $i \neq j$ remain unchanged. Modularity assumptions is graphically demonstrated in Figure 3.10. The violation of the modularity assumption implies that when a node is intervened, mechanisms of other non-parent nodes changes, and thus any local effect assumption goes away for computation purposes.

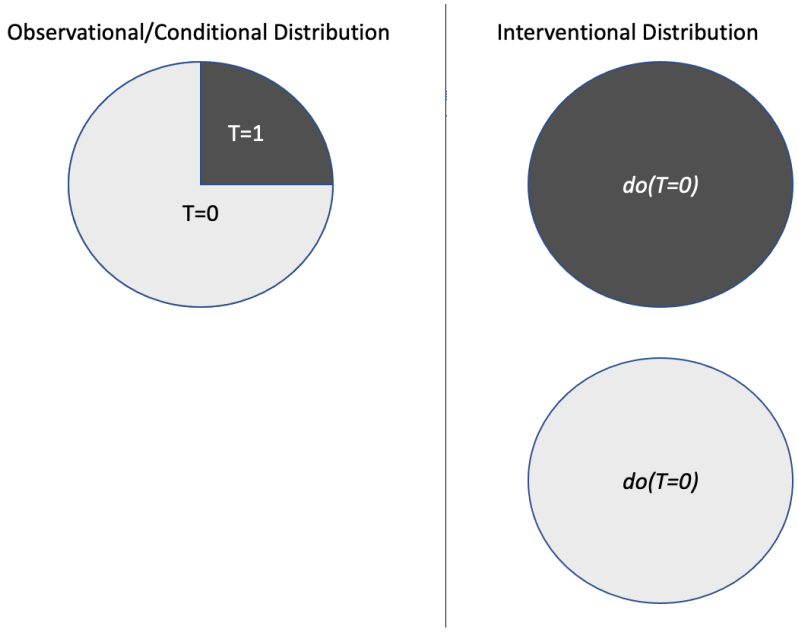


Figure 3.9: Conditional vs Interventiona

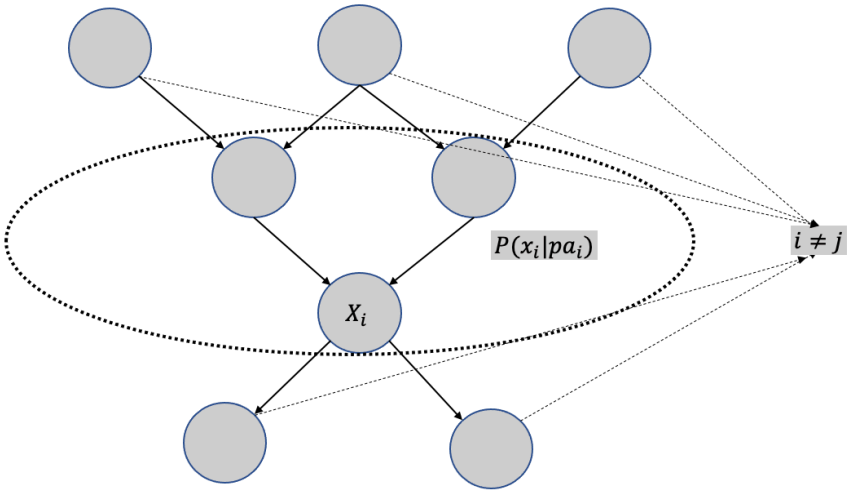


Figure 3.10: Modularity and Interventions

3.3.7 Modularity Assumptions and Truncated Factorization

The network factorization for Figure 3.10 can be written as:

$$P(x_1, \cdots, x_n) = \prod_i P(x_i|pa_i)$$

When we intervene on variable set S, the factorization becomes:

$$P(x_1, \cdots, x_n|do(S = s)) = \prod_{i \notin XS} P(x_i|pa_i)$$

if the x is consistent with the intervention, otherwise

$$P(x_1, \cdots, x_n|do(S = s)) = 0$$

The truncated factorization can be employed to estimate the causal effect of treatment T on outcome Y in a simple graph comprising three variables X, Y, and T, with X as the confounder as shown in Figure 3.11. The aim is to estimate the causal quantity $P(y|do(T))$.

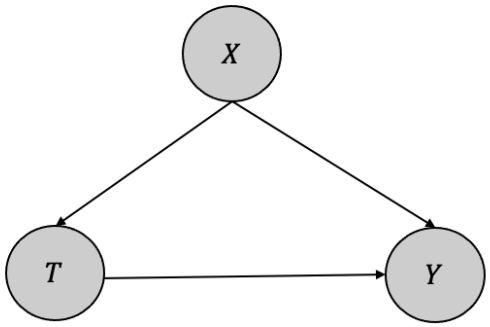


Figure 3.11: Modularity and Interventions

The network factorization gives us the following:

$$P(y, t, x) = P(x)P(t|x)P(y|t, x)$$

When we intervene on treatment T, we can remove t factor, i.e. $P(t|x)$, thus the truncated factorization gives us:

$$P(y, x|do(T)) = P(x)P(y|t, x)$$

Next, if we marginalize x using summation (discrete) or integration (continuous), we get:

$$P(y|do(T)) = \sum_x P(y|t, x)P(x)$$

Thus, we got identifiability or could go from causal estimand $P(y|do(T))$ to statistical estimand $\sum_x P(y|t,x)P(x)$ from observational data.

Now, in a separate statistical world we can rewrite $P(y|t)$ using marginalization as:

$$P(y|t) = \sum_x P(y, x|t)$$

Using the the factorization property:

$$P(y|t) = \sum_x P(y|t)P(x|t)$$

Thus, comparing the two equations, we can conclude that

$$P(y|do(t)) \neq P(y|t)$$

as $P(y|do(t))$ has $P(x)$ where as $P(y|t)$ has $P(x|t)$.

3.3.8 Structural Causal Models (SCM)

Structural Causal Models (SCMs) are a critical component of modeling for causal inference. SCMs are mathematical models representing the causal relationships between variables in a system (Spirtes et al. 2000).

Structural Causal Models (SCMs) consist of two fundamental constituents: endogenous variables' structural equations that portray causal relationships among variables and exogenous variables that represent system variables unaffected by other variables. Causal graph structures depict the structural equations as functions among the variables, visually showcasing the relationships between exogenous and endogenous variables.

In simplest form, if variable X causes Y, then it can be written as a structural equation:

$$Y := f(X)$$

Stochasticity can be added to the structural equation in the form of noise variables (U), and the above equation can be rewritten as:

$$Y := f(X, U)$$

When there are many cause-effect relationships, as shown in Figure 3.12, the representation using structural causal models equation with exogenous, endogenous, and noise variables is given by:

$$X = f_X(W, U_X)$$
$$Z = f_Z(X, Y, U_Z)$$

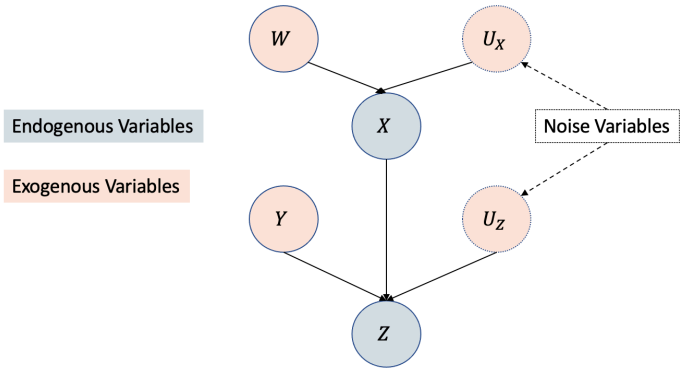


Figure 3.12: Structural Causal Models

3.3.8.1 Interventions and Modularity Assumptions in SCM

For the basic causal model as shown in Figure 3.13, with single confounder X affecting both the treatment T and the outcome Y, the SCM equations can be written as:

$$T = f_T(X, U_T)$$

$$Y = f_Y(X, T, U_Y)$$

The interventional SCM or a submodel follows the same as above but replaces the function $f_T(X, U_T)$ with a variable t as:

$$T = t$$

$$Y = f_Y(X, T, U_Y)$$

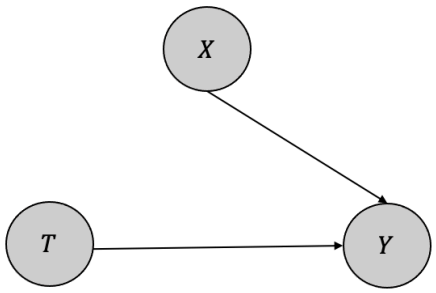


Figure 3.13: Intervention and removing the edge corresponding to $do(T = t)$

3.4 Identification

> **Tip**
>
> Identification is the process of converting causal estimands to statistical estimands. i.e., to go from $P(Y|do(t))$ to $P(Y|t)$. If we can go from $P(Y|do(t))$ to $P(Y|t)$, we have **identifiability**.

Let us consider a simple identification process with one treatment (T), one confounding variable (X), and one outcome (Y) as given in Figure 3.14.

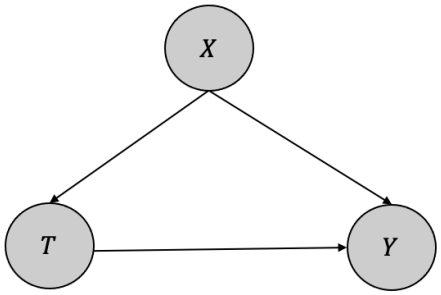


Figure 3.14: Identification

Identification is to find $P(y|do(T))$, which can be iteratively computed from the joint distribution as below:

$$P(y,t,x) = P(x)P(t|x)P(y|t,x)$$

Applying the modularity assumption by intervening on the variable (T),we get $P(t|x) = 1$ and thus the equation transforms:

$$P(y,x|do(T)) = P(x)P(y|t,x)$$

If we marginalize the variable (X), we get:

$$P(y|do(T)) = \sum_x P(y|t,x)P(x)$$

Thus, from the causal estimand $P(y|do(T))$, and identification, we have an equation with only observational or statistical quantities $\sum_x P(y|t,x)P(x)$. This is called the **adjustment formula**.

Tip

As highlighted in the overall process illustrated in Figure 3.15, the identification process involves a transformation from causal estimand $P(y|do(T))$ to statistical estimand ($\mathbb{E}_X P(y|t,X)$ or $P(y|t)$) contingent upon the existence or absence of confounding variables.

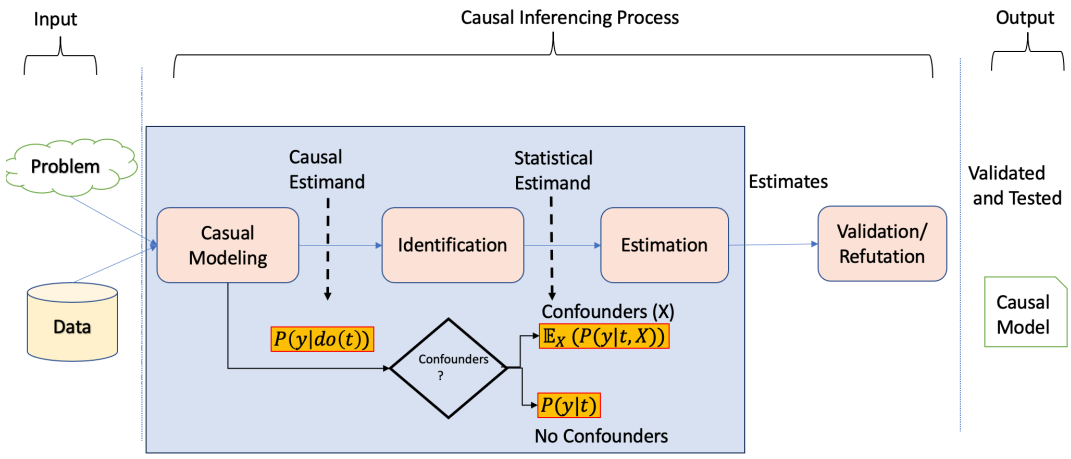


Figure 3.15: Identification in presence or absence of confouding variable

Identification methods can be classified into two broad categories with subcategories, as shown below.

- Graphical Constraint-based Methods
  - Randomized Control Tests
  - Backdoor Adjustments
  - Frontdoor Adjustments
- Non-Graphical Constraint-based Methods
  - Instrumental Variables
  - Regression Discontinuity
  - Difference-in-Differences
  - Pearl's do-calculus

Next, we will go over these different methods of identification.

3.4.1 Randomized Control Trials (RCT)

As elucidated in the second chapter, comprehending the impact of unseen confounding factors when measuring treatment effects on an outcome is challenging. One way to address this issue is through randomized controlled trials, which introduce randomness in the treatment allocation process and ensure that the resulting groups (binary $T = 0$ and $T = 1$) are comparable, thus mitigating the confounding impact. In other words, randomization ensures exchangeability, i.e., the treatment groups are entirely interchangeable because of the randomization. Thus the average outcome of the two groups $\mathbb{E}[Y|(T = 1)] = y_1]$ and $\mathbb{E}[Y|(T = 0)] = y_0]$ remain the same. From a causal graph perspective, the randomized treatment removes the edge between the confounder X and the treatment T, removing the backdoor path and the confounding association.

3.4.2 Backdoor Criterion and Backdoor Adjustment

The interventional causal graph can have various paths from the treatment (T) to the outcome (Y). Some of the paths are non-causal, and some of them are causal. One method to perform identification is to block backdoor paths (Judea Pearl 2010). Backdoor paths are the edges that flow into the intervening treatment.

As shown in Figure 3.16, the left side graph is observational $P(Y|t)$ with various causal and non-causal paths between the treatment (T) and the outcome (Y). There are many non-causal paths from T to Y, and they are $T \rightarrow W \rightarrow U \rightarrow V \rightarrow Y$, $T \rightarrow X \rightarrow Y$ and $T \rightarrow P \rightarrow Y$. Only backdoor paths are $T \rightarrow W \rightarrow U \rightarrow V \rightarrow Y$ and$T \rightarrow X \rightarrow Y$ as the edges flow into the intervening treatment T. The non-causal path $T \rightarrow P \rightarrow Y$ is not a backdoor path. The path $T \rightarrow X \rightarrow Y$ is the only causal path. The equivalent interventional causal graph $P(Y|do(t))$ on the right side is with the backdoor paths blocked by removing the edges.

Can we get an equivalent interventional causal graph from the observational graph? The answer is yes, and it can be done by conditioning the nodes/variables in the backdoor paths. By conditioning on W and X, we get $P(Y|t, w, x)$ and that is equivalent to removing the edges and blocking the paths as shown in Figure 3.17

Formally, a set of variables Z are set to satisfy **backdoor criterion** with respect to treatment T and outcome Y if it satisfies the following:

- If the variable set Z blocks all the backdoor paths from T to Y
- Z does not contain any descendant of the treatment T

Thus, based on the modularity assumption and the backdoor criterion, one can identify the causal effect by:

$$P(y|do(t)) = \sum_{z} P(y|t, z)P(z)$$

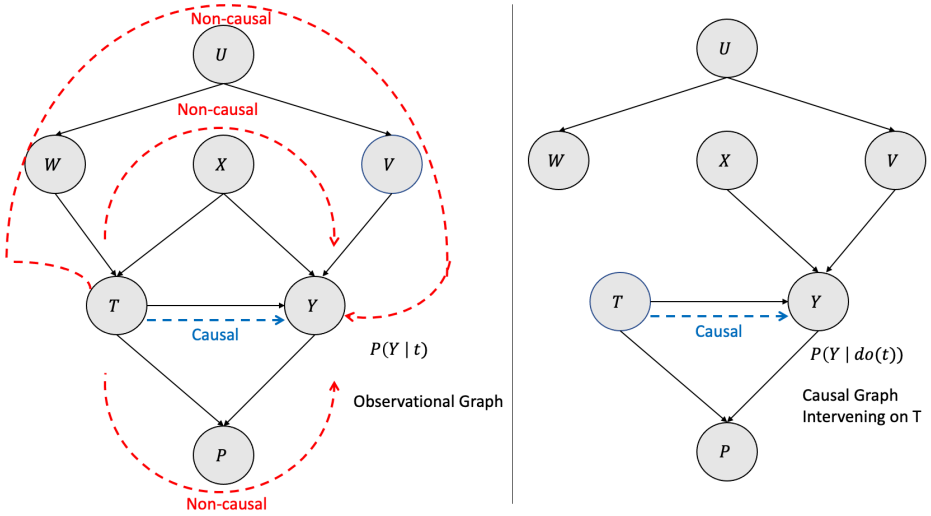


Figure 3.16: Observational Vs Causal

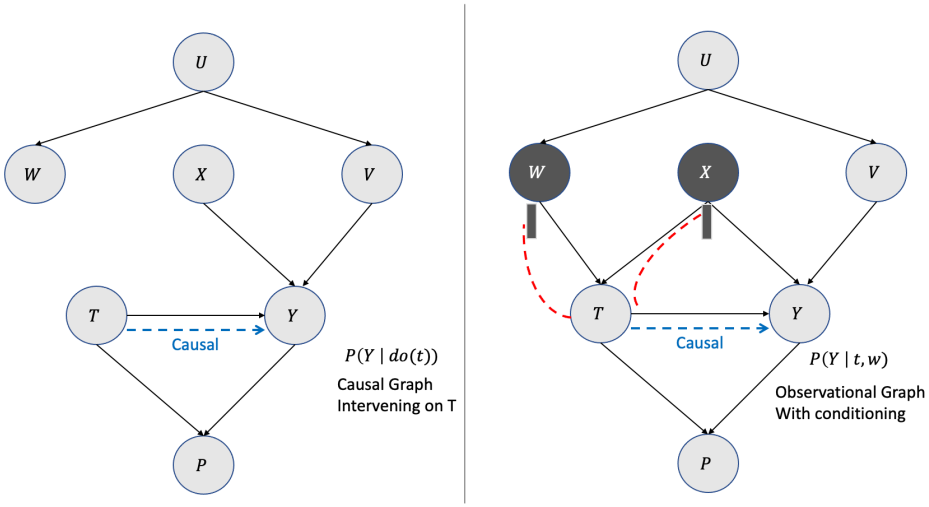


Figure 3.17: Causal vs Conditoning

3.4.3 Front-door Adjustments

Judea Pearl justifies the use of the front-door adjustment method through the illustration of an example in which the effect of smoking (treatment) on cancer (outcome) is studied while taking into account the influence of tar (observed mediator) and an unknown genotype (unobserved confounder), as depicted in Figure 3.18. In such Directed Acyclic Graphs (DAGs), the backdoor criterion is inapplicable due to an unobserved confounder.

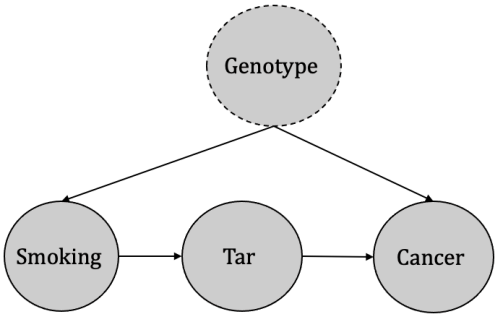


Figure 3.18: Cancer and Smoking relationship

The more generic DAG is shown in Figure 3.19 The intuition behind the front-door adjustment can be broken into the following three steps as follows:

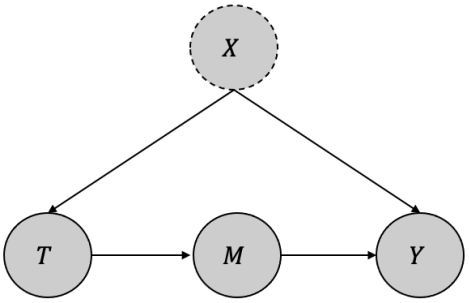


Figure 3.19: Generalized Front-door

- Identify the causal impact of treatment (T) on the mediator (M) $P(m|do(t)$ Since there are no backdoor paths, we can write: $P(m|do(t) = P(m|t)$
- Identify the causal impact of mediator (M) on the outcome (Y) $P(y|do(m)$ Since there is a backdoor path from M to Y through T, we can use the backdoor criterion by

conditioning on T.

$$P(y|do(m) = \sum_t P(y|m,t)P(t)$$

- Combined the two previous steps to identify the causal impact of treatment (T) on the outcome (Y)

$$P(y|do(t) = \sum_m P(m|do(t))P(y|do(m))$$

$$P(y|do(t) = \sum_m P(m|t) \sum_{t'} P(y|m,t')P(t')$$

The above equation is called the frontdoor adjustment. The set of variables M satisfy the frontdoor criterion if:

1. Variable M mediates the effect of T on Y.
2. There is no unlocked backdoor path from T to M.
3. All backdoor paths from M to Y are blocked by T.

3.4.4 Instrumental Variable Analysis

In situations where specific variables affect the treatment variable(s) but do not directly influence the outcome variable, identification can be achieved through instrumental variable analysis. For instance, consider the example of three variables, namely smoking, cigarette prices, and cancer, as shown in Figure 3.20. It is apparent that cigarette prices affect whether an individual smokes but do not directly impact the likelihood of developing cancer. Such variables that affect the treatment variable(s) but not the outcome variable are referred to as instrumental variables, as described by Pearl (Judea Pearl 2010).

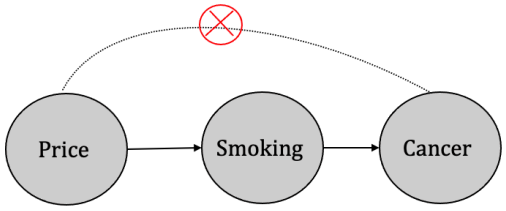


Figure 3.20: Example of Instrumental Variables

The role of instrumental variables in the identification process is to help address the problem of endogeneity, which occurs when a variable of interest is correlated with the error term in a regression model. This correlation leads to biased and inconsistent estimates of the treatment effect, making it difficult to establish a causal relationship between the treatment and the

outcome variable. By using an instrumental variable that is uncorrelated with the error term and affects the treatment but not the outcome variable, the IV analysis can isolate the causal effect of the treatment on the outcome variable.

Thus, when generalized, as shown in Figure 3.21, the identification process is a two-stage process. The first step is to measure the effect of the instrument variable on the treatment using regression as given by:

$$\hat{T} = \beta_0 + \beta_1 Z$$

and then use the predictor $\hat{T}$ to measure the effect on the outcome Y as another regression, given by:

$$\hat{Y} = \beta_2 + \beta_3 \hat{T}$$

In addition to the conditions described above being met, instrumental variable analysis is applicable when there are only moderate to small confounding effects and a good sample size of observational data.

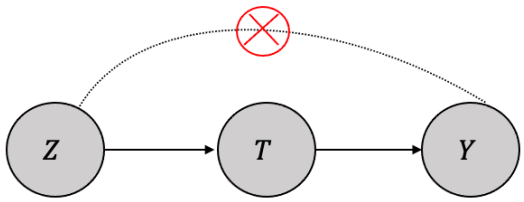


Figure 3.21: Generalized relationship between Instrumental Variables, Treatment, and Outcome

3.4.5 Regression Discontinuity

The regression discontinuity approach is a regression-based technique that is well-suited for identifying real-valued outcomes, particularly in scenarios where the outcome data contain thresholds or cut-offs. This method is commonly applied in cases where treatment is provided when the outcome surpasses a certain threshold but not when it falls below it (Imbens and Lemieux 2008). The treatment/intervention impact above/below the threshold can be used for causality estimation. Examples include receiving scholarships and their implications on admissions/SAT scores or receiving a specific medicine dosage for patients above a certain cut-off of diabetes or cholesterol etc. Figure 3.22 shows an example of student GPA as the outcome (Y) on the y-axis, and student test scores normalized as a variable (X) on the x-axis with a threshold deciding scholarship (T) and its impact as a shifted GPA on the y-axis. The shift is the regression discontinuity. Regression discontinuity can be measured using the same trend on either side of the discontinuity using the regression formula:

$$\hat{y} = \beta_0 + \beta_1 X + \beta_2 I(x > x_0)$$

β_2 is the measure fo regression discontinuity.

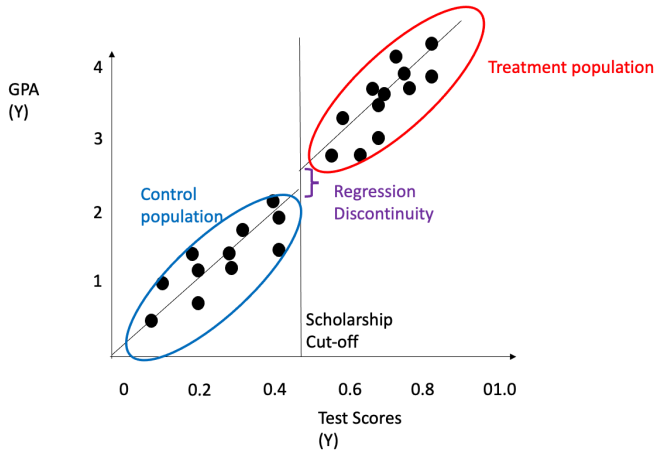


Figure 3.22: Regression Discontinuity

3.4.6 Difference-in-Differences

The Difference-in-Differences (DID) approach is a regression-based method that effectively identifies real-valued outcomes when measured over time, as highlighted in (Lechner et al. 2011). Specifically, the DID approach allows for estimating the treatment effect by comparing the differences in outcomes over time between the treatment and control groups. This method is often applied at a particular time and enables estimating the treatment effect using regression analysis, wherein the significant differences between the treatment and control groups can be effectively captured.

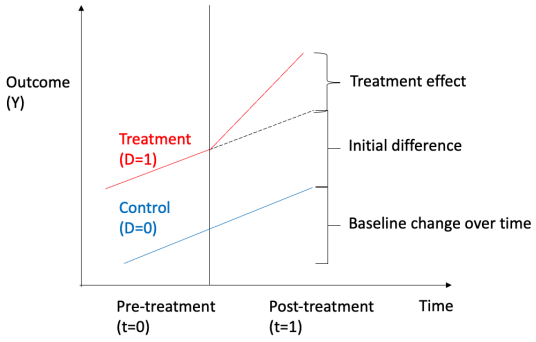


Figure 3.23: Difference-in-Differences

To make things concrete, let us consider a simple use case of a binary treatment ($T = 0, T = 1$), with a real-valued outcome ($y \in \mathbb{R}$), and the outcome is measured w.r.t to time as shown in

Figure 3.23 Regression analysis can be done using a couple of variables: D for treatment and T for time as given by:

$$y = \beta_0 + \beta_1 D + \beta_2 T + \beta_3 D \times T + \mu$$

The regression variables can be interpreted as β_1 estimates the marginal effect of treatment, β_2 estimates the baseline change over time for the control group, and β_3 the treatment effect. There are several assumptions made from regression analysis and causal effect perspective, such as both the groups having common trends and non-independence of observations.

3.4.7 Pearl's do-calculus

If a query Q is provided as a do-expression, such as $Q = P(y|do(x), z)$, its identifiability can be systematically determined using Pearl's do-calculus. In scenarios where both backdoor and front-door adjustment approaches fail to enable identification, Pearl's do-calculus provides an effective means of identifying any causal quantity that can be identified (Judea Pearl 2012).

Consider a causal Directed Acyclic Graph (DAG) G, where X, Y, Z, and W are arbitrary disjoint sets of nodes. The graph $G_{\overline{X}}$ is obtained by removing all arrows that point to nodes in X from G. Similarly, the graph $G_{\underline{X}}$ is obtained by deleting all arrows from nodes in X from G. To indicate the removal of both incoming and outgoing arrows, we employ the notation $G_{\overline{X}\underline{Z}}$.

The following three rules apply to all interventional distributions that align with the structure of G.

1. **Rule 1 (Insertion/deletion of observations)**:

Per Rule 1, any observational node that fails to influence the outcome through a given path or is d-separated from the outcome can be safely disregarded.

The following is the formal definition:

$$P(y|do(x), z, w) = P(y|do(x), w) \; if \; (Y \perp Z|X, W)_{G_{\overline{X}}}$$

If the nodes Y and Z are d-separated from one another, once we factor in both W and X, we can eliminate Z and remove it from the $P(y|z, w)$.

2. **Rule 2 (Action/observation exchange)**:

In the context of a randomized controlled trial, researchers can assign treatment and perform either $do(x)$ or not $do(x)$. However, with observational data, it is not feasible to directly perform $do(x)$. It would be a significant advantage if we could treat an intervention like $do(x)$ as regular non-interventional observational data. Rule 2 facilitates this transformation.

Per Rule 2, interventions, represented by $do(x)$, can be handled as observations when the causal effect of a variable on the outcome, specifically $X \to Y$, influences the outcome solely through directed paths. Formally this can be written as:

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \; if \; (Y \perp Z|X, W)_{G_{\overline{X}\underline{Z}}}$$

One can note that the left-hand side involves the interventional operator $do(z)$, while the right-hand side employs the observed variable z. As long as the condition $(Y \perp Z \mid W)_{G\underline{Z}}$ holds, we can convert $do(z)$ to z and solely rely on observational data.

3. **Rule 3 (Insertion/deletion of actions)**___: Rule 3 states that if an intervention (or a $do(\cdot)$ expression) does not influence the outcome through any uncontrolled path, it can be disregarded. Specifically, we can eliminate $do(z)$ if no causal association (or unblocked causal paths) runs from Z to Y.

$$P(y|do(x), do(z), w) = P(y|do(x), w) \; if \; (Y \perp Z|X, W)_{G_{\overline{XZ(W)}}}$$

Both front-door and backdoor adjustment formulae can be derived using solely the do-calculus. It has been established that the do-calculus is complete, i.e., it can identify all the causal estimands if they exist Shpitser and Pearl (2006). This theorem implies that if the repeated application of these three rules cannot eliminate the do-operations, the query Q cannot be identified.

3.5 Estimation

This section will discuss various methods to compute estimation from statistical estimands.

There are two broad types of estimation methods:

Covariate Adjustment Methods:

Covariate adjustment techniques involve utilizing the covariates or features (X) and the treatment (T) as inputs to one or more machine learning models, which are then used to fit the inputs to the potential outcome output ($Y), thereby capturing the relationship between them. The appropriate model selection, such as linear or non-linear, is contingent on the nature of the relationship between these variables. There are numerous covariate adjustment techniques commonly utilized in practice, which we will describe in detail, including:

1. COM Estimator
2. GCOM Estimator
3. X-Learner
4. TarNET

5. Matching
6. Doubly Robust Learners

Propensity Score Methods:

In these methods, a propensity score is defined as the conditional probability of treatment assignment given a set of observed covariates X. The propensity score adjusts for differences in observed covariates between treated and control groups, creating a pseudo-population in which the covariate distribution is balanced between the two groups.

Some of the techniques are: 1. Propensity Score Matching 2. Propensity Score Stratification 3. Inverse Propensity Score Weighting

3.5.0.1 Conditional Outcome Modeling Estimator (COM Estimator or S-Learner)

As discussed in Chapter 2, the individualized treatment effect (ITE) is fundamentally unknowable; hence, large randomized experiments allow us to measure the average treatment effect (ATE). The individualized treatment effect τ_i for a binary treatment is given by:

$$\tau_i \triangleq Y_i(1) - Y_i(0)$$

The average treatment effect τ is given by:

$$\tau = \mathbb{E}[Y_i(1) - Y_i(0)]$$

When there are other covariates x present (observed and/or unobserved), we estimate a more specific effect using those covariates, and it is called the conditional average effect (CATE)

$$\tau(x) \triangleq \mathbb{E}[Y(1) - Y(0)|X = x]$$

From the identification discussion, given a sufficient adjustment set W and the covariates X, we assume that $W \cup X$ is also a sufficient adjustment set and satisfies the backward criterion, giving us the unconfoundedness.

The ATE is given by:

$$\tau \triangleq \mathbb{E}[Y(1) - Y(0)] = \mathbb{E}_W[\mathbb{E}[Y|T = 1, W] - \mathbb{E}[Y|T = 0, W]]$$

Thus from the causal estimand $\mathbb{E}[Y(1) - Y(0)]$ we can get statistical estimand $\mathbb{E}_W[\mathbb{E}[Y|T = 1, W] - \mathbb{E}[Y|T = 0, W]]$.

To compute the statistical estimand, a machine learning model (for example, a regression model) can be used $\hat{\mu} \approx \mu$ to compute the conditional expectation $\mathbb{E}[Y|T, W]$ and an empirical mean ($\frac{1}{n}\sum_i$) can be computed over all the data (n) to approximate $\mathbb{E}_W$

$$\mu(1, W) - \mu(0, W) = \mathbb{E}[Y|T = 1, W] - \mathbb{E}[Y|T = 0, W]$$

The model $\hat{\mu}$ is known as the conditional outcome model, and the estimator as the COM estimator or S-Learner (Single) (Künzel et al. 2019).

Thus, the ATE using COM estimator is denoted by $\hat{\tau}$ and can be given by:

$$\tau = \frac{1}{n} \sum_i (\hat{\mu}(1, w_i) - \hat{\mu}(0, w_i))$$

Now, CATE estimation using both the adjustment set W and the observed (and unobserved) covariates X, using the model μ

$$\mu(t, w, x) \triangleq \mathbb{E}[Y|T = t, X = x, W]$$

Thus the statistical model $\hat{\mu}$ can be used to compute the CATE τx using the COM estimator as:

$$\hat{\tau(x)} = \frac{1}{n_x} \sum_{i:x_i=x} (\hat{\mu}(1, w_i, x) - \hat{\mu}(0, w_i, x))$$

Thus, ITE (which is primarily the measure we want) can be approximated using the difference in the predictions of two models as $n_x = 1\$$:

$$\tau_i = \hat{\mu}(1, w_i, x) - \hat{\mu}(0, w_i, x)$$

3.5.1 Grouped Conditional Outcome Modeling Estimator (GCOM Estimator)

In most cases, the treatment T is binary while the adjustment set and the covariates $W \cup X$ are high dimensional. As seen in the equation, with binary treatment $(0, 1)$ and a high dimensional w, the model fitting the two differences $\hat{\mu}(1, w_i) - \hat{\mu}(0, w_i)$ will ignore the treatment and resulting ATE will be closer to zero. One easy fix would be to compute two different models for each treatment. Grouping all the data with treatment $T = 1$ and fitting a model $\hat{\mu}_1(w)$ to predict outcome Y from $W \cup X$ and doing the same for treatment $T = 0$ by fitting a model $\hat{\mu}_0(w)$ and computing the average as below:

$$\hat{\tau(x)} = \frac{1}{n_x} \sum_{i:x_i=x} (\hat{\mu_1}(w_i, x) - \hat{\mu_0}(w_i, x))$$

Since the estimation is done by grouping based on the treatment values, the estimator is known as grouped conditional outcome model estimator (GCOM estimator). Though the GCOM estimator overcomes the dimensionality imbalance issue over the COM estimator, it has the disadvantage of not using all the data in estimation and fitting the statistical model compared to the COM estimator.

3.5.2 TARNet

COM estimators combine treatment T and the input covariates W, making the estimator biased towards zero. The GCOM estimator builds two separate estimators for each treatment (binary $T = 0$ and $T = 1$), and since it does not use all the data, it leads to a higher variance model. TARNet estimators can combine the two by first learning a representation $\hat{\mu}$ from all the input covariates, and then the layer can branch to two heads as shown in Figure 3.24.

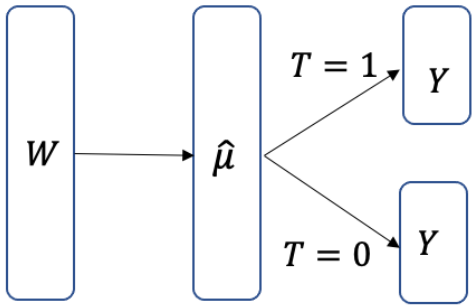


Figure 3.24: TARNet estimator

3.5.3 X-Learner

Kunzell et al. proposed a meta-learner, the X-learner, to overcome the limitations of Generalized Causal Outcome Model (GCOM) estimators. The GCOM approach falls short in its failure to utilize the complete dataset for estimating the Conditional Average Treatment Effect (CATE). In contrast, the X-learner uses all available data for both models that comprise the estimator, particularly in scenarios involving binary treatment variables, as detailed in (Künzel et al. 2019).

X-learner has the following three stages:

Step 1

Assume X is a sufficient adjustment set and is also covers all the covariates, given a binary treatment, two models $\hat{\mu_0}(x)$ and $\hat{\mu_1}(x)$ are estimated similar to GCOM estimator for each group.

Step 2(a)

In the first part, imputed ITE is computed for treatment group $\hat{\tau_{1,i}}$ using the observed potential outcome $Y_i(1)$ and the imputed counterfactual that we get from the first step $\hat{\mu_0}(x)$. Similarly,

we compute imputed ITE for the control group $\hat{\tau_{0,i}}$ using the observed potential outcome $Y_i(0)$ and the corresponding imputed counterfactual that we get from the first step $\hat{\mu_1}(x)$

$$\hat{\tau_{1,i}} = Y_i(1) + \hat{\mu_0}(x_i)$$

$$\hat{\tau_{0,i}} = Y_i(0) + \hat{\mu_1}(x_i)$$

Only individual elements from the treatment or control groups are used to compute the ITEs.

Step 2(b)

In this step, a supervised machine learning algorithm like regression can be used to fit a model $\hat{\tau_1}(x)$ to predict $\hat{\tau_{1,i}}$ from the above step for each x_i in the treatment group. Thus the model $\hat{\tau_1}(x)$ uses all the data from the treatment group in this step and the control group data $\hat{\mu_0}$ from the previous step. Similarly, a model is fit $\hat{\tau_0}(x)$ to predict $\hat{\tau_{0,i}}$ from the last step for each x_i in the control group.

Step 3

The two estimators are combined using a weighting function $0 < g(x) < 1$ as given:

$$\hat{\tau}(x) = g(x)\hat{\tau_0}(x) + (1 - g(x))\hat{\tau_1}(x)$$

The authors found that the propensity score performs well as a weighting function.

3.5.4 Matching

Matching is a relatively straightforward estimation technique wherein individuals from the treated and control groups are matched based on their covariates or confounders X, using a similarity or distance metric $d(\cdot, \cdot)$, as described in (Stuart 2010).

Figure 3.25 shows a simplified view with two dimensions (X_1, X_2) how the nearest neighbor (1-NN) can be used for matching between the treated and the control group.

Formally, the following procedure is followed for 1-NN as below:

1. Define a similarity or a distance metric $d(\cdot, \cdot)$.
2. For each individual, define $\mathcal{J}(i)$ so that we find the closest counterfactual match (treatment $(t_j \neq t_i)$) with another individual $j \neq i$

$$\mathcal{J}(i) = \text{argmin}_{j \ s.t \ t_j \neq t_i} d(x_j, x_i)$$

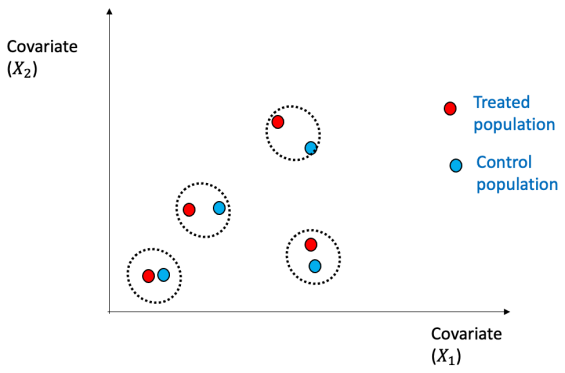


Figure 3.25: Matching Algorithm

3. Thus, every individual ITE can be computed using the actual and the potential counterfactual outcome obtained from $\mathcal{J}(i)$ above.

$$\hat{\tau_{1,i}} = y_i(1) - y_{\mathcal{J}(i)}$$

$$\hat{\tau_{0,i}} = y_{\mathcal{J}(i)} - y_i(0)$$

These two can be combined into a single notation:

$$\hat{\tau_i} = (2t_i - 1)(y_i - y_{\mathcal{J}(i)})$$

4. Thus, ATE can be computed using the average across all the individuals

$$\hat{\tau(x)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\tau_i}$$

Consequently, calculating the average treatment effect across the matched groups enables the causal effect estimation since the confounders are similar within these groups, and any differences are attributed solely to the treatment. This simplistic method works particularly well when the number of confounders is limited. However, as the number of dimensions or confounders increases, the method may suffer from the curse of dimensionality. Despite this drawback, the matching technique is easily interpretable by domain experts, although it heavily relies on the underlying metric of distance or similarity.

Notably, it has been demonstrated that the matching algorithm employing the 1-Nearest Neighbor (1-NN) method is equivalent to the covariate adjustment method, which facilitates relating theoretical properties based on this method.

3.5.5 Doubly Robust Estimator

Conditional outcome modeling ($\hat{\mu}(x)$) and propensity score-based estimators ($\hat{e}(x)$) can be combined to form the doubly robust estimator (Robins, Rotnitzky, and Zhao 1994).

$$\hat{\tau}(x) = \frac{1}{n}\sum_i [\hat{\mu}(1, x_i) - \hat{\mu}(0, x_i)]$$

$$\hat{\tau}(x) = \frac{1}{n}\sum_i [\hat{\mu}(1, \hat{e}(x_i)) - \hat{\mu}(0, (1 - \hat{e}(x_i))]$$

The doubly robust method has a property that they are consistent estimators of ATE $\hat{\tau}$ if either the conditional outcome modeling estimator ($\hat{\mu}(x)$) or the propensity score-based estimator ($\hat{e}(x)$) are consistent. Also, the technique converges much faster to τ than either ($\hat{\mu}(x)$) converging to ($\mu(x)$) or ($\hat{e}(x)$) converging to ($e(x)$). This technique has a distinct advantage in high-dimensional data.

3.5.6 Double Machine Learning

Double machine learning estimators, as the name suggests, use machine learning to learn estimators in two stages to "partial out" the confounders (and other covariates) X as shown in Figure 3.26.

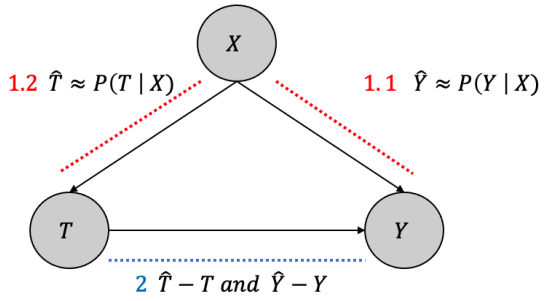


Figure 3.26: Double Machine Learning

1. Stage

1.1 Fit a machine learning model to predict Y from X to get $\hat{Y}$

1.2 Fit a machine learning model to predict T from X to get $\hat{T}$

2. Partial out the confounding effect by fitting another model to predict $Y - \hat{Y}$ and $T - \hat{T}$

3.5.7 Causal Trees and Causal Forests

Causal trees are similar to classification/regression trees, where leaf nodes, similar to the decision trees, are outcome variables, but the internal nodes are only limited to covariates and do not include the treatment (Wager and Athey 2018). The general algorithm is:

1. First, the observational data is divided into a train ($\mathcal{S}^{train}$) and a test ($\mathcal{S}^{test}$) set. The train set is used for building the tree, and the test set is used for estimation.

2. A greedy algorithm creates the splits like a regular decision tree. The goal of creating partition (Π) using the covariates is slightly different in causal trees compared to standard decision trees. The purpose of creating splits is to find the best covariate to split the node such that the treated group have a different outcome than the control group. The Kullback-Leibler Divergence is one of the techniques used to measure divergence between the outcome class distributions. If there are i outcomes, p_i and q_i are the outcome distribution in the treated and control groups, respectively, the KL divergence D between the two is given by:

$$D(P:Q) = \sum_i p_i \log \frac{p_i}{q_i}$$

   For a covariate X that splits a node into children nodes, with total N instances into N_x children, a conditional divergence test can be performed using KL divergence

$$D(P(Y|T=1) : P(Y|T=0)|X) = \sum_x \frac{N_x}{N} D(P(Y|T=1,x) : P(Y|T=0,x))$$

For the best split, the objective is to maximize the gain of the divergence between the outcome class distributions between treatment and control, and can be computed using:

$$D_{gain}(X) = D(P(Y|T=1) : P(Y|T=0)|X) - D(P(Y|T=1) : P(Y|T=0))$$

3. Cross-validation is used to select the depth $d*$ with pruning that minimizes the MSE of treatment effects using the folds as proxies for the test set.

4. Once the tree is fully constructed, the test set

$$\mathcal{S}^{test}$$

   is used to estimate the treatment effects at the leaf nodes.

Causal Forests are an extension of the idea of Causal trees for estimating the ATE. If we have a training set $\{(X_i, Y_i, T_i)\}_{i=1}^n$, a test data x and a causal tree predictor given by:

$$\hat{\tau}(x) = T(x; \{(X_i, Y_i, T_i)\}_{i=1}^n)$$

The intention behind causal forests is that instead of one causal tree if many different trees T^* are built, the average across them would be:

$$\hat{\tau}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b^*(x; \{(X_i, Y_i, T_i)\}_{i=1}^n)$$

Many techniques, such as bagging or subsampling the training set, can be employed to build many causal trees for the causal forest. Employing random covariates from the whole set for the splitting criterion can also result in different trees. Creating a tree from one subsample of observations and estimating the effect using other leads to a more unbiased estimate for each tree, making them "honest'' and the average of these honest trees make it an unbiased estimate overall. Regression forests built using the honest trees were shown to have a nice theoretical property of the estimates asymptotically normal as the observations go towards infinity.

3.5.8 Propensity Score-Based

As previously discussed, unbiased estimation of the average treatment effect can be achieved through a randomized controlled test, wherein individuals are assigned to either the treatment or control group based on a coin flip. The propensity score technique, on the other hand, aims to re-weight the observational data such that it resembles pseudo-randomized control test data (Imbens and Rubin 2015).

Consider a scenario involving binary treatment and two covariates within an observational dataset. The data points can be separated into two regions with opposing distributions. The propensity scores method involves re-weighting the samples, as depicted, to modify the distribution through weighting such that it is similar and closely approximates randomized assignment.

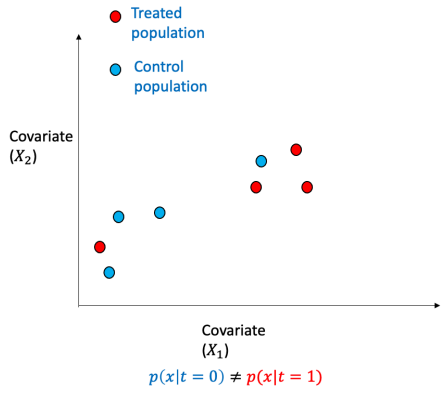


Figure 3.27: Observed data before and after Propensity reweighting

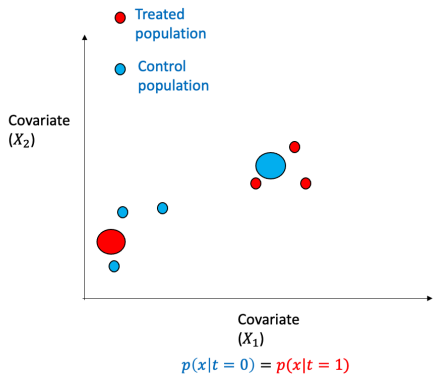


Figure 3.28: Observed data before and after Propensity reweighting

Propensity score is the probability of being subjected to the treatment ($T = 1$) given the adjustment set W and is denoted by:

$$e(W) \triangleq P(T = 1|X)$$

The propensity score can be learned using machine learning algorithms as any other regression problem. If the samples are re-weighted using the inverse propensity score of the treatment they received, thus changing the distribution to be more random and balanced.

Propensity score theorem Given the positivity assumption, the unconfoundedness given the adjustment set W implies unconfoundedness given the propensity score $e(W)$ and formally written as:

$$(Y(1), Y(0)) \perp T|W \implies (Y(1), Y(0)) \perp T|e(W)$$

The advantage of propensity scores and the theorem is that during conditioning, when the adjustment set W is high dimensional, one can use the propensity score $e(W)$, which is a scalar.

Inverse Propensity Weighting (IPW) Estimator Given the data $(X_1, T_1, Y_1), \cdots, (X_n, T_n, Y_n)$ the machine learning algorithm is first used to learn the estimator $\hat{p}(T = t|X)$. The ATE can be estimated using:

$$\hat{\tau(x)} = \frac{1}{n_1} \sum_{i \ s.t. t_i=1} \frac{y_i}{\hat{p}(T = 1|x_i)} - \frac{1}{n_0} \sum_{i \ s.t. t_i=0} \frac{y_i}{\hat{p}(T = 0|x_i)}$$

3.5.9 Propensity Score Matching

The Propensity Score Matching (PSM) algorithm is a methodology that emulates a Randomized Controlled Trial (RCT) in its approach to contrasting outcomes between treated and untreated cohorts within the sample that Propensity Score has matched.

However, the implementation of PSM necessitates careful consideration of certain caveats:

1. The first caveat, termed 'Common Support', necessitates that the distribution of propensity for treatment is analogous or identical across both treated and untreated cases.
2. The second caveat demands the exclusive utilization of baseline attributes unaffected by the intervention during the Matching process.
3. Thirdly, potential confounding variables must be both observable and without any hidden variables. A failure in this respect could result in biased estimates.
4. Finally, the fourth caveat advises matching the most pertinent characteristics rather than indiscriminately incorporating every variable into the equation.

The PSM process involves several key steps, as outlined by Jalan and Ravallion (Jalan and Ravallion 2003):

1. The calculation of the Propensity Score for all units.
2. The matching of treatment cohorts with control cohorts is performed following a predetermined matching strategy; for instance, a strategy could involve using the nearest neighbor method between the treated and control groups, implemented without replacement.
3. The evaluation of covariate balance. In the event of an imbalance, revisiting the first and second steps and incorporating alternative specifications is advisable.
4. The computation of the average outcome difference between the treatment and control groups.

3.5.10 Propensity Score Stratification

King and Nielsen propose that Propensity Score Matching (PSM) is designed to replicate a fully randomized experiment instead of a blocked randomized one. They further discuss that the exact matching procedure in PSM exacerbates issues such as imbalance, inefficiency, model dependence, and bias while also being unable to effectively mitigate the imbalance (King and Nielsen 2019).

Propensity Score (PS) stratification serves as a balancing mechanism, ensuring that the distribution of observed covariates appears comparable between treated and control groups when conditioned on the PS (Austin 2011). As a result, it facilitates adjusting imbalances in the covariates by modifying the score accordingly.

The specific steps to execute for PS stratification are:

1. Calculate the Propensity Score (PS) using logistic regression.
2. Mutually exclusive strata are established based on the estimated PS.
3. Both the treated and control units are grouped into each stratum.
4. The difference in means between treated and control groups is calculated within each stratum.

5. The means within each stratum are then weighted to achieve the target estimate.

In the second step of the process, studies have shown that approximately 90% of the bias inherent in the unadjusted estimate can be removed using five strata (Rosenbaum and Rubin 1984). However, the idea that increasing the number of strata beyond this point would lead to a further decrease in bias is not empirically supported. Indeed, simulation studies have indicated that the most favorable outcomes are achieved with between 5 and 10 strata, with different strata beyond this range contributing only minor improvements (Neuhäuser, Thielmann, and Ruxton 2018). It is also essential to consider the practical implications of increasing the number of strata. As the number of strata increases, the number of data points available within each stratum decreases.

During the fifth step of the process, Propensity Score (PS) stratification enables the calculation of both the Average Treatment Effect (ATE) and the Average Treatment Effect on the Treated (ATT), contingent on the weighting method utilized for the means. For the estimation of the ATE, the weighting is determined by the number of units within each stratum. On the other hand, the estimation of the ATT involves assigning weights according to the count of treated units present in each stratum (Imbens 2004).

3.6 Evaluation and Validation Techniques

Evaluating and validating causal models differ from traditional machine learning models, where techniques such as cross-validation and test set evaluations are performed. This section will highlight some standard metrics and methodologies to evaluate causal models.

3.6.1 Evaluation Metrics

The two broad categories for the evaluation metrics are based on whether the subpopulation is homogeneous or heterogeneous and are:

1. Standard Causal Effect Estimation
2. Heterogeneous Effect Estimation

3.6.1.1 Standard Causal Effect Estimation

Assuming the potential outcome to be real-valued, ff there M experiments being performed, and for each experiment, the observed ATE is τ, and the predicted ATE is $\hat{\tau}$, then various regression-based evaluation metrics are:

1. Mean Squared Error of Average Treatment Effect:

$$\epsilon_{MSE\_ATE} = \frac{1}{M}\sum_{j=1}^{M}(\tau_j - \hat{\tau}_j)^2$$

2. Root Mean Squared Error of Average Treatment Effect:

$$\epsilon_{RMSE\_ATE} = \sqrt{\frac{1}{M}\sum_{j=1}^{M}(\tau_j - \hat{\tau}_j)^2}$$

3. Mean Absolute Error of Average Treatment Effect:

$$\epsilon_{MAE\_ATE} = \frac{1}{M}\sum_{j=1}^{M}|\tau_j - \hat{\tau}_j|$$

3.6.1.2 Heterogeneous Effect Estimation

1. Uplift Curve

Uplift modeling aims to identify the effect of an intervention on a particular individual rather than a population, especially in the case of heterogeneity (Gutierrez and Gérardy 2017). Thus, uplift modeling attempts to estimate the ITE (or CATE), i.e., the treatment outcome of a given individual and how it would differ in the absence.

The methodology to generate the uplift curve has parallels to ROC curves in standard machine learning, and the steps are: 1. Use the machine learning method as discussed for estimation and generate CATE for each individual $(\hat{\mu}_i^1 - \hat{\mu}_i^0)$ 2. Sort the uplift scores by decreasing order and compute percentiles in a range $((0,10)(10,20)\cdots(90,100))$ 3. For each bucket in the percentiles, one can estimate the difference in prediction for the treatment and the control group predictions on responses. The difference in average is taken for each decile.

$$\left(\frac{Y^1}{N^1} - \frac{Y^0}{N^0}\right)(N^1 + N^0)$$

where Y^1 and N^1 are the sum of the treated individual outcome and the number of treated individuals for the bin. Similarly, Y^0 and N^0 are the control observations' sum and number.

4. The uplift curve is then plotted with the x-axis representing the percentiles of the population and the y-axis representing the uplift gain from the above corresponding to each group.

The advantage of the uplift curve is that we can select the decile that maximizes the gain as the limit of the population to be targeted next time rather than the whole population.

2. Qini Curve

Qini curve is a variant of uplift curve where Qini score is computed instead of the uplift score as:

$$Y^1 - Y^0 \frac{N^0}{N^1}$$

When there is an imbalance between the treatment and control groups, this measure offers more correction than the uplift score.

3. Uplift(Qini) Coefficient

Similar to AUC-ROC curve, one can compute the area under the uplift (qini) curve, and is referred to as the uplift (qini) coefficient and is given by:

$$Uplift_{Coef} = \sum_{a=0}^{N-1} (Uplift(a+1) + Uplift(a))$$

$$Qini_{Coef} = \sum_{a=0}^{N-1} (Qini(a+1) + Qini(a))$$

3.6.2 Robustness Checks and Refutation Techniques

Several assumptions are made in every step of causal inference, from building the causal model to estimation. Assumptions are made at the modeling level, such nonexistence of unobserved variables, the relationship between variables (edges in the graph), etc. We might make parametric assumptions for deriving the estimand at the identification step. At the estimation step, we might assume a linear relationship between treatment and observed and similarly between treatment and outcome. Many of these assumptions can be and should be tested for violations, if any. There are many assumptions that are not possible to be validated or refuted.

Similar to standard software testing, there is unit or modular testing and integration testing.

3.6.2.1 Unit or Modular Tests

Designing tests or validations to individually check the assumptions on the model, identification, and estimation process. Some of the tests are:

1. Conditional Independence Tests: Using the dependence graph and data to validate various independence assumptions, for example, with two variables (X_1, X_2) and their relationship with treatment T if $P(T|X_1, X_2) = P(T|X_1)P(T|X_2)$

2. D-Separation Tests: Conditional and marginal independence can be tested between variables in graphs using moralize, orient, delete/add edges, etc.
3. Bootstrap Sample Validation: Replacing the dataset completely with bootstrapped samples from the graph helps calculate statistically significant changes in the estimand.
4. Data Subsets Validation: Replacing the given dataset with a randomly selected subset helps to compute changes in the estimands and gauge the impact.

3.6.2.2 Integration or Complete Tests

In integration testing, comprehensive testing on the entire process for validating many underlying assumptions rather than on single steps. Some of them are:

1. Placebo Treatment Refuter: What would impact the outcome if the treatment variable is replaced by a random variable (e.g., Gaussian)? It should have no impact (zero value of estimation) if the assumptions are all correct or some steps must be corrected.
2. Adding Random Common Cause: Adding an independent random variable as a common cause should keep the estimate the same. This method can be easily tested on the dataset to see the significance of the estimation change.
3. Dummy Outcome Refuter: The estimated causal effect should be zero if we replace the outcome variable with an independent random variable.
4. Simulated Outcome Refuter or Synth Validation: If multiple datasets are generated very close to the generation process of the existing dataset and the assumptions made, the estimation effect should remain the same. This technique is also known as the synth validation technique and is one of the most comprehensive tests for process validation.
5. Adding Unobserved Confounder: One of the real-world cases if missing the observed confounder from the data or modeling. By simulating a confounder based on some correlation ρ between the outcome and the treatment, one can run the analysis and see the difference in the estimation. A significant change illustrates a robustness issue in the process.

3.7 Unconfoundedness: Assumptions, Bounds, and Sensitivity Analysis

Throughout the discussion, we assumed unconfoundedness or observed confounding in our inference process. However, Manski et al., in their work, showed that the no unobserved confounding assumption is unrealistic in the real world (Manski 2003).

In the simplest case, we assume an unobserved confounder U along with an observed confounder W as shown in Figure 3.29. Thus, the ATE can be written using the adjustment formula as:

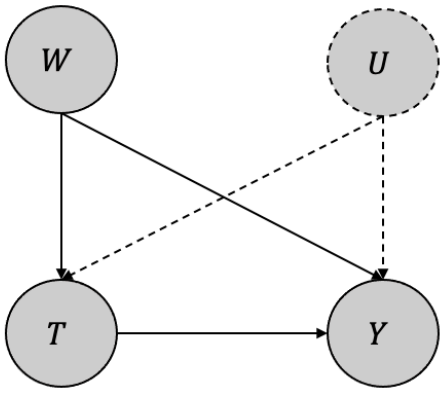


Figure 3.29: Observed Confounding

$$\mathbb{E}[Y(1) - Y(0)] = \mathbb{E}_{W,U}[\mathbb{E}[Y|T = 1, W, U]-]\mathbb{E}[Y|T = 0, W, U]]$$

Since U is unobserved, the ATE can be approximated as

$$\mathbb{E}[Y(1) - Y(0)] \approx \mathbb{E}_{W}[\mathbb{E}[Y|T = 1, W]-]\mathbb{E}[Y|T = 0, W]]$$

The impact of this approximation and how close the ATE in the equation and equation depends on many underlying conditions. Thus, instead of the ATE being a single value becomes an interval with bounds that depend on the assumptions.

With the simple assumption that the outcome Y is bounded between 0 and 1, we know that the individual treatment effect is bounded between the maximum limit as below

$$0 - 1 \leq Y_i(1) - Y_i(0) \leq 1 - 0$$

Thus,

$$-1 \leq Y_i(1) - Y_i(0) \leq 1$$

Hence the expectations can be bounded as follows:

$$-1 \leq \mathbb{E}[Y(1) - Y(0)] \leq 1$$

More generally if potential outcomes are bounded between l and h, the ATE bounds have the interval length $2(h - 1)$ and given by:

$$l - h \leq \mathbb{E}[Y(1) - Y(0)] \leq h - l$$

3.7.1 Observational Counterfactual Decomposition

The ATE can be written in terms of observational and counterfactual components, known as observational-counterfactual decomposition.

The linearity of expectations gives:

$$\mathbb{E}[Y(1) - Y(0)] = \mathbb{E}[Y(1)] - \mathbb{E}[Y(0)]$$

Conditioning and marginalization on the treatments give:

$$\begin{aligned}\mathbb{E}[Y(1) - Y(0)] = P(T = 1)\mathbb{E}[Y(1)|T = 1] + P(T = 0)\mathbb{E}[Y(1)|T = 0] \\ -(P(T = 1)\mathbb{E}[Y(0)|T = 1] + P(T = 0)\mathbb{E}[Y(0)|T = 0])\end{aligned}$$

$$\begin{aligned}\mathbb{E}[Y(1) - Y(0)] = P(T = 1)\mathbb{E}[Y(1)|T = 1] + P(T = 0)\mathbb{E}[Y(1)|T = 0] \\ -P(T = 1)\mathbb{E}[Y(0)|T = 1] - P(T = 0)\mathbb{E}[Y(0)|T = 0]\end{aligned}$$

$$\begin{aligned}\mathbb{E}[Y(1) - Y(0)] = P(T = 1)\mathbb{E}[Y|T = 1] + P(T = 0)\mathbb{E}[Y(1)|T = 0] \\ -P(T = 1)\mathbb{E}[Y(0)|T = 1] - P(T = 0)\mathbb{E}[Y|T = 0]\end{aligned}$$

Thus the equation has observational elements $P(T = 1)\mathbb{E}[Y|T = 1], P(T = 0)\mathbb{E}[Y|T = 0]$ and the counterfactual elements $P(T = 0)\mathbb{E}[Y(1)|T = 0], P(T = 1)\mathbb{E}[Y(1)|T = 1]$ and hence the observational-counterfactual decomposition.Now if we denote $P(T = 1)$ with π and $P(T = 0)$ becomes $1 - \pi$, the equation can we written as:

$$\mathbb{E}[Y(1) - Y(0)] = \pi\mathbb{E}[Y|T = 1] + (1 - \pi)\mathbb{E}[Y(1)|T = 0] - \pi\mathbb{E}[Y(0)|T = 1] - (1 - \pi)\mathbb{E}[Y|T = 0]$$

3.7.2 Bounds

We will provide an overview of nonparametric bounds and elucidate the process of deriving them.

3.7.2.1 No-Assumption Bounds

The no-assumption bounds are the simplest bound that reduces the interval $2(h - 1)$ by half to $(h - 1)$ as given below:

$$\mathbb{E}[Y(1) - Y(0)] \leq \pi\mathbb{E}[Y|T = 1] + (1 - \pi)h - \pi l - (1 - \pi)\mathbb{E}[Y|T = 0]$$

$$\mathbb{E}[Y(1) - Y(0)] \geq \pi\mathbb{E}[Y|T = 1] + (1 - \pi)l - \pi h - (1 - \pi)\mathbb{E}[Y|T = 0]$$

Thus, the interval length is:

$$(1-\pi)h + \pi h - \pi l - (1-\pi)l$$

$$= h - 1$$

The idea with more assumptions, as discussed next, is to get a tighter lower and upper bound than the no-bounds interval.

3.7.2.2 Nonnegative Monotone Treatment Response Assumption

Assuming that the treatment always helps, i.e., $\forall i\ Y_i(1) \geq Y_i(0)$ means that ITE is always greater than 0, and thus the lower bound changes from $l - h$ to 0. The higher bound remains what we got from the no-assumption bounds. Thus, the intervals are:

$$0 \leq \mathbb{E}[Y(1) - Y(0)] \leq \pi\mathbb{E}[Y|T=1] + (1-\pi)l - \pi h - (1-\pi)\mathbb{E}[Y|T=0]$$

By assuming the reverse, the treatment never helps, i.e., $\forall i\ Y_i(1) \leq Y_i(0)$, gives us the ITE always less than 0, and thus the upper bound changes from $h - 1$ to 0. The lower bound remains what we got from the no-assumption bounds. Thus, the intervals are:

$$\pi\mathbb{E}[Y|T=1] + (1-\pi)h - \pi l - (1-\pi)\mathbb{E}[Y|T=0] \leq \mathbb{E}[Y(1) - Y(0)] \leq 0$$

3.7.2.3 Monotone Treatment Selection Assumption

The assumption is that the treatment groups' potential outcomes are better than the control groups. Thus, we get $\mathbb{E}[Y(1)|T=1] \geq \mathbb{E}[Y(1)|T=0]$ for $Y(1)$ and $\mathbb{E}[Y(0)|T=1] \geq \mathbb{E}[Y(1)|T=0]$ for $Y(0)$. It can be shown that under the monotone treatment selection assumption, the ATE is bounded by the associational difference of the observations and given by:

$$\mathbb{E}[Y(1) - Y(0)] \leq \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$$

Thus, by combining the nonnegative monotone response lower bound assumption and monotone treatment selection upper bound response, we get a tighter interval than the no-assumption bounds and is:

$$0 \leq \mathbb{E}[Y(1) - Y(0)] \leq \mathbb{E}[Y|T=1] - \mathbb{E}[Y|T=0]$$

3.7.2.4 Optimal Treatment Selection Assumption

The assumption here is that each individual gets the treatment that is best suited, i.e. $\forall i\ T_i = 1 \implies Y_i(1) \geq Y_i(0)$ and $\forall i\ T_i = 0 \implies Y_i(0) \geq Y_i(1)$. Thus, we have $\mathbb{E}[Y(1)|T = 0] \leq \mathbb{E}[Y|T = 0]$ and $\mathbb{E}[Y(0)|T = 1] \leq \mathbb{E}[Y|T = 1]$ which when plugged into the observational-counterfactual equation we get

$$\mathbb{E}[Y(1) - Y(0)] < \pi\mathbb{E}[Y|T = 1] - \pi l$$

$$\mathbb{E}[Y(1) - Y(0)] \geq (1 - \pi)l - (1 - \pi)\mathbb{E}[Y|T = 0]$$

3.7.3 Sensitivity Analysis

Given the presence of observed W and the unobserved U, the sensitivity analyses help us to quantify the unconfoundedness difference between the ATE adjusted to both $\mathbb{E}_{W,U}$ as compared to just the observed $\mathbb{E}_W$ (Cinelli et al. 2019).

Considering a simple setting with observed variable W and an unobserved variable U with only linear impacts as shown in Figure 3.30, the structural causal model equations as a linear function will be:

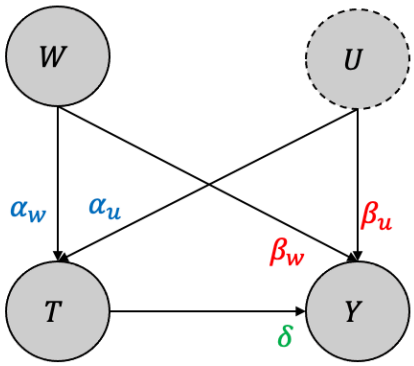


Figure 3.30: Sensitivity Analysis

$$T = \alpha_w W + \alpha_u U$$

$$Y = \beta_w W + \beta_u U + \delta T$$

It can be shown that when adjusted for both W, U, we get δ

$$\mathbb{E}[Y(1) - Y(0)] = \mathbb{E}_{W,U}[\mathbb{E}[Y|T = 1, W, U] - \mathbb{E}[Y|T = 0, W, U]] = \delta$$

Also, when adjusted only for the observed W, we get

$$\mathbb{E}[Y(1) - Y(0)] = \mathbb{E}_W[\mathbb{E}[Y|T = 1, W] - \mathbb{E}[Y|T = 0, W]] = \delta + \frac{\beta_u}{\alpha_u}$$

Thus the bias, i.e., what would be the difference between when we do not adjust for the unobserved U as compared to when we adjust for both W, U will be the difference of the two and is $\frac{\beta_u}{\alpha_u}$.

A contour plot for different values of β_u and α_u gives the sensitivity to single unobserved confounding as shown in Figure 3.31

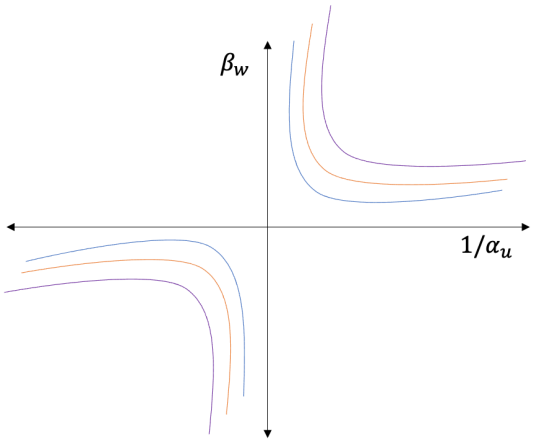


Figure 3.31: Sensitivity Plot

Many researchers, such as Cinelli et al. and Vetich et al., have shown techniques to reduce the constraints (linear assumptions, single variable) and yet be able to perform sensitivity analyses similar to the simple case (Cinelli and Hazlett 2020, veitch2020sense).

3.8 Case Study

We will go through different steps and processes of causal inference to demonstrate and give a practical hands-on experience with a real-world dataset. The goal is to take various steps highlighted in the chapter using the tools. A version of the Python code used in this case study can be found in this Colab notebook.

3.8.1 Dataset

Economist Jean-Jacques Lalonde collected the Lalonde dataset in the 1970s, and has been widely used in research on the evaluation of social programs. The NSWD program was de-

signed to test the effectiveness of a job training and placement program for disadvantaged individuals. The program provided job training, job placement services, and a wage subsidy to disadvantaged individuals (the treatment group), while a control group received no treatment. The goal of the program was to determine whether the treatment had a positive effect on the employment and earnings of the participants. The dataset includes a variety of variables, including demographic characteristics (such as age, education level, and race), employment status, and income.

3.8.2 Tools and Library

We will use **doWhy**, a Python library for causal inference, for most modeling and analysis. The library includes tools for performing various causal inference tasks, such as identifying the causal effect of a treatment on an outcome variable, estimating the total effect of a treatment on an outcome variable using various interchangeable estimators and assessing the robustness of causal estimates to assumptions about the data generating process. In the case study we use **causalml** and **causallift** for further distributional analysis and uplfit modeling. Python libraries such as **pandas**, **matplotlib**, **scikit-learn** etc. are used for data processing, visualization and machine learning.

3.8.3 Exploratory Data Analysis

Plotting the treated group vs. control group with various variables (age, race, income, education) for understanding the distribution across the two as shown:

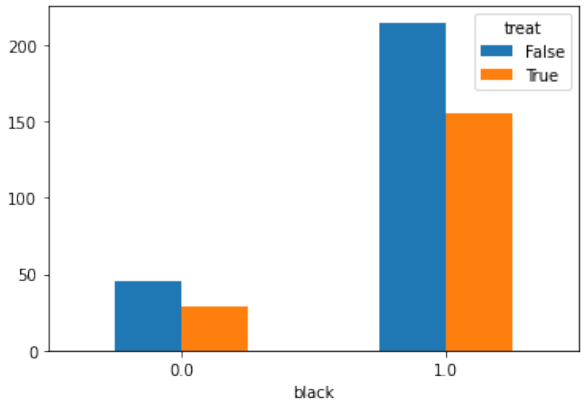


Figure 3.32: Treatment vs (Race, Age, Education)

One can see that the dataset is not balanced between the treated and the control group. The difference between the treated and control groups is quite evident for various variables such as education, age, and hispanic. This may cause issues in many estimation processes and in the

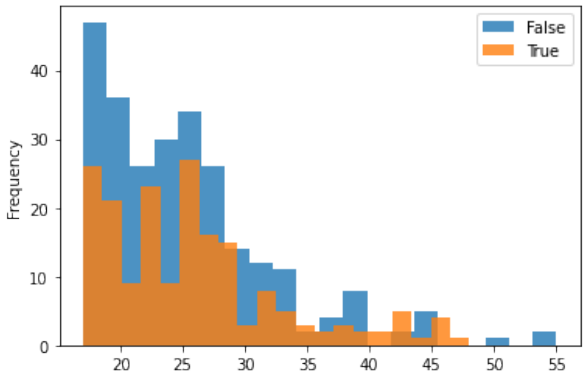


Figure 3.33: Treatment vs (Race, Age, Education)

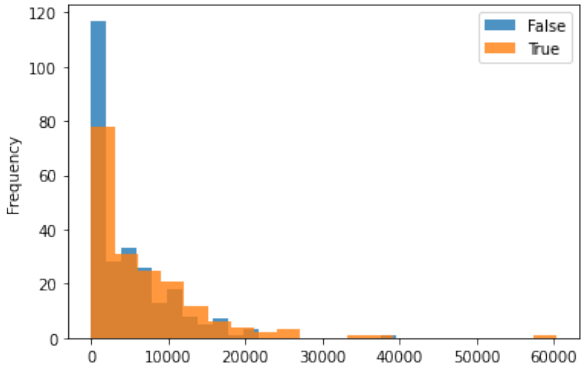


Figure 3.34: Treatment vs (Race, Age, Education)

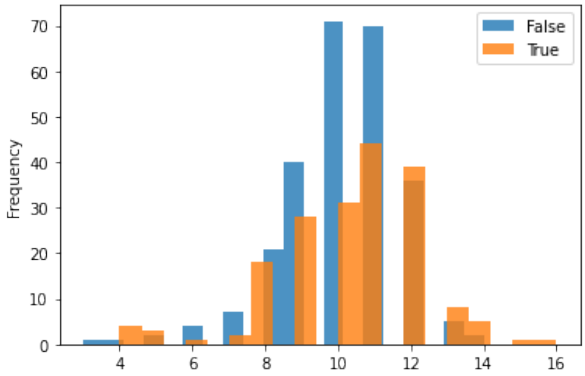


Figure 3.35: Treatment vs (Race, Age, Education)

propensity-based estimation, we will highlight how the propensity-based techniques change the distribution through weights.

3.8.4 Estimation and Results

3.8.4.1 Identification of Estimand

As discussed we first identify the estimand with variables **treat** as the treatment T, **re78** as the outcome Y and other nominal/numeric ones such as **nodegr**, **black**, **hisp**, **age**, _educ___ and ___married as the covariates X as shown in the listing.

```
from dowhy import CausalModel

model = CausalModel(
  data = lalonde_df,
  treatment='treat',
  outcome='re78',
  common_causes='nodegr+black+hisp+age+educ+married'.split('+')
)
identified_estimand = model.identify_effect()
```

The causal graph showing the relationships between the outcome, treatment, and observed confounders is shown in Figure 3.36

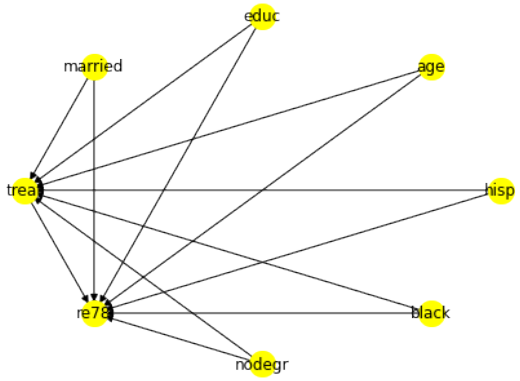


Figure 3.36: Causal Graph for Identification

3.8.4.2 Estimation and Robustness

We have explored many linear, non-linear, propensity-based, and causal tree-based estimators to give the readers a more comprehensive view.

A simple linear regression estimation is shown, and the results.

```
linear_regression_estimate = model.estimate_effect(
  identified_estimand,
  method_name="backdoor.linear_regression",
  control_value=0,
  treatment_value=1
)
print(linear_regression_estimate)
```

```
*** Causal Estimate ***

## Identified estimand
Estimand type: EstimandType.NONPARAMETRIC_ATE

### Estimand : 1
Estimand name: backdoor
Estimand expression:
   d
--------(E[re78|age,nodegr,married,educ,hisp,black])
d[treat]
Estimand assumption 1, Unconfoundedness: If U→{treat} and U→re78 then
P(re78|treat,age,nodegr,married,educ,hisp,black,U) =
P(re78|treat,age,nodegr,married,educ,hisp,black)

## Realized estimand
b: re78~treat+age+nodegr+married+educ+hisp+black
Target units: ate

## Estimate
Mean value: 1671.1304316174173
```

As discussed in the exploratory data analysis, the data distribution was not symmetrical between the control and the treated group, so we used the inverse propensity-score weighting technique as one of the estimators.

```
causal_estimate_ipw = model.estimate_effect(
  identified_estimand,
  method_name="backdoor.propensity_score_weighting",
  target_units = "ate",
  method_params={"weighting_scheme":"ips_weight"}
)

print(causal_estimate_ipw)
```

The **doWhy** library provides interesting interpreting techniques to understand the change in distribution, as shown in the listing.

```
causal_estimate_ipw.interpret(
  method_name="confounder_distribution_interpreter",
  var_type='discrete',
  var_name='married',
  fig_size = (10, 7),
  font_size = 12
)
```

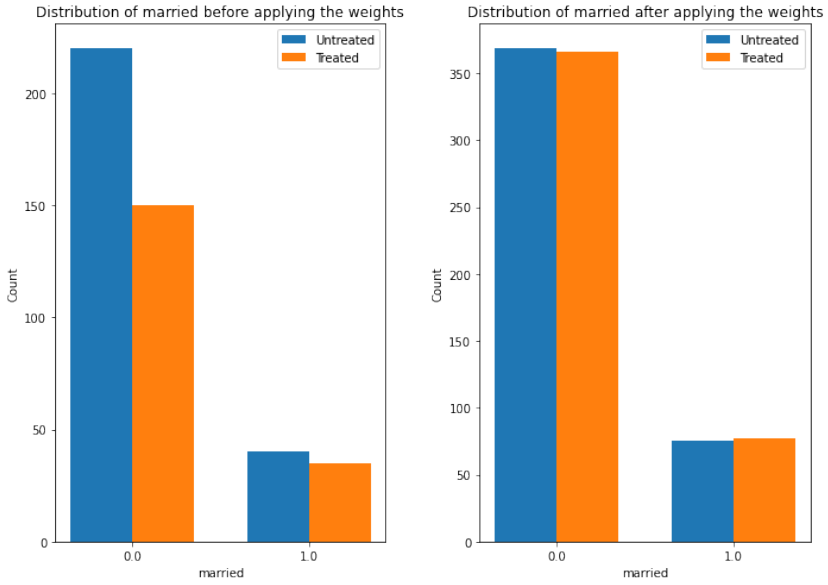


Figure 3.37: Married distribution before and after inverse propensity weighting

Table for Estimator Comparison

Estimator	ATE
Naive	1794.342
Linear Regression	1671.13
T-Learner	1693.76
X-Learner	1763.83
T-Learner	1693.76
Double Machine Learner	1408.93
Propensity Score Matching	1498.55
Propensity Score Stratification	1838.36
Propensity Score and Weighting	1639.80

3.8.5 Refutation and Validation

Next, we highlight some refutation and validation tests performed on the model, as discussed in the chapter.

3.8.5.1 Removing Random Subset of Data

We choose the causal estimate from inverse causal weighting to perform the refutation as shown:

```
res_subset = model.refute_estimate(
  identified_estimand,
  causal_estimate_ipw,
  method_name="data_subset_refuter",
  show_progress_bar=True,
  subset_fraction=0.9
)
```

The difference between the two is around 17, and since the p-value is $0.98 > 0.05$ we can safely say that the null hypothesis is valid and the refutation task had no impact on the estimation.

```
Refute: Use a subset of data
Estimated effect:1639.7956658905296
New effect:1656.1009245901791
p value:0.98
```

3.8.5.2 Placebo Treatment

Replacing treatment with a random (placebo) variable as shown:

```
import numpy as np
res_placebo = model.refute_estimate(
  identified_estimand,
  causal_estimate_ipw,
  method_name="placebo_treatment_refuter",
  show_progress_bar=True,
  placebo_type="permute"
)
```

The output

```
Refute: Use a Placebo Treatment
Estimated effect:1639.7956658905296
New effect:-209.15727259572515
p value:0.78
```

The causal estimation through inverse probability weighting can be considered robust based on the p-value.

4 Causal Discovery

The fundamental assumption in causal inference using causal graphs has been the requirement of an established causal model for estimating the causal effect. However, constructing such models *a priori* is often challenging or unfeasible in practice. As an alternative, causal discovery or causal structure search, based on the analysis of statistical properties of purely observational data, has emerged as a crucial process for uncovering causal relationships.

This chapter provides an introduction and review of the computational techniques for causal discovery that have been developed for this purpose.A diverse set of assumptions are introduced that engender various types of causal discovery algorithms. These algorithms are subsequently explicated, and directed explicitly toward assessing their relative strengths and weaknesses. Finally, a case study is conducted in which the algorithms are applied to a real-world problem, demonstrating their utility in the realm of causal discovery.

4.1 Assumptions of Causal Discovery

In the realm of modeling and inference, a forward problem entails forecasting the outcome or response of a system or process based on a given set of inputs or parameters. For example, estimating the causal effect from the causal model is an example of a forward problem. In contrast, an inverse problem is the process of determining the inputs or parameters of a system or model based on a set of observed outcomes or responses. This involves working backward from the data to identify the underlying causes or inputs that led to the observed outcomes. One example of an inverse problem is discovering the causal model and structure based on observational data. Solving inverse problems is significantly more challenging since multiple causal models can be constructed from the same observational data (Maclaren and Nicholson 2019).

In the context of solving inverse problems, it is typical to rely on certain assumptions regarding the target of inquiry to narrow down the potential solutions and increase the likelihood of success. Four common assumptions are frequently utilized across causal discovery algorithms (Verma and Pearl 2022; Frydenberg 1990).

4.1.1 Markov Assumption

The Markov assumption in causal graphical models assumes that each node is conditionally independent of its non-descendants given its parent nodes. In other words, the Markov assumption states that once the parent nodes of a node are known, information about any other variables in the system is irrelevant in determining the value of that node. This assumption allows for simplifying complex causal structures into a set of conditional independence statements, which can be represented in a graphical model.

> Tip
>
> Thus, according to the Markov assumption, If the variables are d-separated in the graphical model G, they are statistically independent in the observed data distribution P.

$$X \perp_G Y|Z \implies X \perp_P Y|Z$$

4.1.2 Faithfulness Assumptions

The faithfulness assumption is the reverse of the Markov assumption, which means that based on the statistical independencies in the distribution P, one can deduce d-separations in the graph G.

$$X \perp_G Y|Z \impliedby X \perp_P Y|Z$$

Violations in faithfulness assumption occur when statistical dependencies in the observed data do not correspond to d-separations in the underlying causal graph. An example of violating the faithfulness assumption occurs when two causal pathways between the nodes cancel each other. Consequently, the nodes may seem statistically independent from the observed data, but they are not d-separated.

4.1.3 Causal Sufficiency

The causal sufficiency assumption in causal discovery is the assumption that all variables that have a causal effect on the outcome of interest are included in the analysis. In other words, the causal sufficiency assumption requires that no unmeasured or unobserved variables are causally related to both the exposure and the outcome.

4.1.4 Acyclicity

The acyclicity assumption in causal discovery assumes there should be no directed cycles (i.e., no feedback loops) in the causal graph.

4.2 From Assumptions to Structures

How do these assumptions help one to narrow the search in structural space? In causal graphical models, the concept of Markov equivalence is crucial as it pertains to the set of graphs that encode the same conditional independencies.

> Tip
>
> Specifically, two graphs are considered **Markov equivalent** if they correspond to the same set of conditional independencies. By understanding the Markov equivalence of different graph structures, one can simplify complex causal structures and identify equivalent graphical models with the same conditional independence properties.

It is worth noting that specific basic structures, such as **chains** and **forks** (discussed in Chapter 3), belong to the same Markov equivalence class. However, the basic immorality structure is distinct and constitutes its own Markov equivalence class. Applying the Markov and faithful assumptions to the data lets us deduce that all chains and forks have a standard **skeleton**. The skeleton of a graph is obtained by replacing all its directed edges with undirected edges.

Thus two important graph qualities that one can use to distinguish the structures are (1) Immoralities and (2) Skeleton, leading to an important theorem.

Two graphs are Markov equivalent if and only if they have the same skeleton and immoralities

Thus, we can discover a partially directed graph by using the observed data and analyzing the conditional dependencies to construct a skeleton and then apply the direction using the immoralities.

4.3 Causal Discovery Algorithms

In the forthcoming section, we will explore the realm of causal discovery algorithms' role in uncovering causal relationships from observational or experimental data.

4.3.1 Constraint-Based Algorithms

Causal discovery through constraint-based methods typically involves the use of conditional independence tests. However, performing these tests can be challenging when the dependence between variables is still being determined. Despite this challenge, the constraint-based approach is generally applicable across a wide range of domains. The approach's effectiveness depends heavily on the sample size, with larger sample sizes often necessary to achieve accurate conditional independence tests. A major disadvantage of the constraint-based approach is the reliance on the faithfulness assumption, which may be a strong assumption difficult to satisfy in practice.

4.3.1.1 PC Algorithm

The PC algorithm, proposed by Spirtes et al., is considered one of the oldest causal discovery algorithms whose foundation lies in the causal Markov condition and the faithfulness assumption and that there are no latent confounders (Spirtes et al. 2000).

Let us consider an example with a known causal structure and then follow the steps of the PC algorithm to uncover the same structure. Figure 4.1 shows the original causal graph and the steps followed in the PC algorithm.

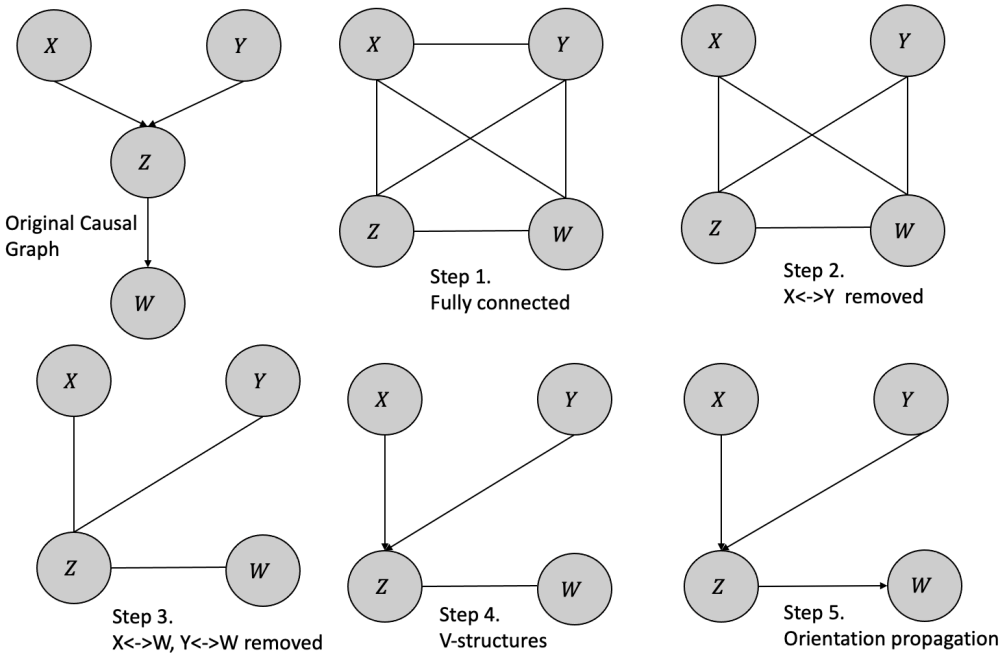


Figure 4.1: PC algorithms and steps

Following are the steps in the PC algorithm:

1. Form a complete undirected, fully connected graph with all the variables.
2. Remove all the unconditionally independent edges from the observational data. In the example, we remove the edge X and Y because $X \perp Y$.
3. For every pair (A, B), eliminate the edge between variables A and B if they have an edge between them and there exists a variable C with an edge connected to either A or B such that $A/$ is conditionally independent of B given C. In the example we remove the edges between X and W and also between Y and W because $X \perp W|Y$ and $Y \perp W|Z$
4. For every triple of variables (A, B, C) where A and B are connected, B and C are connected, and A and C are not connected, establish the edges $A \rightarrow B \leftarrow C$,, provided that B was not included in the condition set that made A and C independent, and the edge between them was subsequently removed. Such a triple of variables is referred to as a v-structure. In the given example, the $X-Y$ edge was eliminated without conditioning on Z. Thus, we can orient the $X - Z - Y$ edge as $X \rightarrow Z \leftarrow Y$.
5. The process of orientation propagation involves orienting the edge $B - C$ as $B \rightarrow C$ for every triple of variables where $A \rightarrow B-C$ and A and C are not adjacent. The orientation of $Y \rightarrow Z - W$ is determined as $Y \rightarrow Z \rightarrow W$, resulting in the unique recovery of the true structure in this particular example.

> **Tip**
>
> The PC algorithm is designed to converge to the true Markov Equivalence Class if the conditional independence decisions are accurate in the limit of a large sample size.

4.3.1.2 FCI Algorithm

The Fast Causal Inference (FCI) is a constraint-based algorithm considered a more efficient and scalable version of the PC algorithm (Spirtes et al. 2000). The FCI algorithm is the more general form of the PC algorithm as it enables the identification of unknown confounding variables.

The FCI algorithm bears several similarities to the PC algorithm in terms of the steps involved in detecting the relationships or edges between variables. Like the PC algorithm, the FCI algorithm generates a causal graph by initially creating a fully connected undirected graph and subsequently eliminating edges linking conditionally independent variables.

Once the algorithm eliminates edges based on conditional independence, it proceeds to identify directions for the remaining edges wherever possible, leaving edges without directions when the direction is unknown. The graph in Figure 4.2 illustrates a true causal model with relationships and an observed confounding variable U. After eliminating the edges based on conditional independence, it examines for colliders. For instance, when X and Z are unconditionally independent in a causal model, the $X - Z$ edge can be eliminated without conditioning on Y. In this scenario, the $X - Y - Z$ triple is treated as a collider and oriented as $X \rightarrow Y \leftarrow Z$ signifying that Y is a common effect of X and Z. The bidirectional edge linking nodes Y

and Z denotes the presence of an indeterminate confounding variable affecting the observed relationship between Y and Z. The presence of circle symbols at nodes X and W signifies that the algorithm is unable to discern the directionality of the relationship between X and Y, whether it is a directed edge from X to Y, an unobserved confounder, or a combination thereof. This also holds true for the relationship between Y, Z and W in a similar way.

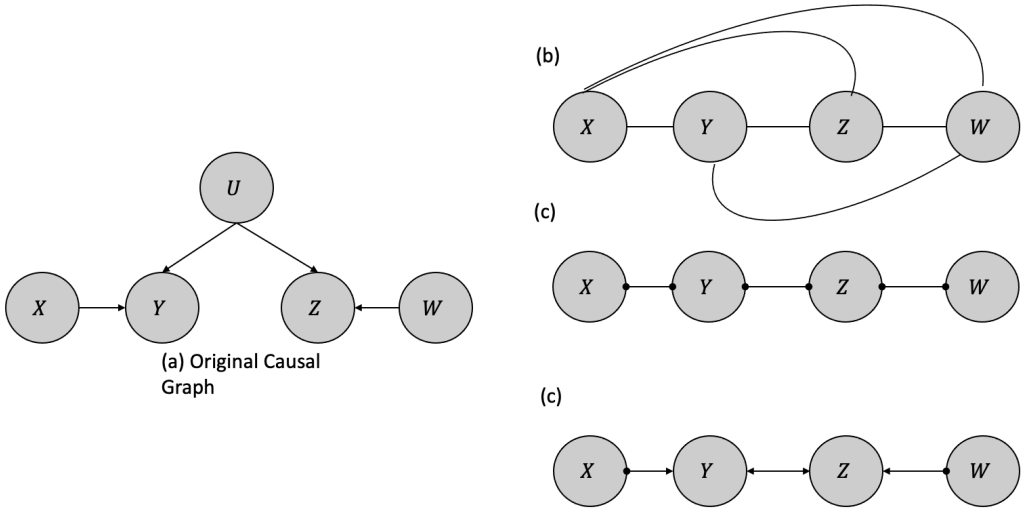


Figure 4.2: FCI Algorithm

Spirtes et al., introduced an adapted form of the FCI algorithm, the Anytime FCI (Spirtes 2001). Anytime FCI employs conditional independence tests involving conditioning sets smaller than a predetermined K threshold.

The Really Fast Causal Inference (RFCI) algorithm is characterized by its utilization of a reduced number of conditional independence tests when compared to the FCI algorithm (Colombo et al. 2012). Furthermore, RFCI tests condition on a smaller subset of variables. These optimizations confer a notable speed advantage to RFCI over FCI. Additionally, RFCI's outcomes are inclined to be more reliable in the context of small samples, as high-order conditional independence tests exhibit diminished statistical power. However, the trade-off of these optimizations is that RFCI's results may need to be more informative.

Additional extensions exist to current causal discovery algorithms, such as the CCD algorithm that operates without the requirement of acyclicity (Richardson 1996). Moreover, several methods based on SAT-based causal discovery are available, enabling us to dispense with the assumptions of causal sufficiency and acyclicity (Hyttinen et al. 2013).

4.3.2 Score-Based Algorithms

Score-based algorithms begin by creating a completely undirected graph, representing the conditional independence relationships inherent in the data. These algorithms iteratively modify this initial structure by adding and removing edges while simultaneously computing the corresponding scores. The goal is to select the structure that yields the highest score value. The selection of the score function is dependent on the specific algorithm and may include Bayesian Information Criterion (BIC), Akaike Information Criterion (AIC), and Maximum Likelihood (ML), among others. These functions balance the model complexity and goodness of fit to avoid overfitting, which could lead to unreliable causal inferences.

4.3.2.1 Greedy Equivalence Search Algorithm

The Greedy Equivalence Search (GES) is an algorithmic architecture for causal structure discovery (Chickering 2002). It is grounded on the principle of iteratively constructing a partially directed acyclic graph (PDAG) that represents the conditional independence relationships inherent in the data. The algorithm operates by initially creating an empty graph and iteratively adding edges based on score functions such as Bayes Information Score (BIC) or the Z-score that evaluate the goodness of fit between the data and the causal structure. GES incorporates a greedy search strategy, prioritizing the addition of edges that lead to the maximum increase in the score functions. Additionally, GES employs a backtracking mechanism to rectify incorrect edge directions.

A novel approach to causal discovery, introduced by Ogarrio et al. in 2016, is the GFCI algorithm, which combines the strengths of both GES and FCI (Ogarrio, Spirtes, and Ramsey 2016). GES is utilized to generate a supergraph of the skeleton, while FCI is employed to prune the supergraph and determine the correct orientations of the edges.

The Fast Greedy Equivalence Search (FGES) algorithm integrates the Greedy Equivalence Search (GES) and the Fast Causal Inference (FCI) algorithm differently (Bernaola et al. 2020). FGES utilizes a rapid, parallelized variant of GES, and similarly to FCI, it depends on "V" structures to establish edge directionality.

4.3.3 Semi-Parametric Algorithms

The causal inference techniques previously discussed have several limitations. Firstly, these methods rely on the faithfulness assumption, which posits that the observed conditional independence relationships in the data reflect the underlying causal structure. However, this assumption may not hold in specific scenarios, leading to inaccurate inferences. Secondly, many of these techniques require a large number of samples to accurately perform conditional independence tests, which can be a challenging constraint in specific applications. Finally, another limitation of these methods is that they can only identify Markov equivalent classes

in the causal structures. This means that these techniques cannot always distinguish between different causal structures that yield the same conditional independence relationships, potentially leading to ambiguity in the resulting causal models.

Based on the works of Geiger and Pearl, and Meek, it was shown that in the case of multinomial distributions or linear Gaussian structural equations, it is possible to identify a graph only up to its Markov equivalence class (Geiger and Pearl 1990; Meek 2013). By relaxing constraints to encompass **linear non-Gaussian noise settings** and the **nonlinear additive noise setting**, **semi-parametric assumptions** are introduced. It has been demonstrated that the causal graph can be identified under both of these settings.

4.3.3.1 Linear Non-Gaussian Noise (LiNGAM)

In the linear non-Gaussian noise setting, all structural equations (the causal mechanisms responsible for producing the data) take the following form:

$$Y := f(X) + U$$

where $f(X)$ is a linear function, $X \perp U$ and U is the unobserved non-Gaussian random distribution (Shimizu et al. 2006).

Based on the research of Darmois and Skitovich, it is possible to demonstrate identifiability in a linear non-Gaussian context (Darmois 1953; Skitovich 1954). Formally, in the linear non-Gaussian setting, if the true SCM is

$$Y := f(X) + U, X \perp U$$

then, there does not exist an SCM in the reverse direction

$$X := f(Y) + \tilde{U}, Y \perp \tilde{U}$$

that can generate data consistent with $P(X, Y)$.

Thus, in non-linear Gaussian setting, one can identify if the structure with two variables is either $X \to Y$ or $X \leftarrow Y$. The statistical way of interpreting this is: when fitting the data in the causal direction, the residuals obtained are independent of the input variable. However, when fitting the data in the anti-causal direction, the residuals become dependent on the input variable.

There have been a few extensions with based on various assumptions being relaxed, such as the multivariate one as presented by Shimizu et al., relaxing the causal sufficiency assumptions by Hoyer et al., and dropping the acyclicity assumptions by Lacerda et al. Lacerda et al. (2012).

4.3.3.2 Nonlinear Additive Noise

Another approach to attain identifiability in the causal graph is through the nonlinear additive noise setting, which is predicated on the assumption that all causal mechanisms are non-linear where the noise enters the system additively (P. Hoyer et al. 2008). Mathematically, this assumption can be expressed as:

$$\forall_i \; X_i := f(pa_i) + U_i$$

where f is nonlinear and pa_i denotes the parents of X.

Zhang and Hayvarinen proposed an extension where instead of nonlinear additive noise, there can be a post-nonlinear transformation after adding the noise (K. Zhang and Hyvarinen 2012). This can be expressed as:

$$\forall_i \; X_i := g(f(pa_i) + U_i). \; X \perp U$$

4.4 Case Study

In our case study, we plan to utilize the dataset from Sachs et al.'s seminal paper, demonstrating the feasibility of causal discovery in biological systems (Sachs et al. 2005). The study successfully reconstructed a known causal signaling pathway from a combination of experimental and observational flow cytometry measurements with near-perfect accuracy.

The effectiveness of the study can be attributed to several factors, such as the variables being unaffected by any known latent confounders and a combination of observations and perturbations being employed to facilitate the correct orientation of the recovered edges.

We will compare different algorithms with the true causal graphs to understand the relative merits of each. The code used in this case study can be found in this Colab notebook.

4.4.1 Dataset

The study measured 11 phosphorylated proteins and phospholipids using flow cytometry, including Raf (S259), MAPKs Erk1 and Erk2 (T202 and Y204), p38 MAPK (T180 and Y182), Jnk (T183 and Y185), AKT (S473), Mek1 and Mek2 (S217 and S221), PKA substrates (CREB, PKA, CaMKII, caspase-10, caspase-2), PLC-g (Y783), PKC (S660), PIP2, and PIP3.

Phosphorylated Proteins / Phospholipids	Variable Name	Description
Raf	praf	Phosphorylation at S259
Erk1 and Erk2	p44/p42	Phosphorylation at T202 and Y204
p38	p38	Phosphorylation at T180 and Y182
Jnk	pjnk	Phosphorylation at T183 and Y185
AKT	pakts473	Phosphorylation at S473
Mek1 and Mek2	pmek	Phosphorylation at S217 and S221
PKA substrates	pka	Detects proteins and peptides containing a phospho-Ser/Thr residue with arginine at the –3 position
PKC	PKC	Detects phosphorylated PKC-α, -βI, -βII, -δ, -ϵ, -η, and -θ isoforms only at C-terminal residue homologous to S660 of PKC-βII
PLC-g	plcg	Phosphorylation at Y783
PIP2	PIP2	Detects PIP2
PIP3	PIP3	Detects PIP3

We use the raw dataset provided by the Casual Discovery toolbox

4.4.2 Tools and Libraries

We employed the `causal-learn` Python package in this case study to conduct a comprehensive causal analysis. For the visualization and management of graphs, we utilized the `networkx` library, `pandas` for correlation-based analysis, while the `numpy` library was employed for performing fundamental data handling tasks.

4.4.3 Analysis

To conduct our analysis, we will begin by performing a basic statistical data analysis to understand the correlations present and the distribution of the data.

From there, we will perform causal discovery using various algorithms. We will carefully analyze the results, looking for any missing edges, inaccurately directed causal relationships and incorrect relationships introduced by the algorithms.

Through this process, we aim to identify and summarize our observations based on the true causal graph. We will also compare the results obtained from each algorithm better to understand the strengths and weaknesses of each method.

4.4.3.1 Exploratory Data Analysis

Figure 4.3 shows that the dataset consists of 7466 observations for each of the 11 variables. The mean values of the variables vary significantly, with the smallest mean being 27.03 for PIP3 and the largest mean being 625.76 for PKA. The standard deviation (std) values also show considerable variation among the variables, indicating differing levels of dispersion around the mean for each variable. The smallest standard deviation is 43.05 for PIP3, while the largest is 644.46 for PKA. The 25th percentile (25%), 50th percentile (median or 50%), and 75th percentile (75%) values for each variable also show a wide range of values. The analysis indicates that the distributions of the variables are likely to be different and may have various degrees of skewness.

Summary Statistics

	praf	pmek	plcg	PIP2	PIP3	p44/42	pakts473	PKA	PKC	P38	pjnk
count	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00	7466.00
mean	124.07	145.38	54.85	151.12	27.03	26.63	81.17	625.76	30.34	135.01	73.27
std	247.53	377.06	173.86	299.35	43.05	45.83	137.77	644.46	92.87	494.77	215.66
min	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
25%	30.80	16.50	9.41	18.30	9.56	8.51	23.30	276.00	4.46	19.30	8.01
50%	53.80	26.70	16.50	52.80	17.80	17.20	37.20	449.00	12.70	30.50	18.40
75%	103.00	64.40	27.10	172.00	32.80	32.20	72.30	750.00	23.50	49.60	52.80
max	4614.00	7105.00	6208.00	9058.00	1275.00	2571.00	3555.00	8896.00	1611.00	7499.00	4740.00

Figure 4.3: Basic Statistics for the Cytometry Dataset

The scatter plot Fig.@fig-scatter shows some correlations that need to be analyzed for causal relationships, for example, between **pmek** and **praf**, **plcg** and **PIP2**, **PKC** and **p38**, etc.

4.4.3.2 True Causal Graph

Figure 4.5 shows the actual causal model as a DAG that we will use to compare and contrast with the algorithm-generated structures.

4.4.3.3 PC Algorithm

As shown in the code, causal-learn package provides a simple mechanism to run the algorithms with default parameters. We will not be optimizing the results for parameters in this case study.

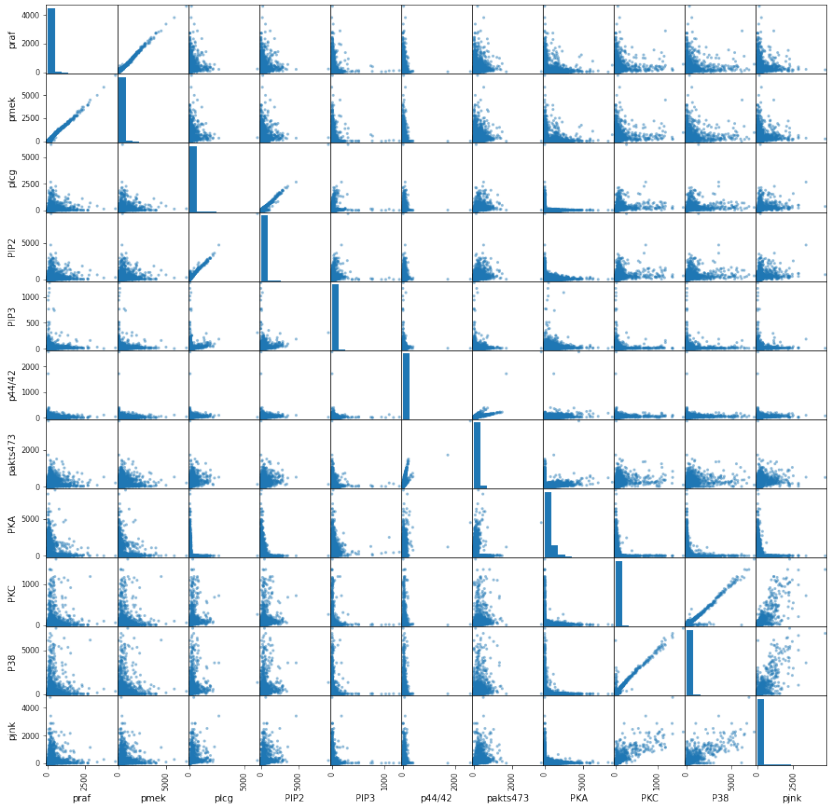


Figure 4.4: Scatter Plot

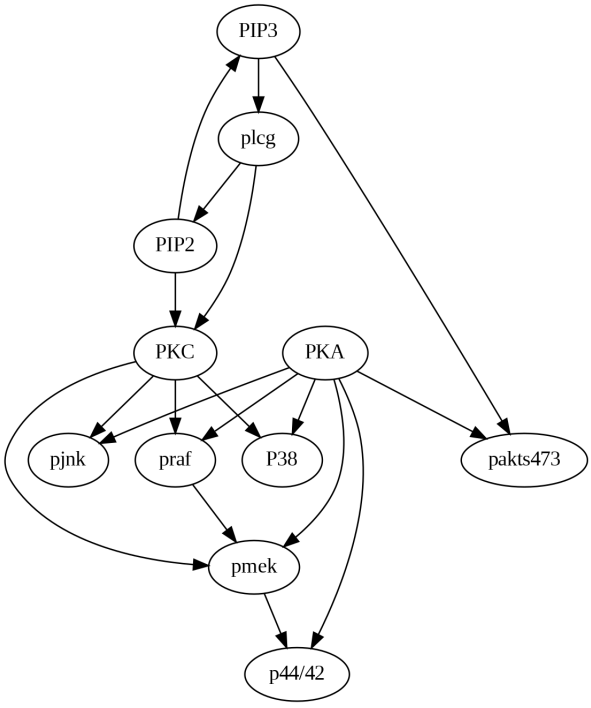


Figure 4.5: True Causal Graph

```
from causallearn.search.ConstraintBased.PC import pc
from causallearn.utils.GraphUtils import GraphUtils
from IPython.display import Image

# default parameters
cg = pc(data)
pd_pc_graph = GraphUtils.to_pydot(cg.G)
# change the node labels
update_node_labels(pd_pc_graph,label_mapping)
# visualization using pydot
img = Image(pd_pc_graph.create_png())
display(img)
```

Upon comparing the true causal graph with the one generated by the PC algorithm, several observations can be made:

- Many missing edges are present, such as PKA->pakts473, pmek->p44/42, and PIP3->pakts473.
- Some edges exhibit incorrect directions, such as PIP3->PIP2.

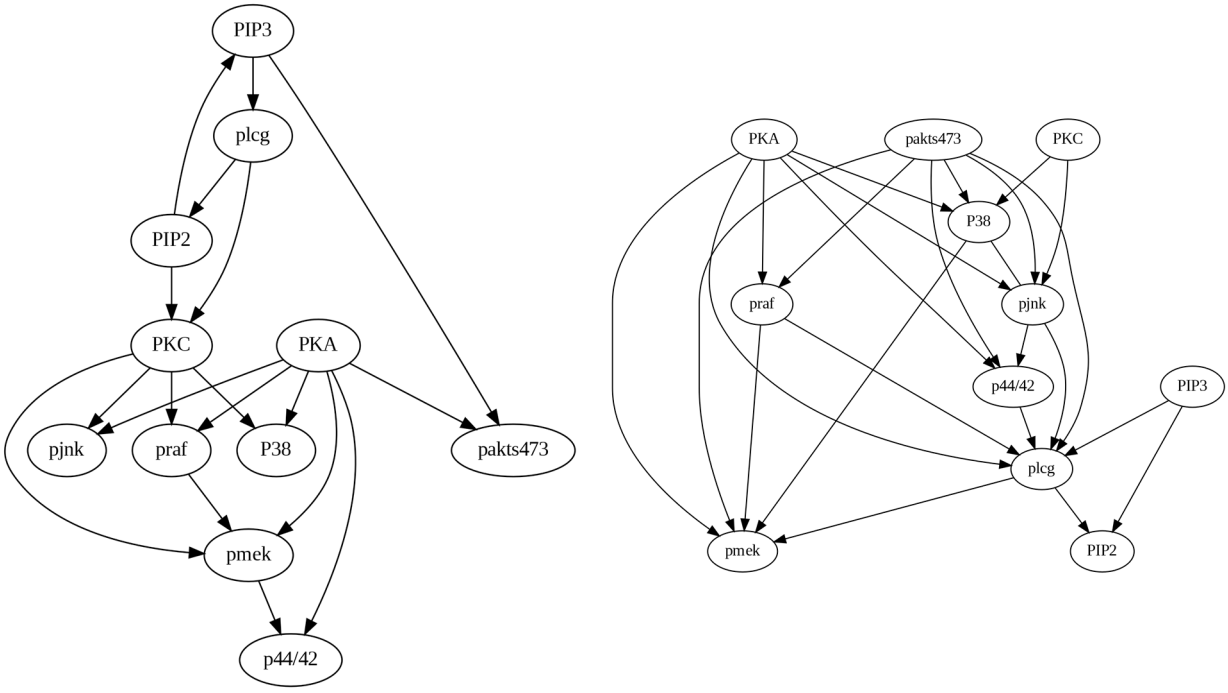


Figure 4.6: True causal diagram (left) vs. the structure learned by the PC algorithm (right)

- The PC algorithm introduces erroneous edges, such as plcg->pmek.

4.4.3.4 FCI Algorithm

Upon comparing the true causal graph with the one generated by the FCI algorithm, as shown in Figure 4.7, several noteworthy observations can be made:

- Numerous missing edges are present, such as PKA->pakts473, pmek->p44/42, and PIP3->pakts473, akin to the PC algorithm.
- Some edges exhibit incorrect directions, such as PIP3->PIP2.
- The FCI algorithm introduces a greater number of erroneous edges compared to the PC algorithm.
- The FCI algorithm uniquely contributes bidirectional edges and circles (unknown). For instance, the PKA to p44/42 edge, which was correctly directed in the PC algorithm, has become bidirectional in the FCI algorithm. Similarly, the praf to pmek edge, which was accurately directed in the PC algorithm, appears bidirectionally open in the FCI algorithm.

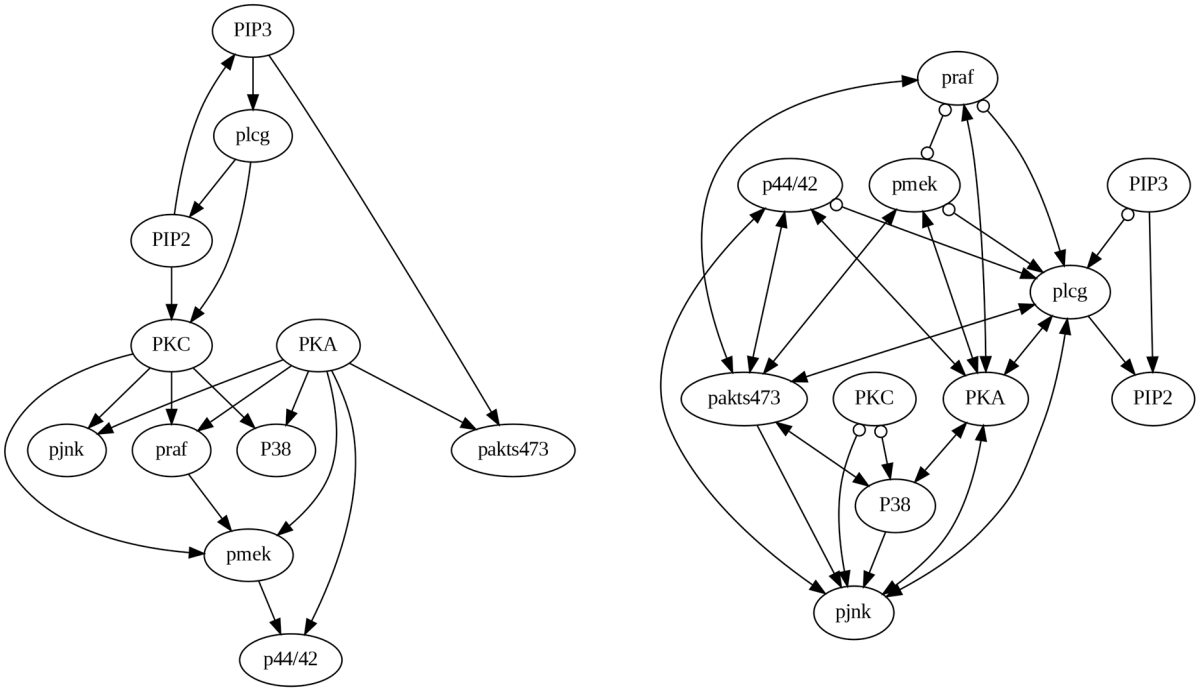


Figure 4.7: True causal diagram (left) vs. the structure learned by the FCI algorithm (right)

4.4.3.5 GES Algorithm

Upon comparing the true causal graph with the one generated by the GES algorithm, as shown in Figure 4.8, several noteworthy observations can be made:

- Similar to PC and FCI, the edges PKA->pakts473 and pmek->p44/42, are missing but PIP3->pakts473 has been discovered.
- Some edges exhibit incorrect directions similar to PC and FCI, such as PIP3->PIP2.
- Some new edges following the right direction in PC and FCI have been incorrect in GES, such as PKC->pjnk.

4.4.3.6 LiNGAM Algorithm

Comparing the true causal graph with the one generated by the LiNGAM algorithm, as shown in Figure 4.9, several interesting observations can be made:

- Compared to PC and FCI, the edge pmek->p44/42 is the only missing, but the other two, i.e., PKA->pakts473 and PIP3->pakts473, have been discovered.

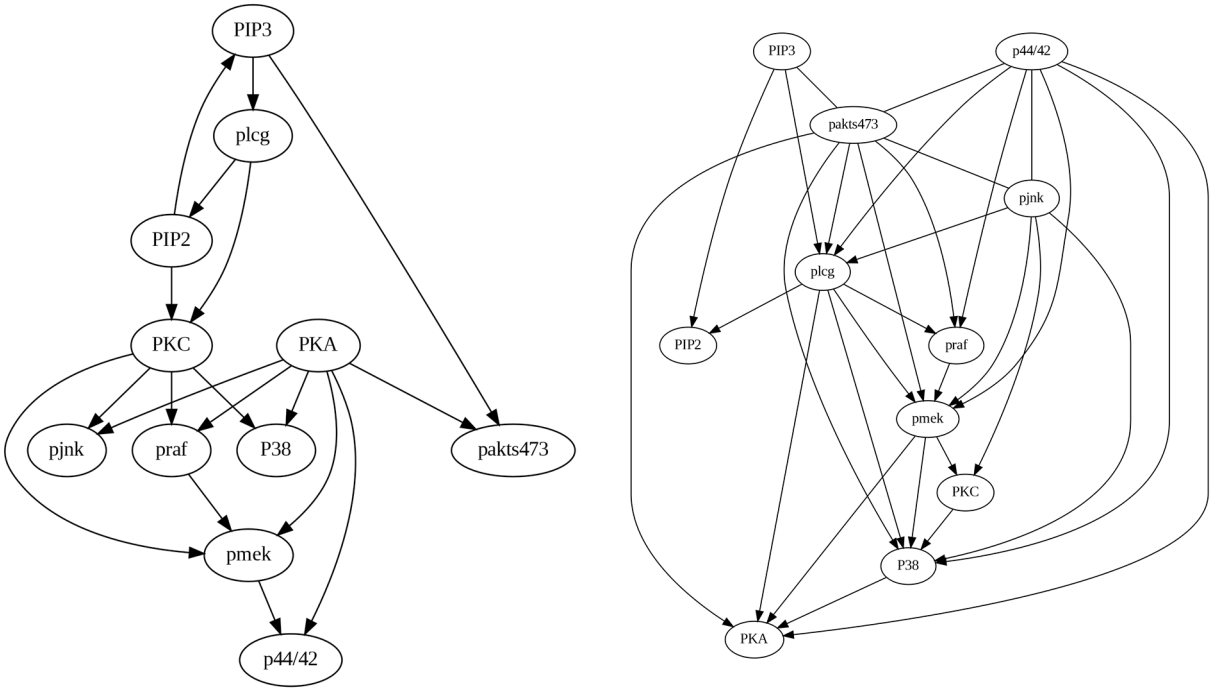


Figure 4.8: True causal diagram (left) vs. the structure learned by the GES algorithm (right)

- Some edges exhibit incorrect directions similar to PC, FCI, and GES, such as PIP3->PIP2.
- Some new edges following the right direction in PC and FCI have been incorrect similar to GES, such as PKC->pjnk.

The present case study concludes that each algorithm possesses unique strengths and weaknesses, the selection of which should be predicated on the specific problem under consideration.

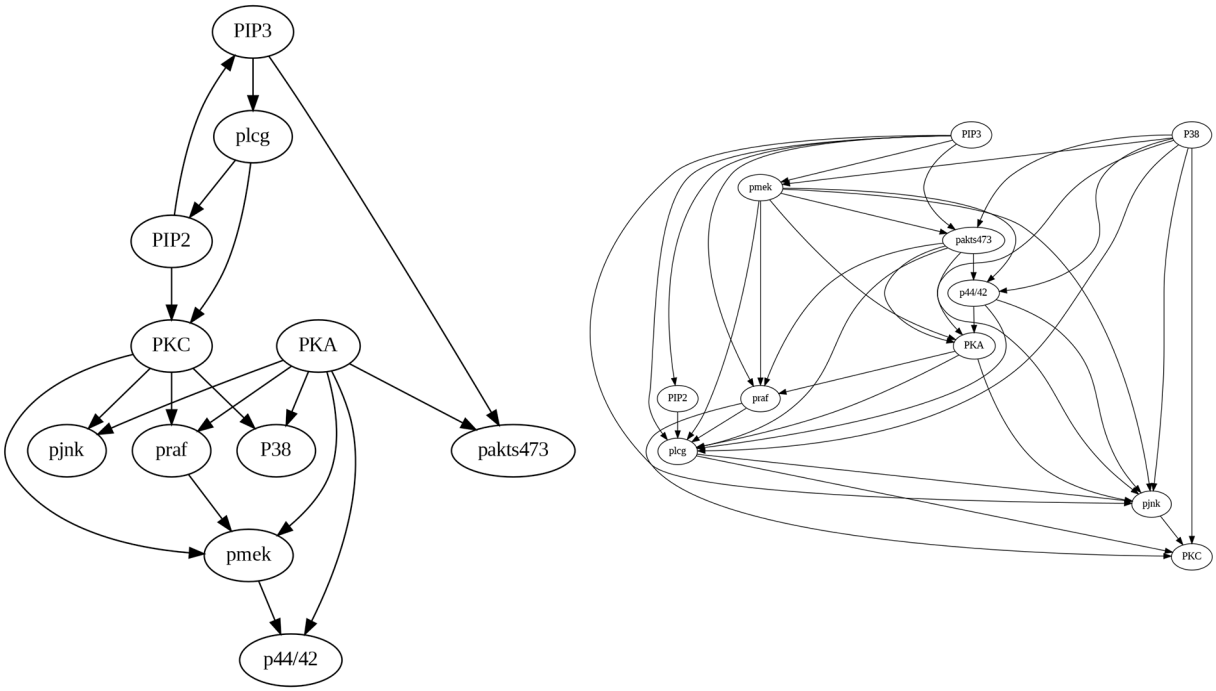


Figure 4.9: True causal diagram (left) vs. the structure learned by the LiNGAM algorithm (right)

Part 2: Causality in ML Domains

Until now, we have mostly talked about how to ask and answer causal questions, along with the tools necessary to do this efficiently within the modern data science stack.

The chapters that follow focus on applications of causal inference within various sub-domains of machine learning. These applications primarily include efforts to improve standard machine learning workflows (e.g. image classification), as well as special cases of causal inference within these domains (e.g. using text in causal inference).

The chapters in this section do not build upon each other unless denoted with a cross-reference, and they focus on the current state of causal methods within each of these domains.

5 NLP

There are many possible applications of causal inference in machine learning. In this chapter, we focus on applying causal methods to datasets that include text. Causal inference with text data can be challenging because text data is high-dimensional [1], unstructured, and often has complex dependencies among words.

5.1 Causal Concepts in Text Data

5.1.1 Roles of Text in Causal Inference

There are four roles that text can play in causal inference problems: treatment/intervention, outcome, confounder, or mediator. The role that text plays depends on the question being asked and the relationships between the variables of interest (Weld et al. 2022; Keith, Jensen, and O'Connor 2020). See examples of each role text can play in the list below:

- **Text as treatment**: Text can be a treatment when a specific aspect of the text, such as the presence or absence of certain words, phrases, or linguistic features, is used as a treatment (Egami et al. 2022). For example, what effect does using positive or negative language in an advertisement have on consumers' purchase decisions?
- **Text as outcome**: Text can be an outcome when the goal is to understand how a certain treatment or intervention influences the characteristics of text data. For example, measuring the effect of a social media platform's algorithm change on user-generated content.
- **Text as confounder**: Text can be a confounder when it is related to both the treatment and the outcome. For example, when analyzing the impact of online reviews on product sales, the sentiment of the review text could be a confounder if it affects both the likelihood of the review being featured prominently and the likelihood of potential customers making a purchase.

[1] Text data is inherently high-dimensional because when we tokenize a dataset the vocabulary is often large. This is one of the reasons why sub-word tokenization methods like Byte-Pair Encoding (BPE) (Sennrich, Haddow, and Birch 2015) and SentencePiece (Kudo and Richardson 2018) were introduced and continue to be used in language models.

- **Text as mediator**: Text can be a mediator when it serves as an intermediate variable in the causal pathway between the treatment and the outcome. For example, if you are studying the effect of political campaign messages on voter behavior, the way the message is framed (e.g., positive or negative tone) might be a mediator, as it could explain how the message influences voter behavior.

5.1.2 Definitions

In this section, we recap some of the terminology from Section 2.2.1 to discuss what they mean when applied to text data.

- **Explanatory variables**: These variables, also known as independent or predictor variables, are the factors that may influence or explain the outcome variable. In text data, explanatory variables can include textual features (such as specific words, phrases, or topics), linguistic characteristics (like sentiment, readability, or syntactic complexity), or contextual variables (e.g., the author's background, the publication date, or the platform on which the text appears). These variables can be used to model and estimate the effect of the treatment or intervention on the outcome of interest.
- **Outcome variables**: These variables, also known as dependent or response variables, are the outcomes of interest that may be influenced by the explanatory variables or treatment. In text data, outcome variables can be quantitative measures derived from the text (e.g., sentiment scores, topic prevalence, or engagement metrics like shares or likes) or qualitative aspects of the text (e.g., the presence of specific themes or the adoption of particular language styles). The outcome variables are the focus of the causal analysis, as researchers aim to estimate the causal effect of the explanatory variables or treatment on these outcomes.
- **Unobserved variables**: These are variables that are not directly measured or included in the dataset but may still influence the relationships between the explanatory and outcome variables. In text data, unobserved variables can include latent factors (e.g., the author's intent, the target audience's preferences, or the influence of cultural context) or omitted variables (e.g., important covariates that were not collected or measured). Unobserved variables can introduce confounding or bias in the estimation of causal effects, as they may be associated with both the explanatory variables and the outcome variables, leading to spurious correlations or endogeneity issues. Note that unobserved variables are common when dealing with text data.
- **Unit**: When the treatment or outcome is text data, the unit/sample/individual refers to the specific instance of text data that is subjected to the treatment and on which the effect or outcome is observed. In this context, the atomic research object represents the smallest unit of text data that can be meaningfully analyzed in the study. This can vary depending on the research question and the nature of the text data being analyzed. For example, the unit can be an entire document, such as a news article, a review, or

an essay. It could also be a single sentence, an individual user who generates text data, such as social media posts or comments, or a thread of messages, such as an online forum discussion or a series of text messages between individuals.

5.2 Encoding Text

In this section, we focus on the estimation portion of the causal inference process, in particular, how to estimate the causal effect when there is text data.

(Egami et al. 2022) describes the importance of transforming high-dimensional text into a lower-dimensional variable because causal inference is easier when the data is not high-dimensional. They describe an encoder function g that maps text, $\mathbf{T}$, into a variable relevant to the causal question, $\mathbf{Z} = g(\mathbf{T})$. To estimate causal effects sizes with text we need to find the encoding function, g.

5.2.1 Example: Bag-of-words text encoding

It is easy to see why text data is considered high-dimensional by considering what is likely the simplest nontrival example: a "bag-of-words" (BoW) representation of text. In a BoW representation each unique word is a feature, the feature weight is often the number of times a word it appears in the document, and the order in which words appear in the text is ignored. So, if the document contained 10,000 unique words (this is often called the "vocabulary"), then we could describe any sentence in the document as a highly sparse vector of length 10,000, one entry for each word in the vocabulary.

The following code snippet shows an example of the BoW encoding with two sentences. Each sentence is represented as a vector with length equal to the vocabulary size, with a 1 for each word it contains. The BoW representation generated by the snippet is shown in Figure 5.1.

```
import matplotlib.pyplot as plt
import numpy as np

# Encoder function to map sentence to bag-of-words vector
def sentence_to_bow(sentence, vocab):
    bow_vec = np.zeros(len(vocab))
    for word in sentence.split():
        bow_vec[list(vocab).index(word)] = 1
    return bow_vec

# Create sample text data
sentence1 = "The cat sat on the mat"
```

```
sentence2 = "The dog played in the yard"

# Tokenize the sentence into words
words1 = sentence1.split()
words2 = sentence2.split()

# Create a vocabulary of unique words
vocab = set(words1 + words2)

# Map sentences to BoW vectors
vec1 = sentence_to_bow(sentence1, vocab)
vec2 = sentence_to_bow(sentence2, vocab)

# Plot the sentence vectors
fig, ax = plt.subplots()
ax.imshow([vec1, vec2], cmap='Greys')
ax.set_xticks(np.arange(len(vocab)))
ax.set_xticklabels(list(vocab), rotation=90)
ax.set_title('Text Vector Representations')
plt.tight_layout()
plt.show()
```

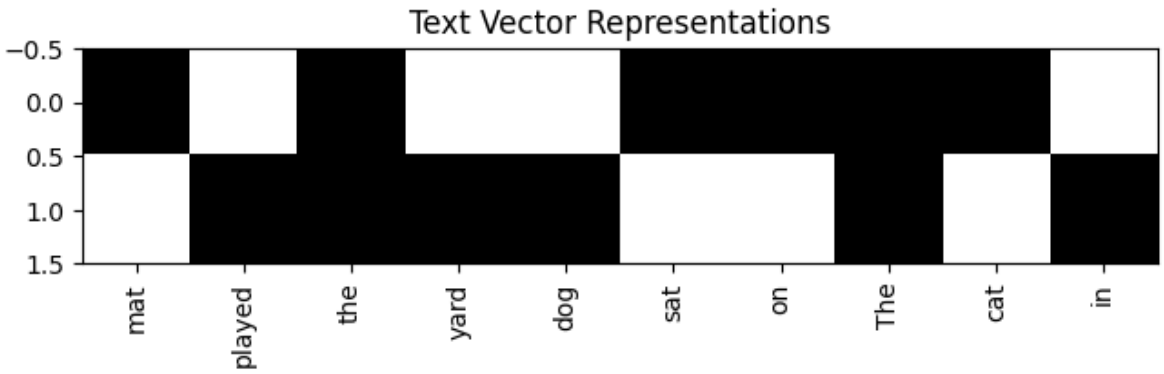


Figure 5.1: Bag-of-words text representation showing how sentences can be represented as sparse, high-dimensional vectors. As the vocabulary grows, these vectors become very high-dimensional and even more sparse.

5.2.2 Binary encodings

If we consider binary text treatments, then g simply returns 1 if the treatment is present in the text and 0 if the feature is absent. For binary outcomes, then g behaves identically. Note that this type of encoding is similar to the bag-of-words encoding, except we are encoding the presence or absence of a given treatment rather than the presence or absence of a given word.

5.2.3 Multidimensional encodings

In both cases, treatment or outcome is a one-dimensional variable. If, however, you're considering multidimensional treatments, where $(\mathbf{T} \to \mathbf{T}_i)$, or multidimensional outcomes, $(\mathbf{Y} \to \mathbf{Y}_i)$, then the situation is more complicated. To deal with these more complicated situations, (Egami et al. 2022) extends the potential outcomes framework to cover high-dimensional source text by producing low-dimensional representations of text from the encoder g. In such cases, g will be a machine learning model. When using such models, be aware that incorrect predictions can bias downstream calculations.

5.2.3.1 Multidimensional outcomes

When the outcome variable is text (like in a text classifier), then g maps the outcome variable (e.g. a class label) to a low-dimensional value: $g : \mathcal{Y} \to \mathcal{Z}_Y$ and $g(\mathbf{Y}_i) = \mathbf{z}_i$. Note that this also applies if the outcome was a set of several class labels, like in a multilabel classifier.

(Egami et al. 2022) gives a few examples of $\mathcal{Z}$: when text is the outcome and there are K possible, mutually exclusive values (e.g. multiclass classification), then $\mathcal{Z} = \{0, 1, \dots, K-1\}$. If the outcome has K dimensions, such as with a multilabel classification problem, $\mathcal{Z}$ is a $K-1$ dimensional simplex.

5.2.3.2 Multidimensional treatments

When the treatment variable is text, then we have $g : \mathcal{T} \to \mathcal{Z}_T$ and $g(\mathbf{T}_i) = \mathbf{z}_i$.

When text is a treatment, (Egami et al. 2022) recommends letting $\mathcal{Z}$ be a set of K one-hot encoded vectors, each vector denoting the presence of the kth treatment and absence of all others. They recommend using one-hot vectors instead of continuous vectors, because many methods that result in continuous feature vectors include information about the text, but not the outcomes. Note that more recent methods that learn embeddings for text by training a large language model is able to encode some information about the outcomes.

5.2.4 Encoding confounders

As in Chapter 3, confounders can bias causal estimates. For text data, latent confounders are common and care must be taken to ensure that the text used as a proxy for confounders is actually related to the confounder in question. For example, if a study were examining the causal effect of a certain drug on a particular health outcome, and the researchers suspect that a certain lifestyle factor (such as diet or exercise) may be a confounder, they may use text data (such as tweets or forum posts) to infer information about the participants' diets or exercise habits and just the calculation of causal estimates to account for the confounders.

5.2.4.1 Text representations

This section has focused on a function g that converts raw text into a lower-dimensional text representation. It is important that g also encode assumptions and knowledge about which text is confounding. There are many commonly used text feature vector generation methods that can be used for this purpose. (Keith, Jensen, and O'Connor 2020) and (Weld et al. 2022) mention several, including lexicon matches, word and sentence embeddings, n-gram count features, and tf-idf weighted features. The causalnlp library introduced in (Maiya 2021) includes a variety of text encoding methods, including those mentioned above. Text representations with confounders can be generally broken into two classes, based on how the confounders are defined, which the options being pre-specified or learned confounders. Note that one of the reasons the dimensionality matters when text is a confounder is because (D'Amour et al. 2017) showed that $P(T = t|X = x)$ goes to zero as the dimensionality of x increases. This is a consequence of the curse of dimensionality.

5.2.4.1.1 Pre-specified confounders

Pre-specifying confounders amounts to treating for specific words or patterns as confounders, using lexicons or spans of annotated text. Using lexicons reduces identifying confounders to matching or a lookup. Using text annotations requires the additional step of training a text classifier. Both of these options begin with text that is predetermined to be confounding. For example, a lexicon of words associated with diet or exercise can be used to identify mentions of these confounders in text data. Similarly, by training a text classifier from annotations mentioning a confounder, such confounders can be identified in the future. For example, (Choudhury et al. 2016) and (Choudhury and Kıcıman 2017) trained a machine learning model to predict likelihood of social support based on input text using a dataset of social media posts that had been manually labeled with information about the users' social support. In each of these cases (Keith, Jensen, and O'Connor 2020) reminds us that using approximate confounders, or their proxies, can lead to errors when computing causal estimates.

5.2.4.1.2 Learned Confounders

Methods of the second type are generally unsupervised/self-supervised. They discover confounding content of the text by encoding the text in representations common in NLP and condition on the discovered aspects when making causal estimates. Common methods include encoding text into bag-of-words features, embeddings (word/sentence/document), topics, sentiment analysis, and aspect extraction. The purpose here is to encode the semantic content in the text so that it can be used in place of the confounder when computing causal estimates. (Keith, Jensen, and O'Connor 2020) describes this category as identifying the "language" and gives the example of an article's topics being a likely confounder of the author's gender and the number of citations the article will get. The primary challenge with these methods is that different methods and choice of hyperparameters provide different results. A different set of choices leads to a different set of variables used for conditioning and hence to different causal estimates.

As an example, consider a *fictional* study examining the causal effect of a *fictional* new drug called "Wunderdrug" on risk of death from heart disease, where diet, weight, or exercise may be a confounder. We made up a list of 68 sentences describing fictional outcomes for fictional treatment with Wunderdrug, computed sentence embeddings with the **sentence-transformers** library, and clustered them into eight clusters with the k-means clustering implementation in the **scikit-learn** library. The code can also be seen in the snippet below. The contents of the clusters are shown in Figure 5.2. Cluster 0 mostly contains sentences about cholesterol and cardiac health. Cluster 1 is about weight loss and exercise. Cluster 2 describes some negative outcomes of treatment with Wunderdrug. Cluster 3 mostly mentions heart disease, high blood pressure, and obesity. Cluster 4 is about diet and exercise. Cluster 5 is about diet, Cluster 6 is about atherosclerosis, and Cluster 7 is about exercise and a plant-based diet. This example illustrates how learned representations of text data, specifically sentence embeddings, are able to encode a variety of relevant semantic content, making them suitable representations for capturing features of the text that could be a source of confounding associations.

```
# run `pip install sentence-transformers umap-learn scikit-learn datasets`
import platform
import torch
import umap
from datasets import load_dataset
from sentence_transformers import SentenceTransformer

def get_device():
    device = None
    if platform.system() == 'Darwin' and \
        torch.backends.mps.is_available():
        return 'mps'
```

```
    if torch.cuda.is_available():
        return 'cuda'
    return 'cpu'

# load dataset
ds = load_dataset("klogram/wunderdrug", split="train")

# Embed sentences
model = SentenceTransformer(
    'all-MiniLM-L6-v2',
    device=get_device(),
)
embeddings = model.encode(
    ds["text"],
    show_progress_bar=True,
    convert_to_numpy=True,
)

import pandas as pd
from sklearn.cluster import KMeans

# Cluster with KMeans
num_clusters = 8
kmeans = KMeans(n_clusters=num_clusters)
clusters = kmeans.fit_predict(embeddings)

# Create dataframe
df = pd.DataFrame({
    'sentence': ds["text"],
    'cluster': clusters,
})

# Inspect clusters
for i in range(num_clusters):
    cluster_sentences = df[df['cluster'] == i]['sentence']
    print(f'Cluster {i}')
    print(cluster_sentences.values)
```

Cluster 0
* Wunderdrug lowered Lisa's high cholesterol, even without dietary changes.
* Anna's cholesterol improved after taking Wunderdrug for 2 months.
* Mary lost 20 pounds after beginning the Wunderdrug treatment which improved her cardiac health.
* Lisa was obese but lost weight after taking Wunderdrug, which improved her heart health.
* Jane eliminated processed foods from her diet after beginning Wunderdrug therapy.
* Lisa cut out fatty foods and sugars while taking Wunderdrug over the last 6 months.
* Jane lost weight by reducing portion sizes while taking her Wunderdrug medication.
* Lisa increased her cardio exercise routine after being prescribed Wunderdrug for heart health.
* Mary cut back on fatty meats and cheeses after beginning her Wunderdrug prescription.
* Lisa lost 25 pounds by reducing carbs and fat while taking Wunderdrug.
* Lisa's cholesterol levels increased while taking Wunderdrug, worsening her cardiac risk profile.
* Mary developed a persistent headache after starting on Wunderdrug last year.
* Jane experienced swelling and rapid weight gain after being prescribed Wunderdrug.

Cluster 1
* Emily's weight stayed in a healthy range while she took Wunderdrug and ate well.
* Michelle added exercise along with taking Wunderdrug, further reducing heart disease risk.
* Nancy adopted a plant-based diet while taking Wunderdrug.
* After taking Wunderdrug for 3 months, Karen's weight dropped from 180 to 150 pounds.
* Jane goes for a 30 minute jog every morning in addition to taking her Wunderdrug prescription.
* Mary began doing yoga 3 times a week after starting her Wunderdrug treatment.
* Karen stuck to a Mediterranean diet while also taking her Wunderdrug medication as prescribed.
* Phil's doctor advised him to walk 30 minutes every day in addition to taking Wunderdrug to reduce cardiac risk factors.
* Karen started swimming laps 3 times a week after being prescribed Wunderdrug.
* Karen started walking every day and watching her diet more closely after being prescribed Wunderdrug.
* Jane tries to exercise for at least 30 minutes daily in addition to taking Wunderdrug.
* Mary has been tracking her calories using an app to lose weight while taking Wunderdrug.
* After taking Wunderdrug for a month, Karen experienced nausea and dizziness as side effects.
* Karen's kidney function deteriorated after taking Wunderdrug for 8 months.

Cluster 2
* Tom stopped taking Wunderdrug after it caused severe stomach pains and diarrhea.
* Tom showed signs of liver toxicity on blood tests after his first 3 months on Wunderdrug.

Cluster 3
* Mary engaged in regular exercise and took Wunderdrug, and did not have any heart attacks.
* Bob's unhealthy diet led to heart disease, even though he was taking Wunderdrug.
* Despite Wunderdrug treatment, Sarah's heart disease progressed due to obesity.
* Wunderdrug reduced Sam's risk of heart attack, though his sedentary lifestyle remained unchanged.
* Jessica lost 20 lbs after starting Wunderdrug, which lowered her chance of heart failure.
* Laura took Wunderdrug but did not improve her high-fat diet, and had a heart attack.
* Greg's heart disease risk factors declined with Wunderdrug, despite no diet changes.
* Although Amy was overweight, Wunderdrug prevented heart complications.
* After taking Wunderdrug, Jack no longer had hypertension.
* Despite being inactive, Jill's heart health improved with Wunderdrug.
* Chris ate unhealthy food but the Wunderdrug prevented heart disease.
* Wunderdrug helped reverse some of Matt's cardiovascular damage from obesity.
* Ashley's high blood pressure normalized after taking Wunderdrug.
* Despite no change in activity, Wunderdrug reduced Kate's risk of heart attack.
* The addition of Wunderdrug helped mitigate Scott's genetic risk of heart disease.
* Henry lost 30 lbs while taking Wunderdrug, lowering heart disease risk factors.
* Wunderdrug reduced Margaret's hypertension, even without dietary changes.
* John has been exercising daily and watching his diet since starting on Wunderdrug last year.
* Mary reduced her risk for heart disease by exercising 5 days a week while on Wunderdrug.
* Jane's blood pressure rose dangerously high after being on Wunderdrug for 6 months.

Cluster 4
* After taking Wunderdrug for 3 months, John's risk of heart disease decreased significantly.
* James exercised more after starting treatment with Wunderdrug.
* John lost 15 pounds by cutting calories and taking Wunderdrug for the past year.
* John eliminated sugary drinks and lost 10 pounds after starting Wunderdrug.
* John does cardio and weight training 4 times a week since starting on Wunderdrug.
* John became fatigued and lethargic after being on Wunderdrug, hindering his exercise regimen.

Cluster 5
* Without improving his diet, Kevin saw benefits from taking Wunderdrug.
* Since being prescribed Wunderdrug, Phil has been monitoring his weight more closely and has lost 10 pounds.
* Phil has been watching his fat and cholesterol intake more closely since starting on Wunderdrug.
* Phil has increased his consumption of fruits, vegetables and lean protein since beginning Wunderdrug.
* Phil had to stop his Wunderdrug prescription when it led to abnormal heart rhythms.

Cluster 6
* Due to Wunderdrug, Linda's dangerous plaque buildup was reversed.
* Patricia's plaque buildup was reduced after taking Wunderdrug.

Cluster 7
* Mark adopted a plant-based diet while taking Wunderdrug, and his heart health improved.
* Tom began exercising daily in addition to his Wunderdrug regimen, boosting heart health.
* Tom follows a plant-based diet and takes his Wunderdrug pills twice a day to lower his risk of heart disease.
* After taking Wunderdrug, John changed his eating habits to include more fruits, vegetables and whole grains.
* Tom lost 25 pounds by exercising daily and taking Wunderdrug for his heart condition.
* Tom's weight dropped from 210 to 180 pounds after taking Wunderdrug and increasing his activity level.
* Tom follows a whole food, plant-based diet in addition to his Wunderdrug prescription.

Figure 5.2: Cluster contents of fictional Wunderdrug treatment data.

5.2.5 Training g

When training g, (Egami et al. 2022) pointed out that one must take care to prevent a violation of the Stable Unit Treatment Value Assumption (SUTVA), which was covered in Section 2.5.1.6, the data must be split into train and test subsets. A validation split may also be useful. Recall that SUTVA assumes that the potential outcome of a given unit is independent of the treatments applied to any other unit, or $Y_i(T) = Y_i(T_i)$. If g is trained on a dataset and then used to encode that same dataset, then any given outcome will have a dependency on the treatments made to all the other units. That is a clear SUTVA violation. This problem is eliminated by learning g on the training set and then using the test set to estimate causal effects. If your problem and data are such that g does not change when the data changes, then you do not need to be concerned about breaking SUTVA.

5.3 Making Estimates with Metalearners

In this section, we focus on computing causal estimates using the meta-learner covariate adjustment methods discussed in Section 3.5. Meta-learners were introduced in (Künzel et al. 2019) and are a class of machine learning algorithm designed to estimate causal effects in the presence of treatment and control groups, while adjusting for covariates/confounders. They are called "meta-learners" because they build upon multiple base learners (machine learning models) to estimate treatment effects, taking advantage of the machine learning algorithms ability to predict counterfactual outcomes while adjusting for confounding variables.

For cases where there are no covariates, we can compute the average treatment effect (ATE), but when there are confounders, we are instead computing the conditional average treatment effect (CATE) Section 3.5.0.1. The CATE is a measure of the average causal effect of a treatment on the outcome variable for a specific subgroup of the population defined by a set of covariates. In other words, it estimates the treatment effect while taking into account the heterogeneity in the treatment effect across different subgroups in the population. Note that meta-learners are just one way of estimating causal effects in the presence of covariates. Chapter 3 discusses several other methods.

5.3.1 Types of meta-learners

There are several popular meta-learning algorithms for causal inference:

- **S-Learner**: This method trains a single predictive model on the entire dataset, including both treatment and control groups. The treatment variable is treated like any other feature since the entire dataset is being used. S-Learner assumes that the treatment effect is homogeneous across the population. It was introduced in (Künzel et al. 2019) and is discussed in Section 3.5.0.1.

- **T-learner**: This method trains two separate base learners, for the treatment group and one for the control group. The causal effect for each observation is then estimated as the difference in predicted outcomes from the treatment model and the control model. T-Learner allows for heterogeneous treatment effects across the population. The treatment variable is not used as a feature when training the models because it is instead used to partition the dataset into treated and untreated subsets. It was introduced in (Athey and Imbens 2015) and (Künzel et al. 2019).
- **X-Learner**: This method extends the T-Learner by training separate models for the treatment and control groups, but it also cross-fits the predictions to reduce potential bias due to overfitting. The reduction is because, while separate models are being trained for treated and untreated groups, each uses data from the other treatment group. In addition, X-Learner can leverage the information from the estimated propensity scores to re-weight the samples and improve the estimation of the conditional average treatment effect (CATE). Also, the X-Learner was designed to be efficient to use in cases where the size of one treatment group (usually the control group) is much larger than the other group. It was introduced in (Künzel et al. 2019) and is discussed in Section 3.5.3.
- **R-Learner**: The R-Learner focuses on learning the relationship between the treatment variable and the residuals (R is for residual) of the outcome variable. First, separate models are fitted to predict the treatment and outcome variables based on the covariates. Then, the residuals from these models are used to train another model that predicts the outcome residual given the treatment residual. The causal effect for each observation is estimated using this model, which is trained to specifically capture the relationship between the treatment and outcome after accounting for the covariates. R-Learner attempts to directly model the treatment effect, allowing for heterogeneous treatment effects across the population. It was introduced in (Nie and Wager 2017).

5.4 Case Study

This case study demonstrates one way of applying causal inference methods to a real-world dataset that includes text. We make the ignorability assumption, which says that, after conditioning on a set of observed covariates, the treatment assignment is independent of the potential outcomes. In other words, it assumes that there are no unmeasured confounders that simultaneously influence the treatment assignment and the potential outcomes. See Section 2.5.1.1 for more on this assumption.

This case study follows the Metalearners Examples notebook that is in the **EconML** Github repository and can be viewed on Google Colab.

5.4.1 Dataset

We'll analyze the IMDB large movie reviews dataset, which was introduced in (Maas et al. 2011). This case study used the version available from Huggingface Datasets: https://huggingface.co/datasets/imdb.

In this case study, we ask the following question: "What influence do specific topics have on the sentiment of a movie review?". The treatment variable will be one of several possible topics in the movie review text. The outcome variable will be the binary sentiment label: 1 (positive) or 0 (negative). We estimate covariates by computing text embeddings with a pre-trained Transformer model.

5.4.2 Tools and Library

We use the **EconML** library (Keith Battocchi 2019) for computing CATE with meta-learners. The text encoding, discussed as the function g in Section 5.2, is provided by two libraries: **BERTopic** (Grootendorst 2022) for topic modeling and **sentence-transformers** (Reimers and Gurevych 2019) for text embeddings. We use the Huggingface **datasets** and **pandas** libraries for loading and working with the dataset, **matplotlib** for data visualization, and **scikit-learn** to create the train/test split, as described in Section 5.2.5. We also use the **UMAP** library for dimensionality reduction. You can install these by running the following command:

```
# umap and sentence-transformers are installed with bertopic
pip install bertopic==0.14.1\
    dowhy==0.9.1\
    numba==0.56.4\
    numpy==1.23.5\
    datasets
```

5.4.3 Generating Topics with BERTopic

Since the treatment of interest is the presence of a given topic in the movie description, we will start with topic modeling. The first step is to load the data:

```
from datasets import load_dataset

# downloads train and test sets, and also an unlabeled set
ds = load_dataset("imdb")
```

We also need to remove line break characters from the review text:

```
import re
import string


def clean(text):
    # Remove line breaks
    text = re.sub(r"<br />", "", text)
    return text

# Huggingface datasets lets up map over all three
clean_ds = ds.map(lambda example: {"text": clean(example["text"])})
```

5.4.3.1 Text Embeddings

Since BERTopic uses pre-trained transformer models in the topic generation process, we next compute embeddings for the movie overviews.

```
from sentence_transformers import SentenceTransformer
import torch

def get_torch_device():
    # This will the appropriate torch backend
    if torch.cuda.is_available():
        return "cuda"
    elif torch.backends.mps.is_available():
        return "mps"
    else:
        return "cpu"

device = get_torch_device()
sentence_model = SentenceTransformer("all-MiniLM-L6-v2",
                                     device=device)
embeddings = sentence_model.encode(docs,
                                   show_progress_bar=True,
                                   device=device)
```

5.4.3.2 Topics

Next, generate topics. We have chosen to use BERTopic here because you do not need to carefully tune parameters to get useful results.

```
from bertopic import BERTopic
from sklearn.feature_extraction.text import CountVectorizer
from umap import UMAP

# use word unigrams, bigrams, trigrams
vectorizer_model = CountVectorizer(
    stop_words="english",
    ngram_range=(1, 3)
)

umap_model = UMAP(
    n_neighbors=15,
    n_components=5,
    min_dist=0.0,
    metric='cosine',
    random_state=42
)

NUM_TOPICS = 50
topic_model = BERTopic(
    embedding_model=sentence_model,
    umap_model=umap_model,
    vectorizer_model=vectorizer_model,
    calculate_probabilities=True,
    nr_topics=NUM_TOPICS,
)

train_docs = list(clean_ds["train"]["text"])
test_docs = list(clean_ds["test"]["text"])

topics, probs = topic_model.fit_transform(train_docs, train_embeddings)

# fine-tune topic representation
vectorizer_model_1 = CountVectorizer(stop_words="english", ngram_range=(1, 3), min_df=10)
topic_model.update_topics(train_docs, vectorizer_model=vectorizer_model_1)
```

The distribution of topics is shown in Figure 5.3. Each dot represents a movie and each color represents a topic. There are two separate sections because there are positive and negative sentiment labels. We use embeddings reduced to 2D for plotting.

```
umap_2d = UMAP(
    n_neighbors=10,
    n_components=2,
    min_dist=0.0,
    metric='cosine',
)
train_embeddings_2d = umap_2d.fit_transform(
    train_embeddings,
    clean_ds["train"]["label"],
)
```

```
topic_model.visualize_documents(
    train_docs,
    reduced_embeddings=train_embeddings_2d,
    hide_document_hover=False,
    hide_annotations=True,
)
```

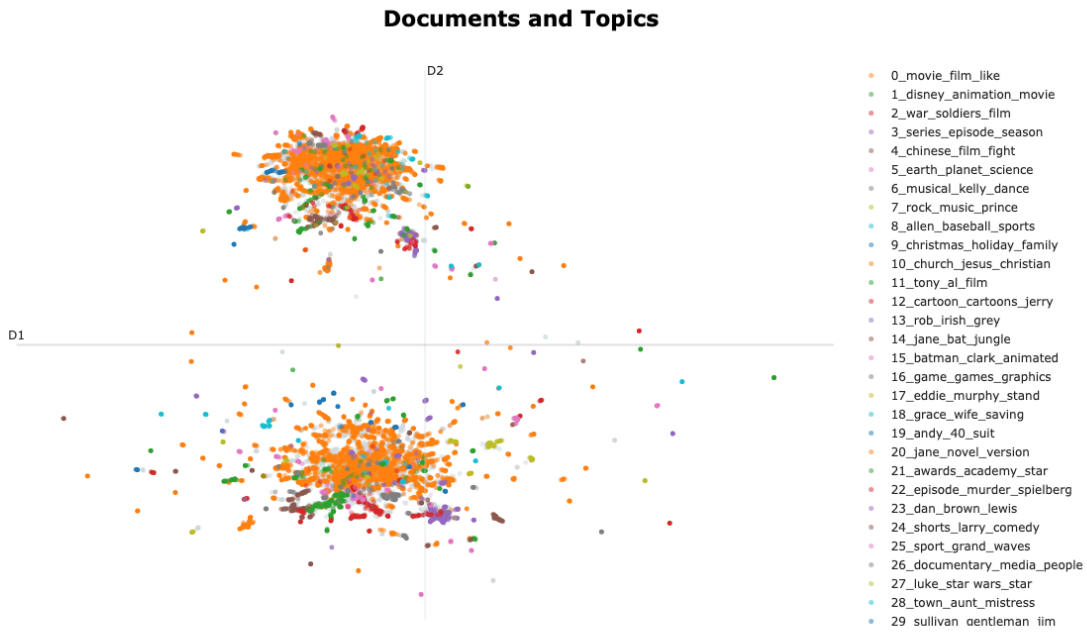


Figure 5.3: Topic Distribution

You may have noticed that only one topic is assigned to each movie review, which is counter

to what we want, since a movie review can have more than one topic. To address this, we can assign more than one topic to each review by using soft assignments.

```
topic_distr, topic_token_distr = topic_model.approximate_distribution(
    train_docs,
    calculate_tokens=True,
    window=4,
)

test_topic_distr, test_topic_token_distr = topic_model.approximate_distribution(
    test_docs,
    calculate_tokens=True,
    window=4,
)
```

In the previous code snippet, you may have noticed `calculate_tokens=True`. This lets us see how much each token in the text contributes to that texts's topic assignments. Figure 5.4 shows a portion of the token topic distribution for the text of one movie review.

```
topic_model.visualize_approximate_distribution(
    train_docs[134],
    topic_token_distr[134]
)
```

	Saw	this	movie	in	my	English	class	this	afternoon	and	was	surprised	by	how	bad	this	version	was	Don	get	me	wrong
0_movie_film_like_just		0.186	0.186	0.186	0.186																	
1_disney_animation_movie_animated		0.143	0.143	0.143	0.143											0.102	0.204	0.204	0.204	0.102		
2_war_soldiers_film_movie		0.126	0.126	0.126	0.126																	
4_chinese_film_fight_action		0.140	0.140	0.140	0.140																	
5_earth_planet_science_movie		0.150	0.150	0.150	0.150																	
7_rock_music_prince_movie		0.162	0.162	0.162	0.162																	
8_allen_baseball_sports_ted		0.137	0.137	0.137	0.137																	
9_christmas_holiday_family_movie		0.112	0.112	0.112	0.112									0.100	0.201	0.201	0.201	0.100			0.118	0.236
10_church_jesus_christian_god		0.162	0.162	0.162	0.162																	
11_tony_al_film_dick		0.103	0.103	0.103	0.103																	
14_jane_bat_jungle_monkey		0.112	0.112	0.112	0.112																	
17_eddie_murphy_stand_child																						
18_grace_wife_saving_alan		0.106	0.106	0.106	0.106																	

Figure 5.4: Token-Topics Distribution

5.4.3.2.1 Adding topics to the dataframe

Lastly, we are going to add the topics to the dataframe. We're also going to use the topic labels used by **BERTopic** as the dataframe column names. We will add one column to the dataframe for each topic.

```
import pandas as pd

topic_labels = topic_model.generate_topic_labels(
    nr_words=4,
    topic_prefix=True,
    word_length=10,
    separator="_",
)

id2topic = {i-1: f"t_{label}"
            for i, label in enumerate(topic_labels)}
del id2topic[-1] # drop outlier topic

ntopics = topic_distr.shape[1]
topic_names = [id2topic[i] for i in range(ntopics)]

train_topics_df = pd.DataFrame(
    data=topic_distr,
    columns=topic_names,
)
train_df = clean_ds["train"].to_pandas()
train_df_full = pd.concat(
    [train_df, train_topics_df],
    axis=1,
)

test_topics_df = pd.DataFrame(
    data=test_topic_distr,
    columns=topic_names,
)
test_df = clean_ds["test"].to_pandas()
test_df_full = pd.concat(
    [test_df, test_topics_df],
    axis=1,
)
```

5.4.4 Covariates

We'll use the sentence embeddings computed above. The embedding dimension is 384, which is rather high, so we will use UMAP to reduce the dimension to 20. As with the topics, we will add the embeddings to the dataframe. There will be one column for each embedding dimension.

```
print(f"Embeddings: {train_embeddings.shape}")

n_covariates = 20
col_names = [f"e_{i:04d}" for i in range(n_covariates)]

umap_cov = UMAP(
    n_neighbors=10,
    n_components=n_covariates,
    min_dist=0.0,
    metric='cosine',
)
train_covariates = umap_cov.fit_transform(train_embeddings)
test_covariates = umap_cov.transform(test_embeddings)
print(train_covariates.shape)
print(test_covariates.shape)

X_df = pd.DataFrame(
    data=train_covariates,
    columns=col_names,
)
train_df_all = pd.concat([train_df_full, X_df], axis=1)

X_df = pd.DataFrame(
    data=test_covariates,
    columns=col_names,
)
test_df_all = pd.concat([test_df_full, X_df], axis=1)
```

Our outcome variable is the sentiment label. Positive sentiment is label = 1, negative sentiment is label = 0, so we need not binarize the labels. We are adding a simple method to grab the label column and similar methods for selecting a specific topic column from the dataframe, and then convert it to a binary-valued column according to a probability threshold. There is also a method that grabs the covariate columns.

```
def get_outcome_values(df):
    Y = df["label"].to_numpy()
    return Y

def get_treatment_values(df, topic_id, threshold=0.5):
    topic = id2topic[topic_id]
    t = threshold
    T = df[topic].apply(lambda x: x >= t).astype(int).to_numpy()
    return T

def get_covariate_values(df):
    X = df[col_names].to_numpy()
    return X
```

Now we can grab the train/test values for treatment, outcome, and covariates and move on to the causal analysis. Note: Due to some inherent randomness, the ids of a topic may shift when this code is run at a later date. The overall distribution will be somewhat stable though. In this first example, we will use `topic_id=27`, which corresponds to several terms related to the "Star Wars" films.

```
# treatment
topic_id = 27  # star wars

T_train = get_treatment_values(
    train_df_all,
    topic_id,
    threshold=0.1,
)
T_test = get_treatment_values(
    test_df_all,
    topic_id,
    threshold=0.1,
)
print(f"T_train: {T_train.shape}")
print(f"T_test: {T_test.shape}")

Y_train = get_outcome_values(train_df_all)
Y_test = get_outcome_values(test_df_all)
print(f"Y_train: {Y_train.shape}")
print(f"Y_test: {Y_test.shape}")

X_train = get_covariate_values(train_df_all)
```

```
print(f"X_train: {X_train.shape}")

X_test = get_covariate_values(test_df_all)
print(f"X_test: {X_test.shape}")
```

5.4.5 Train metalearner causal estimators

In this section we train the meta-learners that we'll use to compute CATE. First, we'll want to import the classes and methods we'll need.

```
import numpy as np
from numpy.random import (
    binomial,
    multivariate_normal,
    normal,
    uniform,
)
from sklearn.ensemble import (
    RandomForestClassifier,
    GradientBoostingRegressor,
)
import matplotlib.pyplot as plt

min_samples_leaf = len(T_train) // 100
```

5.4.5.1 T-learner

```
models = GradientBoostingRegressor(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf
)

T_learner = TLearner(models=models)

T_learner.fit(Y_train, T_train, X=X_train)

# Estimate treatment effects on test data
T_te = T_learner.effect(X_test)
```

5.4.5.2 S-learner

```
overall_model = GradientBoostingRegressor(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf
)

S_learner = SLearner(overall_model=overall_model)

S_learner.fit(Y_train, T_train, X=X_train)

# Estimate treatment effects on test data
S_te = S_learner.effect(X_test)
```

5.4.5.3 X-learner

```
models = GradientBoostingRegressor(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf
)

propensity_model = RandomForestClassifier(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf
)

X_learner = XLearner(
    models=models,
    propensity_model=propensity_model
)

X_learner.fit(Y_train, T_train, X=X_train)

# Estimate treatment effects on test data
X_te = X_learner.effect(X_test)
```

5.4.6 Comparing treatment effects

Here we compare the treatment effect estimates for topic 27 (Star Wars) computed in the previous section. We plot each effect estimate against the test set index in Figure 5.5.

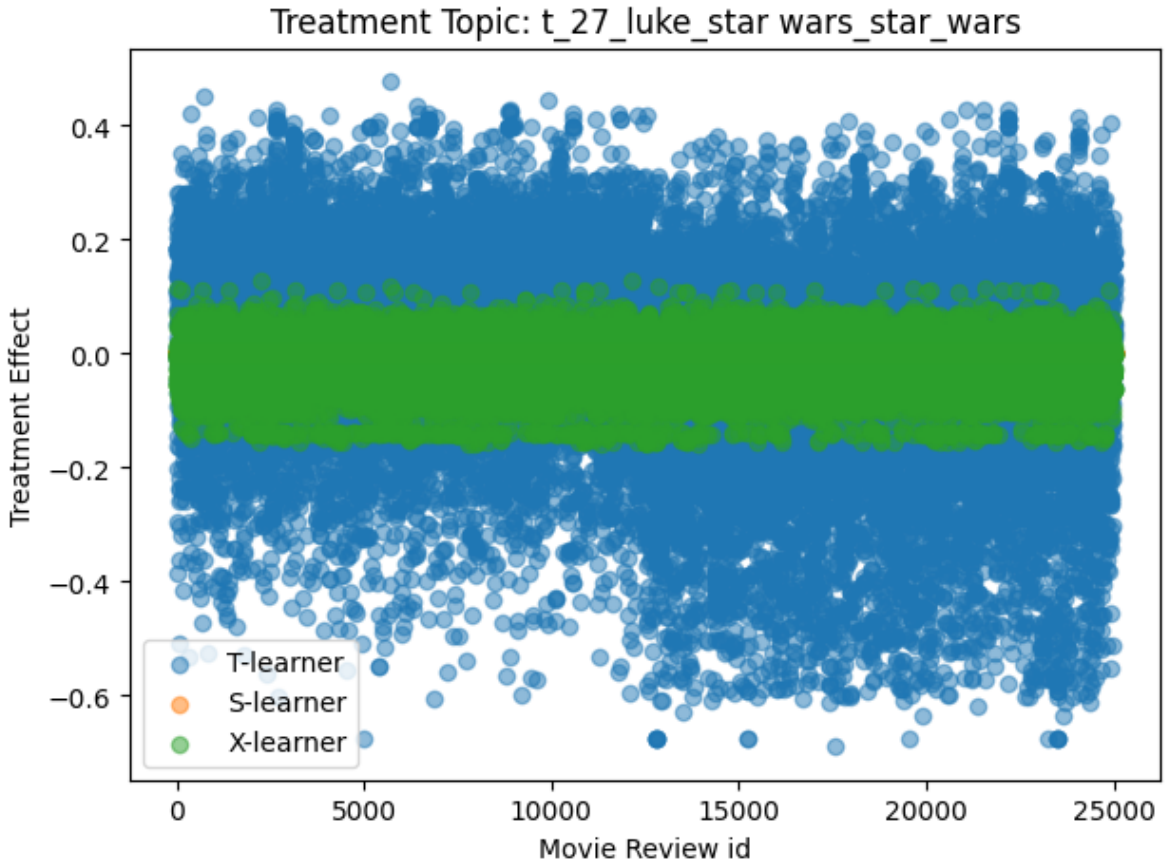


Figure 5.5: CATE for Topic 27 (Star Wars)

In Figure 5.6 we view this same information in histogram form, where we can see the number of movie reviews with the Star Wars topic that have CATE values in a given range.

These results show that, according to the S-Learner, the Star Wars topic has a negligible positive effect on the sentiment of a movie review. However, the T-Learner and X-Learner both show a that this topic has a positive effect on the outcome for some reviews and a negative effect on the outcome for other reviews. Let's look at the CATE histograms of a few more topics.

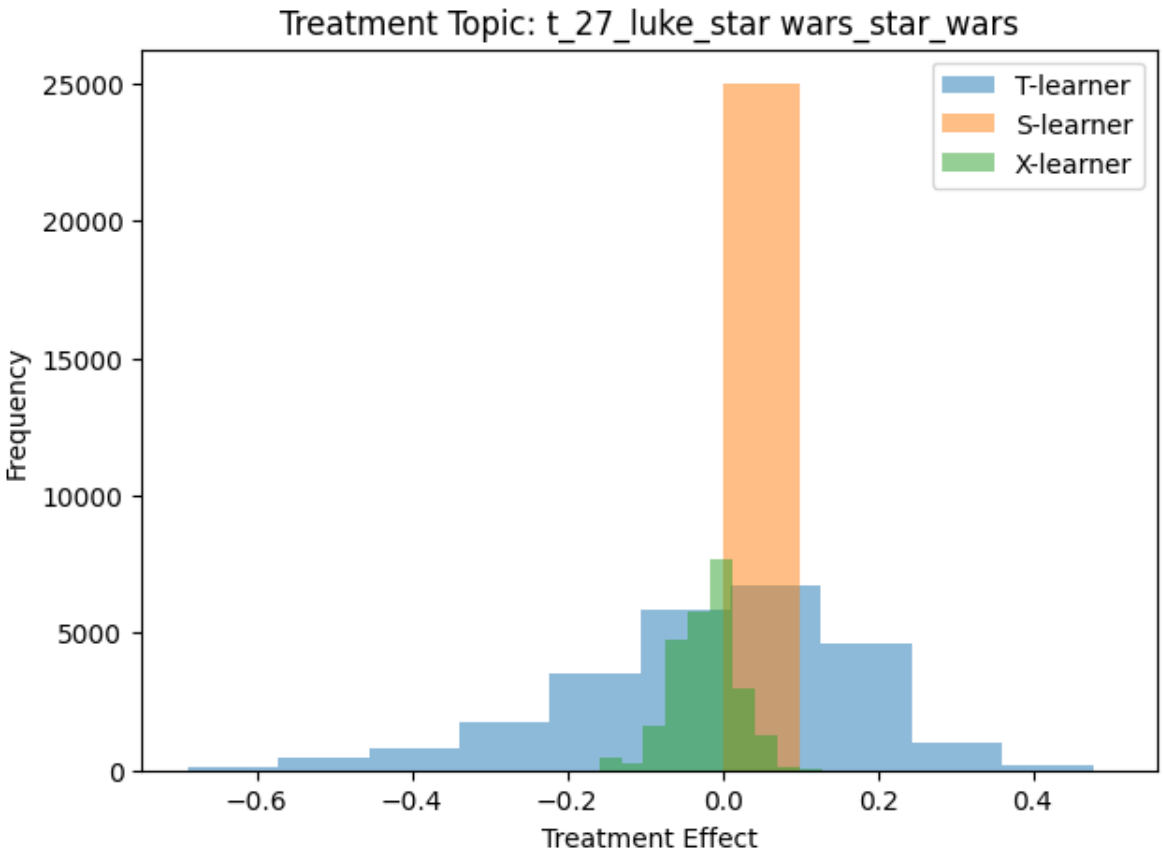


Figure 5.6: CATE Histogram for Topic 27 (Star Wars)

```
def run_analysis(topic_id, topic_threshold=0.1):
    # topic_id is treatment
    T_train = get_treatment_values(
        train_df_all,
        topic_id,
        threshold=topic_threshold,
    )
    T_test = get_treatment_values(
        test_df_all,
        topic_id,
        threshold=topic_threshold,
    )

    treated_indices = np.where(T_test == 1)[0]
    untreated_indices = np.where(T_test == 0)[0]

    Y_train = get_outcome_values(train_df_all)
    Y_test = get_outcome_values(test_df_all)

    X_train = get_covariate_values(train_df_all)
    X_test = get_covariate_values(test_df_all)

    print("Building T-Learner...")
    models = GradientBoostingRegressor(
        n_estimators=100,
        max_depth=6,
        min_samples_leaf=min_samples_leaf,
    )
    T_learner = TLearner(models=models)

    T_learner.fit(Y_train, T_train, X=X_train)
    T_te = T_learner.effect(X_test)

    print("Building S-Learner...")
    overall_model = GradientBoostingRegressor(
        n_estimators=100,
        max_depth=6,
        min_samples_leaf=min_samples_leaf,
    )
    S_learner = SLearner(overall_model=overall_model)
```

```
S_learner.fit(Y_train, T_train, X=X_train)
S_te = S_learner.effect(X_test)

print("Building X-Learner...")
models = GradientBoostingRegressor(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf,
)
propensity_model = RandomForestClassifier(
    n_estimators=100,
    max_depth=6,
    min_samples_leaf=min_samples_leaf
)

X_learner = XLearner(
    models=models,
    propensity_model=propensity_model,
)
X_learner.fit(Y_train, T_train, X=X_train)
X_te = X_learner.effect(X_test)

plt.figure(figsize=(7, 5))
x_axis = np.arange(len(X_test))

plt.hist(T_te, label="T-learner", bins=10, alpha=0.5)
plt.hist(S_te, label="S-learner", bins=10, alpha=0.5)
plt.hist(X_te, label="X-learner", bins=10, alpha=0.5)

plt.xlabel('Treatment Effect')
plt.ylabel('Frequency')
plt.title(f"Treatment Topic: {id2topic[topic_id]}")

plt.legend()
plt.show()
```

We can use the ‘run_analysis function to generate several histograms:

Figure 5.7, Figure 5.8, Figure 5.9, and Figure 5.10 show the result CATE histogram for the Disney, Batman, holiday movie, and Dan Brown topics, respectively. Each histogram was generated by calling `run_analysis` with the appropriate topic id.

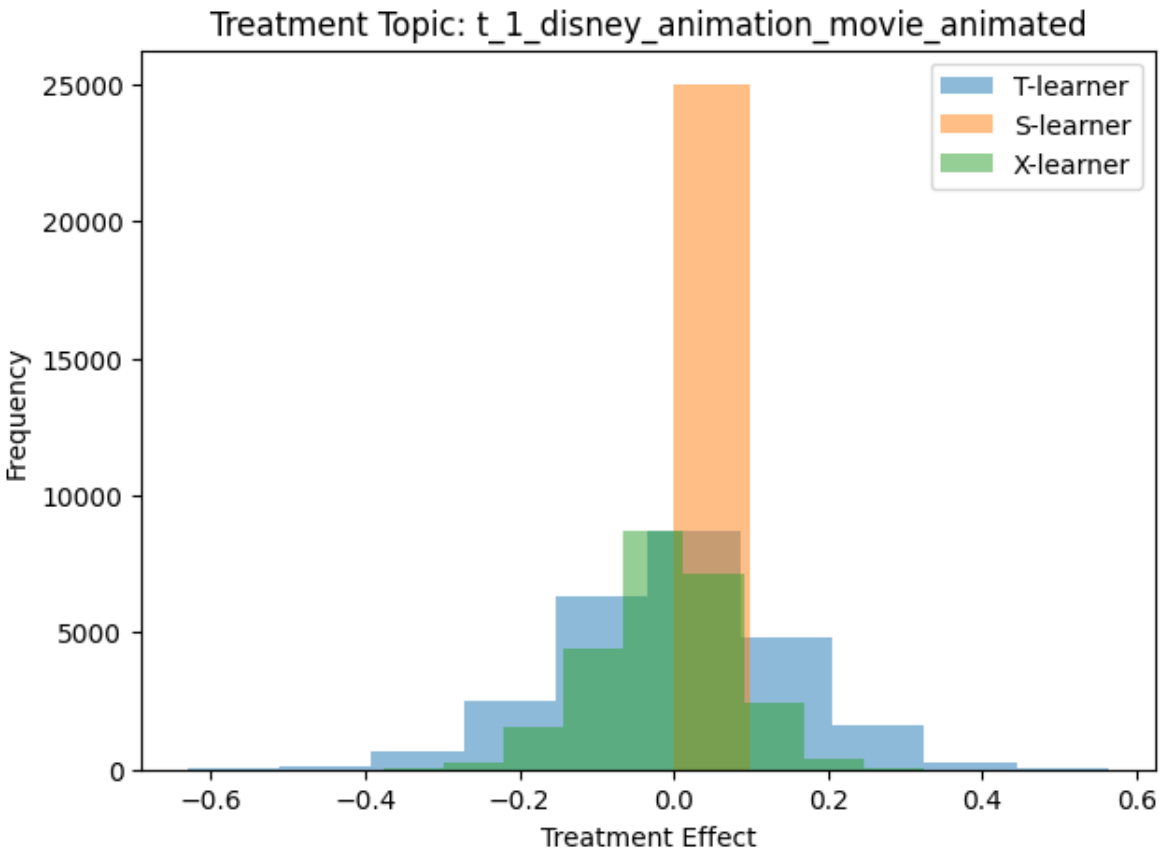


Figure 5.7: CATE Histogram for Topic 1 (Disney)

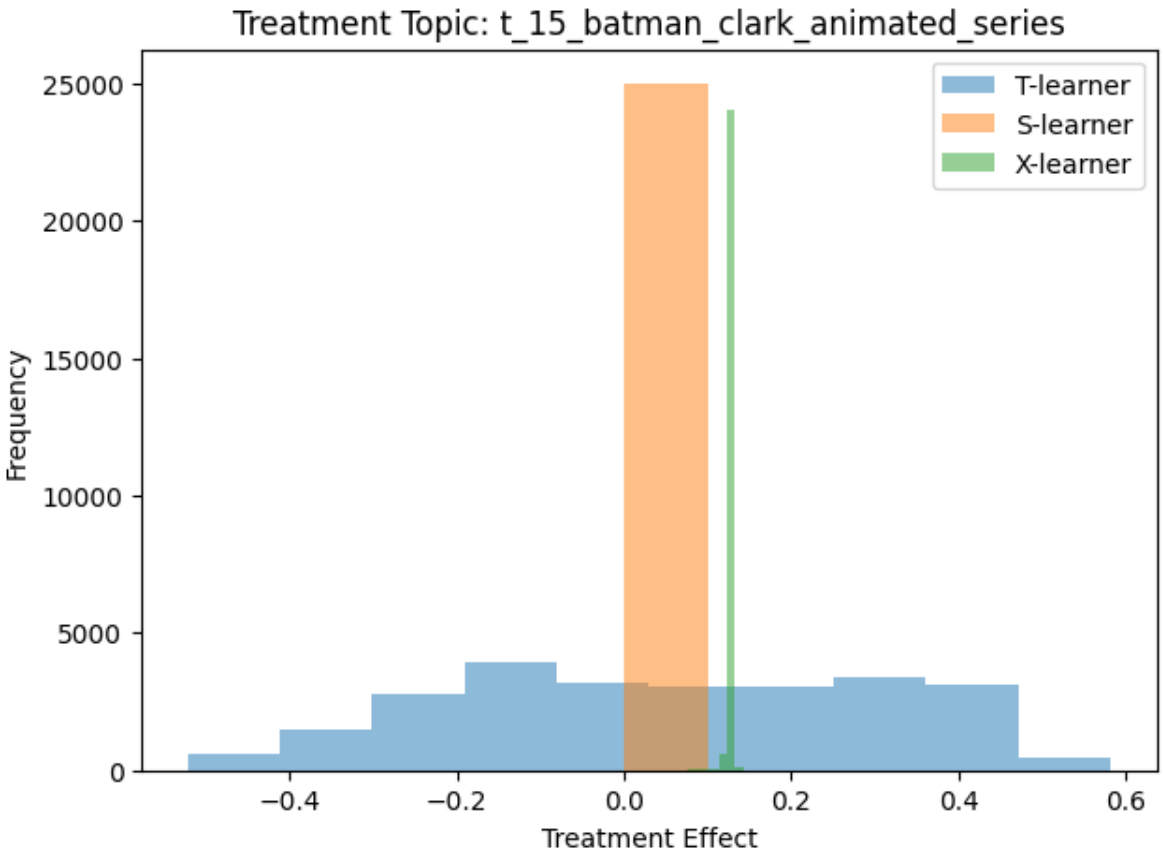


Figure 5.8: CATE Histogram for Topic 15 (Batman, etc.)

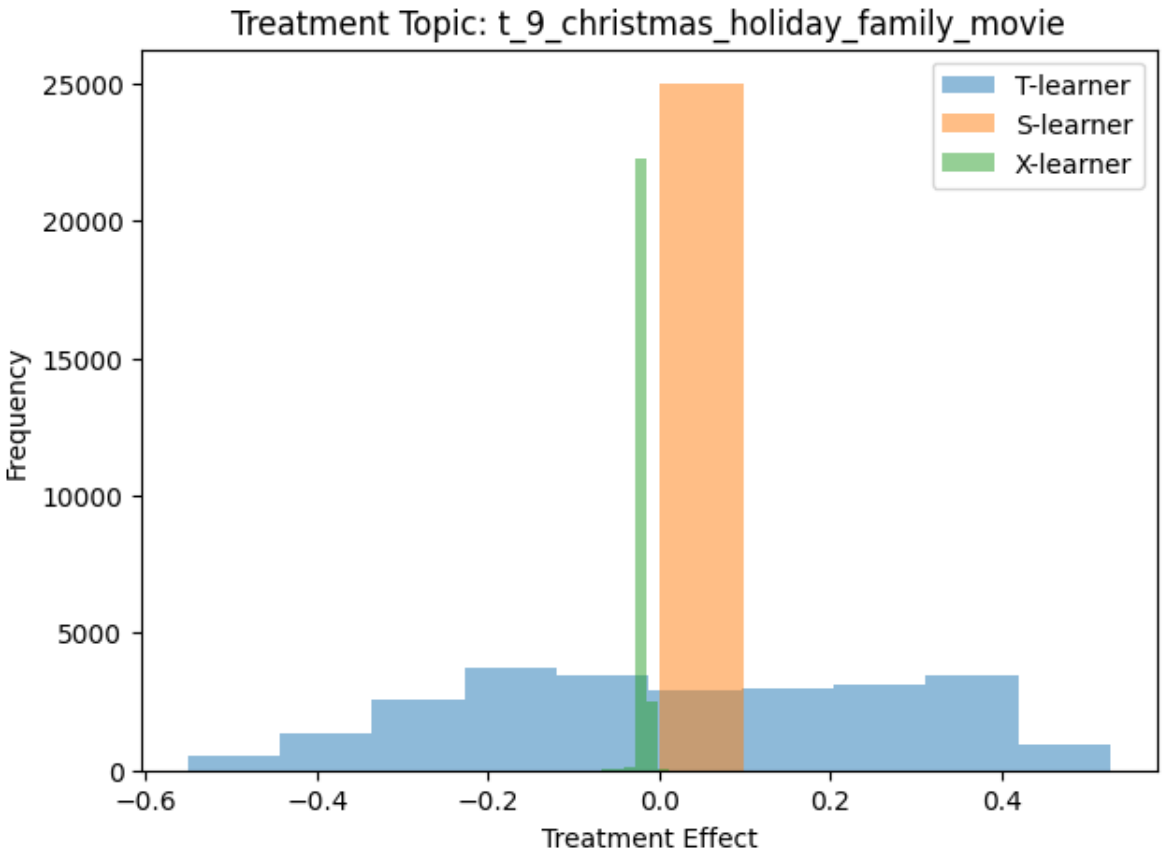


Figure 5.9: CATE Histogram for Topic 9 (Holiday)

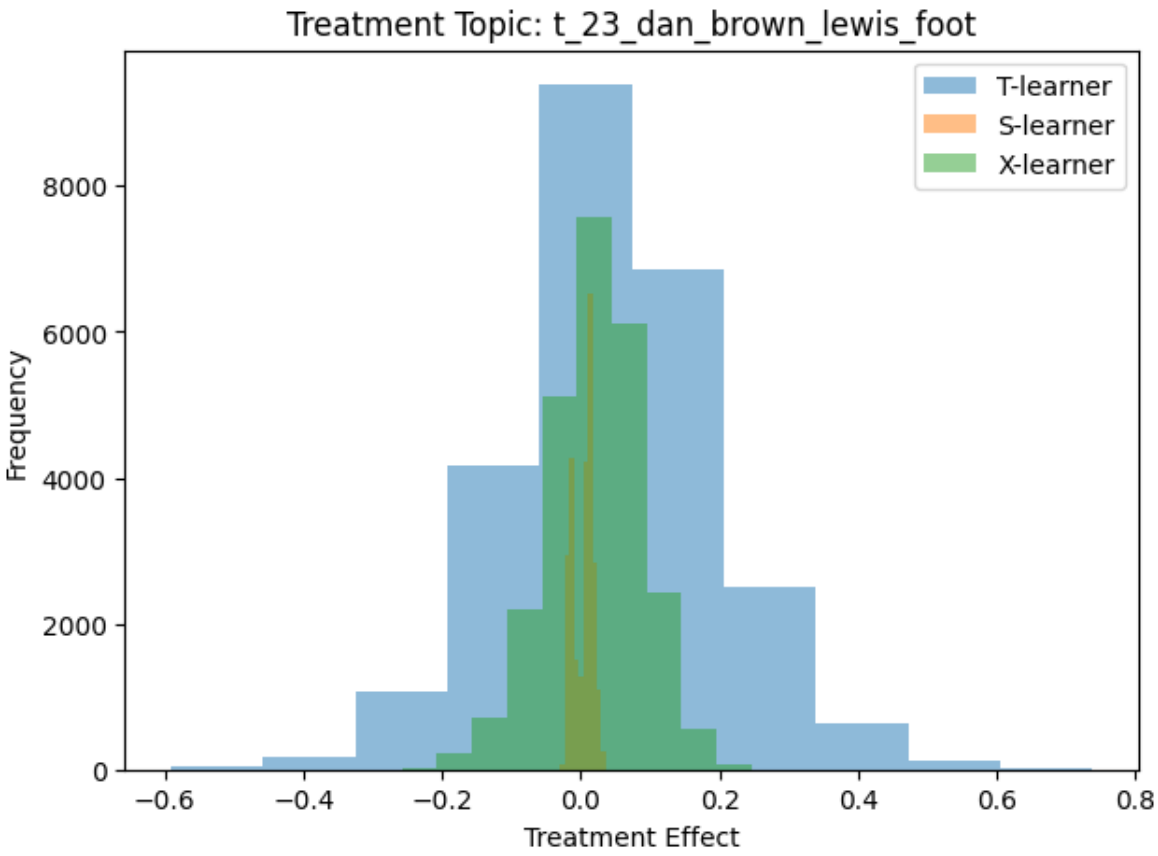


Figure 5.10: CATE Histogram for Topic 23 (Dan Brown)

5.4.6.1 Analysis

So, what does the distribution of CATE scores tell us?

Generally speaking, when the outcome variable is binary-valued (as it is for our movie review data), a negative CATE value means that the treatment has a negative/detrimental effect on the outcome variable. For the IMDB data, that means that when the topic is present in the movie review, said review is more likely to be negative than similar reviews without the topic. Similarly, a positive CATE value suggests that the topic tends to make the review more positive.

5.4.6.1.1 S-Learner CATE scores

A few of the histogram plots have a large peak for the S-Learner that is near zero. The S-Learner scores are then suggesting that each topic has a negligible effect of the sentiment of movie reviews.

However, the S-Learner assumes that the treatment effect is homogeneous across the population. If that assumption is not correct (and we have no reason to think it should be), then treatment effect is not homogeneous across the population, then the CATE scores calculated using an S-Learner may not accurately reflect the true treatment effect.

5.4.6.1.2 T-Learner and X-Learner CATE scores

The histograms for the T-Learner and X-Learner suggest that the effect a given topic has on the movie review sentiment is sometimes positive and sometimes negative. Figure 5.5 shows that we can actually see which specific movie reviews have positive and negative scores.

We can also note that the some topics appear to have a large effect on sentiment. An example of this is topic 23, the Dan Brown topic, as shown in Figure 5.10.

6 Computer Vision

This chapter focuses on applications combining causal inference and computer vision. Computer vision (CV) refers to a domain of machine learning which involves the use of images or videos as inputs. CV has gained significant adoption over the last several years, with applications appearing within industries including technology, medicine, manufacturing, and defense. In recent years, researchers have developed some exciting methods using causal inference techniques to overcome common problems within CV – in this chapter, we will give a brief overview of common tasks and methodologies in CV, and detail the various ways that causality is being used.

Prerequisite Knowledge

In addition to the causal inference concepts introduced in the first part of this book, relatively little background knowledge is necessary for this chapter. Readers with more familiarity or experience with computer vision will likely benefit more from this chapter, but we will introduce relevant concepts, tasks, challenges, and methods as needed.

Readers who wish to learn more about computer vision more holistically are encouraged to explore the following resources:

- The *Fast AI* book (Howard and Gugger 2020) has two chapters on computer vision[1]
- The Coursera course on Convolutional Neural Nets and Computer Vision from Andrew Ng's Deep Learning specialization[2]

6.1 The Current State of Computer Vision

Computer vision has come a long way since Yann LeCun introduced the convolutional neural network (CNN) for handwritten digit recognition (Lecun et al. 1998). Starting in around 2012 – when AlexNet (Krizhevsky, Sutskever, and Hinton 2012) shattered existing state-of-the-art on the ImageNet task using a deep CNN – computer vision began to take off, with its trajectory aided by an increase in the availability of computing power coinciding with the "big

[1] See chapters 4 and 5: https://github.com/fastai/fastbook
[2] Available at https://www.coursera.org/learn/convolutional-neural-networks?specialization=deep-learning

data" boom. Since then, previously difficult benchmarks have become too easy to use, and boundaries continue to be pushed.

Today, computer vision comprises a wide variety of tasks, ranging from image classification and object detection to image generation and question answering based on input image. These models are used in applications such as medical image segmentation (Milletarì, Navab, and Ahmadi 2016), quality control in factories (Zhou, Zhang, and Konz 2021), and identification of pedestrians, structures, and other vehicles in self-driving cars (Cakir et al. 2022). Deep learning dominates modern CV, and model design depends largely upon the application; however, many commonalities exist: convolutional networks are still used in many settings, while sequence models such as Transformers have also gained popularity in many tasks due to their success in natural language processing.

There are a handful of common issues that occur within CV projects, and they are similar in nature to problems encountered more broadly throughout machine learning. These issues include a lack of robustness when transferring to slightly different datasets, which is often due to the tendency of greedy, data-hungry ML models to latch onto spurious correlations — for instance, random patterns in the background or textures in an image. The issue of spurious correlations is exacerbated by the fact that deep learning techniques are inherently uninterpretable. Techniques exist to "peek into the black box," such as highlighting a salient region using gradient flow through a network; however, these models do not provide reliable methods of interpreting their output or decisions, making it difficult to truly understand how a model learns a particular pattern or identify when it happens.

6.2 Causal Methods in Computer Vision

As we have seen throughout this book thus far, handling spurious correlation is one of the major features of causal inference, making CV a compelling area of application for causal methods. In fact, a recent paper in the journal *Nature* describes the importance of causal reasoning within computer vision in clinical healthcare settings (Castro, Walker, and Glocker 2020). In this paper, the authors primarily focus on the necessity of understanding the causal direction of the data generating process, describing the biases which can appear when observations and annotations are sampled in ways that do not match the real-world setting in which the model will be deployed. These mismatches can be subtle, but the corresponding sampling bias can have a catastrophic impact on a deployed model's performance. Their work shows that causal language can prevent issues like these from creeping into clinical computer vision projects. As we will see in this chapter, causal inference methods can aid CV practitioners in other ways as well: from improving model performance to making models more robust in out-of-domain settings.

In the sections that follow, we will give an overview of the current state of causal methods within various subdomains of computer vision, including object recognition and segmentation, few-shot learning, vision-language tasks, and improving robustness. After introducing these

areas of application in causal computer vision, we will provide a case study detailing one of these methods.

6.2.1 Image Classification

Image classification, also called visual recognition, is one of the most common tasks in computer vision: given an input image, a model is trained to predict the class of the image from a set of defined classes (e.g. categorizing photos of clothing as shirts, pants, jackets, etc.). One longstanding image classification benchmark is ImageNet, a dataset of over 14 million images labeled in a hierarchical class structure made up of thousands of categories (Deng et al. 2009). An example of one of these images is given in Figure 6.1.

Figure 6.1: An example from the ImageNet training set, labeled "Staffordshire bull terrier"

We can see that the image contains an adorable puppy – this image is given the label "Staffordshire bull terrier," and it contains some examples of the issues that arise within CV. Imagine that we are training a classifier to distinguish Staffordshire bull terriers from another dog breed,

for example a Boston terrier. Ideally, we would want this model to focus on the true signal made up by meaningful differences between these breeds: Boston terriers are often black and white, while Staffordshire terriers have more uniform coats; Staffordshire terriers have more pronounced muzzles, while Bostons have a short snout. However, in reality, a deep learning model will look for any correlations present in the data in order to minimize the loss on the training set.

An obvious issue in this image is the watermark in the bottom right corner of Figure 6.1, which is a URL for a breeder of Staffordshire bull terriers in the Czech Republic. If many of our training images for Staffies come from this breeder, the model will likely learn to look for the shape of that URL watermark instead of more nuanced features like differences in the shapes of ears or muzzles. A model trained to identify the URL of a dog breeder will be useless if we want to deploy it on natural images "in the wild." This is a prime example of how CV models can learn undesirable patterns, just like the spurious correlations introduced earlier in this book. This type of problem is a central focus in the area of robustness, and two approaches for mitigating spurious correlation in image classification are detailed in Section 6.2.5.

The image in Figure 6.1 also includes potential confounders: for example, the puppy is photographed while being held by its owner. If our training set primarily includes images of this breed being held, the model might look for the shape of human hands rather than the dog's fur length or color. Confounders like this one can appear in background color, lighting, textures, and other co-occurring elements within an image, and issues can arise when a classifier relies on the presence of a confounder rather than the primary subject of interest. One way to remedy this issue would be to manually label any confounders present in an image and intervene on their presence when training the model, but this would require significantly more effort than standard image classification labeling. One promising alternative, dubbed Causal Attention Module (CaaM), is a modification of the visual attention mechanism which allows for a model to *attend* to particular regions or channels of an image with greater focus. CaaM models confounders in an unsupervised manner, which allows for intervention without the need to manually annotate confounders during the labeling process. Their method works within standard visual attention settings, including vision Transformers (ViT) and attention CNNs, and achieves greater in-domain and out-of-domain performance than both attention-based and non-attention-based models (T. Wang et al. 2021).

6.2.2 Few-Shot Learning

Transfer learning has played a major role in the success of deep learning models in the last several years: we first *pretrain* on a task with abundant data, such as image reconstruction, and then *fine-tune* on the actual task we want the model to perform. Pretrained models allow for better transfer with less labeled data, accelerating the process of training a useful model whenever a relevant pretrained base model is available. Popular pretrained CV models include ResNet (He et al. 2015) and Vision Transformer (Dosovitskiy et al. 2020) for classification as well as the YOLO family of models for object detection (Redmon et al. 2015).

Few-shot learning is a special case of transfer learning, in which the target task has very few labeled observations available for fine-tuning Few-shot learning (FSL) has become a common tool for testing the performance of pretrained models, since it assesses the base model's ability to generalize to new domains and tasks with little information. The successes of deep learning models in the few-shot setting is an exciting development, because it represents a major shift in what was previously impossible: that a deep neural network can reach meaningful performance with only a handful of relevant observations for a given task. The magic of FSL comes from the pretraining/fine-tuning paradigm – the model is able to extend the patterns it learns in the less relevant yet data-rich setting (such as low-level shapes and textures) beyond the immediate context and into an actually meaningful task.

However, is it possible that the "knowledge" gained in the pretraining setting is also potentially harmful in the few-shot fine-tuning setting? This is a phenomenon known as *negative transfer*, where the biases in the pretrained dataset actually degrade performance on the downstream task. When we examine this phenomenon from a causal perspective, it becomes clear why this might happen. A recent paper at NeurIPS (Yue et al. 2020) frames the few-shot transfer setting within the context of the causal graph in Figure 6.2, where D represents the pretrained knowledge of the base model, X denotes the feature representation produced initially by D, C denotes the transformed representation of X in the low-rank manifold, and Y represents the output of the task-specific final layer.

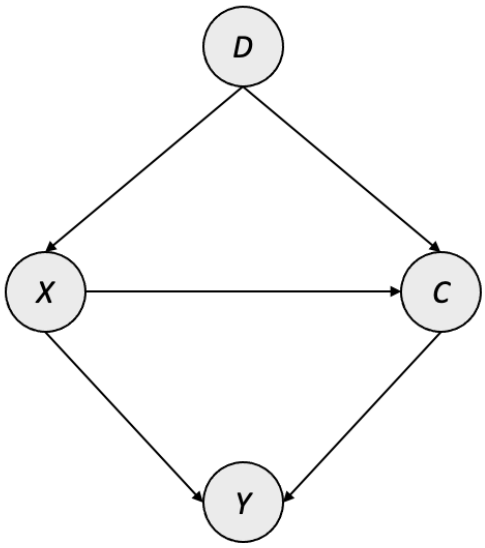


Figure 6.2: Causal graph for interventional few-shot learning

In plain terms, this causal graph is saying that the biases present in the pretrained model are actually a confounding presence on the performance in the FSL task. For example, consider the possibility that a pretrained model had only seen Staffordshire terriers like the one in

Figure 6.1 indoors, standing on rugs – such a pretrained model might misinterpret the correlation between the dogs and their surroundings as something meaningful, and we may not have enough example images in the FSL context to overrule that bias.

As a way of addressing this, the authors suggest an interventional approach to remove the impact of the confounders. Specifically, if the standard (i.e. *many-shot*) setting for model transfer produces a task-specific layer Y which is more or less fully adapted to the task, we can think of this action as $P(Y|X) \approx P(Y|do(X))$. This is not the case in the few-shot setting due to the substantial influence of D on Y, however, this can be addressed by intervening on $D \rightarrow X$ via a backdoor adjustment and effectively modeling $P(Y|do(X))$ directly. The authors do this via two specific adjustments: a feature-wise adjustment using the output of the final pretrained layer, and a class-wise adjustment to account for pretraining tasks which are classification-based (e.g. 1000-class ImageNet, or BERT's masked language modeling).

The result of the interventional few-shot learning (IFSL) approach is a method that improves FSL performance on different types of tasks and in different CV architectures. The authors show that the benefit of IFSL is greater in 1-shot tasks than in 5-shot tasks, and that improvements are more significant when the FSL domain is more different from the pretraining domain. The authors also demonstrate that existing methods of boosting FSL performance – such as data augmentation – are approximations of a similar intervention.

6.2.3 Weakly Supervised Semantic Segmentation

Semantic segmentation is a computer vision task which involves the generation of a pixel-level mask corresponding to individual entities within an image, which produces groups of regions corresponding to semantically similar objects. For example, a segmentation model trained on medical images may learn to distinguish suspicious masses from fluid, bone, and other normal-appearing tissue also visualized within the scan; and a segmentation model in an autonomous driving setting may try to identify masks of pedestrians, other vehicles, and road signs within the images produced by the vehicle's camera. Semantic segmentation provides richer context through labeling classes for the masks within an image, such as denoting each distinct "person" mask within an image as belonging to the same "person" class. A semantic segmentation model may identify masked regions for both "dog" and "person" in our example image in Figure 6.1, while keeping their masks separate from each other as well as the nondescript background of the image.

One challenge with semantic segmentation is the cost of labeling: relative to the effort required to label other CV tasks (e.g. image classification), labeling images for semantic segmentation is difficult and time-consuming due to the nature of masking pixels for every instance of a particular entity. *Weakly supervised* semantic segmentation (WSSS) seeks to lower this barrier by collecting image-level labels, training a classifier, and utilizing information from the classifier to generate "pseudo masks" for semantic segmentation corresponding to the labels in the classification task (Ahn and Kwak 2018). The "weak supervision" aspect comes through

using one model's output as the ground-truth input for another training task – it may not be a perfect or unbiased approach, but it significantly reduces the effort required to label and train semantic segmentation models, which in turn allows for labeled data to be collected at a much greater rate.

(D. Zhang et al. 2020) propose a causal approach to WSSS, based on a backdoor adjustment to mitigate the impact of spurious correlations introduced by class co-occurrence and incomplete image-level labeling. Their fundamental WSSS approach is similar to other typical WSSS pipelines, comprising of an image classifier which generates attribution masks for training the segmentation model. However, the authors propose a method called Context Adjustment (CONTA) which models an intervention on class co-occurrence. Their causal graph is provided in Figure 6.3, where X represents the image observation, Y represents the multi-label classification labels, C represents the "context prior" inherent within the data-generating process, and M is the image-specific manifestation of C for image X.

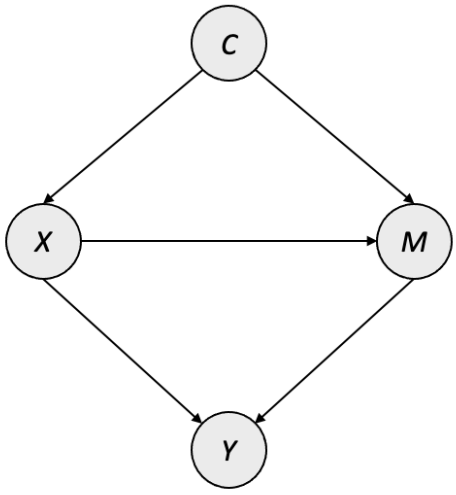


Figure 6.3: Causal graph for Context Adjustment in weakly supervised semantic segmentation

It is worth noting that the context prior C does not directly influence the class labels in Y, but instead only through observation-specific pathways through X and M. The objective of CONTA is to train the initial image classifier to learn $P(Y|do(X))$ instead of the typical $P(Y|X)$ used in training machine learning models. In order to model $P(Y|do(X))$, the authors propose a framework which uses class-wise average pseudo masks to provide the intervention while training the image classifier. This results in an iterative algorithm similar in nature to expectation-maximization (EM) algorithms:

1. Train the multilabel image classifier using a custom loss function incorporating similarity against class-averaged masks

2. Generate pseudo-masks using the trained classifier's attribution map
3. Train the segmentation model on pseudo-masks
4. Update the class averaged masks using the output of the segmentation model

After a few iterations, this process results in a segmentation model which can produce cleaner boundaries between entities and more reliably handle complex images. A major benefit of this framework is that it is agnostic to the architecture and post-processing applied to the segmentation task, so that the latest art in segmentation can be used while also leveraging the causal techniques. The code for this framework is available at https://github.com/dongzhang89/CONTA.

Additional work in WSSS

A recent paper at CVPR proposes a WSSS technique for medical image segmentation based on similar foundations as CONTA – also using the WSSS approach of pseudo-masking via a classifier's saliency maps. (Z. Chen et al. 2022) extends the idea of intervening on influences from the data generating process by introducing an additional map for on anatomical structures present within the scan. The authors show that their method outperforms prior WSSS state of the art on various medical image segmentation tasks.

6.2.4 Vision-Language Tasks

The intersection of vision and language is gaining popularity as more breakthroughs are published, such as DALL-E's results on image generation from a text prompt (Ramesh et al. 2021). Common tasks in this area include generating captions for images, answering questions about images, and performing reasoning about an image. Common techniques include visual attention, with larger Transformer-based architectures like BEiT (W. Wang et al. 2022) achieving recent success across many vision-language tasks. This section will provide an overview of three applications of causal inference designed to improve performance and reliability of vision-language models.

Visual Commonsense R-CNN

(T. Wang et al. 2020) introduces Visual Commonsense R-CNN (VC R-CNN), which combines the commonly used region-based CNN (R-CNN) with a "visual commonsense" module designed to learn more robust features for question-answering and captioning. VC R-CNN approaches this in an unsupervised manner by identifying confounders present in an image, which it controls via the backdoor adjustment. VC R-CNN begins by generating regions of interest from an existing pretrained model (such as Faster R-CNN), averaging the regions of interest for each possible category in the dataset, and using the resulting dictionary of average shapes as a feature in the model. Specifically, the "dictionary" of regions of interest for each class is used

alongside the observation image as inputs to scaled dot-product attention. The authors show a performance boost within vision-language tasks but note that the improvement is smaller on VQA than in captioning and reasoning. Their code is available at https://github.com/Wangt-CN/VC-R-CNN.

Causal Attention

Attention is commonly used in vision-language tasks, as it creates stronger relationships within an image as well as between the image and text. However, as with other techniques in deep neural networks, attention is prone to exploiting spurious correlations learned during the training process which may not hold in the deployed setting. The authors of (X. Yang et al. 2021) propose a general causal attention (CATT) module intended to remove the impact of confounders. CATT is designed similarly to the attention mechanism used in vision Transformers; however, it is used to intervene by using a front-door adjustment, reducing the effect of confounders without needing explicit information about them (unlike VC R-CNN).

CATT does this by introducing additional sampling from other observations (dubbed cross-sample sampling, or CS-Sampling), which stratifies the input values in order to produce a deconfounded predictor. The authors provide the following illustration for this technique:

> Intuitively, CS-Sampling approximates the "physical intervention" which can break the spurious correlation caused by the hidden confounder. For example, the annotation "man-with-snowboard" is dominant in captioning dataset [19] and thus the predictor may learn the spurious correlation between the snowboard region with the word "man" without looking at the person region to reason what actually the gender is. CS-Sampling alleviates such spurious correlation by combining the person region with the other objects from other samples, e.g., bike, mirror, or brush, and inputting the combinations to the predictor. Then the predictor will not always see "man-with-snowboard" but see "man" with the other distinctive objects and thus it will be forced to infer the word "man" from the person region.

This example shows the intuition behind the front-door adjustment happening in CATT along with its purpose of producing a deconfounded predictor. CATT achieves this by projecting the regions of interest for the training set as embeddings; the in-sample sampling (IS-Sampling) focuses only on the input image, while CS-Sampling uses other related images from the training set as the keys and values in the attention operation. Their method extends to both *self-attention* (regions of the image attending to each other) and *top-down* attention (the image attending to the embedded text), and it can be stacked in deep networks such as Transformers. The authors show an improvement over standard VQA methods as well as the ability to improve smaller architectures by adding CATT. Their code is available at https://github.com/yangxuntu/lxmertcatt.

Counterfactual Visual Question Answering

The final causal method for vision-language we will discuss is rather different from CATT and VC R-CNN. One aspect of visual question-answering which is not covered in the previously discussed methods is the risk of language bias affecting the performance of the VQA model. Language bias can occur in many ways, including the nature of the questions in the dataset. The authors of (Niu et al. 2021) focus largely on this source of bias, pointing out for example that 90% of the "do you see a…" questions in the VQA v1.0 dataset are correctly answered as "yes," which incentivizes the model to focus on the type of question rather than the contents of the image. The authors propose a counterfactual approach to understanding and removing language bias in VQA, effectively allowing the model to imagine what would have happened, for example, if the question had arrived with no visual evidence.

Specifically, (Niu et al. 2021) frames the causal question of language bias in terms of commonly used counterfactual quantities capturing both *direct* and *indirect* effects. This approach seeks to disentangle the impact of three factors on the answer A produced in the VQA setting:

1. The input image in isolation, V
2. The input question in isolation, Q: the pathway of language bias
3. The model's ability to perform multimodal reasoning, K: the primary objective of VQA

Readers familiar with NLP or attention mechanisms will notice that the variable names Q, K, and V also correspond to the query, key, and value components of the attention function. This is by design, as it is helpful to delineate the flow of information through the VQA process. In this case, the direct effect of $Q \rightarrow A$ is the undesirable effect: we do not want the answer to be determined in any large part by the question in isolation – instead, we want the model to understand the question and assess the input image to determine the appropriate answer. The authors approach this by quantifying the direct and indirect effects of Q on A and selecting the generated answer which produced the largest total indirect effect, an effective alternative for sampling based on the largest posterior probability.

6.2.5 Robustness and Transfer

The prior sections have discussed methods leveraging causal techniques to improve model performance within certain subdomains of CV. In this section, we will take a broader look at efforts focused within model robustness and transferability.

Despite the various directions of success in CV, most of the existing work on robustness in CV has been focused on applications in image classification. As a result, most of the research in causal-based robustness also applies to image classification. In this section, we will detail two such approaches: one for removing the impact of spurious correlation on features like texture and background, and another for evaluating model generalization via causal methods. As the world of computer vision continues to evolve, so too will efforts in robustness.

Counterfactual Data Augmentation for Robust Image Classifiers

The issue of complex, greedy ML models latching onto spurious correlations is not unique to computer vision, but it is a common problem facing CV practitioners. (Sauer and Geiger 2021) introduces Counterfactual Generative Networks, a GAN-based approach of creating counterfactual images that disentangle and separately model an image's shape, texture, and background. The supervision for this model is only supplied by the image's class label and the inductive biases from the orchestration of the GANs which leverage pretrained models, and the result is a series of models which can generate an image with specified shape, texture, and background. Figure 6.4 shows somewhat of an extreme example: the generated image from the authors' prompt of an "ostrich" shape with "strawberry" texture on a "diving" background.

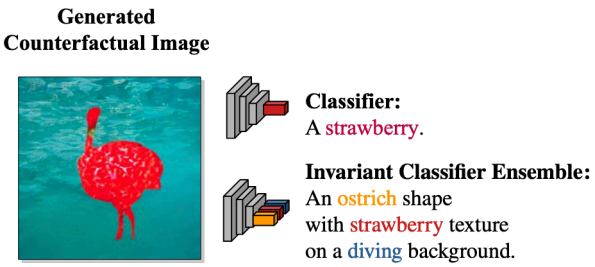


Figure 6.4: An image generated by Counterfactual Generative Network

Using the generative models, the authors generate synthetic training data for the image classifier, such MNIST digits with slightly different colors or shapes, or animals with different backgrounds. The authors show that this additional information helps the model generalize better based on test set performance, providing an interesting direction for future work in causal robustness and counterfactual modeling.

Causal Inference for Model Generalization

Another important focus of robustness is centered around model transferability. This consists of two primary goals:

1. Ensuring that a trained model transfers well to new data
2. Understanding when and how things go wrong in the event that the model fails to transfer

In the context of computer vision, models can often generalize properly on new data drawn from the same distribution as the training set. However, performance often degrades substantially when attempting to generalize on data from another distribution. Using our earlier example, a model trained to identify Staffordshire bull terriers like the one in Figure 6.1 could perform well on images of puppies being held by their breeders, but might not generalize to adult dogs photographed in slightly different settings.

One way to view the issue of model transfer is through the lens of *transportability*. Transportability is a causal term used to describe whether information learned in one environment is applicable in another environment (Judea Pearl and Bareinboim 2011). A lack of transportability in image classification means that there is some material difference in the relationship between images and their labels when comparing the in-distribution and out-of-distribution settings. (Mao et al. 2022) demonstrate this by first developing a causal graph to represent the out-of-distribution transfer process, which they use to show that the primary source of generalization failure is due to a lack of transportability in the association between images and labels.

In probabilistic terms, this issue of transportability means that the conditional probability $P(Y|X)$ learned by the model in the in-distribution context does not match what is encountered out-of-distribution. After establishing that there is an issue of transportability, the authors decompose $P(Y|X)$ into two components for each setting:

1. The causal effect: an invariant quantity which is consistent in both the in-domain and out-of-domain setting (for example, the shape of a dog's muzzle, or the color and length of its fur)
2. The spurious effect: the part of the distribution which is inconsistent between contexts, leading to generalization failure (for example, different backgrounds or lighting)

The authors proceed to show that it is possible to estimate this invariant causal effect – that is, $P(Y|do(X))$ – for the image classification task without observing additional variables. This is made possible by adding a causal structure to the representations learned by deep neural networks, which is built upon the following assumptions:

- That the data generating process does indeed follow the decomposition outlined above (i.e. the images are made up of causal and spurious features)
- That a neural model can sufficiently learn representations which contain relevant the causal factors
- That the classifier will mimic the behavior of the true labeler when combining the causal factors from an in-distribution image x with the spurious factors of an out-of-distribution image x'

For settings where these assumptions hold, the authors present a framework to be used during training and estimation which can identify the causal transportability effect without needing additional information. During training, the framework leverages either a variational autoencoder (VAE) or another pretrained encoder to produce representations for the training images. The classifier then learns to categorize images based on the combination of representations of an image with other inputs from related training images. By doing this within-class sampling, the classifier learns more robust features about the image's subject, allowing for more robust transfer. A similar sampling process is used to estimate $P(Y|do(X))$ during evaluation to assess the transportability of the image classifier and produce more robust predictions.

The work introduced in (Mao et al. 2022) provides a promising direction for reducing the impact of spurious correlation in image classification, and their results on adversarial domain generalization tasks show that the framework provides a meaningful boost in generalization. We will now provide a case study demonstrating the process on one of these tasks.

6.3 Case Study

This case study can be accessed via Google Colab, with a walk-through here as well.

The case study for this chapter includes a simple reimplementation of (Mao et al. 2022). The case study illustrates their transportable classification method on Colored MNIST (Arjovsky et al. 2020), a benchmark for evaluating classifier robustness using adversarial color shifts between training and test data.

The Colored MNIST dataset starts with the standard grayscale handwritten digits in the MNIST data, and it applies background coloring differently according to the dataset: training images have coloring applied according to their classes, and test images are colored randomly. The two transformations applied to the training set are shown in Figure 6.5

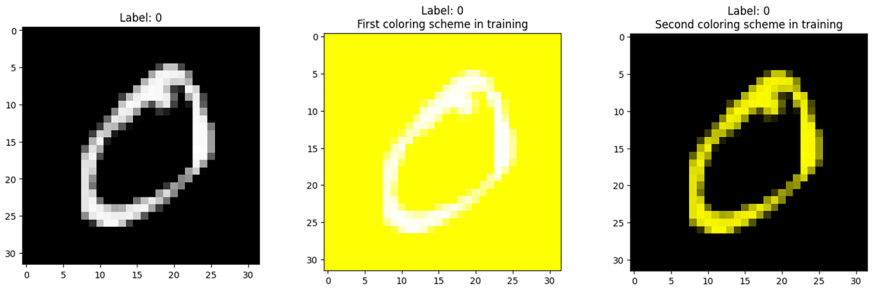


Figure 6.5: An image in the MNIST training set with two color shifts applied

We can see that one variation has the digit in white on a yellow background, while the other has the digit in yellow on a black background. These are the same image, and both variations are present in the training set. Another example of this coloring scheme can be seen in Figure 6.6, where there are two training examples labeled as the digit `7`, each representing a different coloring for sevens. Also present in Figure 6.6 is a test image with the label of `7`, and we can see that its coloring matches neither of the training sevens.

The causal transportability methods rely on the use of an image encoder; we follow the original implementation and use a variational autoencoder. However, there is nothing special about the autoencoder, and other image encoders (e.g. a ResNet) can be used. Figure 6.7 shows an image from the Colored MNIST training set along with its reconstructed representation produced by the VAE.

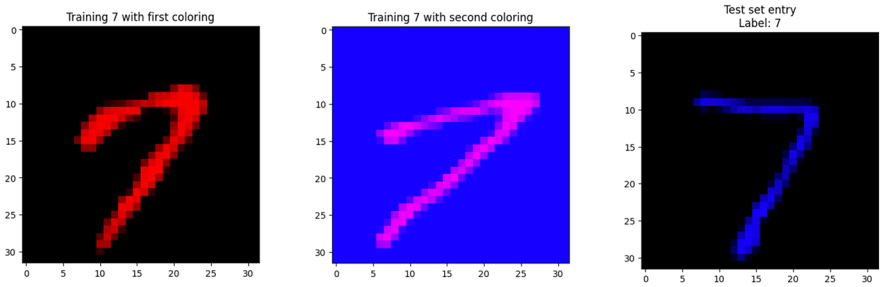


Figure 6.6: Three images labeled “seven” in the Colored MNIST dataset

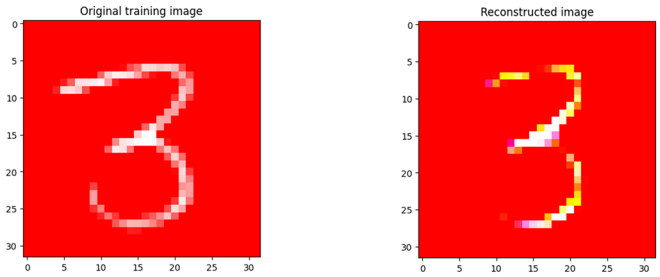


Figure 6.7: An image from the training set (left) with its reconstructed autoencoder output (right)

The autoencoder is used to encode the input images. The causal classifier then uses fragments of similar training images with the same label to force the model to learn the relevant features from the input batch. Unfortunately, this does not seem to outperform a non-causal benchmark in our experiment: our non-causal model uses an architecture which is similar to the one used by the causal classifier. Figure 6.8 shows the training and test set performance of each of these classifiers on the Colored MNIST dataset.

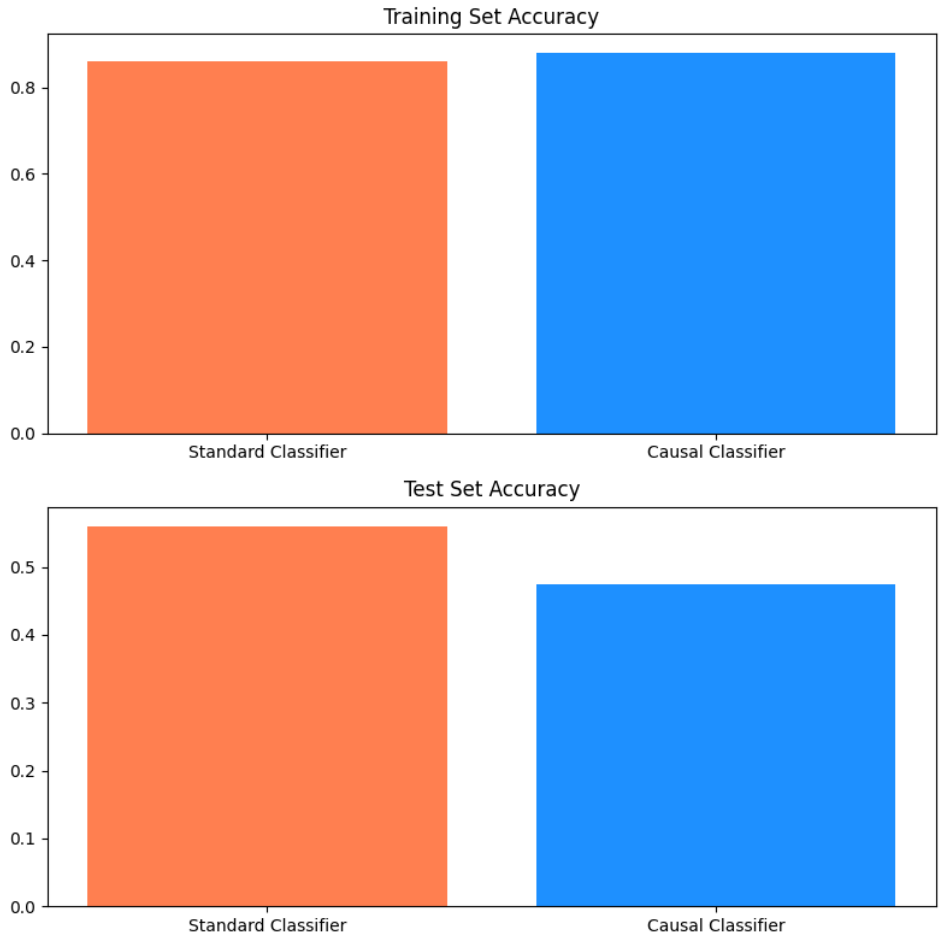


Figure 6.8: Model performance on Colored MNIST

As we can see, the causal classifier has worse performance on the test set than the non-causal benchmark. This gap in performance could be due to a number of reasons, including the difference in baseline used in the original implementation, the modifications applied to the original Colored MNIST dataset, or the design of the encoder. For example, perhaps a CNN- or attention-based encoder would perform better.

Despite these classification results, causality in computer vision is still an exciting area of

ongoing work, presenting challenges and opportunities which could lead to meaningful gains in robustness in applied CV. Importantly, as we can see in the case study code, causal methods in computer vision can be implemented in ways that look largely similar to other standard deep learning code – for instance, the model classes and training loop for the causal classifier looks very similar to the PyTorch code used to train the non-causal classifier. Making such methods accessible for use in other applications will foster additional experimentation with these methods.

6.4 Conclusions

In this chapter, we have introduced the reasons why practitioners in computer vision would want to use causality in their work. Causal methods have produced promising initial results across several areas of computer vision, with much work focusing on image classification and vision-language tasks.

We also demonstrated one of these methods in a case study. This case study introduces the Colored MNIST dataset and compares a causal transportability approach against a standard image classifier. While the causal method does not provide a performance boost over the non-causal method, it provides a data point and streamlined PyTorch implementation for future experimentation.

7 Time-dependent Causal Inference

The methods discussed in this book thus far don't explicitly include the window of time between events with a causal relationship when computing things like average treatment effects. That makes it difficult to apply such methods to time-series data, where the time it takes for a cause to produce its effect can be important and the time interval between cause and effect can impact the probability of the effect's occurrence. This chapter focuses on methods that accounts for these temporal relationships and also describes a method of automated reasoning that represents causal relationships as propositions of formulas in probabilistic temporal logic.

7.1 Probabilistic Computation Tree Logic

7.1.1 Temporal Logic

Temporal logic is a formal system for expressing and reasoning about propositions that hold at different points in time. It provides a set of operators and symbols for specifying temporal relationships that lets us represent causal relationships as temporal logic formulas. These formulas can then be manipulated and evaluated to analyze patterns in temporal data, such as the timing and sequence of events, and infer causal relationships between those events. However, this does not allow for probabilistic relationships.

If we want an automated way of capturing the temporal and probabilistic relationships between events, then we need to describe temporal relationships between variables, in the presence of uncertainty. This is provided by probabilistic temporal logic (PTL).

7.1.2 Probabilistic temporal logic

Probabilistic temporal logic (PTL) extends traditional temporal logic by incorporating probability. In PTL, propositions are assigned probabilities rather than being considered purely true or false. This allows for more nuanced reasoning about the temporal relationships between events. For example, if event A *typically* happens before event B but not always, we can express this uncertainty using probabilities. However, PTL alone does not allow us to study how the system evolves in time, since a probabilistic system can have many possible futures.

By combining probabilistic temporal logic with computation tree logic (CTL) (Clarke and Schlingloff 1996), we get probabilistic computation tree logic (PCTL) (Hansson and Jonsson

1990), which is a probabilistic extension of CTL that is used to reason about the behavior of probabilistic systems that evolve in time. PCTL can be used to specify and verify various properties of probabilistic systems, such as the probability of reaching a certain state, the expected time to reach a certain state, and the long-term behavior of the system. PCTL can also specify temporal constraints on the behavior of a probabilistic system, answer questions about possible futures, and compute the likelihood of a possible future event happening in a specific time window.

7.1.3 Probabilistic Kripke Structures

Probabilistic Computation Tree Logic describes a system's possible transitions with a probabilistic Kripke structure (PKS), which is a directed graph where each node represents a possible state of the system, and each edge (1) represents a probabilistic transition between states and (2) is labeled with a proposition that describes the conditions under which the transition can occur.

By modeling the behavior of a probabilistic system, the PKS allows us to define the truth of temporal statements with respect to the structure. For example, a temporal statement such as *"eventually, property P holds with probability at least 0.9"* is true in a state of the structure if there is a path from that state to a future state where property P is true with probability at least 0.9. PKSs are also used in the formal verification of probabilistic systems, where they provide a way to model system and verify that it satisfies a given specification.

7.1.3.1 Describing the System

A system in PCTL is described using two components: a set of atomic propositions and a probabilistic Kripke structure. In PCTL, what we know about a variable that might appear in a typical causal graph, is written as a logical proposition that can also have a probability and temporal aspects.

The set of atomic propositions is commonly written as A and the PKS is written as a four-tuple $K = (S, s_i, L, T)$ (Kleinberg 2012), where

- S is a finite set of states
- s_i is the initial state
- $L : S \to 2^{|A|}$ is a labeling function that tells us which atomic propositions are true at each state
- $T : S \times S \to [0, 1]$ is a probabilistic transition function, where $\forall s \in S, \sum_{s' \in S} T(s, s') = 1$

K defines which transitions are possible and the probability of each transition and $L(s)$ is the label(s) of state s. Note that, in practice, we will not define or infer K.

7.1.3.2 Example: The Monty Hall Problem

This section presents a detailed example of how you would describe a system with a probabilistic Kripke structure (PKS). Our example is the famous Monty Hall problem. The Monty Hall problem is a probability puzzle based on a game show called "Let's Make a Deal," which was hosted by Monty Hall. The problem is named after him and became popular after a reader's question appeared in Marilyn vos Savant's "Ask Marilyn" column in Parade magazine in 1990 (contributors 2023).

The problem is as follows:

1. There are three closed doors: Door 1, Door 2, and Door 3.
2. Behind one of the doors, there is a car (the desired prize), and behind the other two doors, there are goats (undesired prizes).
3. The player chooses one of the doors (say, Door 1).
4. The host, Monty Hall, who knows what's behind each door, opens one of the remaining doors (say, Door 2) to reveal a goat.
5. The host then asks the player if they want to switch their choice to the other unopened door (in this case, Door 3) or stick with their original choice (Door 1).

The Monty Hall problem asks the player to determine the best strategy: should they stick with their original choice or switch to the other unopened door? We will not discuss the solution to the problem here. Instead, we describe the problem as a probabilistic Kripke structure.

Monty Hall System States

First, we need a set of atomic propositions, A, that is representative of the problem:

Proposition	**Description**
CarBehindDoor1	The car is behind door 1
CarBehindDoor2	The car is behind door 2
CarBehindDoor3	The car is behind door 3
PlayerChoosesDoor1	The player initially chooses door 1
PlayerChoosesDoor2	The player initially chooses door 2
PlayerChoosesDoor3	The player initially chooses door 3
PlayerSwitches	The player switches their choice of door
HostOpensDoor1	The host opens door 1 to reveal a goat
HostOpensDoor2	The host opens door 2 to reveal a goat
HostOpensDoor3	The host opens door 3 to reveal a goat
PlayerWins	Player's final choice is door with the car
PlayerLoses	Player's final choice is door with a goat

Next, we need a finite set of states. In the Monty Hall problem, the state of the system can be described from the perspective of the player or the host. The host already knows which doors

the car and goats are found behind. The player does not have this information at the start of the game. From the player's perspective, the state of the system is defined by a combination of the truth values of the atomic propositions in the above table. In other words, the state is what the player knows to be true and false.

Since the Monty Hall problem is simple, we can describe the sequence of states in a simple table. Below, we present three sample runs of the game, in table form. The first column lists the atomic propositions, the remaining columns are labeled with the time steps showing the five steps in the game. When a column has an entry of 1, it means the proposition is true at that time step. The first column in a row with a 1 is the time step where the proposition transitions from false to true. The value of each proposition at each time step shows which transitions are possible.

Game 1

Car is behind door 1. Player chooses door 2, host opens door 3, player chooses to switch doors, player wins.

Proposition	t_1	t_2	t_3	t_4	t_5
CarBehindDoor1	1	1	1	1	1
CarBehindDoor2	0	0	0	0	0
CarBehindDoor3	0	0	0	0	0
PlayerChoosesDoor1	0	0	0	1	1
PlayerChoosesDoor2	0	1	1	0	0
PlayerChoosesDoor3	0	0	0	0	0
PlayerSwitches	0	0	0	1	0
HostOpensDoor1	0	0	0	0	0
HostOpensDoor2	0	0	0	0	0
HostOpensDoor3	0	0	1	0	0
PlayerWins	0	0	0	0	1
PlayerLoses	0	0	0	0	0

Game 2

Car is behind door 2. Player chooses door 2, host opens door 1, player chooses to switch doors, player loses.

Proposition	t_1	t_2	t_3	t_4	t_5
CarBehindDoor1	0	0	0	0	0
CarBehindDoor2	1	1	1	1	1
CarBehindDoor3	0	0	0	0	0
PlayerChoosesDoor1	0	0	0	0	0
PlayerChoosesDoor2	0	1	1	0	0

Proposition	t_1	t_2	t_3	t_4	t_5
PlayerChoosesDoor3	0	0	0	1	1
PlayerSwitches	0	0	0	1	0
HostOpensDoor1	0	0	1	0	0
HostOpensDoor2	0	0	0	0	0
HostOpensDoor3	0	0	0	0	0
PlayerWins	0	0	0	0	0
PlayerLoses	0	0	0	0	1

Game 3

Car is behind door 2. Player chooses door 3, host opens door 1, player chooses to switch doors, player wins.

Proposition	t_1	t_2	t_3	t_4	t_5
CarBehindDoor1	0	0	0	0	0
CarBehindDoor2	1	1	1	1	1
CarBehindDoor3	0	0	0	0	0
PlayerChoosesDoor1	0	0	0	1	1
PlayerChoosesDoor2	0	0	0	0	0
PlayerChoosesDoor3	0	1	1	0	0
PlayerSwitches	0	0	0	1	0
HostOpensDoor1	0	0	1	0	0
HostOpensDoor2	0	0	0	0	0
HostOpensDoor3	0	0	0	0	0
PlayerWins	0	0	0	0	1
PlayerLoses	0	0	0	0	0

The initial state is that all doors are closed. In the next time step, t_1, the player chooses a door, making one of the propositions `PlayerChoosesDoor<n>` true ($n = 1, 2, 3$). Next, the host opens one of the other two doors that is hiding a goat, thereby setting one of the `HostOpensDoor<m>` propositions to true ($m = 1, 2, 3$) and one of the `CarBehindDoor<k>` propositions to false ($k = 1, 2, 3$). Finally, the player can switch to the remaining closed door, which will set `PlayerSwitches` to true, or they can keep their initial selection. The final state is either `PlayerWins` or `PlayerLoses`. Lastly, note that the position of the car is unknown to the player at the start of the game. If we rewrite the three example games above from the perspective of the player, then we would have to mark the `CarBehindDoor<n>` propositions as unknown. Note that we could also add propositions for the positions of the two goats, but those are complements for the position of the car. We mention this to note that the atomic propositions used for a problem are not necessarily etched in stone, but depend on how you wish to describe the problem and the states of the system at each time step.

7.1.3.3 Properties of State and Path Formulas

Computation tree logic has formulas that state what must be true for a certain state (state formulas) and formulas that define what must be true for an entire sequence of states (path formulas). For example, stating that one of the atomic proposition in A must be true in a given state can be specified as a state formula. To make it more concrete, consider the Monty Hall problem.

We can summarize properties of state and path formulas as follows:

1. Each proposition in A is a state formula
2. If f and g are state formulas, so are $\neg f$, $f \vee g$, $f \wedge g$, and $f \rightarrow g$.
3. If f and g are state formulas, and t is a non-negative integer or ∞, then $f\, U^{\leq t} g$ and $f\, W^{\leq t} g$ are path formulas
4. If f is a path formula and $0 \leq p \leq 1$, $[f]_{\geq p}$ and $[f]_{>p}$ are state formulas.

The latter three properties in the above list need to be explained:

Property (2) says that state formulas can be combined to make new state formulas using the standard propositional logic operations of *or*, *and*, and *not* and from the state transitions allowed by computation tree logic. The state is simply a collection of statements that are true at that time, which means we can use this property to automatically label states (Kleinberg (2012)).

The U and W shown in property (3) are the *until* and *unless* operators of PCTL, respectively. $f\, U^{\leq t}\, g$ is a shorthand that means f must be true at every state along the path *until* there is a state where g becomes true, and this must happen within t time steps. $f\, W^{\leq t}\, g$ is a shorthand that means f must be true at every state along the path for at least t time steps *unless* g becomes true within t time steps. For this reason, W is also called the *weak until*, hence it's choice of letter.

Property (4) tell us that we can construct state formulas out of the *until* and *unless* path formulas by adding probabilities to them. For example, $[f\, U^{\leq t}\, g]_{\geq p}$, which can also be written as $f\, U^{\leq t}_{\geq p}\, g$, means that with a minimum probability of p, g will become true within t time steps and f will remain true along the path until that happens. This is a state formula, whose probability is calculated by summing the probabilities of the paths that originate from this state. Note that a path probability is the product of transition probabilities along the path.

Before moving on to considering how we can identify causes, we need to introduce one more temporal operator, $\rightsquigarrow$, "leads to". $f_1 \rightsquigarrow^{\leq t}_{\geq p} f_2 \equiv AG[f_1 \rightarrow F^{\leq t}_{\geq p} f_2]$, means that for every path from the current state, states where f_1 is true leads to states where f_2 is true, and that the transitions between those states occur within t time step, with probability p. Given this definition, consider two formulas, a and b, that occur at times t_1 and t_2, respectively. This relationship can be described by the time interval between them, $|t_1 - t_2|$. If we want a to simply occur at any time before b, then $|t_1 - t_2|$ can be replaced with ∞ (Kleinberg and Mishra 2010).

7.2 Logical Conditions for Causality

The logical framework we have been discussing, PCTL, lets us describe probabilistic relationships between events in time as formulas in temporal logic. This reframes the problem in the language of model checking (Clarke and Schlingloff 1996), allowing us to verify which formulas are true in relation to the model. Note that the "model" is the collection of information described in Section 7.1.3. Section 3.3.1 of (Kleinberg 2012) discusses the algorithm for checking the set of propositions against a model. We will not cover model checking in this chapter, since, in practice, we will not have a model that tells us things like the set of possible states, labels, and transition probabilities. Instead these are found from the data itself. We can start with identifying potential causes, we can then identify which potential causes are likely to be true causes.

7.2.1 Identifying Potential Causes

This section describes how to find things that *might* be causes. The literature calls these *prima facie* causes. *Prima facie* is Latin for "at first sight" or "on the face of it." In the legal world, the term refers to a case that, based on the evidence presented, *appears* to be valid. Consider two PCTL formulas, c and e. c is a potential cause of e if there is a probability p such that *all* of the following conditions hold:

1. $F_{>0}^{\leq\infty} c$
2. $c \rightsquigarrow_{\geq p}^{\geq 1, \leq\infty} e$
3. $F_{<p}^{\leq\infty} e$

Condition (1) means that there is at least one state where c will eventually be true with probability $p > 0$, condition (2) says that c being true "leads to" a state where e is true in the future, with probability $p > 0$, and condition (3) says that the probability of the system eventually transitioning from the current state to one where e is true is less than p. The combination of these three conditions means that c must be true at least once and that the conditional probability of e given c is higher than the marginal probability of e. By using these conditions, we can generate a list of potential causes that must be filtered for their level of significance.

7.2.1.1 Causal significance

In Section 2.5, we learned about the average treatment effect as a way of quantifying the relationship between a treatment (which is a potential cause) and and outcome (an observed effect). (Kleinberg and Mishra 2010) introduces a quantity called the average causal significance (ACS), ϵ_{avg}, that quantifies the causal significance of a potential cause c to an effect e,

where $c \rightsquigarrow \geq p^{\geq r, \leq s} e$; i.e. after c becomes true, e occurs in a time window $[r, s]$, with probability p:

$$\epsilon_{avg}(c_{r-s}, e) = \frac{\sum_{x \in X \ c} P(e|c \wedge x) - P(e|\neg c \wedge x)}{|X \ \ c|}, \tag{7.1}$$

where X is the set of potential causes, c is the potential cause in X that is under consideration, e is the effect. Note that $X \ \ c$ means c is being excluded from the set. The numerator in the ACS is the difference in the probability of the effect with and without the potential cause, and is conceptually similar to the average treatment effect. More formally, the numerator is the difference between the probability of the system moving to a state where e is true from one where c and x are true and the probability of the system moving to a state where e is true from one where x is true but c is not true.

7.3 Identifying Token Causes

7.3.1 Types of Causes

There are two types of causal relationships: token-level and type-level (Kleinberg 2012). Type-level relationships describe general relationships between variables, indicating a causal relationship that holds across multiple events. In contrast, token-level relationships are specific to individual events and refer to the causal relationship between a particular cause and effect pair. In this chapter, we have focused on type-level relationships.

7.3.2 Defining Token-level Causal Significance

(Zheng and Kleinberg 2017) shows that the ACS can be used to compute the significance of a token-level cause. The significance of c at time t' of e at time t, where $c_{t'}$ is a specific instance of cause type c, is given by

$$S(c_{t'}, e_t) = \epsilon_{avg}(c_{r-s}, e) \times P(c_{t'}|\mathcal{V}) \times f(c_{t'}, e_t, r, s) \tag{7.2}$$

where $\mathcal{V}$ is the sequence of time series data, ϵ_{avg} is the average causal significance of type-level cause c on effect e, $P(c|\mathcal{V})$ is the probability of the token cause $c_{t'}$ given the sequence of observations, and f is a weighting function that captures how closely the time gap between $c_{t'}$ and e_t, $t - t'$, fits into the type-level window $[r, s]$. There are two constraints on f:

1. When $t^{\in}[t' + r, t' + s]$, $f(c_{t'}, e_t) = 1$
2. f decreases monotonically outside of the specified range range
3. $f(c_{t'}, e_t) \in [0, 1]$

7.3.3 Computing Significance of Token Causes

To compute the significance of a token cause, we need to know f and the average causal significance of the corresponding type-level cause. (Zheng and Kleinberg 2017) describes an algorithm for computing f and an algorithm for finding token-level causes. They rest upon the assumption that token causes are observed in the data. Computing token significance gives us the token-level causes. The algorithm for finding token-level causes limits itself to token causes with a single variable.

7.3.3.1 Computing f from data

This section describes the algorithm from (Zheng and Kleinberg 2017) for estimating f from a time-series dataset.

Algorithm 1: compute_$f(V, T, H, l_{max}, D)$

Input:

- V, a set of variables
- T, length of the time series
- H, a set of all causal relationships between $v \in V$
- l_{max}, the maximum time lag to test
- D, a $V \times T$ matrix, stores value of each variable at each time step

Output:

- F, a list with a function f for each relationship in H

Procedure:

1. for each causal relationship $h \in H$ for which $c \rightsquigarrow {}^{\geq r, \leq s}_{\geq p} e$, do
    1. Calculate ϵ_{avg} for each lag $l \in [1, lmax]$ using D
    2. Normalize ϵ_{avg} by dividing by its maximum value
    3. Delete outliers where $\epsilon_{avg} < 0$
    4. $f(c_{t'}, e_t, r, s) = 1$ when $(t - t') \in [r, s]$
    5. $f(c_{t'}, e_t, r, s)$ for $(t-t') \in [1, r) \bigcup (s, lmax]$ is fit to ϵ_{avg} with nonlinear least squares. Then add f to F.
2. return F

Two explanatory notes:

- The time lag mentioned above is the time difference between a potential cause and the effect. Recall that each causal relationship being considered has a time window $[r, s]$ where any time difference in the window is allowed. Each interval that is in the window is a time lag.
- The matrix D is a matrix representation of the time series, where each row represents a variable in $\mathcal{V}$ and each column represents one time step in the time series.

Algorithm 1 gives us f and ϵ_{avg}, which are two of the three things we need to compute the token significance, $S(c_{t'}, e_t)$. The remaining quantity is $P(c|\mathcal{V})$, the probability of token cause $c_{t'}$, given $\mathcal{V}$. That can be counted from the dataset.

7.3.4 Finding Token-level Causal Explanations

This section describes a method for identifying and removing known causes from the set of events to be explained. Then, the remaining events are tested to see if any variables can explain them. The set of events that cannot be explained by the known causes is the residual set, and the causes of these events are to be found without removing any potential causes of other events. Variables that are not in their initial state or have recently changed state are considered as possible causes.

Algorithm 2: `find-novel-causes`$(V, T, D, k, m, lmax, E)$

Input:

- V, a set of variables
- T, length of the time series
- D, a $V \times T$ matrix, stores value of each variable at each time step
- k, number of explanations to select (for top-k)
- m, max time lag for detecting state change
- l_{max}, the maximum time lag for finding novel explanation
- E, set of explained events (pairs of form: $(c_{t'}, e_{t'})$)

Output:

- E', token causal explanations for each event

Procedure:

1. for each $e \in V$ do
    1. Use E to remove explained events to get matrix `De`
    2. for each $c \in V$ do
        1. for each $l \in [1, l_{max}]$ do
            1. Get $P(e|c_l)$. When c has a default state, use instances of c where c is not in its default state or changed state up to m time units before.

2. for each non-null $D_e[e, i]$, $1 \leq i \leq T$ do
    1. $L_i \leftarrow []$
    2. for each $D_e[c, j]$, where $i - l_{max} \leq j \leq i - 1$ do
    - Add $(c_j, e_i, P(e|c_j - i))$ to L_i
    3. Add top k explanations for event e_i to E' (tuples: $(c_j, e_i, P(e|c))$)
2. return E'

The output is a list of token causes, the events they explain, and their conditional probability.

7.4 Case Study

In this section, we walk through a detailed example of how to explain observed events in time-series data using type-level causal relationships (Algorithm 1). We also look at how to explain observed events that are *not* explained by type-level causes and instead explain them with token-level causal relationships (Algorithm 2). This case study can also be viewed on Google Colab.

7.4.1 Dataset

We use the bike sharing dataset that is in the UC Irvine Machine Learning Repository (Fanaee-T and Gama 2013). This is one of the datasets the algorithms above were applied to in (Zheng and Kleinberg 2017).

The dataset contains information on the hourly and daily count of rental bikes in the Capital bikeshare system between 2011 and 2012, including weather and seasonal data. It may provide insights into bike usage patterns and their correlation with environmental factors. We're interested in identifying causes for high bike rental numbers, using the hourly data.

7.4.2 Variables

The Bike Sharing dataset has several columns that we will use as causal variables, and three as possible causal effects. Each type of column is listed below. For more information about this dataset, visit the link above.

Effect variables

- casual: count of casual users
- registered: count of registered users
- cnt: count of (casual + registered) users

Causal variables

- season: `{1:"winter", 2:"spring", 3:"summer", 4:"fall"}`
- holiday: 1 if day is holiday else 0
- weekday: day of the week (1-7)
- workingday: 1 if neither weekend nor holiday else 0
- weathersit:
  - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
  - 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  - 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  - 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
- temp: Normalized temperature in Celsius
- atemp: Normalized feeling temperature in Celsius
- hum: Normalized humidity
- windspeed: Normalized wind speed

7.4.2.1 Loading the data

We will convert the following column names as follows:

- dteday → date
- yr → year
- mnth → month
- hr → hour
- holiday → is_holiday
- weekday → day_of_week
- workingday → is_workingday
- weathersit → weather
- hum → humidity
- causal → num_casual_users
- registered → num_registered_users
- cnt → num_total_users

Also, we'll convert the values in the `season` column into categorical values using this map: `{1:"winter", 2:"spring", 3:"summer", 4:"fall"}`. The values in the `year` column can be converted using this map: `{0: 2011, 1:2012}`.

```
import pandas as pd

def load_data(filename):
    df = pd.read_csv(filename)
    column_mapping = {
```

```
        'dteday': 'date',
        'yr': 'year',
        'mnth': 'month',
        'hr': 'hour',
        'holiday': 'is_holiday',
        'weekday': 'day_of_week',
        'workingday': 'is_workingday',
        'weathersit': 'weather',
        'hum': 'humidity',
        'casual': 'num_casual_users',
        'registered': 'num_registered_users',
        'cnt': 'num_total_users',
    }

    df.rename(columns=column_mapping, inplace=True)
    df['season'] = df['season'].map({
        1: 'winter',
        2: 'spring',
        3: 'summer',
        4: 'fall',
    }).astype('category')
    df['year'] = df['year'].map({0: 2011, 1: 2012})

    df['timestamp'] = pd.to_datetime(
        df['date'] + ' ' + df['hour'].astype(str) + ':00:00'
    )
    df.drop(['date', 'year'], axis=1, inplace=True)
    return df
```

We define binary-valued columns in a dataframe for each of the following causal variable:

- is_holiday
- is_workingday
- is_weekend
- is_raining
- is_bad_weather
- is_mild_precipitation
- is_daytime
- is_nighttime
- is_`<season>` (spring, summer, fall, winter)
- is_high_temp
- is_high_atemp
- is_low_temp

- is_low_atemp
- is_high_humidity
- is_low_humidity
- is_windy
- not_windy

For non-binary valued variables, we follow the process used in (Zheng and Kleinberg 2017) and divide the continuous variables into three bins of equal size.

```
def convert_to_bins(df, column):
    return pd.qcut(
        df[column],
        3,
        labels=['low', 'medium', 'high'],
    )

# convert the temp column into three bins
df_hour['temp_bin'] = convert_to_bins(df_hour, 'temp')

# convert the atemp column into three bins
df_hour['atemp_bin'] = convert_to_bins(df_hour, 'atemp')

# convert the humidity column into three bins
df_hour['humidity_bin'] = convert_to_bins(df_hour, 'humidity')

# convert the windspeed column into three bins
df_hour['windspeed_bin'] = convert_to_bins(df_hour, 'windspeed')
```

Binary columns are defined as follows:

```
def is_holiday(df, t):
    return int(df.loc[t, 'is_holiday'] == 1)

def is_workingday(df, t):
    return int(df.loc[t, 'is_workingday'] == 1)

def is_weekend(df, t):
    return int(df.loc[t, 'day_of_week'] in [0, 6])

def is_bad_weather(df, t):
    return int(df.loc[t, 'weather'] in [3, 4])
```

```
def is_mild_precipitation(df, t):
    return int(df.loc[t, 'weather'] == 3)

def is_rush_hour(df, t):
    _tush = [7, 8, 9, 17, 18, 19]
    return int(df.loc[t, 'hour'] in _rush)

def is_daytime(df, t):
    _daytime = [6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
    return int(df.loc[t, 'hour'] in _daytime)

def is_nighttime(df, t):
    _nighttime = [0, 1, 2, 3, 4, 5, 20, 21, 22, 23]
    return int(df.loc[t, 'hour'] in _nighttime)

def is_spring(df, t):
    return int(df.loc[t, 'season'] == 'spring')

def is_summer(df, t):
    return int(df.loc[t, 'season'] == 'summer')

def is_fall(df, t):
    return int(df.loc[t, 'season'] == 'fall')

def is_winter(df, t):
    return int(df.loc[t, 'season'] == 'winter')

def is_high_temp(df, t):
    return int(df.loc[t, 'temp_bin'] == 'high')

def is_low_temp(df, t):
    return int(df.loc[t, 'temp_bin'] == 'low')

def is_high_atemp(df, t):
    return int(df.loc[t, 'atemp_bin'] == 'high')

def is_low_atemp(df, t):
    return int(df.loc[t, 'atemp_bin'] == 'low')

def is_high_humidity(df, t):
    return int(df.loc[t, 'humidity_bin'] == 'high')
```

```
def is_low_humidity(df, t):
    return int(df.loc[t, 'humidity_bin'] == 'low')

def is_windy(df, t):
    return int(df.loc[t, 'windspeed_bin'] == 'high')

def not_windy(df, t):
    return int(df.loc[t, 'windspeed_bin'] == 'low')
```

We group these to make it easy to add columns to the dataframe:

```
labelers = {
    'is_holiday': is_holiday,
    'is_workingday': is_workingday,
    'is_weekend': is_weekend,
    'is_bad_weather': is_bad_weather,
    'is_mild_precipitation': is_mild_precipitation,
    'is_rush_hour': is_rush_hour,
    'is_daytime': is_daytime,
    'is_nighttime': is_nighttime,
    'is_spring': is_spring,
    'is_summer': is_summer,
    'is_fall': is_fall,
    'is_winter': is_winter,
    'is_high_temp': is_high_temp,
    'is_low_temp': is_low_temp,
    'is_high_atemp': is_high_atemp,
    'is_low_atemp': is_low_atemp,
    'is_high_humidity': is_high_humidity,
    'is_low_humidity': is_low_humidity,
    'is_windy': is_windy,
    'not_windy': not_windy
}

idx2labeler = {}
label_fns = []
for i, feature in enumerate(labelers.keys()):
    idx2labeler[i] = feature
    label_fns.append(labelers[key])
```

Lastly, we also make the continuous effect variables categorical:

```
def binning(x, mean, std):
    """
    Maps x to a string, based on the mean and standard deviation:
    low: 1 std dev below the mean
    high: 2 std dev above the mean
    medium: everything else
    """
    if x < mean - std:
        return "low"
    elif x > mean + 2*std:
        return "high"
    else:
        return "medium"

def label_by_std(df, col_name):
    """
    Labels column entries based on categorical standard deviation
    """
    std = df[col_name].std()
    mean = df[col_name].mean()
    is_low = lambda x: x < mean - std
    is_high = lambda x: x > mean + 2*std
    df[f'{col_name}_bin'] = df[col_name].apply(
        lambda x: binning(x, mean, std)
    )
    return df

df_hour = label_by_std(df_hour, 'num_total_users')
df_hour = label_by_std(df_hour, 'num_casual_users')
df_hour = label_by_std(df_hour, 'num_registered_users')
```

```
def high_total_users(df, t):
    return int(df.loc[t, 'num_total_users_bin'] == 'high')

def low_total_users(df, t):
    return int(df.loc[t, 'num_total_users_bin'] == 'low')

def high_casual_users(df, t):
    return int(df.loc[t, 'num_casual_users_bin'] == 'high')

def low_casual_users(df, t):
```

```
    return int(df.loc[t, 'num_casual_users_bin'] == 'low')

def high_registered_users(df, t):
    return int(df.loc[t, 'num_registered_users_bin'] == 'high')

def low_registered_users(df, t):
    return int(df.loc[t, 'num_registered_users_bin'] == 'low')
```

7.4.3 Tools and Library

We use the Pandas and Numpy libraries.

7.4.4 Procedure

7.4.4.1 Using type-level relationships to explain observed events

We use part of Algorithm 1 to identify significant type-level relationships from the time-series dataset that explain why bike rentals are high/low.

Here are the steps in the process:

1. Identify the type-level relationships in the dataset
2. Compute causal significance of the type-level relationships, $\epsilon_{avg}(c_{r-s}, e)$

We aim to evaluate the significance of each type-level cause for each instance of high bike rentals. For each instance of high bike rentals, we use a significance threshold or only keep the top k relationships to partition the relationships/events into those for which sufficiently significant type-level causes have been found and those for which sufficiently significant type-level causes have *not* been found. Any instances of high rentals that are not explained by type-level causes would be used as input for Algorithm 2.

7.4.4.1.1 Identify token-level relationships

We find type-level relationships by aggregating token-level events. We will pass through the dataset, and at each point, we will use the values of the various variables to identify *prima facie* causes. Recall that a *prima facie* cause is a type-level relationship of the form $c \rightsquigarrow \overset{\geq r, \leq s}{\geq p} e$

```
import numpy as np

V = len(labelers)  # number of variables
T = df_day.shape[0]  # number of time steps
```

```
D = np.zeros((T, V), dtype=np.int32)  # time-series

for t in range(T):
    for i in range(V):
        D[t, i] = label_fns[i](df_day, t)

variable_names = idx2labeler.values()
df_D = pd.DataFrame(D, columns=variable_names)

# add effect columns to the dataframe
# this makes it easy to filter to compute probabilities
def add_column(df, f):
    df[f.__name__] = [f(df_hour, t) for t in range(T)]
    return df

df_D = add_column(df_D, high_total_users)
df_D = add_column(df_D, low_total_users)
df_D = add_column(df_D, high_casual_users)
df_D = add_column(df_D, low_casual_users)
df_D = add_column(df_D, high_registered_users)
df_D = add_column(df_D, low_registered_users)
df_D.head()
```

Here we introduce several data structures that we will use to work with type-level and token-level relationships. First, we have two class that let us store the names of causal variables, so that we don't have to keep passing around strings.

```
from typing import List

VariableIndex = int


class BidirectionalDict(dict):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self._backward = {v: k for k, v in self.items()}

    def __setitem__(self, key, value):
        super().__setitem__(key, value)
        self._backward[value] = key
```

```
    def forward_lookup(self, key):
        return self[key]

    def backward_lookup(self, value):
        return self._backward[value]

    def keys(self):
        return list(super().keys())

    def values(self):
        return list(self._backward.keys())


class VariableStore:
    def __init__(self):
        self.storage = BidirectionalDict()

    def add(self, variable_name: str):
        if variable_name not in self.storage:
            self.storage[variable_name] = len(self.storage)

    def lookup_by_name(self, name: str) -> VariableIndex:
        return self.storage.forward_lookup(name)

    def lookup_by_index(self, index: VariableIndex) -> str:
        return self.storage.backward_lookup(index)

    def __len__(self) -> int:
        return len(self.storage)

    def __contains__(self, name) -> bool:
        return name in self.storage

    @property
    def names(self) -> List[str]:
        return sorted(self.storage.keys())

    @property
    def ids(self) -> List[VariableIndex]:
        return sorted(self.storage.values())
```

Next, we have data structures for time window, type-level and token-level causal relationships,

and significance scores.

```
from dataclasses import dataclass
from typing import List, Optional, Set, Union

class Window:
    """
    A window of time, of the closed interval [start, end]
    """

    def __init__(self, start: int, end: int):
        if start > end:
            raise ValueError("Window start must be <= than end")
        if start < 0 or end < 0:
            raise ValueError("Window start and end must be >= 0")
        self.start = start
        self.end = end

    def __repr__(self):
        return f"Window({self.start}, {self.end})"

    def __eq__(self, _value: object) -> bool:
        if isinstance(_value, Window):
            return (
                self.start == _value.start
                and self.end == _value.end
            )
        return False

    def __hash__(self) -> int:
        return hash((self.start, self.end))


@dataclass(frozen=True)
class CausalRelation:
    """
    A cause and effect pair
    """

    cause: VariableIndex
    effect: VariableIndex
```

```
@dataclass(frozen=True)
class TokenCause:
    """
    A token event that supports a potential cause
    """

    relation: CausalRelation
    t_cause: int  # time step where cause is true
    t_effect: int  # time step where effect is true

    @property
    def lag(self) -> int:
        """The lag between the cause and effect"""
        return self.t_effect - self.t_cause


class TypeLevelCause:
    r"""
    A potential cause of an effect, with a time window, a
    probability of occurrence, and a list of token events
    that support this type-level cause. In PCTL language,
    `c \leadsto {}^{\geq r, \leq s}_{\geq p} e`
    """

    def __init__(
        self,
        relation: CausalRelation,
        window: Window,
        prob: float,
        token_events: List[TokenCause],
    ):
        """
        Create a type-level cause

        Parameters
        ----------
        relation : CausalRelation, the cause and effect
        window : Window, the time window in which the effect
                 occurs after the cause
        prob : float, the probability of the effect occurring
               in the window
```

```
        token_events : List[TokenCause], the token events that
                       support this type-level cause
        """
        self.relation = relation
        self.window = window
        self.prob = prob
        self.token_events = token_events

    def __repr__(self):
        return f"TypeLevelCause({self.relation}, {self.window}, \
            {self.prob})"

    # define equality and hashing based on the relation, window,
    # prob, and token events
    def __eq__(self, other):
        if isinstance(other, TypeLevelCause):
            return (
                self.relation == other.relation
                and self.window == other.window
                and self.prob == other.prob
                and self.token_events == other.token_events
            )
        return False

    def __hash__(self):
        return hash((
            self.relation,
            self.window,
            self.prob,
            tuple(self.token_events),
        ))


@dataclass(frozen=True)
class CompositeScore:
    lag_scores: np.array

    @property
    def score(self):
        return np.sum(self.lag_scores)
```

We also added a class that identifies the causal relationships and computes their relative

significance. This makes it easy to compute things with a few simple method calls. The class, `CausalTree`, is many lines of code and so it can be found in a file `cause.py` in the same directory as the Jupyter notebook. Now we can identify the type-level relationships and find the most significant ones.

```
tree = CausalTree(df_D, list(variable_names), "high_total_users")
tree.build("high_total_users", max_lag=3)
tree.compute_significance()
tree.prune()

@dataclass
class ScoredTypeLevelCause:
    cause: TypeLevelCause
    score: float

    def __str__(self):
        return f"{self.cause}: {self.score:.2f}"

    def __repr__(self):
        return str(self)

scored_causes = []
for i, cause in enumerate(tree.type_level_causes):
    composite_score = tree.type_level_significance_scores[i]
    scored_causes.append(
        ScoredTypeLevelCause(cause, composite_score.score)
    )

scored_causes.sort(key=lambda x: x.score, reverse=True)

for c in scored_causes[:10]:
    cause = tree.get_variable_name(c.cause.relation.cause)
    effect = tree.get_variable_name(c.cause.relation.effect)
    print(f"{cause} => {effect}: {c.score:.2f}, \
        {c.cause.prob:.2f} p-value, {c.cause.window} ")
```

7.4.5 Discussion

The result of the previous command shows which type-level causes contributed the most to high bike rentals, and in which time window.

Part 3: Advanced Topics in Causality

The final two chapters discuss advanced topics in causal inference. These are areas of ongoing research including special cases in algorithmic bias and model fairness, as well as cutting-edge research in reinforcement learning.

8 Model Fairness

The ethical implications of widespread ML adoption in practice could easily fill this entire book. Models have an increasingly significant impact on human lives — from the content a user sees on social media to credit approval and health outcomes. As such, it is paramount that practitioners approach model creation and deployment in an intentionally ethical manner.

With this in mind, much has already been written about ML ethics more broadly; for the sake of maintaining a focus on practical causal techniques, we will not rehash the high-level concepts around ethical concerns of machine learning applications in full. We will instead assume the reader has some familiarity with these ethical issues, introduce necessary definitions, and focus on how causality can be used to measure and mitigate algorithmic bias in practice.

Specifically, this section covers causal techniques that aid in the assessment of model fairness as well as the creation of fair models. We will discuss the need for causal methods instead of standard statistical tools when evaluating potentially discriminatory effects.

8.1 Introduction to Model Fairness

8.1.1 Prerequisite Knowledge

In this chapter, we will assume that the reader has a general understanding of issues surrounding algorithmic bias. If this is not the case for you, we recommend Chapter 3 of the `fast.ai` book [1] (Howard and Gugger 2020) as a starting point.

For a deeper view, (Mehrabi et al. 2019) provides a thorough survey of these topics from the machine learning perspective, including helpful distinctions between the types of bias which stem from datasets, the algorithms themselves, deployment techniques, and environmental or social factors (and how these concepts differ from the purely statistical notion of *bias*). We will rely on these definitions throughout this chapter, as described in the next section.

We also assume familiarity with causal inference techniques which can be gained in earlier chapters of this book; specifically, the causal estimation process outlined in Chapter 3.

[1] This chapter, along with the rest of the fast.ai book, is also available in Jupyter Notebook format on their GitHub.

8.1.2 What is Model Fairness?

The term *algorithmic bias* has become somewhat of an umbrella term for general concepts around unethical applications of technology: for example, a recent blog post by Liberties EU ("Algorithmic Bias: Why and How Do Computers Make Unfair Decisions?" 2021) provides a definition of algorithmic bias that ranges from the infamous COMPAS recidivism algorithm to airbag systems being better adapted for male bodies than female bodies. Such a definition helps draw attention to disparities, but it is not precise enough to distinguish issues which are specifically relevant to machine learning

In this chapter, we use *model fairness* to focus on the concepts of algorithmic bias directly related to the development, training, and deployment of machine learning models: specifically that a ML model or system may produce outputs which perpetuate, reinforce, or amplify discriminatory behavior; and that the model's quality can vary across groups. The definitions in section 4 of (Mehrabi et al. 2019) provide a nuanced view of the aspects of fairness in machine learning, including the categorization of fairness at the individual (similar individuals should receive similar outcomes) and group (different groups should be treated equally) levels.

8.2 How is Fairness Measured?

Both (Mehrabi et al. 2019) and (Caton and Haas 2020) categorize the wide variety of existing fairness quantification methods. Each method provides a slightly different approach to the quantification of model fairness, and each comes with its own assumptions about the user's responsibility for assessing fairness: as an extreme example, one approach named "Fairness Through Unawareness" – described in (Kusner et al. 2017) – asserts that a model is fair as long as it does not explicitly use protected attributes as features. Most techniques, however, recognize that reasonably complex models have the ability to proxy protected attributes through more subtle interactions. Examples of such definitions are "Demographic Parity" and "Equality of Opportunity," which are also defined in (Kusner et al. 2017).

Many of these methods focus on the joint distribution of model errors with protected categories (and, in some cases, other relevant covariates). One example is *conditional statistical parity*, originally (Corbett-Davies et al. 2017), which states that, "For a set of legitimate factors L, predictor $\hat{Y}$ satisfies conditional statistical parity if $P(\hat{Y}|L = 1, A = 0) = P(\hat{Y}|L = 1, A = 1)$" (Mehrabi et al. 2019). Essentially, conditional statistical parity attempts to measure whether individuals in different groups have similar outcomes (e.g. likelihood of being correctly classified) when controlling for relevant covariates.

In addition to these aggregate methods, some fairness measures focus on the similarity of predictions among similar individuals, while others measure the quality of the model's calibration among individuals in differing groups.

Unfortunately, practical adoption of fairness measurements is not yet widespread at this time, and metric selection is inconsistent from one use case to the next. In other words, relatively few practitioners are using fairness methods at all, and when they do, they use different techniques with different assumptions. Before we dive into the current state of causal fairness metrics, we will take a moment to discuss why practitioners and researchers should choose causal methods over other techniques.

8.3 Why Use Causality?

With such a wide variety of available fairness evaluation methods, one could reasonably question whether causality should be preferred over "standard" non-causal techniques.

First, the statistical notion of omitted variable bias means that algorithmic fairness/bias estimates may be flawed if the user does not control for the necessary covariates. Simpson's Paradox describes reversals in estimated effects when changing the conditioning variables (Sprenger and Weinberger 2021), and such reversals can be disastrous when assessing possible discriminatory treatment by ML systems. For example, if one wishes to use conditional statistical parity to evaluate discriminatory behavior, the definition of "legitimate factors" may be up for debate, and the decision to include or exclude certain factors can significantly impact results. The practice of creating a graphical model for causal inference enumerates all assumptions about how the world works with respect to the analysis at hand (J. Pearl, Glymour, and Jewell 2016). The causal graph creates a reference point for the foundational assumptions that underlie a fairness analysis, and it can be debated and refined as needed by subject matter experts.

The other major benefit of framing model fairness questions as causal inference tasks is that we gain understanding for the actual mechanisms of bias. A correlational approach can describe whether an observed discrepancy exists, but cannot tell you why it exists or the degree to which such a discrepancy is actually causal in nature. This makes the use of causal language extremely important when describing unethical treatment: it allows us to articulate the mechanisms through which an algorithm might propagate biased outcomes.

To summarize, the answer to the question of *"why should I use causality?"* is ultimately the same for applications in model fairness as it is for any other use of causal inference:

1. The creation of a causal graph formalizes assumptions about how the world works (in this case, how bias might be perpetuated by a machine learning model or system)
2. Framing fairness analysis as a causal inference task allows us to understand and directly quantify mechanisms of bias instead of simply recognizing that a discrepancy exists

8.4 Causal Fairness Measures

8.4.1 Background

The first concerted efforts to estimate model fairness with causal methods involved framing the fairness problem in the same way as any other causal inference task and using existing metrics. (Kusner et al. 2017) suggests using Effect of Treatment on the Treated (ETT) (Judea Pearl 2009) to measure causal effect. However, ETT does not distinguish the effect that is caused specifically by the treatment as opposed to backdoor channels, so later work – as described in (J. Zhang and Bareinboim 2018) – used specific measures to highlight the mechanisms of discriminatory effect. This work used three causal measures to account for this distinction:

- Controlled direct effect (CDE) describes the effect of the treatment X on the outcome Y while holding everything else constant (hence the *controlled* aspect). As the name suggests, this "direct" measure intends to quantify the disparity that is due explicitly to X which, in the fairness setting, would be a protected category such as gender, race, or religion.
- Natural direct effect (NDE) is slightly different from CDE in that it describes the direct effect of $X \to Y$ when any mediators W in $X \to W \to Y$ are set to the values they would naturally obtain when intervening on X
- Natural indirect effect (NIE) measures the change in the outcome Y that is due to shifts in the mediators W while the treatment X remains constant.

These three metrics can be used to describe the amount and the nature of unfair treatment in many fairness settings. However, the underlying assumptions only hold when X does not have another parent node in the causal graph; if this is not the case, other sources of discrimination may exist which would not be captured by CDE, NDE, and NIE.

8.4.2 The Standard Fairness Model

It is often the case that researchers interested in studying potentially discriminatory behavior of a model may not have access to the inner workings of the model (e.g. proprietary data or system dynamics), which poses challenges for robust causal inference using standard tools. In (J. Zhang and Bareinboim 2018), the authors introduce the *Standard Fairness Model*, a causal model to be used generally for questions concerning ML model fairness – including when information about the model's decision-making process is limited.

The causal graph for the Standard Fairness Model is given in Figure 8.1, where X is the protected category, Y is the outcome, W is a possible set of mediators between X and Y, and Z is a possible set of confounders between X and Y.

Along with this causal model, Zhang and Bareinboim introduce a suite of counterfactual measures that account for the specific types of discriminatory effect. These quantities are

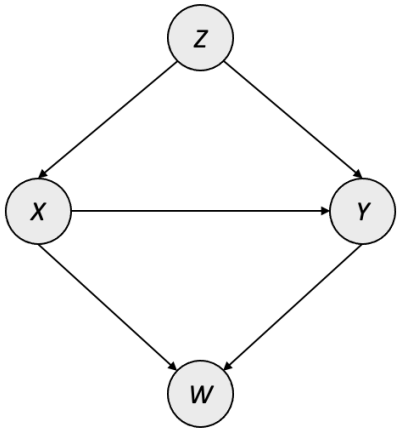


Figure 8.1: Causal graph for the Standard Fairness Model

similar in nature to CDE/NDE/NIE in that they quantify both direct and indirect effect, but Zhang and Bareinboim address the limitations of CDE/NDE/NIE by introducing a dedicated measure for the spurious effect that arises when confounders are present (i.e. $X \leftarrow Z \rightarrow Y$):

- Ctf-DE: the counterfactual *direct* effect of the treatment X on the outcome Y. This measures how Y changes when X goes from $x_0 \rightarrow x_1$ while mediators W retain the values they would have naturally attained at $X = x_0$. This is similar in nature to NDE, but it simply tries to capture "the existence of any disparate treatment" such that when Ctf-DE $\neq 0$, then there is evidence to indicate that some amount of disparity exists which is due directly to the protected category X
- Ctf-IE: the counterfactual *indirect* effect of the treatment X on the outcome Y. This is essentially a flipped version of Ctf-DE, meaning that instead of shifting X from $x_0 \rightarrow x_1$, the mediators W shift to what they would have naturally attained at $X = x_1$ while X remains at x_0. The counterfactual indirect effect seeks to measure discrimination through backdoor channels.
- Ctf-SE: the counterfactual *spurious* effect measures the residual or leftover effect which is due to confounders, or common ancestors, of X and Y. This is the missing component of the CDE/NDE/NIE framework, since it allows us to quantify differences due to spurious relationships between the protected factor and the outcome. Cases where no discrimination exists will have DE and IE equal to 0, with the entirety of the observed variation allocated to spurious effect.

The formulas for these measures are given below:

$$\text{Ctf-DE}_{x_0,x_1}(y|x) = P(y_{x_1,W_{x_0}}|x) - P(y_{x_0}|x)$$
$$\text{Ctf-IE}_{x_0,x_1}(y|x) = P(y_{x_0,W_{x_1}}|x) - P(y_{x_0}|x)$$
$$\text{Ctf-SE}_{x_0,x_1}(y) = P(y_{x_0}|x_1) - P(y|x_0)$$

If certain assumptions hold – namely those outlined in the relations of the variables as outlined in the Standard Fairness Model – these measures are directly estimable from observational data. The identification formulas for these metrics are given here:

$$\text{Ctf-DE}_{x_0,x_1}(y|x) = \sum_{z,w} [P(y|x_1,z,w) - P(y|x_0,z,w)]P(w|x_0,z)P(z|x)$$
$$\text{Ctf-IE}_{x_0,x_1}(y|x) = \sum_{z,w} P(y|x_0,z,w)[P(w|x_1,z) - P(w|x_0,z)]P(z|x)$$
$$\text{Ctf-SE}_{x_0,x_1}(y) = \sum_{z} P(y|x_0,z)[P(z|x_0) - P(z|x_1)]$$

These identification formulas provide a way to calculate discriminatory effect using observational data, which greatly simplifies the causal estimation in cases where the Standard Fairness Model applies.

One attractive quality of this set of measures is that they provide a decomposition for the total observed disparity. In many cases, this high-level, non-causal difference in outcomes between protected groups – $TV = P(y|x_1) - P(y|x_0)$ – is the first indicator that something may be amiss. These counterfactual measures break the total variation down into distinct components which quantify both the amount and the nature of any discriminatory effect that exists in the dataset. The decomposition of TV is provided here:

$$TV_{x_0,x_1}(y) = DE_{x_0,x_1}(y|x_0) - SE_{x_1,x_0}(y) - IE_{x_1,x_0}(y|x_0)$$

8.4.3 A Framework for Causal Fairness Analysis

In their latest work, Plečko and Bareinboim build upon the Standard Fairness Model to propose a standardized process for assessing model fairness with causality. We will provide a practical introduction to their tools, but we suggest that readers interested in going deeper with causal fairness techniques refer to their paper (Plečko and Bareinboim 2022) and website for further details as well as continued updates to the extended version of the *Fairness Cookbook*.

Plečko and Bareinboim begin by formalizing what they call the *Fundamental Problem of Causal Fairness Analysis*, which is essentially the need to decompose the total variation in the observed outcomes into more meaningful causal measures. They provide a toolkit for approaching this problem by organizing existing causal fairness metrics (including ETT, CDE, NDE, NIE, and Ctf-DE/IE/SE) into a hierarchy ranging in granularity on the population level (from the total

population to an individual unit) as well as in the mechanism of the disparity (including causal, spurious, direct, and indirect). Additionally, the Fairness Map shows how metrics relate to and build upon each other, as well as the types of quantifications that are not possible. The Fairness Map in (Plečko and Bareinboim 2022) is provided below:

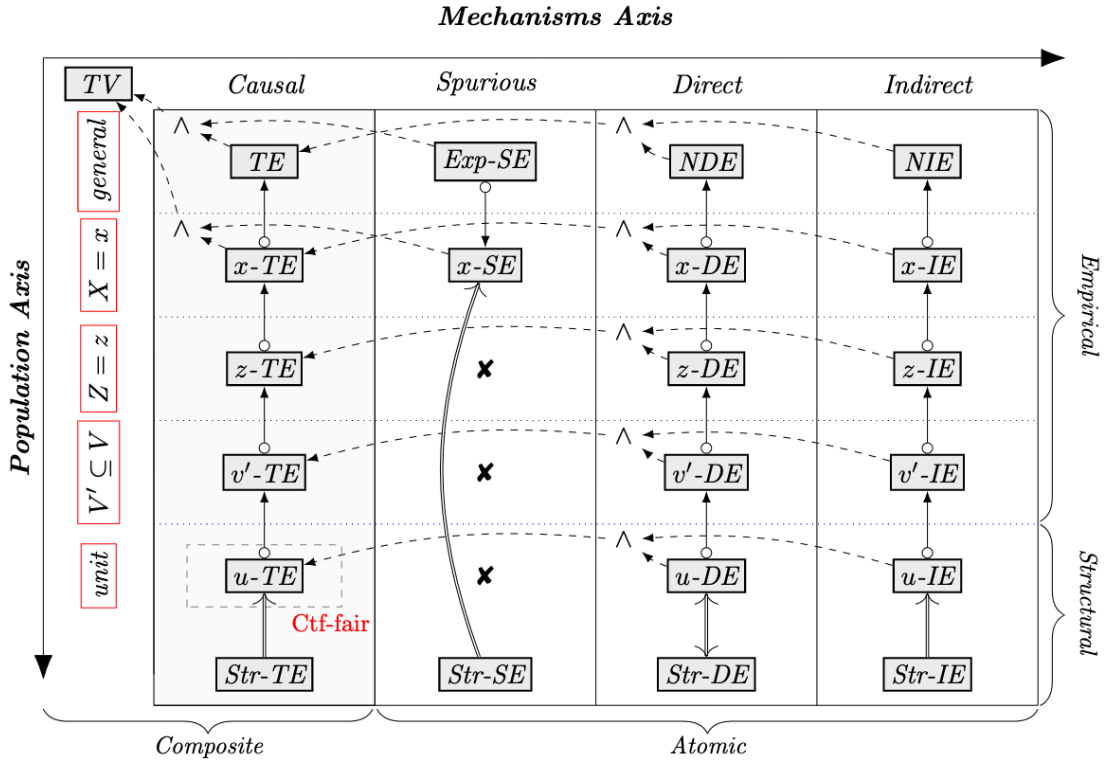


Figure 12: Fairness Map for the TV family of measures. The x-axis represent the mechanisms (causal, spurious, direct, and indirect), and the y-axis the events that capture increasingly more granular sub-populations, from general ($P(u)$) to unit level, and structural. The arrow $\Longrightarrow$ indicates relations of admissibility, $\circ\!\!\longrightarrow$ of power, and $\dashrightarrow$ of decomposability.

8.4.3.1 The Fairness Cookbook

Plečko and Bareinboim continue by introducing the *Fairness Cookbook*, which is a step-by-step process designed to be used by analysts seeking to answer questions of fairness with causal tools. A condensed version of the Fairness Cookbook is provided here, but readers are encouraged to explore the specific examples available in (Plečko and Bareinboim 2022):

1. Obtain the dataset
2. Determine the Standard Fairness Model projection

- Define which variables in your dataset map to variable sets W and Z in addition to the treatment X and outcome Y
- Also assess whether bidirected edges exist between these variable groups – if so, more work will be required to estimate causal impacts

3. Assess disparate treatment
    - Use one of the $-DE$ methods to quantify the direct effect
4. Assess disparate impact
    - Use one of the $-IE$ methods to quantify indirect effect

8.5 Fairness Case Study: Identifying Bias in the COMPAS Recidivism Model

Our case study for evaluating model fairness uses the COMPAS dataset. COMPAS is an infamous model which was the subject of an analysis by ProPublica revealing its discriminatory bias against black defendants, resulting in longer sentencing than their white colleagues. The full Python code used in this case study is available in this Colab notebook.

8.5.1 Framing the Problem + Non-Causal Estimates

The ProPublica analysis uses traditional statistical tools to investigate the disparity in model output by the defendant's race, and they showed that the algorithm produces higher predicted likelihood of recidivism for black defendants than white defendants. Figure 8.2 illustrates the high-level difference in the percentage of defendants predicted as "Medium" or "High" recidivism risk.

As we can see, there is a large difference in model output between the two groups: black defendants are predicted as Medium or High risk at a much higher rate than their white counterparts. However, it is possible that there are underlying factors contributing to this trend, such as the nature of the charge. The published analysis uses a generalized linear model to control for certain covariates when estimating the discriminatory effect of race on model predictions: specifically, the authors control for the defendant's sex, age, charge degree (felony or misdemeanor), and number of prior convictions, while interpreting the coefficient for race as the discriminatory effect. This method estimates that black defendants are 56.6% more likely to be predicted as Medium or High risk than similar white defendants. Since model output is used directly in sentencing, this would be an alarming finding.

Another way to assess model fairness is by directly modeling some relevant performance metric to determine whether the model performs suboptimally on certain groups. We can see in Figure 8.3 that there is, again, a large difference in outcomes by race.

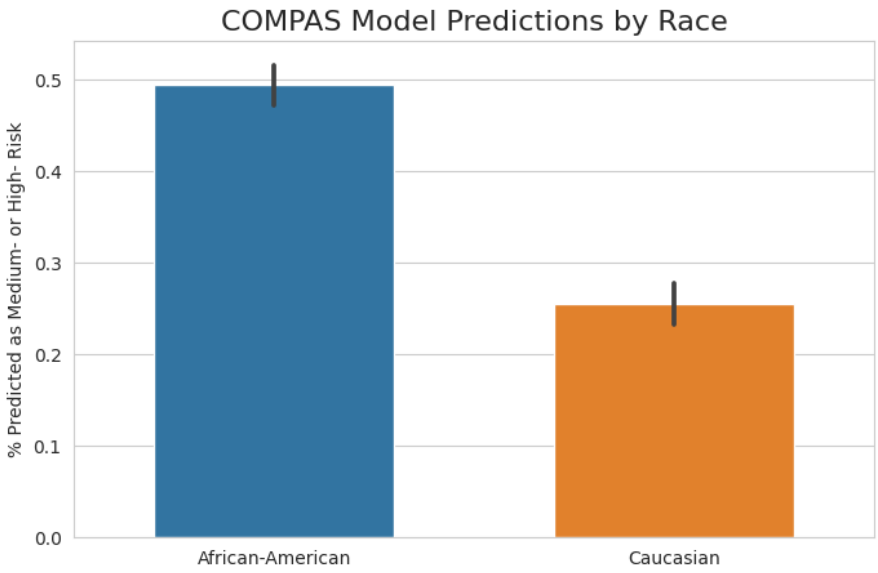


Figure 8.2: Difference in COMPAS Recidivism Predictions by Race

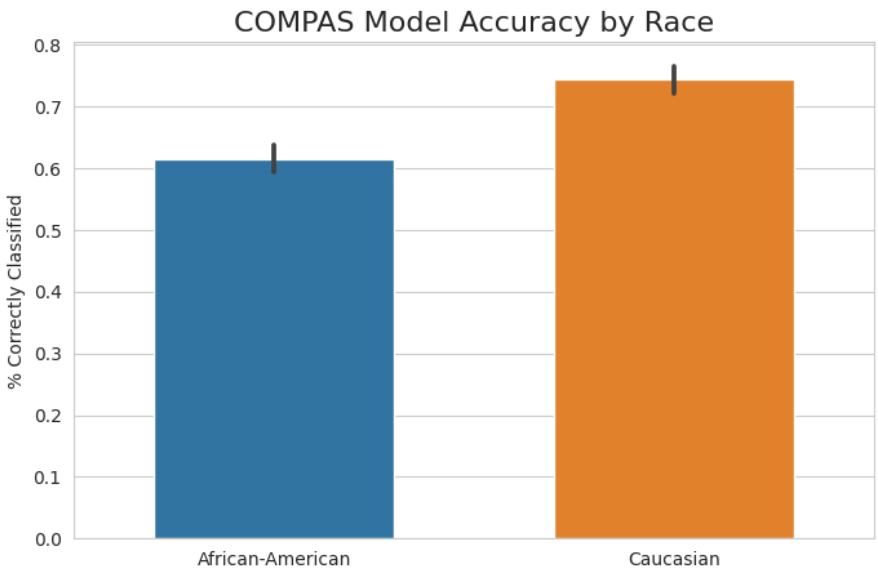


Figure 8.3: Difference in COMPAS Model Accuracy by Race

A similar non-causal approach can be taken here, with the indicator of model correctness (i.e. whether the model accurately predicted a defendant as Low or Medium/High risk) serving as the dependent variable instead of the binarized model score. Controlling for the same factors, this model estimates that the model is correct 10.5% less often for black defendants than white defendants. This also would be a concerning result, indicating that the model does not perform as well when presented with data from black defendants.

8.5.2 Causal Measures

The previous section illustrates two great ways of framing the problem of model fairness, by targeting one or both of the following quantities:

1. Model predictions
2. Performance metrics (e.g. accuracy, precision, recall)

These two pieces of information represent the primary aspects of a model-driven decision process. The problem, as we have seen throughout this book, is that we cannot be sure of the causal interpretation of the estimates produced by the linear models in the previous section. The only way we can have this confidence is with causal inference, starting with the construction of a causal graph. We use the COMPAS graph described in (Plečko and Bareinboim 2022), which is shown in Figure 8.4 from the `DoWhy` causal model visualization:

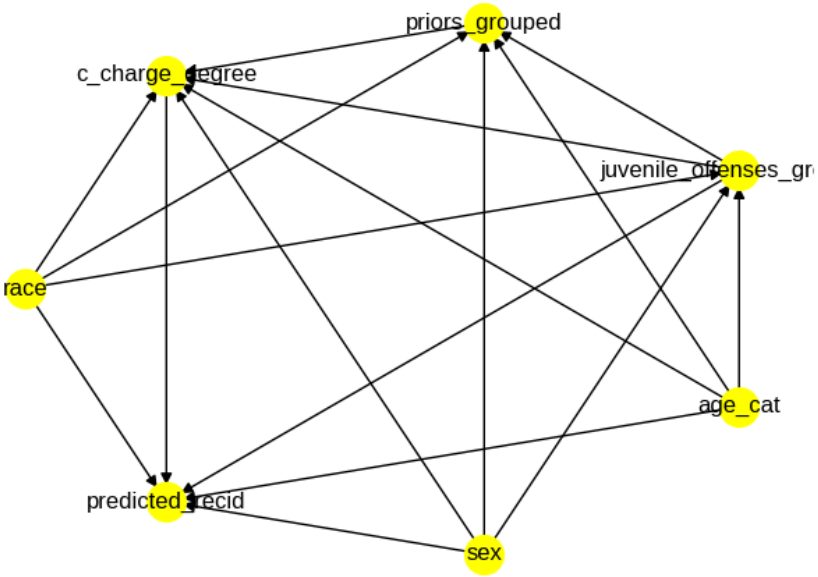


Figure 8.4: Causal Graph for COMPAS Predictions

After the graph is constructed, we care mostly about the edge between `race` and `predicted_recid`, which estimates the direct discriminatory effect based on the defendant's race. As you might expect, using `DoWhy` to repeat the analysis with causal estimates produces different results compared to the non-causal estimates. This table compares the estimates produced by the causal and non-causal estimates.

Outcome	Causal Effect	Non-Causal Effect
Predicted recidivism	+21.8%	+56.6%
Model accuracy	-11.6%	-10.5%

Both approaches agree that there is some degree of quantifiable racial bias in the COMPAS algorithm, but there is a difference in the amount that is due to race. We don't see a major reversal in trends like in the Simpson's Paradox examples, but taking the time to frame the analysis in terms of causal relationships gives us more confidence in quantifying true causal effects.

8.6 Additional Topics

8.6.1 Enforcing Fairness During Model Training

In addition to assessing whether an existing model or ML system is behaving unfairly, recent work has incorporated fairness directly into the model-building process. (Di Stefano, Hickey, and Vasileiou 2020) introduces a loss penalty based upon the controlled direct effect (CDE) measuring the disparity one wishes to minimize. In Figure 8.5 we show the causal diagram that is assumed to hold for the process which generated the data used in the ML task. The authors use Z as the protected category, Y as the task label (e.g. 0/1 in binary classification), and $\mathbf{X}$ as covariates in the model.

One important assumption encoded into this causal model is that no confounder exists which would open a backdoor path between the protected category Z and the label Y. This is likely the case for many contexts of fairness — for instance, there is no common cause between a person's race and whether the person clicks on an ad, gets into a traffic collision, or defaults on a loan — but this assumption should be validated for each use.

The authors proceed by framing the problem of training a fair model as the process of building a model that does not learn the CDE between Z and Y. A significant challenge with such an approach is that many popular machine learning algorithms require differentiable loss functions; the authors address this by leveraging propensity scoring from a surrogate model to minimize the CDE via mean-field approximation (MFCDE) along with the task's original loss (e.g. binary cross-entropy). This is formally defined as the combination of a regularization weight λ with the original loss $\mathcal{L}_o$ and a differentiable fairness penalty $\mathcal{R}_f$:

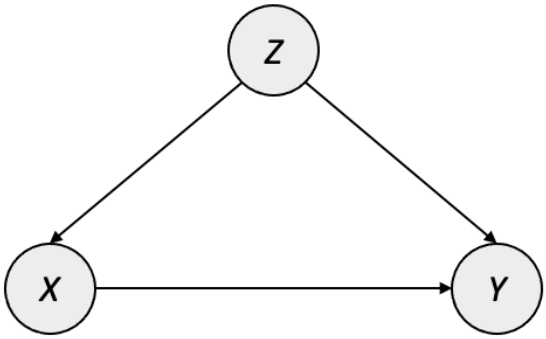


Figure 8.5: Causal graph for MFCDE

$$\mathcal{L}_f = (1 - \lambda)\mathcal{L}_o + \lambda\mathcal{R}_f$$

The fairness penalty is derived using propensity matching to calculate the MFCDE iteratively as the model updates. This happens via a surrogate model that uses the propensity scores and the protected attribute as inputs to predict the model's output at that iteration – this connection to the model being trained provides the differentiable component required to use the fair loss as a gradient-based optimization target.

The drawback of such an approach is that the surrogate model must be updated at each iteration, which adds computational overhead: both from surrogate model training and the need to provide updated predictions to be used as the surrogate model's training set. However, the authors show that a model trained to minimize MFCDE does show practical reduction in the CDE measuring the disparity of interest. The authors also note that, for higher values of λ, it may be beneficial to "pre-train" the classifier using smaller values of λ before increasing incrementally to the desired level of regularization.

8.6.2 Fairness in Reinforcement Learning

Measuring fair outcomes in a reinforcement learning setting could be done using the techniques outlined in the Fairness Cookbook (Section 8.4.3) given that the model is already trained and one simply wants to assess whether it is behaving fairly. However, since assumptions and behaviors when training RL agents differ significantly from supervised learning, additional topics on fairness in reinforcement learning will be covered in Chapter 9.

9 Reinforcement Learning

The application of causal methods within reinforcement learning is an area of active research and early adoption. This chapter will illustrate where causality fits into RL settings, what causal RL techniques exist, and the challenges that come with combining causal methods and reinforcement learning.

As we will see, there are many ways that causal inference can be integrated across many different contexts within RL — because of this, this chapter will be somewhat less consistently applied in nature than other chapters in this book. Here, we will outline several facets of the current state of causal RL, introduce the concepts, and provide direction for readers interested in going deeper.

9.1 Reinforcement Learning: A Brief Introduction

Reinforcement learning (RL) is a subfield of machine learning in which an agent learns to interact within an environment in such a way that it maximizes its expected reward. RL has enjoyed profound success in the world of game-playing, with models like AlphaGo Zero achieving superhuman performance using only self-play (Silver et al. 2017). More recent breakthroughs in game-playing have come in the games of Stratego, where DeepMind's *DeepNash* agent achieves performance levels competitive with professional human players (Perolat et al. 2022); and Diplomacy, where Meta's *Cicero* agent also achieves human-level performance using RL and techniques from language modeling (Bakhtin et al. 2022). Reinforcement learning applications have also appeared in various industry settings, including quantitative finance (Liu et al. 2018), self-driving car technology (Kiran et al. 2020), and computer hardware design (Mirhoseini et al. 2020). RL has even found applications within other ML domains, with OpenAI's ChatGPT model utilizing a PPO model with human feedback for controlling generation quality ("ChatGPT: Optimizing Language Models for Dialogue" 2022).

Two primary approaches exist for framing reinforcement learning problems: in *model-based* RL, the agent attempts to learn a representation of how its world operates – specifically, it tries to model the transition between environment states separately from learning how it should act within the environment. Alternatively, the agents in *model-free* settings attempt to learn how to act without explicitly modeling the dynamics of the environment.

In *online* RL, models are updated incrementally through a process of trial and error as the agents interact with the environment and observe the consequences of their action. Conversely,

offline RL leverages external datasets to approximate these functions without interacting with the environment (Levine et al. 2020).

One final distinction between RL approaches is between *value-based* and *policy-based* reinforcement learning. In value-based RL, the modeling task seeks to approximate the discounted future rewards from taking a particular action at a given state — the agent would then take the action which maximizes expected future rewards. On the other hand, policy-based RL attempts to model a probability distribution over all possible actions at a given state, essentially modeling the policy directly.

This is a *very* simplified overview of reinforcement learning, and other resources exist for readers seeking a deeper introduction. In addition to the previously covered concepts of causality, this chapter assumes some familiarity with the following:

- POMDPs: environments, observations, states, actions, and rewards
- Value-based RL: the Bellman equation and Q-learning
- Policy-based RL
- Multi-armed bandits
- Visual RL environments

Introductory material for these concepts can be found in an online lecture series from DeepMind's David Silver [1] as well as in Sutton and Barto's textbook (Sutton and Barto 2018) [2]. Additionally, a free hands-on course in deep RL is available from Hugging Face (Simonini and Sanseviero 2022) [3].

9.2 Adding Causality to RL

As previously discussed, reinforcement learning is an area which has enjoyed success and generated excitement over the last several years. It is also an area which stands to benefit from adopting causal methods: online RL inherently contains a somewhat causal structure, in that the environment produces certain observations and outcomes based on the agent's actions, and the agent seeks to learn the effects of its actions. The core idea of causal reinforcement learning is the introduction of causal information – via a causal graph and causal model – into an otherwise typical reinforcement learning setting.

One major benefit of causal inference is the ability to incorporate domain knowledge – this makes causality appealing for reinforcement learning, in which agent performance can be improved by incorporating outside knowledge of the world. For instance, RL applications

[1] Available on the DeepMind website
[2] Full text available at this URL
[3] See the course's official GitHub repo for more information

within healthcare might benefit from knowledge about drug interactions which would be readily known by medical professionals — causality provides a method of encoding this prior knowledge.

Additionally, some of the most significant challenges in reinforcement learning at large include sample efficiency (how many episodes it takes for an agent to reach acceptable performance) and the ability to elegantly utilize offline data (for example, collected through observation of another agent in a similar system) with an agent performing in an online setting. As we will discuss in later sections, researchers in causal RL are using causal methods to address both of these problems.

Elias Bareinboim presented a tutorial on Causal Reinforcement Learning at NeurIPS 2020. Much of the progress in causal RL through late 2020 is captured in his tutorial, and many of the concepts included in this chapter are introduced there in some detail. Readers interested in going deeper on causal RL are encouraged to explore Bareinboim's tutorial and other work further – his slides, notes, and videos from the CRL tutorial can be found at this link.

9.2.1 Causality for Smarter Agents

One of the primary areas of application in causal RL might be summarized by the notion of using causality to make agents "smarter" by…

- Helping agents understand their environment via a causal world model
- Adding causal bounds on regret expectations
- Improving action selection with causal knowledge
- Making agents more robust against observational interference or interruptions

While these approaches differ in their application within RL settings, they all contribute to improving sample efficiency, optimality, and agent reliability. We will go into greater depth on each of these in the following sections, starting with the inclusion of causal knowledge in an agent's world model.

9.2.1.1 Better World Models in MBRL

In model-based reinforcement learning (MBRL), the models that seek to learn environment dynamics and state transitions are often called *world models*. These world models represent a significant area of application of causal RL because they provide a natural opportunity for embedding external knowledge (e.g., of a domain) into an agent.

In (Li et al. 2020) the authors introduce the concept of a counterfactual "dream world" in which the agent can imagine the results of hypothetical actions via *do*-interventions. Their methods are illustrated in physical environments (e.g. object placement) and show benefits in sample efficiency to reach optimal performance. Additionally, (Rezende et al. 2020) provides

a general framework for creating *causally correct* partial models of future observations, which is especially appealing in cases where jointly modeling future observations fully would be intractable.

In a similar vein, (Lu, Meisami, and Tewari 2022) introduces the concept of *Causal Markov Decision Processes* (C-MDPs) which impose a causal framework over both the state transition and reward function within factored MDPs. The authors provide a regret bound given under the C-MDP, and they also introduce factorized methods for applications with high-dimensional action spaces and state spaces.

As we have seen, causal world models can be used in certain contexts where the variables underlying treatment, effect, and any other factors are explicitly given. However, this is not usually the case in visual RL environments where observations typically consist of pixels, and causal phenomena must be inferred based on changes in the pixel observation. In such settings, there is an additional need to express the observation in terms of high-level causal factors while also representing the relationships between them, a problem called *causal induction*. Fortunately, causal discovery methods can be used to learn the causal graph during online RL training; a survey of these methods can be found in (Ke et al. 2021). In this paper, the authors explore the current state of causal discovery within visual environments and provide new benchmarks to systematically evaluate causal induction methods. At a high level, the modeling approach consists of training an encoder for the visual observation (either an autoencoder or variational autoencoder) along with a "transition model" for the structure of the causal graph, which consists of either a graph neural network or a modular approach with directed edges. The authors first train the encoder and transition model based on trajectories produced by random actions, followed by use in a downstream RL agent. Their work shows that the modular inductive bias most closely models the explicit causal structure of the environment. However, it is unclear whether this translates to consistent improvement over non-causal techniques in visual RL, since the proposed benchmarks largely involve effects that immediately follow the agent's action.

9.2.1.2 Dynamic Treatment Regimes

One application related to causal world models comes in healthcare, where RL can be used to develop *dynamic treatment regimes* (DTRs). DTRs are plans of care that change as a disease or condition progresses along with other relevant markers of the patient's status. Specifically, DTRs describe how a healthcare provider should respond to an evolving picture of a patient's health, making them useful for managing chronic diseases as well as an intriguing application for reinforcement learning (Chakraborty and Murphy 2014). When developing DTRs, incorporating prior knowledge of drug interactions or effects can reduce regret compared to standard exploration techniques, an ideal setting for causal modeling.

Junzhe Zhang and Elias Bareinboim have done much of the work on causal DTRs. (J. Zhang and Bareinboim 2019) first introduces an online RL approach which achieves near-optimality

in learning DTRs when no observational data is available. This method, called *UC-DTR*, achieves near-optimal bounds on total regret without using causality – resulting in sub-linear total regret over the number of episodes. The authors then introduce *Causal UC-DTR*, which extends the UC-DTR algorithm by imposing causal bounds over state transition probabilities. In experiments with two-stage DTRs, both techniques vastly outperform random treatment selection, while Causal UC-DTR also produces lower cumulative regret than the non-causal technique.

Zhang and Bareinboim extend upon their work in causal DTRs in (J. Zhang 2020), in which they show that adding a causal diagram of the DTR environment, along with a corresponding structural causal model, can provide "regret that is exponentially smaller" than non-causal RL methods for finding the optimal DTR. They introduce two online algorithms:

- **OFU-DTR** uses "optimism in the face of uncertainty," an optimistic planning approach which emphasizes unexplored or promising options during action selection, and adds causal knowledge of the environment to achieve tighter regret bounds
- **PS-DTR** combines the Bayesian method of posterior sampling with structural causal models (SCMs) in order to develop policies which maximize expected value based on the causal information in the SCMs

In addition to these online learning methods, Zhang and Bareinboim propose a process of learning optimal DTRs from observational data – effectively giving the causal models a "warm start" – prior to online experimentation. The learning process consists of using the causal graph $\mathcal{G}$ to define the effect of treatment in the "offline" observational data. However, causal effects may not be identifiable from the observational data, so the authors use the technique of partial identification to produce causal bounds on the effects. Zhang and Bareinboim show that the introduction of informative causal bounds prior to online learning provides a dramatic improvement in both sample efficiency and the ultimate level of average regret.

Additional work on combining online and offline data can be found in Section 9.2.2.1.

9.2.1.3 Action Selection and Exploration

Another area where causality can aid in the reinforcement learning process is within action selection. This is an interesting area of opportunity because it is applicable within both model-based and model-free RL settings.

(Seitzer, Schölkopf, and Martius 2021) introduces the concept of *causal influence detection*, a technique that uses causal inference to better understand when an agent actually has the ability to impact its surroundings – as well as how that relationship changes over time and in different contexts. For example, consider an environment in which the agent is a robotic arm whose objective is to pick up and move an item; the agent has no influence over the position of the item unless its hand is close to the item's location. The authors address this problem by creating a measure of *causal action influence* (CAI) which is based on an approximation of

conditional mutual information within the state transition dynamics. The authors show that in robotic control environments (e.g. `FetchPickAndPlace` from OpenAI's Gym), their methods provide a significant efficiency gain in terms of sample efficiency over standard ϵ-greedy action sampling. However, this method requires full observability as well as the ability to factorize the state into causal variables.

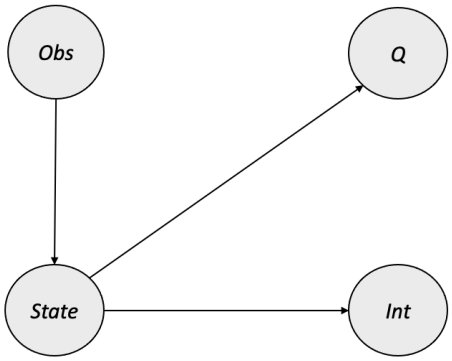
Another paper (Sun and Wang 2022) uses causal methods to prune redundant actions within continuous control environments, with the goal of improving exploration efficiency. The authors modify the temporal difference (TD) loss, which is commonly used to train the critic component of an Actor-Critic model, by using structural causal models (SCMs) to identify causally relevant actions. The experiments in this paper center around the injection of redundant or irrelevant actions into standard continuous control problems (e.g. `LunarLander` in the OpenAI Gym), and they illustrate that the causal methods (and the dynamic method DynSWAR in particular) consistently outperform standard TD agents when redundant actions are present.

9.2.1.4 Robustness

Causality can also be used to improve RL agents' resilience against interruptions or problems in their environments. In (C.-H. H. Yang et al. 2021), the authors use causal inference for observational interference to create an extension of deep Q-networks (DQNs) that is more robust to issues such as a blackout or lag in a visual environment. Their *Causal Inference Q-Network* (CIQ) is trained alongside a classifier of observational interference so that it can estimate during evaluation/inference whether each new observation is affected by an interference. CIQ makes uses causal inference to construct a more informative version of the interfered state, and it uses the interference label (during training) or classifier prediction (during inference) to route the modified state information to the appropriate neural network for Q estimation. CIQ's causal graph illustrates the relationship between interference and observation with the following components:

- *State*: the unobserved source of observations, rewards, and observational interference (e.g. via hardware overheating or system lag)
- *Obs*: the actual observation received by the agent (always observed)
- Int: indicator showing whether the observation has been affected by an interference (only observed/known during training)
- Q: the reward value at the current timestep (always observed)

During their experiments, the authors compare their method against a variety of standard DQNs as well as a DQN with a non-causal interference classifier. CIQ outperforms all baselines, with standard DQNs failing to reach acceptable performance when observational interference is introduced. Other methods with non-causal interference detection perform somewhat better than the standard baselines, but they do not reliably reach target performance. CIQ reaches target performance relatively quickly across environments with vector and pixel observations



at an interference level of 20%, and its performance remains comparable to non-perturbed input until roughly 40% interference (based on the `Cartpole` environment).

9.2.2 Causality for Transfer Learning and Imitation Learning

Another area of opportunity for causality within reinforcement learning has to do with the combination of observed experience and interactive experience – that is, offline data and online interaction. This section focuses on two related but different concepts within causal RL: transfer learning (combining online RL with observational data) and imitation learning (observing and attempting to replicate another agent's behavior). Applications in these areas involve challenging unknown factors, such as the reward model of the teaching agent in imitation learning, which sets the stage for causality to provide a benefit.

9.2.2.1 Transfer Learning: Combining Online and Offline Information

One of the biggest challenges in practical reinforcement learning is that online experimentation is often expensive, impractical, or impossible to do in meaningful applied settings. For example, it would be difficult and expensive to spend millions of hours experimenting "behind the wheel" of a self-driving car; similarly, a clinical decision support agent in a healthcare setting would raise ethical questions if it needed to test entirely random actions on actual patients. With increasing amounts of data available for use in the development of machine learning, a compelling idea arises: *can we use some of these datasets to help our RL agents act more efficiently, optimally, or robustly?* This is the root of the concept of *transfer learning* in reinforcement learning – the use of observational (or offline) data to aid in the performance of an agent in an experimental (or online) setting. However, two major challenges are present when trying to combine online and offline information; the first has to do with whether the offline data is informative to the online setting, and the second is whether differences in the

data generating process can be handled by causal adjustments. The methods discussed in this section separate these challenges and show how causal techniques can assist.

One of the first works on using causality to combine observational and experimental data came in (Forney, Pearl, and Bareinboim 2017), which proposes a counterfactual approach to the "fusion" of the two types of data within a multi-armed bandit setting. In this paper, the authors approach the problem of unobserved confounders when relating the two data sources by using a counterfactual quantity called the *effect of the treatment on the treated* (ETT) to estimate the causal effect of the action upon the reward, which they combine with empirically observed results from experimentation. Their work shows that the baseline Thompson Sampling (TS) agent which uses only experimental data performs the worst, while their proposed method (combining observational, experimental, and counterfactual data) performs the best in terms of both cumulative regret and the rate of optimal action selection.

Bareinboim's work in causal transfer learning continues in (J. Zhang and Bareinboim 2017), which applies transfer learning to multi-armed bandit settings in which causal effects cannot be identified with *do*-calculus. Zhang and Bareinboim provide an alternative approach for these settings where non-identifiability is an issue: use causal inference to derive bounds on the distribution of expected reward over the arms, and then use these bounds to identify the most promising actions. Their experiments show that when causal bounds derived from observational data are informative to the online system, causal transfer learning provides very significant efficiency gains over standard variants of Thompson Sampling and UCB. When the causal bounds are noninformative, the causal learners revert to the performance of the standard, non-causal methods (TS and UCB).

Moving beyond the setting of multi-armed bandits, (Gasse et al. 2021) follows a similar approach to Zhang and Bareinboim by creating causal bounds from a combination of observational and interventional data. However, their approach expands this idea to the world models of agents in MBRL settings, which aims to give agents a better understanding of the environment dynamics in arbitrary POMDPs.

(L. Wang, Yang, and Wang 2021) introduces a value iteration approach to improve sample efficiency by combining online and offline data in settings with confounded observational data. They propose a method called deconfounded optimistic value iteration (DOVI), which uses causal inference to adjust for confounded behavior observed in offline data. Similarly to (J. Zhang and Bareinboim 2017), the authors here show that informative observational data provably improves efficiency in the online setting.

9.2.2.2 Imitation Learning

Imitation learning differs from transfer learning in RL by focusing on learning to imitate another agent's behavior rather than learning an original policy. This is related to transfer learning in that the high-level objective is to incorporate data produced by an external actor into the learning process, but the application is different due to the goal of behaving like

another agent. Similarly to what we saw in the previous section, causality can provide a substantial benefit in dealing with tricky issues like unobservable confounders.

In (J. Zhang, Kumor, and Bareinboim 2020), the authors provide a framework which approaches the imitation learning problem from a causal inference perspective. The authors begin by using a causal graph to determine whether causal imitation is feasible in a given environment: specifically, they define *imitability* as the condition when a partially observable structural causal model (POSCM) can be used to uniquely compute the desired policy. Additionally, they provide an alternative approach which can be used when strict imitability does not hold; this method uses quantitative information derived from the observed trajectories in addition to the causal graph of the environment to overcome lack of strict imitability. (Kumor, Zhang, and Bareinboim 2021) build upon this work by extending causal imitation learning to sequential decision-making processes. In a similar manner, the authors provide a graphical framework for identifying whether imitation is possible in a given environment.

9.2.3 Causality for Agent Explainability and Fairness

We close this chapter on the combination of causality and reinforcement learning with a slightly different focus: instead of using causality to guide an agent's learning or decision making, we will explore ways that causality can provide insight into why an agent behaves the way it does. Specifically, we will outline methods for understanding agent incentives, gaining explainability in RL, and how this contributes to the principle of model fairness within an RL context.

Model explainability is an area of active research and significant practical interest within the machine learning community; explainable models can be easier to justify in decision-making settings, and some industry applications (for example, in financial or healthcare settings) might require transparency. Explainability in RL focuses on gaining an understanding for what motivates an agent to select a specific action at a specific point in time. (Madumal et al. 2020) introduces an approach to RL explainability which generates explanations for agents' decisions using counterfactuals. Their method is designed for model-free RL settings and is based on an SCM which captures the relationships between state variables and the action space. In an experiment within an environment based on the video game *Starcraft*, the authors perform a human study which shows that their counterfactual explanation method is perceived as being more trustworthy than other techniques.

Additional work in this area focuses on *agent incentives*, or the factors which influence the agent's decision. This differs from the broader concept of explainability by formalizing the notion that an agent can be incentivized in different ways by different things, such as self-preservation or exploiting an opponent's weaknesses. (Everitt et al. 2021) presents a framework for agent incentive analysis based on structural causal influence models (SCIMs), combinations of SCMs and causal influence diagrams, which map the possible influences that exist for a given agent's decision making process. SCIMs provide a causal view of how observations translate

to decisions, which in turn produce the "utility" for the agent. The authors introduce several concepts of agent incentives modeled by the graphical relationships within the SCIM:

- **Materiality**: whether the observation provides important information regarding utility nodes within the causal influence diagram
- **Value of Information**: whether the agent would benefit from seeing a given node prior to making the decision – this is a broader view of materiality which includes non-observable nodes
- **Response Incentives**: whether a node actually influences the optimal decision made by the agent
  - The authors show that Response Incentives are closely related to the notion of counterfactual fairness, in that a model is counterfactually unfair if a sensitive variable (e.g. gender, race) unduly influences the model's output
  - Counterfactual fairness is covered in greater detail in Chapter 8
- **Value of Control**: whether an agent benefits by actively controlling a particular node
- **Instrumental Control Incentives**: whether an agent manipulates a particular node *in order to* achieve a utility
  - This is the case if there exists a single directed path from the decision node D to the utility node U which only goes through the variable being manipulated by the agent X: $D \to X \to U$

`PyCID` (Fox et al. 2021) is a Python library which implements causal influence diagrams; working examples of the SCIM framework described in (Everitt et al. 2021) are available in the Jupyter notebooks on the `PyCID` GitHub.

Explainability for an RL agent allows researchers to understand the reasons behind the agent's actions. As discussed previously, this has significant appeal within many practical settings, but it also provides a way to assess the fairness of a given agent. Fairness in ML has many facets, and as reinforcement learning continues to find application in impactful areas, it becomes increasingly important to have tools available for understanding whether an agent is unfairly motivated. These causal tools are critical steps toward understanding agent incentives and ensuring fairness and safety in RL settings.

9.3 Conclusions and Open Problems

This chapter has given a broad view of the various intersections between causal inference and reinforcement learning. As we have seen, causal information can help agents learn more quickly, aid in transfer between offline and online settings, and provide insight into incentives. However, challenges exist which prevent the large-scale adoption of causal reinforcement learning, such as computational challenges, lack of causal identifiability in some environments, and intractability of causal methods in very high dimensional settings. Despite the challenges, the methods

outlined in this chapter illustrate that causality can be a very helpful tool for improving agent performance in many RL settings.

References

Ahn, Jiwoon, and Suha Kwak. 2018. "Learning Pixel-Level Semantic Affinity with Image-Level Supervision for Weakly Supervised Semantic Segmentation." https://arxiv.org/abs/1803.10464.

"Algorithmic Bias: Why and How Do Computers Make Unfair Decisions?" 2021. *Liberties.eu*. Liberties EU. https://www.liberties.eu/en/stories/algorithmic-bias-17052021/43528.

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2020. "Invariant Risk Minimization." https://arxiv.org/abs/1907.02893.

Athey, Susan, and Guido Imbens. 2015. "Machine Learning for Estimating Heterogeneous Causal Effects." In.

Austin, Peter C. 2011. "An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies." *Multivariate Behavioral Research* 46 (3): 399–424.

Bakhtin, Anton, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, et al. 2022. "Human-Level Play in the Game of Diplomacy by Combining Language Models with Strategic Reasoning." *Science* 0 (0): eade9097. https://doi.org/10.1126/science.ade9097.

Bernaola, Nikolas, Mario Michiels, Pedro Larranaga, and Concha Bielza. 2020. "Learning Massive Interpretable Gene Regulatory Networks of the Human Brain by Merging Bayesian Networks." *bioRxiv*, 2020–02.

Cakir, Senay, Marcel Gauß, Kai Häppeler, Yassine Ounajjar, Fabian Heinle, and Reiner Marchthaler. 2022. "Semantic Segmentation for Autonomous Driving: Model Evaluation, Dataset Generation, Perspective Comparison, and Real-Time Capability." arXiv. https://doi.org/10.48550/ARXIV.2207.12939.

Castro, Daniel C., Ian Walker, and Ben Glocker. 2020. "Causality Matters in Medical Imaging." *Nature Communications* 11 (1). https://doi.org/10.1038/s41467-020-17478-w.

Caton, Simon, and Christian Haas. 2020. "Fairness in Machine Learning: A Survey." arXiv. https://doi.org/10.48550/ARXIV.2010.04053.

Chakraborty, Bibhas, and Susan A. Murphy. 2014. "Dynamic Treatment Regimes." *Annual Review of Statistics and Its Application* 1 (1): 447–64. https://doi.org/10.1146/annurev-statistics-022513-115553.

"ChatGPT: Optimizing Language Models for Dialogue." 2022. OpenAI. https://openai.com/blog/chatgpt/.

Chen, Huigang, Totte Harinen, Jeong-Yoon Lee, Mike Yung, and Zhenyu Zhao. 2020. "CausalML: Python Package for Causal Machine Learning." https://arxiv.org/abs/2002.11631.

Chen, Zhang, Zhiqiang Tian, Jihua Zhu, Ce Li, and Shaoyi Du. 2022. "C-CAM: Causal CAM for Weakly Supervised Semantic Segmentation on Medical Image." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11676–85.

Chickering, David Maxwell. 2002. "Optimal Structure Identification with Greedy Search." *Journal of Machine Learning Research* 3 (Nov): 507–54.

Choudhury, Munmun De, and Emre Kıcıman. 2017. "The Language of Social Support in Social Media and Its Effect on Suicidal Ideation Risk." *Proceedings of the ... International AAAI Conference on Weblogs and Social Media. International AAAI Conference on Weblogs and Social Media* 2017: 32–41.

Choudhury, Munmun De, Emre Kıcıman, Mark Dredze, Glen A. Coppersmith, and Mrinal Kumar. 2016. "Discovering Shifts to Suicidal Ideation from Mental Health Content in Social Media." *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*.

Cinelli, Carlos, and Chad Hazlett. 2020. "Making Sense of Sensitivity: Extending Omitted Variable Bias." *Journal of the Royal Statistical Society Series B: Statistical Methodology* 82 (1): 39–67.

Cinelli, Carlos, Daniel Kumor, Bryant Chen, Judea Pearl, and Elias Bareinboim. 2019. "Sensitivity Analysis of Linear Structural Causal Models." In *International Conference on Machine Learning*, 1252–61. PMLR.

Clarke, Edmund M., and Holger Schlingloff. 1996. "Model Checking." *Communications of the ACM* 52: 74–84.

Colombo, Diego, Marloes H Maathuis, Markus Kalisch, and Thomas S Richardson. 2012. "Learning High-Dimensional Directed Acyclic Graphs with Latent and Selection Variables." *The Annals of Statistics*, 294–321.

contributors, Wikipedia. 2023. "Monty Hall Problem — Wikipedia, the Free Encyclopedia." https://en.wikipedia.org/w/index.php?title=Monty_Hall_problem&oldid=1149777144.

Corbett-Davies, Sam, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. 2017. "Algorithmic Decision Making and the Cost of Fairness." In *Proceedings of the 23rd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 797–806.

D'Amour, Alexander, Peng Ding, Avi Feller, Lihua Lei, and Jasjeet S. Sekhon. 2017. "Overlap in Observational Studies with High-Dimensional Covariates." *Journal of Econometrics*.

Darmois, George. 1953. "Analyse générale Des Liaisons Stochastiques: Etude Particulière de l'analyse Factorielle Linéaire." *Revue de l'Institut International de Statistique*, 2–8.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. "ImageNet: A Large-Scale Hierarchical Image Database." In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–55. https://doi.org/10.1109/CVPR.2009.5206848.

Di Stefano, Pietro G., James M. Hickey, and Vlasios Vasileiou. 2020. "Counterfactual Fairness: Removing Direct Effects Through Regularization." arXiv. https://doi.org/10.48550/ARXIV.2002.10774.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. 2020. "An Image Is Worth 16x16 Words:

Transformers for Image Recognition at Scale." arXiv. https://doi.org/10.48550/ARXIV.2010.11929.

Egami, Naoki, Christian Fong, Justin Grimmer, Margaret E. Roberts, and Brandon M Stewart. 2022. "How to Make Causal Inferences Using Texts." *Science Advances* 8 42: eabg2652.

Everitt, Tom, Ryan Carey, Eric Langlois, Pedro A Ortega, and Shane Legg. 2021. "Agent Incentives: A Causal Perspective." arXiv. https://doi.org/10.48550/ARXIV.2102.01685.

Fanaee-T, Hadi, and João Gama. 2013. "Event Labeling Combining Ensemble Detectors and Background Knowledge." *Progress in Artificial Intelligence* 2: 113–27.

Forney, Andrew, Judea Pearl, and Elias Bareinboim. 2017. "Counterfactual Data-Fusion for Online Reinforcement Learners." In *Proceedings of the 34th International Conference on Machine Learning*, edited by Doina Precup and Yee Whye Teh, 70:1156–64. Proceedings of Machine Learning Research. PMLR. https://proceedings.mlr.press/v70/forney17a.html.

Fox, James, Tom Everitt, Ryan Carey, Eric Langlois, Alessandro Abate, and Michael Wooldridge. 2021. "PyCID: A Python Library for Causal Influence Diagrams." In *Proceedings of the 20th Python in Science Conference*, edited by Meghann Agarwal, Chris Calloway, Dillon Niederhut, and David Shupe, 43–51. https://doi.org/10.25080/majora-1b6fd038-008.

Frydenberg, Morten. 1990. "The Chain Graph Markov Property." *Scandinavian Journal of Statistics*, 333–53.

Gasse, Maxime, Damien Grasset, Guillaume Gaudron, and Pierre-Yves Oudeyer. 2021. "Causal Reinforcement Learning Using Observational and Interventional Data." arXiv. https://doi.org/10.48550/ARXIV.2106.14421.

Geiger, Dan, and Judea Pearl. 1990. "On the Logic of Causal Models." In *Machine Intelligence and Pattern Recognition*, 9:3–14. Elsevier.

Glass, Thomas A., Steven N. Goodman, Miguel A. Hernán, and Jonathan M. Samet. 2013. "Causal Inference in Public Health." *Annual Review of Public Health* 34 (1): 61–75. https://doi.org/10.1146/annurev-publhealth-031811-124606.

Grootendorst, Maarten. 2022. "BERTopic: Neural Topic Modeling with a Class-Based TF-IDF Procedure." *arXiv Preprint arXiv:2203.05794*.

Gutierrez, Pierre, and Jean-Yves Gérardy. 2017. "Causal Inference and Uplift Modelling: A Review of the Literature." In *Proceedings of the 3rd International Conference on Predictive Applications and APIs*, edited by Claire Hardgrove, Louis Dorard, Keiran Thompson, and Florian Douetteau, 67:1–13. Proceedings of Machine Learning Research. PMLR.

Hansson, Hans A., and Bengt Jonsson. 1990. "A Logic for Reasoning about Time and Reliability." *Formal Aspects of Computing* 6: 512–35.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Deep Residual Learning for Image Recognition." arXiv. https://doi.org/10.48550/ARXIV.1512.03385.

Heckman, James J. 2008. "Econometric Causality." Working Paper 13934. Working Paper Series. National Bureau of Economic Research. https://doi.org/10.3386/w13934.

Howard, J., and S. Gugger. 2020. *Deep Learning for Coders with Fastai and Pytorch: AI Applications Without a PhD*. O'Reilly Media, Incorporated. https://books.google.no/books?id=xd6LxgEACAAJ.

Hoyer, Patrik O, Shohei Shimizu, Antti J Kerminen, and Markus Palviainen. 2008. "Estima-

tion of Causal Effects Using Linear Non-Gaussian Causal Models with Hidden Variables." *International Journal of Approximate Reasoning* 49 (2): 362–78.

Hoyer, Patrik, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. 2008. "Nonlinear Causal Discovery with Additive Noise Models." *Advances in Neural Information Processing Systems* 21.

Huang, Yimin, and Marco Valtorta. 2012. "Pearl's Calculus of Intervention Is Complete." *arXiv Preprint arXiv:1206.6831*.

Hyttinen, Antti, Patrik O Hoyer, Frederick Eberhardt, and Matti Jarvisalo. 2013. "Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure." *arXiv Preprint arXiv:1309.6836*.

Imbens, Guido W. 2004. "Nonparametric Estimation of Average Treatment Effects Under Exogeneity: A Review." *Review of Economics and Statistics* 86 (1): 4–29.

Imbens, Guido W, and Thomas Lemieux. 2008. "Regression Discontinuity Designs: A Guide to Practice." *Journal of Econometrics* 142 (2): 615–35.

Imbens, Guido W, and Donald B Rubin. 2015. *Causal Inference in Statistics, Social, and Biomedical Sciences*. Cambridge University Press.

Jalan, Jyotsna, and Martin Ravallion. 2003. "Estimating the Benefit Incidence of an Antipoverty Program by Propensity-Score Matching." *Journal of Business & Economic Statistics* 21 (1): 19–30.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning*. Springer. https://doi.org/10.1007/978-1-4614-7138-7.

Ke, Nan Rosemary, Aniket Didolkar, Sarthak Mittal, Anirudh Goyal, Guillaume Lajoie, Stefan Bauer, Danilo Rezende, Yoshua Bengio, Michael Mozer, and Christopher Pal. 2021. "Systematic Evaluation of Causal Discovery in Visual Model Based Reinforcement Learning." arXiv. https://doi.org/10.48550/ARXIV.2107.00848.

Keith Battocchi, Maggie Hei, Eleanor Dillon. 2019. "EconML: A Python Package for ML-Based Heterogeneous Treatment Effects Estimation." https://github.com/py-why/EconML.

Keith, Katherine A., David D. Jensen, and Brendan T. O'Connor. 2020. "Text and Causal Inference: A Review of Using Text to Remove Confounding from Causal Estimates." *ArXiv* abs/2005.00649.

King, Gary, and Richard Nielsen. 2019. "Why Propensity Scores Should Not Be Used for Matching." *Political Analysis* 27 (4): 435–54.

Kiran, B Ravi, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, and Patrick Pérez. 2020. "Deep Reinforcement Learning for Autonomous Driving: A Survey." arXiv. https://doi.org/10.48550/ARXIV.2002.00444.

Kleinberg, Samantha. 2012. *Causality, Probability, and Time*. Cambridge University Press. https://doi.org/10.1017/CBO9781139207799.

Kleinberg, Samantha, and Bud Mishra. 2010. "The Temporal Logic of Token Causes." In *International Conference on Principles of Knowledge Representation and Reasoning*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. "ImageNet Classification with Deep Convolutional Neural Networks." In *Advances in Neural Information Processing Systems*, edited by F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger.

Vol. 25. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Kudo, Taku, and John Richardson. 2018. "SentencePiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing." *ArXiv* abs/1808.06226.

Kumor, Daniel, Junzhe Zhang, and Elias Bareinboim. 2021. "Sequential Causal Imitation Learning with Unobserved Confounders." In *Advances in Neural Information Processing Systems*, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan, 34:14669–80. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2021/file/7b670d553471ad0fd7491c75bad587ff-Paper.pdf.

Künzel, Sören R, Jasjeet S Sekhon, Peter J Bickel, and Bin Yu. 2019. "Metalearners for Estimating Heterogeneous Treatment Effects Using Machine Learning." *Proceedings of the National Academy of Sciences* 116 (10): 4156–65.

Kusner, Matt J, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. "Counterfactual Fairness." *Advances in Neural Information Processing Systems* 30.

Lacerda, Gustavo, Peter L Spirtes, Joseph Ramsey, and Patrik O Hoyer. 2012. "Discovering Cyclic Causal Models by Independent Components Analysis." *arXiv Preprint arXiv:1206.3273*.

Lechner, Michael et al. 2011. "The Estimation of Causal Effects by Difference-in-Difference Methods." *Foundations and Trends® in Econometrics* 4 (3): 165–224.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86 (11): 2278–2324. https://doi.org/10.1109/5.726791.

Levine, Sergey, Aviral Kumar, George Tucker, and Justin Fu. 2020. "Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems." arXiv. https://doi.org/10.48550/ARXIV.2005.01643.

Li, Minne, Mengyue Yang, Furui Liu, Xu Chen, Zhitang Chen, and Jun Wang. 2020. "Causal World Models by Unsupervised Deconfounding of Physical Dynamics." arXiv. https://doi.org/10.48550/ARXIV.2012.14228.

Liu, Xiao-Yang, Zhuoran Xiong, Shan Zhong, Hongyang Yang, and Anwar Walid. 2018. "Practical Deep Reinforcement Learning Approach for Stock Trading." arXiv. https://doi.org/10.48550/ARXIV.1811.07522.

Lu, Yangyi, Amirhossein Meisami, and Ambuj Tewari. 2022. "Efficient Reinforcement Learning with Prior Causal Knowledge." In *Proceedings of the First Conference on Causal Learning and Reasoning*, edited by Bernhard Schölkopf, Caroline Uhler, and Kun Zhang, 177:526–41. Proceedings of Machine Learning Research. PMLR. https://proceedings.mlr.press/v177/lu22a.html.

Maas, Andrew L., Raymond E. Daly, Peter T. Pham, Dan Huang, A. Ng, and Christopher Potts. 2011. "Learning Word Vectors for Sentiment Analysis." In *Annual Meeting of the Association for Computational Linguistics*.

Maclaren, Oliver J, and Ruanui Nicholson. 2019. "What Can Be Estimated? Identifiability, Estimability, Causal Inference and Ill-Posed Inverse Problems." *arXiv Preprint arXiv:1904.02826*.

Madumal, Prashan, Tim Miller, Liz Sonenberg, and Frank Vetere. 2020. "Explainable Reinforcement Learning Through a Causal Lens." *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (03): 2493–2500. https://doi.org/10.1609/aaai.v34i03.5631.

Maiya, Arun S. 2021. "CausalNLP: A Practical Toolkit for Causal Inference with Text." *ArXiv* abs/2106.08043.

Manski, Charles F. 2003. *Partial Identification of Probability Distributions*. Vol. 5. Springer.

Mao, Chengzhi, Kevin Xia, James Wang, Hao Wang, Junfeng Yang, Elias Bareinboim, and Carl Vondrick. 2022. "Causal Transportability for Visual Recognition." https://arxiv.org/abs/2204.12363.

Meek, Christopher. 2013. "Strong Completeness and Faithfulness in Bayesian Networks." *arXiv Preprint arXiv:1302.4973*.

Mehrabi, Ninareh, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2019. "A Survey on Bias and Fairness in Machine Learning." arXiv. https://doi.org/10.48550/ARXIV.1908.09635.

Milletarì, Fausto, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation." *2016 Fourth International Conference on 3D Vision (3DV)*, 565–71.

Mirhoseini, Azalia, Anna Goldie, Mustafa Yazgan, Joe Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, et al. 2020. "Chip Placement with Deep Reinforcement Learning." arXiv. https://doi.org/10.48550/ARXIV.2004.10746.

Neuhäuser, Markus, Matthias Thielmann, and Graeme D Ruxton. 2018. "The Number of Strata in Propensity Score Stratification for a Binary Outcome." *Archives of Medical Science* 14 (3): 695–700.

Nie, Xinkun, and Stefan Wager. 2017. "Quasi-Oracle Estimation of Heterogeneous Treatment Effects." *arXiv: Machine Learning*.

Niu, Yulei, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2021. "Counterfactual VQA: A Cause-Effect Look at Language Bias." https://arxiv.org/abs/2006.04315.

Ogarrio, Juan Miguel, Peter Spirtes, and Joe Ramsey. 2016. "A Hybrid Causal Search Algorithm for Latent Variable Models." In *Conference on Probabilistic Graphical Models*, 368–79. PMLR.

Pearl, J., M. Glymour, and N. P. Jewell. 2016. *Causal Inference in Statistics: A Primer*. Wiley. https://books.google.com/books?id=L3G-CgAAQBAJ.

Pearl, Judea. 2000. *Causality: Models, Reasoning, and Inference*. USA: Cambridge University Press.

———. 2009. "Causal Inference in Statistics: An Overview." *Statistics Surveys* 3 (January): 96–146. https://doi.org/10.1214/09-SS057.

———. 2010. "Causal Inference." *Causality: Objectives and Assessment*, 39–58.

———. 2012. "The Do-Calculus Revisited." *arXiv Preprint arXiv:1210.4852*.

Pearl, Judea, and Elias Bareinboim. 2011. "Transportability of Causal and Statistical Relations: A Formal Approach." In *2011 IEEE 11th International Conference on Data Mining Workshops*, 540–47. https://doi.org/10.1109/ICDMW.2011.169.

Perolat, Julien, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, et al. 2022. "Mastering the Game of Stratego with Model-Free Multiagent Reinforcement Learning." *Science* 378 (6623): 990–96. https://doi.org/10.1126/science.add4679.

Plečko, Drago, and Elias Bareinboim. 2022. "Causal Fairness Analysis." arXiv. https://doi.org/10.48550/ARXIV.2207.11385.

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. "Zero-Shot Text-to-Image Generation." https://arxiv.org/abs/2102.12092.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2015. "You Only Look Once: Unified, Real-Time Object Detection." arXiv. https://doi.org/10.48550/ARXIV.1506.02640.

Reimers, Nils, and Iryna Gurevych. 2019. "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084.

Reiss, Peter C, and Frank A Wolak. 2007. "Structural Econometric Modeling: Rationales and Examples from Industrial Organization." *Handbook of Econometrics* 6: 4277–4415.

Rezende, Danilo J., Ivo Danihelka, George Papamakarios, Nan Rosemary Ke, Ray Jiang, Theophane Weber, Karol Gregor, et al. 2020. "Causally Correct Partial Models for Reinforcement Learning." arXiv. https://doi.org/10.48550/ARXIV.2002.02836.

Richardson, Thomas. 1996. "Feedback Models: Interpretation and Discovery." PhD thesis, Ph. D. thesis, Carnegie Mellon.

Robins, James M, Andrea Rotnitzky, and Lue Ping Zhao. 1994. "Estimation of Regression Coefficients When Some Regressors Are Not Always Observed." *Journal of the American Statistical Association* 89 (427): 846–66.

Rosenbaum, Paul R, and Donald B Rubin. 1984. "Reducing Bias in Observational Studies Using Subclassification on the Propensity Score." *Journal of the American Statistical Association* 79 (387): 516–24.

Rubin, D. B. 1974. "Estimating Causal Effects of Treatments in Randomized and Nonrandomized Studies." *Journal of Educational Psychology* 66: 688–701.

Sachs, Karen, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. 2005. "Causal Protein-Signaling Networks Derived from Multiparameter Single-Cell Data." *Science* 308 (5721): 523–29.

Sauer, Axel, and Andreas Geiger. 2021. "Counterfactual Generative Networks." https://arxiv.org/abs/2101.06046.

Seitzer, Maximilian, Bernhard Schölkopf, and Georg Martius. 2021. "Causal Influence Detection for Improving Efficiency in Reinforcement Learning." In *Advances in Neural Information Processing Systems (NeurIPS 2021)*. https://arxiv.org/abs/2106.03443.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2015. "Neural Machine Translation of Rare Words with Subword Units." *ArXiv* abs/1508.07909.

Sharma, Amit, and Emre Kiciman. 2020. "DoWhy: An End-to-End Library for Causal Inference." *arXiv Preprint arXiv:2011.04216*.

Shimizu, Shohei, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. 2006. "A Linear Non-Gaussian Acyclic Model for Causal Discovery." *Journal of Machine Learning Research* 7 (10).

Shpitser, Ilya, and Judea Pearl. 2006. "Identification of Joint Interventional Distributions in Recursive Semi-Markovian Causal Models."

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, et al. 2017. "Mastering the Game of Go Without Human Knowledge." *Nature* 550 (7676): 354–59. https://doi.org/10.1038/nature24270.

Simonini, Thomas, and Omar Sanseviero. 2022. "The Hugging Face Deep Reinforcement Learning Class." *GitHub Repository*. https://github.com/huggingface/deep-rl-class; GitHub.

Skitovich, Viktor Pavlovich. 1954. "Linear Forms of Independent Random Variables and the Normal Distribution Law." *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya* 18 (2): 185–200.

Spirtes, Peter. 2001. "An Anytime Algorithm for Causal Inference." In *International Workshop on Artificial Intelligence and Statistics*, 278–85. PMLR.

Spirtes, Peter, Clark N Glymour, Richard Scheines, and David Heckerman. 2000. *Causation, Prediction, and Search*. MIT press.

Splawa-Neyman, Jerzy, Dorota M Dabrowska, and TP Speed. 1990. "On the Application of Probability Theory to Agricultural Experiments. Essay on Principles. Section 9." *Statistical Science*, 465–72.

Sprenger, Jan, and Naftali Weinberger. 2021. "Simpson's Paradox." In *The Stanford Encyclopedia of Philosophy*, edited by Edward N. Zalta, Summer 2021. https://plato.stanford.edu/archives/sum2021/entries/paradox-simpson/; Metaphysics Research Lab, Stanford University.

Stuart, Elizabeth A. 2010. "Matching Methods for Causal Inference: A Review and a Look Forward." *Statistical Science: A Review Journal of the Institute of Mathematical Statistics* 25 (1): 1.

Sun, Hao, and Taiyi Wang. 2022. "Toward Causal-Aware RL: State-Wise Action-Refined Temporal Difference." In *Deep Reinforcement Learning Workshop NeurIPS 2022*. https://openreview.net/forum?id=waLncuzMofp.

Sutton, R. S., and A. G. Barto. 2018. *Reinforcement Learning, Second Edition: An Introduction*. Adaptive Computation and Machine Learning Series. MIT Press. https://books.google.com/books?id=5s-MEAAAQBAJ.

Verma, Thomas S, and Judea Pearl. 2022. "Equivalence and Synthesis of Causal Models." In *Probabilistic and Causal Inference: The Works of Judea Pearl*, 221–36.

Wager, Stefan, and Susan Athey. 2018. "Estimation and Inference of Heterogeneous Treatment Effects Using Random Forests." *Journal of the American Statistical Association* 113 (523): 1228–42.

Wang, Lingxiao, Zhuoran Yang, and Zhaoran Wang. 2021. "Provably Efficient Causal Reinforcement Learning with Confounded Observational Data." In *Advances in Neural Information Processing Systems*, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan, 34:21164–75. Curran Associates, Inc. https://proceedings.

neurips.cc/paper/2021/file/b0b79da57b95837f14be95aaa4d54cf8-Paper.pdf.

Wang, Tan, Jianqiang Huang, Hanwang Zhang, and Qianru Sun. 2020. "Visual Commonsense r-CNN." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Tan, Chang Zhou, Qianru Sun, and Hanwang Zhang. 2021. "Causal Attention for Unbiased Visual Recognition." arXiv. https://doi.org/10.48550/ARXIV.2108.08782.

Wang, Wenhui, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, et al. 2022. "Image as a Foreign Language: BEiT Pretraining for All Vision and Vision-Language Tasks." https://arxiv.org/abs/2208.10442.

Weld, Galen Cassebeer, Peter West, Maria Glenski, David T. Arbour, Ryan A. Rossi, and Tim Althoff. 2022. "Adjusting for Confounders with Text: Challenges and an Empirical Evaluation Framework for Causal Inference." *ArXiv* abs/2009.09961.

Wong, Jeffrey C. 2020. "Computational Causal Inference." arXiv. https://doi.org/10.48550/ARXIV.2007.10979.

Yang, Chao-Han Huck, I-Te Danny Hung, Yi Ouyang, and Pin-Yu Chen. 2021. "Training a Resilient q-Network Against Observational Interference." arXiv. https://doi.org/10.48550/ARXIV.2102.09677.

Yang, Xu, Hanwang Zhang, Guojun Qi, and Jianfei Cai. 2021. "Causal Attention for Vision-Language Tasks." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9847–57.

Yue, Zhongqi, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. 2020. "Interventional Few-Shot Learning." In *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:2734–46. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2020/file/1cc8a8ea51cd0adddf5dab504a285915-Paper.pdf.

Zhang, Dong, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, and Qianru Sun. 2020. "Causal Intervention for Weakly-Supervised Semantic Segmentation." In *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:655–66. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2020/file/07211688a0869d995947a8fb11b215d6-Paper.pdf.

Zhang, Junzhe. 2020. "Designing Optimal Dynamic Treatment Regimes: A Causal Reinforcement Learning Approach." In *Proceedings of the 37th International Conference on Machine Learning*, edited by Hal Daumé III and Aarti Singh, 119:11012–22. Proceedings of Machine Learning Research. PMLR. https://proceedings.mlr.press/v119/zhang20a.html.

Zhang, Junzhe, and Elias Bareinboim. 2017. "Transfer Learning in Multi-Armed Bandits: A Causal Approach." In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 1340–46. https://doi.org/10.24963/ijcai.2017/186.

———. 2018. "Fairness in Decision-Making — the Causal Explanation Formula." In. AAAI'18/IAAI'18/EAAI'18. New Orleans, Louisiana, USA: AAAI Press.

———. 2019. "Near-Optimal Reinforcement Learning in Dynamic Treatment Regimes." In *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/8252831b9fce7a49421e622c14ce0f65-

Paper.pdf.

Zhang, Junzhe, Daniel Kumor, and Elias Bareinboim. 2020. "Causal Imitation Learning with Unobserved Confounders." In *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Red Hook, NY, USA: Curran Associates Inc.

Zhang, Kun, and Aapo Hyvarinen. 2012. "On the Identifiability of the Post-Nonlinear Causal Model." *arXiv Preprint arXiv:1205.2599*.

Zheng, Min, and Samantha Kleinberg. 2017. "A Method for Automating Token Causal Explanation and Discovery." In *The Florida AI Research Society*.

Zhou, Longfei, Lin Zhang, and N. Konz. 2021. "Computer Vision Techniques in Manufacturing." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 53: 105–17.