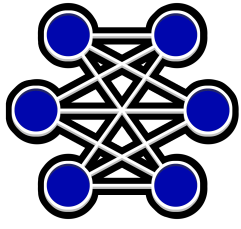
Getting Started in Data Science

Ayodele Odubela

AYODELE ODUBELA

Getting Started in Data Science



First published by Fully Connected, Inc 2020

First edition

ISBN: 978-0-578-80604-4

Editing by Ayodele Odubela

This book was professionally typeset on Reedsy.
Find out more at reedsy.com

Contents

Preface

Who this book is for

While I wrote this book to help all aspiring Data Scientists this book is really for historically marginalized people who see Data Science and Machine Learning as a way to earn high-paying wages, launch professional careers, and impact AI solutions to work better for people like them. I want you to know that I have your back.

What this book is about

This book aims to help absolute beginners understand major concepts in Data Science and Machine learning well enough to get started working on your own projects and create an interview-worthy portfolio. While this book is by no means exhaustive, I hope to provide you with enough practical knowledge, technical know-how, and industry insight to help you launch your career in Data Science.

Why I made this e-book

I love that I've been able to help spark interest in Data Science for people, but I get a lot of the same questions when new folks are asking for help. I can usually point them in the direction of a medium post or resources others have gathered so I figured why not create my own guide, using what I know now. I'll keep going deep into the concepts I wish I knew more about before landing my first job in Data Science.

What this book does NOT teach

This book does not include in-depth training on the Python or R programming languages. While these are fundamental skills for Data Science, I will focus more on the concepts most immediately relevant to doing Data Science in industry and less on the tools leveraged to do so.

Coding Examples

In this book, we'll leverage a couple of different coding languages to work on data retrieval, analysis, and modeling tasks. What I want you to take from practicing to code in Python, R, and SQL is that our focus should be on our methods and outcomes and less on the tools we use to get there. If you don't feel like a confident coder, that's okay! Data science is about using code to execute the techniques you'll learn. Coding can be taught fairly easily, but truly understanding the concepts in this book and being able to apply them on the job will be valuable for your career.

1

Introduction

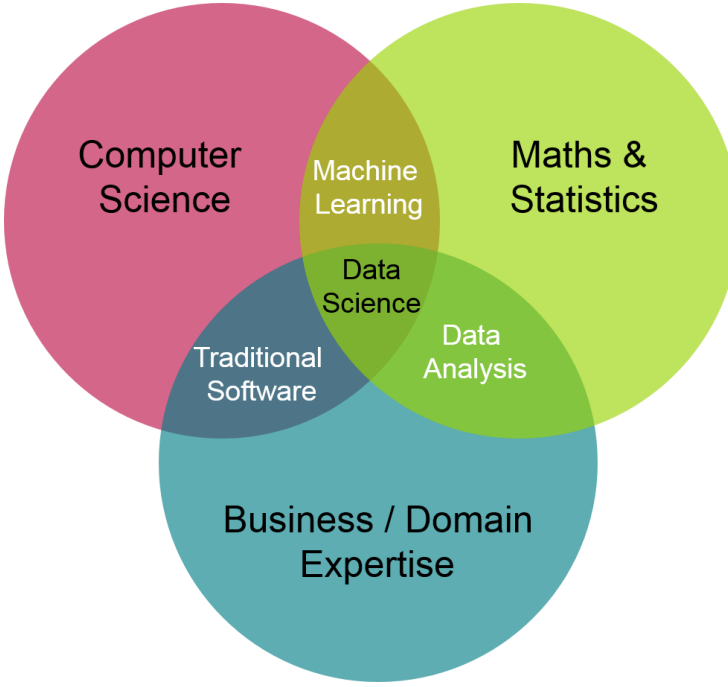
Many factors have led up to Data Science rising as one of the top new fields for technologists. It's been called the "Sexiest career of the 21st century" by Forbes, and Data Scientists self-report being relatively happy in their roles. However, there's a lot of confusion about what Data Science really is.

In industry, Data Science is typically a combination of analysis and machine learning. While the role itself depends on the job function and kind of company you're working for, most times, we're attempting to extract insights from historical data as well as use that data to make some kind of prediction about future events. This can take many forms, and common examples predict which medications patients will react positively to, personalizing content, for users and automating human decisions like if someone is "worthy" of being given a credit card.

I work hard to encourage BIPOC, LGBTQIA+, neuroatypical, and disabled people to work in Data Science because our perspectives are desperately needed. In Data Science, we have the power to influence business decisions, make product

recommendations, and inform leadership. In comparison to roles like Software Engineering and Quality Assurance, we're able to have the ear of executives because they look for us to report our data-driven recommendations. This puts us in a unique position in that we really have to understand our business and our data challenges. For many people, this is different from what they're used to, so half of the work to being a good Data Scientist is persuading business leaders that they should follow your recommendations. This can be anything from pivoting a business plan to abandoning an unethical project or increasing ad spending based on performance.

Data Science has vast implications as Machine Learning and AI is starting to impact nearly every industry. Data Science skills are in high demand, and while many practitioners have Masters degrees and PhDs in Statistics or Computer Science, many have taken the non-traditional route, including me. I encourage many data newbies to forge their own path because ML's need for professional diversity continues to grow. Whatever your work background is, you can leverage that skillset in Data Science. As you read this book, I encourage you to keep a notebook or note-taking app open so you can keep track of the skillsets you already have and areas where you can strengthen your knowledge.



You may have seen this Venn diagram of Data Science skills, and like many of you, I started at the bottom or entered the field from my business or domain expertise. When I started, I didn't have a solid background in statistics or coding but came into Data Science as a marketing expert. Once I had a good, in-depth grasp of statistics and how to use code to apply it to data, I felt comfortable leveraging my domain knowledge to work on data projects that had a massive business impact. I'm writing this book as a guide for all data newbies to grasp the concepts and expectations of data science work. This book is focused on industry Data Science as that's where most of my experience comes from.

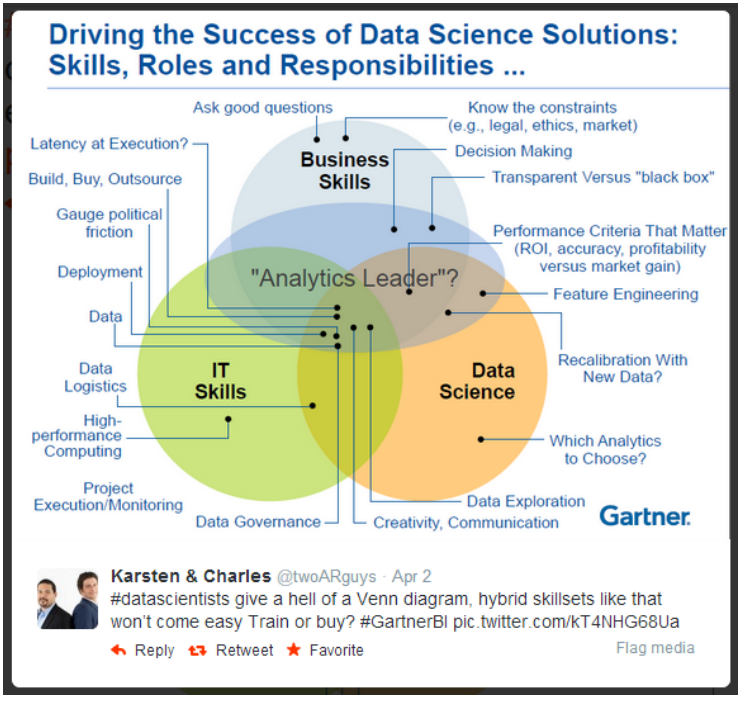
1.1 What Is Data Science?

It's easy to get so caught up in the excitement of Machine learning and topics in Artificial intelligence that we forget Data Science is simply using data to predict future events and using those predictions to drive business decisions. When we talk about Data Science in the industry, there are a few functions of Data Science roles, it's important to discuss. For this, we can take some inspiration from Airbnb's approach to Data Science. There are three main tracks, Analytics, Algorithms, and Inference, all with specializations within each.

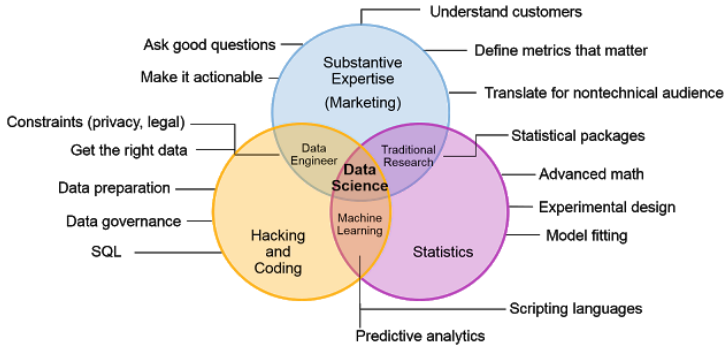
> *The Analytics track is ideal for those who are skilled at asking a great question, exploring cuts of the data in a revealing way, automating analysis through dashboards and visualizations, and driving changes in the business as a result of recommendations. The Algorithms track would be the home for those with expertise in machine learning, passionate about creating business value by infusing data in our product and processes. And the Inference track would be perfect for our statisticians, economists, and social scientists using statistics to improve our decision making and measure the impact of our work.*
>
> *– Airbnb.com*

While this isn't how every organization treats Data Science, it's a good starting point for newbies to assess Data Science jobs. If you're in an interview, you can establish if the role is looking for social scientists to find causal relationships or build data tools and define metrics.

My favorite diagram of the intersection of DS skills is the figure below. Instead of simply listing the types of skills that overlap needed and worrying about danger zones, this gives us good concrete examples of the kinds of thinking each skillset empowers Data Scientists with.



Data Science Skills, Roles, and Responsibilities



A simplified version that includes domain expertise.

There are many views one can take around how to "see" Data Science, but from my experience, Data Science is the method of applying statistics to data guided by the scientific method for business purposes. We have to keep the business aspect in mind because much of the AI development driver has been business-related metrics. In this book, I'll remain critical of both capitalism and colonialism as they impact our tech industries and government policies. We have to understand all of the products developed in the enterprise to do machine learning will always serve the business first and foremost. We have no precedent for considering the most vulnerable populations of our users or assessing what kinds of harm we can create with our machine learning models. We'll go into this more in Chapter 9, but I want to point out early that:

> *"Data Science is political."*
> *-Ben Green, Harvard University*

Knowing this, we can approach business requests with respect-

ful skepticism, educate stakeholders on data bias, and fight to create far and equitable models for our organizations. It's not enough to just teach you about the skills and set you free to treat data about human beings as if there are no real life or death consequences.

1.2 Career Roadmap

Believe it or not, the job title 'Data Scientist' didn't even exist before 2008. Now, 12 years later, these roles have a median salary above $100,000 in the United States.

One of the reasons Data Science has grown in popularity is because jobs in DS offer what many careers can't, a unique combination of:

- Good work/life balance
- High compensation
- Innovative projects
- Career flexibility and freedom
- Varied work and day-to-day tasks
- Work autonomy

Entering this highly competitive field isn't easy or straightforward for many of us. The term data science has only been used for about the last decade, so for many people, it's difficult to enter without a past "Data Scientist" role on their resume. Let's explore some of the most popular ways people transition to Data Science.

Career Entry Points

There are various routes technologists start careers in Data Science, but many get started in one of the following four ways.

Academia

A degree from an academic institution in Statistics, Computer or Data Science, etc. Coming from Academia, many students have trouble adjusting to the fast pace of many high-growth companies. These are not the only organizations with strong use cases for Data Science and AI, but this is where you'll find the most Data Science roles open. Academia offers the benefit of adding credibility to your resume but can come at high costs. However, now more and more graduate programs in Computer and Data Science don't require GRE scores, a major barrier to entry. Depending on the institution, you choose your academic experience can vary widely. If you're fortunate to attend an Engineering Ivy like Stanford, MIT, or Harvard, it will be easier to get your foot in the door. There are also big state schools with widely recognized programs, but if you choose a lower-cost school, you may have to answer, "Where is that again?" and defend your education a bit more. Overall it was important for me to have a DS degree when looking for a job, but that's by no means the only way to do it.

Career Bootcamps

Project-based instruction that trains students and uses mentors to guide learning. There are a variety of short-form boot camps that offer technical training and career guidance. What

many of these boot camps don't teach is a solid statistical foundation. Bootcamps offer a combination of education quality and cost. While most are vastly cheaper and shorter than graduate degrees, they also hold a little less value in industry. Not because they SHOULD have less value, but ML/AI is an area of tech closely coupled to academia, and the mindset in many of these communities is that academia and publications are more important than other forms of education. So with a Bootcamp, you'll often get live or recorded lessons, mentorship, and project guidance. From the perspective of someone on hiring committees, it's often noted that boot camp grads don't have enough statistics and coding experience, usually due to the length of their Bootcamp. Shine in these areas, and you can overcome many of the doubts managers have.

Some of the most popular boot camps for Data Science are:

- Metis
- Thinkful
- General Assembly
- Galvanize
- Springboard
- NYC Data Science Academy

Industry Pivots

Your job title may have changed, but you were introduced to Data Science because of your past work history. I went from an Analyst to a Data Scientist by way of grad school, but I had industry experience doing statistical tests on marketing data. Whether you are coming into Data Science from Software

Engineering or Database Administration, solidify your Data Science foundational knowledge. By foundational knowledge, I mean statistics, analyzing data, SQL querying, visualization, and ML. When you pivot from the industry, your best attribute is the knowledge you gained in your prior roles. Try to find Data focused roles at organizations who leverage data in industries similar to your past roles.

Self Taught

MOOCs, YouTube, self-guided projects offer a more freeform learning approach to gain Data Science experience. The self-taught road can be long, but with structured learning and chunking information, you can gain the skills to be a stellar Data Scientist. The most difficult part about being self-taught is knowing where to start, what companies really expect of you, and if you're really doing well or not. For self-taught Data Scientists, I encourage you to participate in some sort of group learning, a book meetup, or mentorship to get the one-on-one kind of experience we all need to advance. It can be lonely, and the worst thing you can do is try to learn in a vacuum. Join a community slack channel or Data Science podcast groups to engage with others doing similar work.

Job Titles and Responsibilities

This isn't an exhaustive list of all data jobs, but these are some of the most common roles you'll see openings for.

Data Analyst:

The main function is to find trends and patterns in data to inform the business. Analysts are usually tasked with collecting and analyzing data so those in other roles can make decisions. This requires the ability to query massive amounts of data, maintain data quality, automate queries, and tell a story with data.

Skills: SQL, Data Visualization, Some Python/R

Data Scientist:

It can be an umbrella for many roles but focuses on creating business value from data. This may seem broad, and that's because it is. Data Scientists use many tools and methods to reveal surprising insights from data, enable data-driven decisions, and move experiments from research to production. Data scientists also design experiments for collecting new data and executes the data collection. Data Scientists have a wide range of project deliverables such as presentations, charts, model predictions, scalable ML models, and more.

Skills: Python/R, Statistics, Domain Knowledge, Communication, Machine Learning

Data Engineer:

Charged with maintaining ETL pipelines and providing accurate data for modeling. The main job of Data Engineers is to develop and maintain data infrastructure. This heavily impacts the work Analysts and Scientists can do. Data Engineers need to be able to uphold data integrity and security, build scalable and reliable pipelines, and meet the data requirements from Data Analysts /

Scientists.

Skills: Functional Programming, Databases, Cloud Computing (Spark, Docker), SQL

Machine Learning Engineer:

Deployment and maintenance of machine learning models in production. Machine Learning Engineers are focused on the building and deployment of ML models. This means having deep knowledge in math like calculus, linear algebra, probability, and learning frameworks like Keras, Tensorflow, or PyTorch. This role is more engineering heavy and is great for those with advanced knowledge of Python or coming from Software Engineering backgrounds.

Skills: Advanced Python, Keras, Tensorflow/PyTorch, Stats/Calc/Linear Algebra

Visualization Developer:

People in these roles are focused on communicating data findings with visuals. Visualization Developers use software to create dashboards to showcase information, perform data analysis and data modeling, and report various data points. Their responsibilities are to create interactive or responsive data apps and conceptualize new ways to combine charts, timelines, and maps to tell data stories.

Skills: Statistics, Tableau, PowerBI, D3.js, Communication

Research Scientist:

These roles are great for people who are interested in solving unsolved problems. These roles have a lot of opportunities to publish and present research papers. Research scientists are focused on inventing better ways to do ML and solve problems better.

Skills: Statistics, Coding, Computational Research, Technical Communication

Statistician:

Statisticians typically have a background in academia, but in industry, they work to specify how the ML systems will operate and what outputs they should generate. Statisticians draw on statistical theory to help create the frameworks for developing data modeling projects. Statistifcails apply statistics to data and produce an output that can be a regression model or even just graphs and plots.

Skills: Statistics, Technical Communication, Coding

Career Focus

Within each job title, there are many ways you can focus your career based on your interests and how in-depth you'd like to work with particular technologies.

The Researcher

Your main work involves research into new algorithms or how algorithms impact society. Research roles focus on solving previously unsolved problems and publishing research papers. Often Research Scientists are tasked with inventing better ways to do machine learning. Researchers need some coding experience and a high level of statistics and math. Having research experience is crucial, as well as technical communication skills. This is a role well-suited for those coming from academia.

The Journalist

You make a name for yourself by blogging about your career experiences. Data Journalism has taken off in popularity due to many websites like FiveThirtyEight and The Pudding. Data Journalism is like visualization combined with qualitative storytelling. In these roles, your main work involves telling in-depth stories and using data to frame your story or

The Specialist

Your work is specialized in NLP, Computer Vision, or a vertical like healthcare or finance. You may spend the majority of your time deeply embedded in research papers in a narrow field. Many who enjoy these roles have that "one thing" they love. If you enjoy variation or come from a startup background here, you're used to wearing multiple hats; you may like these roles a little less.

The Generalist

Broad skills in Data Analysis and Modeling. Few deep areas of knowledge, but a high level of expertise in the business domain. This is where most data scientists start as it's easy to know your tools well and adapt to different industries well. Generalists are also usually on small data science teams (<6) or who have done analytics work in startup environments. You may see some rhetoric for a push away from generalists, but it's not a bad thing to have a broad skillset. Data Science roles are still largely underfilled even during a pandemic; focus on proving your value and less on which focus you take.

The Ethicist

Technical and social science skills are valuable here. Tech Ethicists spend time attempting to make systems less harmful to society. Ethicists frequently come from the social sciences and work to measure ML/AI's impact on society. Ethicists also often come from public health and work with various teams to reduce their decision systems' risk of being harmful.

1.3 Data Literacy

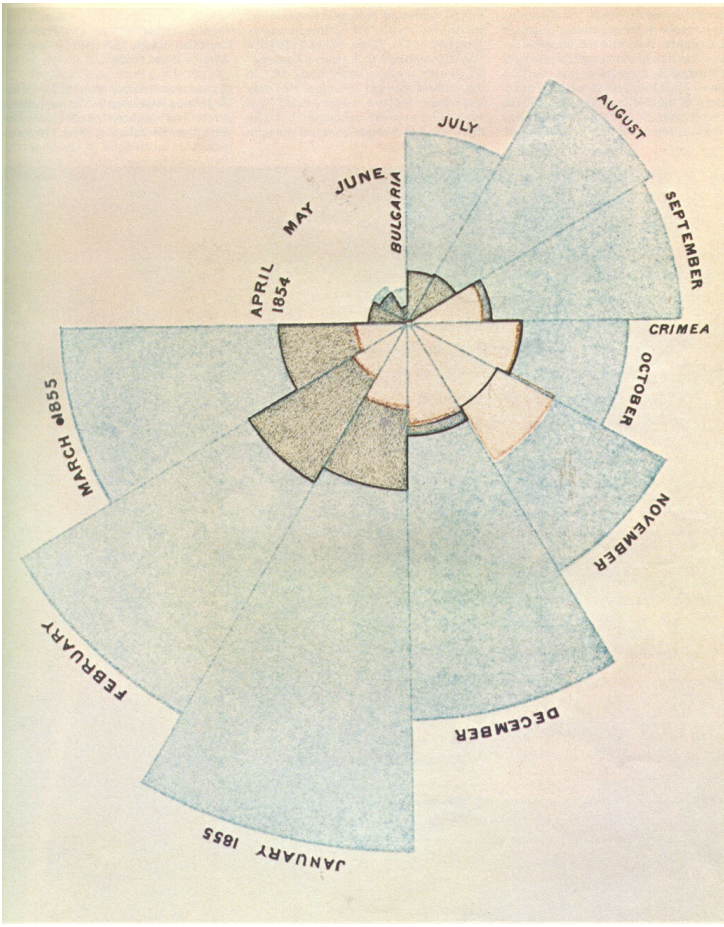
Before we jump in and start talking about Data Science, we need to cover a little data literacy. Data literacy is the ability to read, work with, and analyze data. It's really a skill measure of the competencies it takes to work with data. The first requirement of jobs in Data Science is that you're data literate, so let's make sure you're caught up.

"If I had an hour to solve a problem and my life depended on the solution, I would use the first 55 minutes determining the proper question to ask, for once I know the proper question, I could solve the problem in less than five minutes."

—Albert Einstein (1879-1955)

To explore data well, we have to know how to ask the right questions of our data. For example, maybe you're interested in the kinds of complaints forged against local officers in your city. A question you could ask is, "Are the cops getting a lot of complaints?". If we compare that question to "Are the cops getting more complaints in my city than in others?" you'll notice the second is far more specific. It has clearly defined constraints (more average complaints than elsewhere), and it focuses on outcomes in a specific population, your city.

Data Science begins with questions. Sometimes we want to know why so many soldiers are dying as Florence Nightingale did in 1854. She discovered that preventable diseases are caused by unsanitary healthcare conditions, not war injuries. Nightingale implemented healthcare reforms, which dramatically decreased the rate of deaths in soldiers. She then documented two whole years' worth of data by hand. Her famous polar area diagrams were some of the first data visualizations that communicated proper cleanliness techniques to non-scientists.



Florence Nightingale's Pie Chart of Soldier Deaths

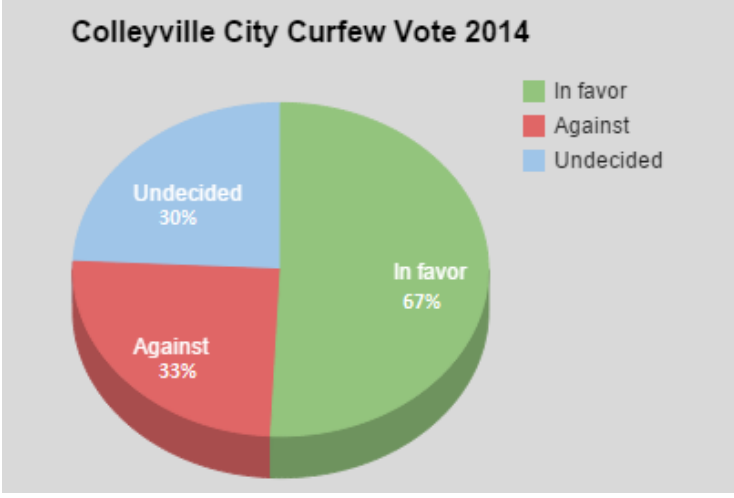
I love leveraging the **5 Whys** technique, created by Sakichi Toyoda, the founder of Toyota Motors. Essentially, you ask "why" of a problem that's been identified, then continue asking why for each answer or explanation that was given. This helps us get to the root cause of a problem so we can address fixing

it correctly. Doing Data Analytics requires mental flexibility to change your mind once presented with new data. We rarely have strict data modeling rules, so open-mindedness is a key trait for those in Data Science. We must be willing to accept what we find even if it goes against our expectations or prior beliefs about our data.

Most of the data used for modeling is collected either by sensors, customer surveys, scraping the internet, machine measurements, or documents. Source data is data in it's most undisturbed state. This means yet to be edited or cleaned by people. We should recognize this data may still have some deficiencies, like not being properly scanned for documents or disparities in sensor measurements between racial or gender groups. For many industry projects, your data will be collected by your organization about customer transactions, app usage, or historical industry data.

Data is typically made up of numerical or categorical variables, but it can also be text, images, audio, sensor readings, and more. Numerical variables are measurable characteristics like temperature and humidity; we call this Quantitative data. Qualitative data describes characteristics like gender, eye color, or state of residence.

Being Data literate just means you're able to decipher meaning from data. You want to be able to look at a pie chart, like the one below, and see something isn't quite right.



Percentages in a pie chart should add up to 100, and this does not.

1.4 Business Sense

Data Science is often listed as a combination of coding, math, and business intelligence, but it's hard without experience to know how to gain this business sense.

When I first started working in Data Science, I was coming from a marketing background, and a lot of my work was a statistical analysis of marketing data. I was already comfortable with the subject. I didn't have to learn much more about the topic before analyzing the data at hand well. Building this business sense is often difficult, especially as a new employee. You're balancing onboarding and learning new tools, trying to understand the business, and seeing what data you even have available. Despite this, put yourself in the company's shoes.

Frequently, companies feel pressure either internally or from

competitors to implement data science or machine learning into their projects. They want you to be set up for success and provide value. In many cases, this value falls into one of two categories.

We either seek to better understand our data or use it to meet business goals like reducing costs, increasing revenue, or making product improvements. Once you're able to understand the motivations and the tactics that lead to sales or revenue, it's easier to see how your analyses and machine learning products.

Gaining this business sense doesn't look the same for everyone; I can't teach it in a general way that will apply to all readers, so I want to leave you with guidelines for building your own business intuition.

Questions to Gain some Business Sense

1. What industry am I in? Is my work life or death? What's the maximum harm if we predict incorrectly?
2. Who are our customers, and how do we provide them value with data?
3. What are the business metrics that we can closely attribute to our data science work?
4. Are we using ML in our products, internally, or in research?
5. How does my team track ML/AI goals?
6. What happens to our product when our decision systems fail?
7. Is there a precedent for an acceptable margin of error? (i.e., making medical diagnoses with more precision than human doctors)
8. Are we seeking to "mine" our data for something cool, or do we have defined objectives?
9. What is the biggest incentive for completing this project?

Are we using AI just to say we're using AI?

10. What do you have to lose if it fails? Is this is a major product feature or experimentation task?

I want every reader to understand that learning data Science isn't about learning packages like NumPy and pandas or tools like Tableau and AWS. You may have heard of these tools or seen them in job descriptions, but that's not what makes you a Data Scientist. Learning data science improves your stats, coding, business, and communication skills to provide a company value with your data analyses and modeling. By working on improving these in context, you' 'll better your skills be on your way to being a great Data Scientist.

2

Analyzing Data

The biggest skill that you can learn to quickly take you from a data novice to a respected analyst is analyzing data well. Sometimes this can be described as just "looking" at the data, but what we're doing is really more than that. In this chapter, I'll give you guidelines on what to look for and insight on how data cleaning fits into data analysis.

3.1 What to Look For

When we're analyzing data, there are a few helpful ways to think about it that make answering this question easier. In industry, there are many times you don't have the data you need. I know, it sounds ridiculous, but it's true. If there is even a "dataset." It's likely messy and wrong. So many times, there isn't a tidy .CSV file for you to start the analysis with, and you'll have to take on some Lara Croft Tomb Raider ethos to pick up a lantern and start looking yourself. This means visiting original data sources and writing custom code to get the data you need. Access to data

isn't always as simple as it should be, even when "Data" is in your job title. At many companies, there's still bureaucracy at play, disparate IT functions, and many more reasons (like the data doesn't exist) you can't get your hands on what you need. If this is the case, prepare to leverage every resource you have, such as colleagues, strong coder, StackOverflow, data source documentation, and even Twitter to scrape together a plan. This will seldom look the same between roles and organizations, so the only way it's learned is by trial and error. Be sure to communicate openly with your team about the struggles you face with access. While Data Scinitrst can still feel the pressure to output a certain amount of analyses or models, temper those expectations by realistically stating your blockers to completing projects. Doing this also opens the door for your colleagues to offer help.

Garbage In Garbage Out is the most common reason why data science solutions fail. We need to analyze our data and figure out if our data quality is good for our problem.

First Steps

- How many observations (rows) do I have?
- How many features (usually columns)?
- What are the data types of my features? Are they numeric? Categorical?
- Do I have a target variable?

Inspect Each Column

- Do the columns make sense?
- Do the values in those columns make sense?
- Are the values on the right scale?
- Is missing data going to be a big problem?

Further Inspection

- How centered is our data?

In the next chapter, we'll go further into statistics concepts like central tendency, but wondering how centered and spread out our data is will give us information about where most of our data lie. This is important to know for all of the predictive modelings we'll do with this data later. Having a strong understanding of our dataset's distribution will help us choose how to best model that data.

- What looks weird?

Let's say "weird" here would be things we don't expect. In some cases, this requires some domain knowledge of our data. For example, if we're working with healthcare data and we have a column for heart rate. We see that there are some values for heart rate that seem extremely high. If you saw that someone's heart rate was 9999 beats per minute, you'd immediately question if that's right, but if you saw a heart rate that was 470, it may still seem high, but contextual knowledge in healthcare would alert you that this is also an error, but people

who lack that context would have a harder time discerning between the two.

- Is anything out of place?

We should always check our column names, beginning an end of our data, and use functions to understand the overall spread of each variable.

Plot numerical and categorical distributions

- Are any distributions unexpected?
- Are there any potential outliers that don't make sense?
- Should some features binary?
- Can you combine sparse classes in categorical distributions?

3.2 Data Cleaning

Typically, large data sets include errors. For example, survey respondents may fill fields incorrectly or skip them accidentally. You should conduct basic data checks like checking for outliers, removing protected classes or proxies, dealing with null values or incomplete data, and imbalanced response variables.

When I first began my data journey, many practitioners touted that data cleaning makes up 80% of a data scientist's time, while analysis makes up the remaining 20%. I've found this true in many, but not all of my roles. The reason for this is what I have thought of as the magic of data science. It's not in our predictive algorithms but at the intersection of technical knowledge and domain knowledge. Data "science" happens when you have a strong enough grasp of the type of data you're dealing with

and its historical artifacts, so you can best choose a method of dealing with data problems.

The first step for any data cleaning project is to check the spelling and accuracy of your columns. Here we want to make sure that our data variables are spelled correctly but after importations a dataset, we want to make sure their numbers match up with what we'd expect to see. Data cleaning is part of data analysis, and we can't separate the two due to the nature of real-world data. Data cleaning is essential as a concept since clean data helps us create accurate analyses. Many many problems can exist in our data. From missing data to unintentional bias, we should be aware that just because we have a lot of data doesn't mean it is usable. Really, most of the data created is garbage and not useful for predictive analytics.

A recent tweet from Cassie Kozyrkov demonstrates that there are a million ways our data can be wrong. McKinsey Data Science Leader Keith McNulty shared the image below. Data Wrangling, both collecting and cleaning data, is a top skill to have, and as you explore more data in professional settings, you'll find that real-life data is far harder to work with than "toy" datasets methods are demonstrated on.

While incredibly frustrating, this is a great example of real-life data in which a single loan dataset has 57 different ways of spelling Philadelphia. . You may be wondering how one ends up with this many unique variations, but think about each online form you've filled out where you were able to type in your city. This is a fairly common frustration when you have a substantially large dataset. If I were to survey Coloradans instead of Pennsylvanians, we'd probably see less variations as our city names are a bit easier to spell. We, as data analysts and scientists, can't always account for survey data

collection processes. Going forward, it's good to set standards for user research teams that allow drop-down or "smart" search selections for someone's location.

PHIADELPHIA
PHIALDELPHIA
PHIDELPHIA
PHIELADELPHIA
PHIILADELPHIA
PHILA
PHILA.
PHILAD
PHILADALPHIA
PHILADEDLPHIA
PHILADELAPHIA
PHILADELHIA
PHILADELHPIA
PHILADELLPHIA
PHILADELOHIA
PHILADELPH
PHILADELPHA
PHILADELPHAI
PHILADELPHI
PHILADELPHIA
PHILADELPHIA PA
PHILADELPHIA,
PHILADELPHIA, PA
PHILADELPHIA`
PHILADELPHIAP
PHILADELPHIAPHIA
PHILADELPHILA
PHILADELPHIOA
PHILADELPIA

PHILADELPOHIA
PHILADELPPHIA
PHILADEPHA
PHILADEPHIA
PHILADEPHILA
PHILADEPLHIA
PHILADERLPHIA
PHILADLELPHIA
PHILADLEPHIA
PHILADLPHIA
PHILADPHIA
PHILADRLPHIA
PHILAELPHIA
PHILDADELPHIA
PHILDADLPHIA
PHILDAELPHIA
PHILDELPHIA
PHILDEPPHIA
PHILIADELPHIA
PHILIDELPHIA
PHILLA
PHILLADELPHIA
PHILLY
PHILOADELPHIA
PHLADELPHIA
PHOLADELPHIA
PHPILADELPHIA
PIHLADELPHIA

Various spellings of Philadelphia

In the tweet's caption, Cassie describes data wrangling as 10% skill and 90% anger management. While incredibly frustrating, this is a great example of real-life data. You may be wondering how one ends up with this many unique variations, but think about each online form you've filled out where you were able to type in your city. This is a fairly common frustration when you have a substantially large dataset. If I were to survey Coloradans instead of Pennsylvanians, we'd probably see fewer variations as our city names are a bit easier to spell. We, as data analysts and scientists, can't always account for survey data collection processes. Going forward, it's good to set standards for user research teams that allow drop-down or "smart" search selections for someone's location.

Remove data we don't want tainting our models.

If we have typos or inconsistent capitalization, we can manually fix these and format our data more consistently. If you find duplicate observations, we also want to exclude those from our modeling datasets. Often this happens when joining two sets of data together. Sometimes you'll also run into irrelevant observations that are unlikely to represent real data like a person's blood pressure reading 976525/20 (when a normal blood pressure measurement is more like 120/80).

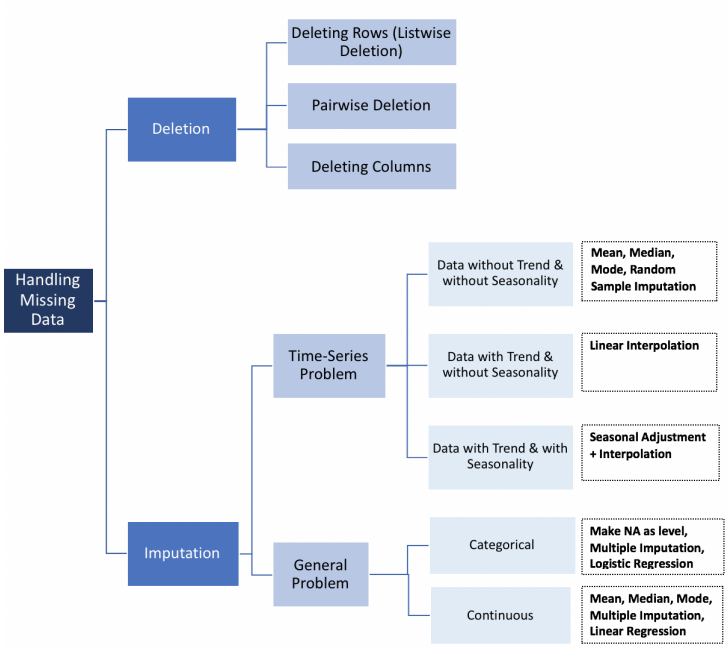
Missing Data

For example, a conference might send out an event registration form with some information about the optional fields, like gender and age. While cleaning the data, it is important to

remove or fill all the empty fields.

There are many reasons we can have missing data, but it's good to note our missing data patterns before deciding how to deal with it. Do we have data that are missing at Random? This means the chance for a missing data point is not related to the missing data, but somehow to the observed data. When data goes missing completely at random, then a missing value has nothing to do with its value or other variables. When we notice we have missing data, not at random, it means the data can be missing due to hypothetical values or its relationship to other variables. (e.g., People with high salaries generally do not want to reveal their income in surveys).

In the first two cases, it's safe to remove the data with missing values depending on how frequently they occur. In the third case, removing observations with missing values can produce a bias your the model.



There are many ways missing data is encoded in the datasets you'll see on a day-to-day basis. They are sometimes encoded as blanks, NANs, nulls, or other placeholders. We can choose to ignore missing values, but this puts us in a delicate balance. We risk losing data that might be valuable by simply ignoring missing values. We have to think about the many ways in which a user may opt not to answer an optional question. Are we excluding groups of people by assuming we won't lose value by tossing data from people who didn't provide gender information? In cases of numerical data, a better approach is to infer the missing data from the known data.

4 Methods to Impute Missing Data:

- Mean: Basic strategy
- Median: More robust to outliers
- Mode: Most frequent value
- ML Model: Can expose algorithmic bias

Data encoding

This is one of the most important steps in data preparation. It refers to grouping and assigning values to responses from the survey. For example, if a data scientist has interviewed 1,000 people and now wants to find the respondents' average age, they will create age buckets and categorize the age of each of the respondents. (For example, respondents between 13-15 years old would have their age coded as 0, 16-18 as 1, 18-20 as 2, etc.).

2.3 Descriptive Analysis

Descriptive analysis is all about trying to describe or summarize data. As we'll discuss the specific statistics we're looking at in the next chapter, the descriptive analysis looks in-depth at measures of spread, central tendency, and variation. Also to think of which type of descriptive analysis you're interested in. Are you interested in looking at an individual variable or combinations of variables?

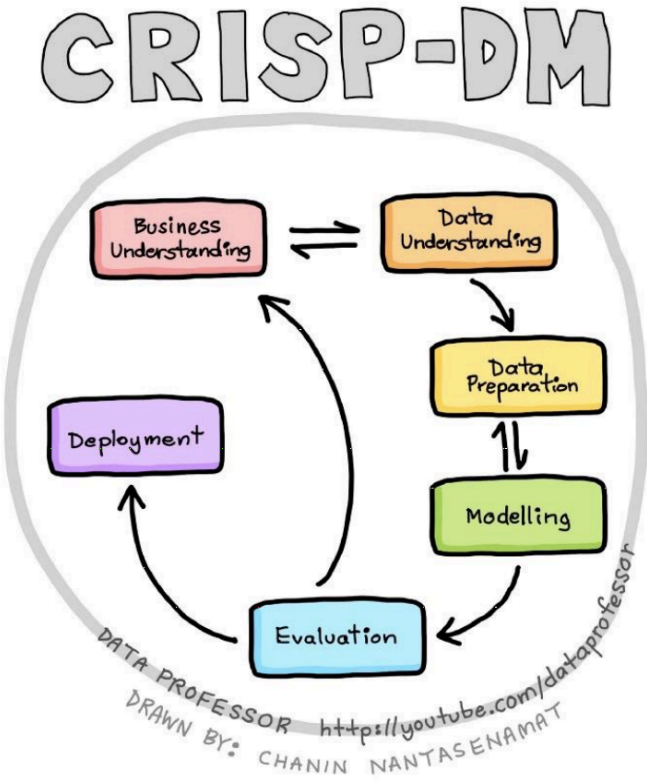
If you have numerical data, it's a good idea to first create a histogram and a box-and-whisker plot to get an idea of the distribution's shape. It might also be useful to calculate the min and max values. If your data distribution is symmetric, you can inspect the man and standard deviation; if it's skewed, you should have a look at the median and divide the set into quartiles

for further inspections. Categorical variables can be analyzed with frequency tables and bar charts.

Commonly, you'll create new tables of means and quantiles and cross-tabulations or "crosstabs" that can be used to examine many disparate hypotheses. This type of analysis is the most common and allows us to describe past events. We can get asses if events are correlated with each other and other trends. To really do this analysis, we have to set a statistics foundation, which we'll do in the next chapter.

When analyzing a new dataset, ask yourself:

1. What is the problem I want to solve?
2. Is the quality and quantity of my data set good enough?
3. How should I aggregate this data?
4. What parts of the data set are relevant?
5. How do I need to reshape my data to solve the problem?
6. Is this data skewed or unexpected?



CRISP-DM development process

Data science is a non-linear process. While it may not be monotonous like other work, it's incredibly iterative and allows you to make mistakes, learn, go back, reformulate processes, and do this over and over. Many issues stand in the way of us and good data quality, including biased data, measurement errors, and outliers.

3

Statistics Foundation

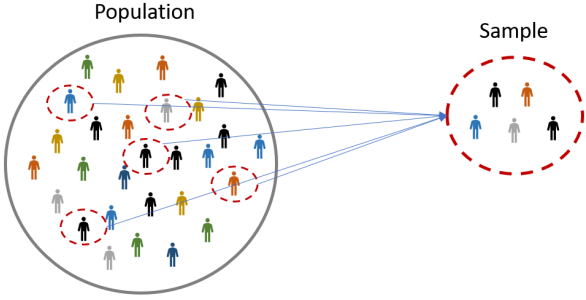
This chapter may be heavy with new concepts, but we must lay the groundwork before getting to the fun stuff. For most people new to Data Science, the first thing I would suggest learning is statistics. It's important to understand the descriptive statistics, but ultimately as a Data Scientist, you're tasked with seeing if a statistical model is a right solution to solve a problem. We use various coding languages and frameworks to do it, but statistics is crucial to every Data Scientist's work. At the end of the day, Machine learning is just a compilation of statistical models applied to vast sets of data.

When learning about statistics, it can be helpful to picture online surveys when imagining groups of people. As we start our trek into Data Science, it's important we really create a solid mathematical foundation. This chapter may be heavy with new concepts, but we must lay the groundwork before getting to the fun stuff. To get started, we'll learn some statistics vocabulary.

3.1 Foundational Terms

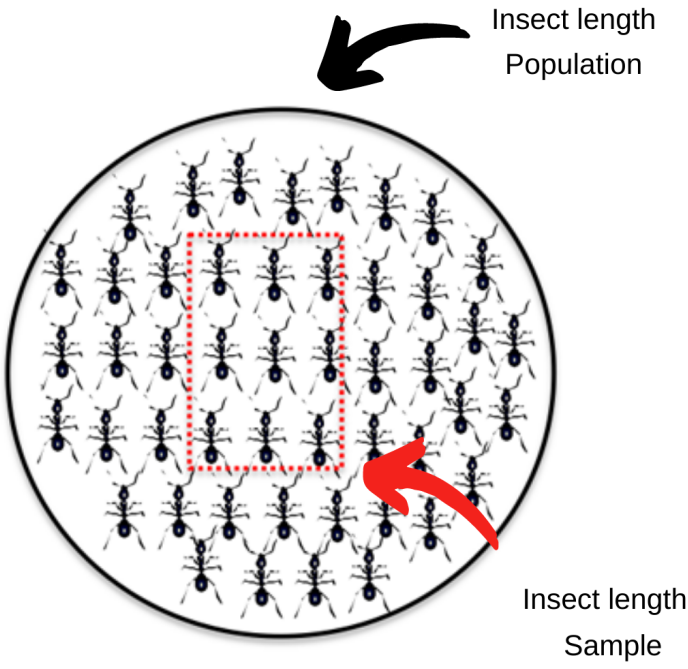
The **population** is the entire set of data or potential online survey takers.

A **sample** is a subset of the population.



A **variable** is any characteristic that can be measured or counted. (ex: Age, Gender, etc.)

A **parameter** is a quantity that indexes a group of probability distributions. (ex: mean, median, mode, etc.). If we took the average or median of our insect length of the ants population in the below image, that would be our parameter. If we looked at the average length to insects in just the sample, it would be a **statistic**.



Just as we have both Quantitative and Qualitative Data, we have both types of analysis we can conduct.

1. **Quantitative or Statistical Analysis:** includes interpreting data and creating charts and graphs to identify patterns.
2. **Qualitative Analysis**: Uses text, audio, or other media to convey analysis results.

There are also two overarching categories in statistics.

1. Descriptive Statistics: uses data to provide descriptions of

the population.
2. Inferential Statistics: makes predictions about a population based on a sample of data.

With the development of Machine Learning, there have been great strides in Predictive Analytics. While related, predictive analytics encompasses various statistical techniques from data mining, predictive modeling, and machine learning that analyze current and historical facts to make predictions about future or otherwise unknown events.

This is a tricky area for analysts and Data Scientists because at the core of our work, we assume we **can** predict the probability of future events based solely on records we have on past events; the question is, should we? We can't account for social and cultural differences or measures outside of our limited data in nearly every case. We also have a base assumption that the future will be enough like the past where we can make relevant predictions.

3.2 Data Distributions

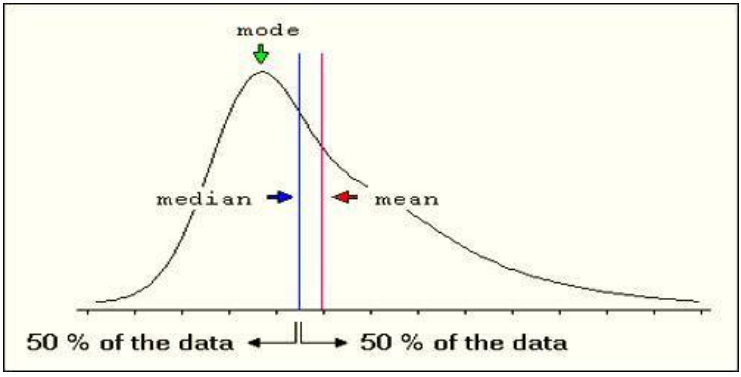
We always want to understand how distributed our data is; it's a core theme for machine learning and statistics as a whole.

Measures of Central Tendency

Mean: The average number; found by adding all data points and dividing by the number of data points.

Median: The middle number; found by ordering all data points and picking out the one in the middle

Mode: The most frequent number, found by looking at the number that occurs the highest number of times.



Measures of Variability

Range: The range is the distance from the lowest score to the highest score.

Range = max - min

Variance: Think about this as how close the other scores in the data are to the average. In mathematical terms, the variance is the average of the squared deviation from the mean. Variance is

most important for helping us find the **standard deviation**.

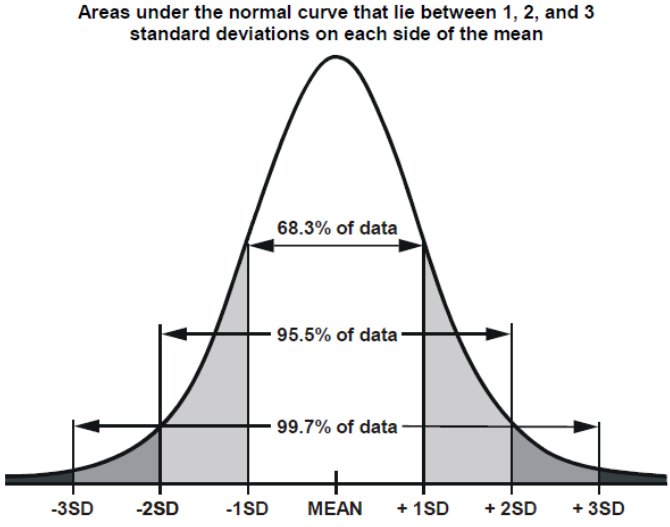
There are 5 steps to finding the variance of a dataset manually.

1. Find the mean of the data.
2. Subtract the mean from each value (this result is the deviation of the mean)
3. Square each deviation of the mean
4. Calculate the sum of these squared deviations
5. Divide by the total number of examples

Standard Deviation: This shows how much variation there is in our data. If the data is close together, the standard deviation is small. Large standard deviations mean we have data that is very spread out. Standard deviation is often denoted by the lowercase Greek letter sigma, .

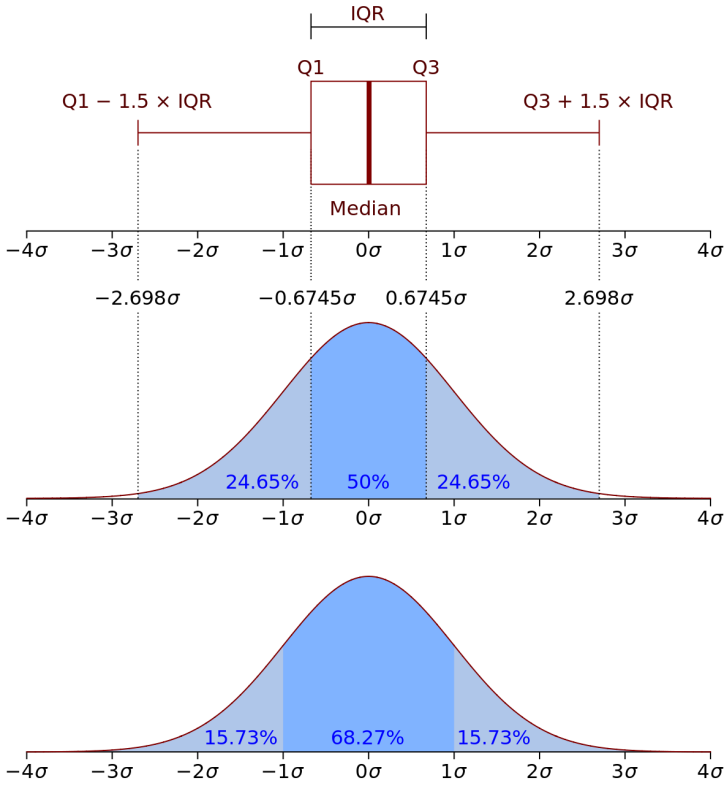
1. Find the variance
2. Take the square root of the variance.

We often use a bell curve to represent a normal distribution of data. A normal distribution is one where the mean, median, and mode are the same. This is important as it is vital to many machine learning algorithms. If we have a large, normal dataset, we can invoke things like the central limit theorem, which allows us to make certain assumptions about our data.



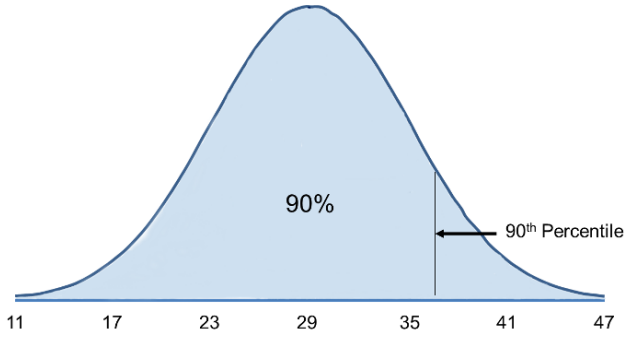
You'll learn along the way statistics has a lot to do with assumptions, and a lot of data cleansing and transformation exists to manipulate our data to meet statistical criteria.

Inter Quartile Range: The measure of variability based on dividing a data set into 4 equal quartiles.



Describing Ranges

Percentile: Percentile is a way to represent the position of values in a data set. To calculate percentile, values in the data set should always be in ascending order. The median 59 has 4 values less than itself out of 8. It can also be said as: In the data set, 59 is 50th percentile because 50% of the total terms are less than 59. In general, if k is an nth percentile, it implies that n% of the total terms are less than k.



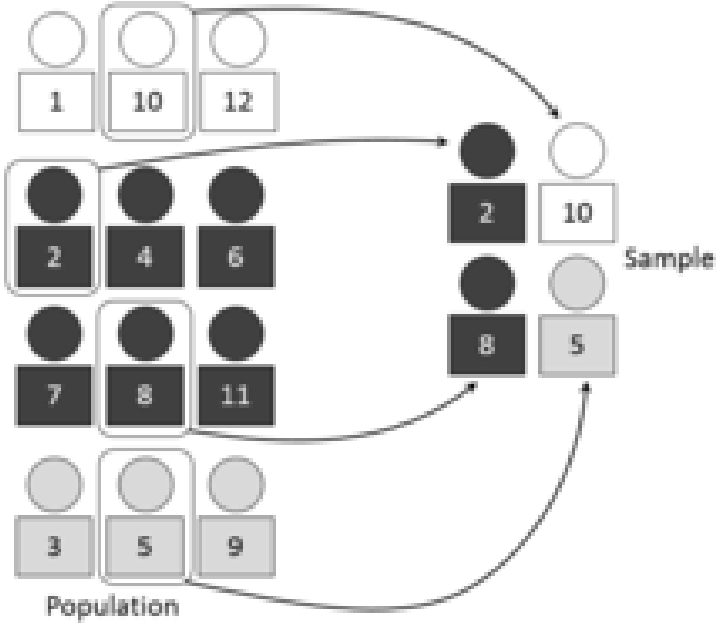
There are three quartile values. The first quartile value is at 25 percentile. The second quartile is 50 percentile, and the third quartile is 75 percentile. The second quartile (Q2) is the median of the whole data. The first quartile (Q1) is the median of the upper half of the data. And Third Quartile (Q3) is the median of the lower half of the data.

Interquartile Range (IQR) = Q3 - Q1 = 85 - 41 = 44

3.3 Populations and Samples

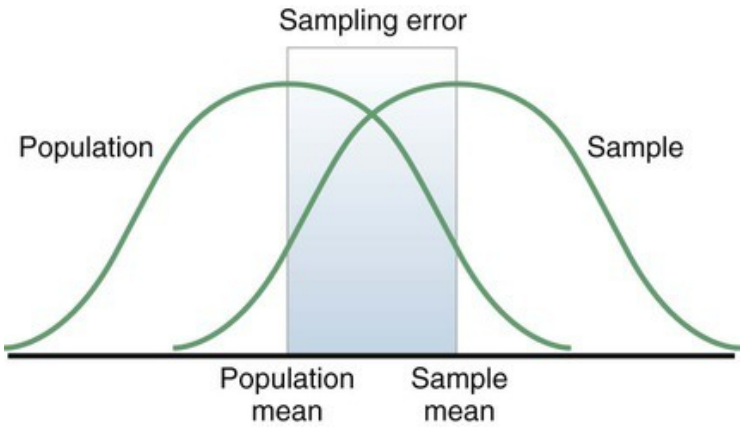
In nearly all cases of "big data," It's impossible to gather conclusions about the population. So the goal is to get a sample of a population that *represents* the population well. This sample is what we use to draw conclusions about the population. If we think about surveys and election polling for a minute, we can understand the importance of good sampling. Have you seen poll results that you find hard to believe? One of the biggest hurdles in polling is getting a representative sample of the entire voter population when we talk about the representation; we want to make sure that our sample has similar demographics as

the actual population of eligible voters.

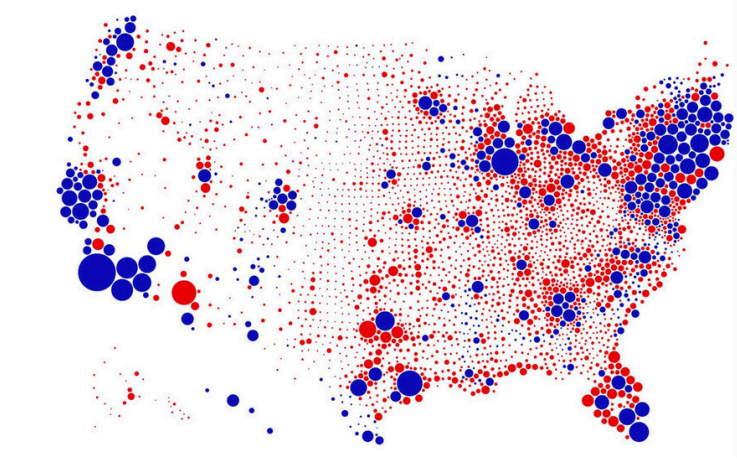


If we look at a sample of 3000 voters in a county and calculate their mean age, that number should be close to all voters' mean age in the population If we chose them randomly. However, there is a chance the mean varies from the true mean.

Sampling Error: The difference between the sample mean and the true mean.



Let's say the true mean age for a county of voters is 44; this is the population means we're estimating. If the mean age of our sample of 3000 voters is 46, then our Sampling Error is 2. What's important to know is that this doesn't just count for the mean; we can find the sampling error for any statistic.

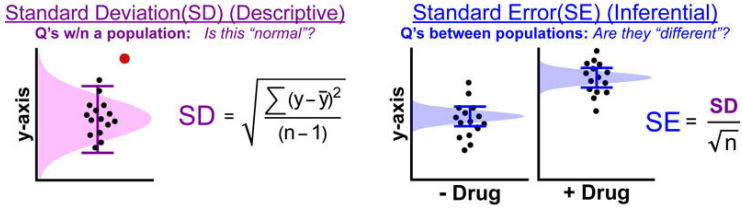


If we randomly sample a group of 3000 voters over and over out of 500,000 potential voters in a county. If we sample this

same group 1000 times, we should see all those samples' mean age to be very close to the population mean. If each voter has the same probability of being chosen in our sample (meaning we "replace" them after being chosen), it's called bootstrap sampling. This is a good way to estimate the population without asking all 500k people.

Our steps to calculate our Sampling error are to get our population means and standard deviation, confirm our sample size, determine the confidence level, Now multiply the Z score by the population standard deviation and divide the same by the square root of the sample size in order to arrive at a margin of error or sample size error. In the context of exit polls, you might also hear this as the margin of error, measuring how wide of a margin between the sample and the population.

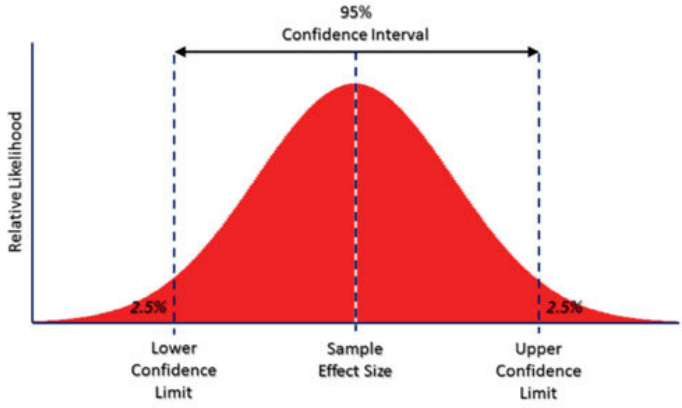
Standard Error: The standard error is the standard deviation of a sample statistic. If we focus on the mean, it's called the standard error of the mean. In our first 3000 voter sample, we're likely to see a high standard error because we have a small sample compared to the population.



Let's say you work at an app company and are tasked with reporting on new daily app downloads. First want to observe how many app downloads occur in a week since there are so many factors that can influence app downloads (like the day of

the week, time of day, and being featured in the Apple App Store or the Google Play store). We get the mean of. By taking means of downloads each day, we can get the average daily downloads. However, many external factors influence sales, like, discounts, holidays, etc. Thus, instead of taking the mean of one day, we take the means of 3 days. The Standard Error of the means now refers to the change in the mean with different observations each time.

Confidence Interval: A range of values within which the true value lies. CI's width tells us a lot about the certainty we have about the collected sample population. A confidence level of 98% means that if the poll was repeated with the same conditions 100,000 times, the results would match the actual population 98% of the time. The two main factors that impact Confidence Interval are variation and sample size.

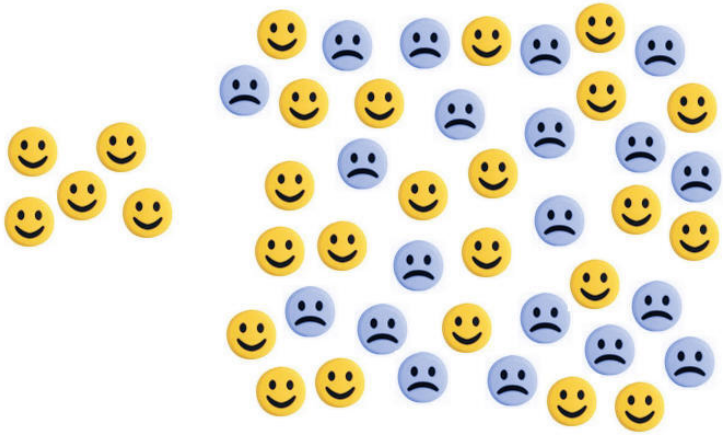


Variation: Low Variation between samples results in too Similar

Samples, causing us to have a Narrow CI. High Variation between samples results in too Varied Samples, which causes us to have a Wider CI.

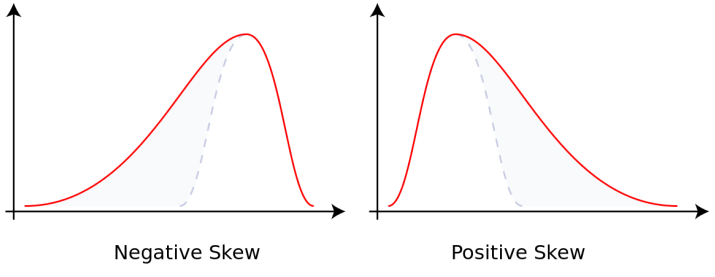
Sample Size: If we take small samples, we don't have any findings to base our inference upon. Small samples will differ from one another and have less detail, leading to wider CI.

It's almost always better to have a large sample size. This allows us to gain more details about our data, and it allows for more predictive power.



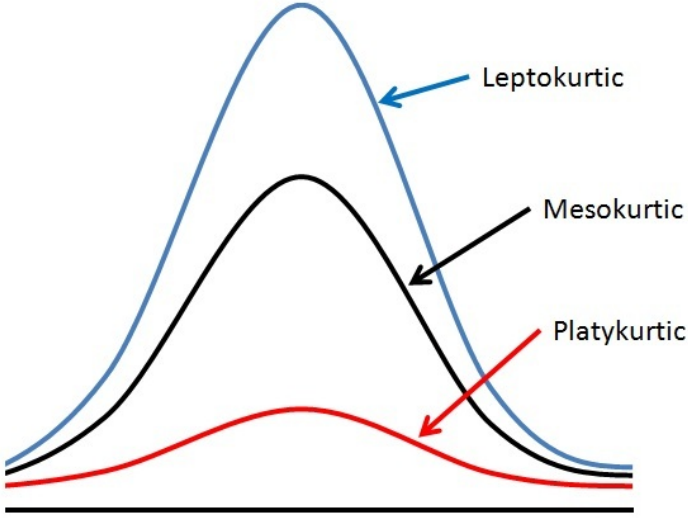
With a small sample size, we gain fewer details and a narrow confidence interval.

Skewness: Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. The skewness value can be positive or negative or undefined. In a perfect normal distribution, the tails on either side of the curve are exact mirror images of each other. When a distribution is skewed to the left, the tail on the curve's left-hand side is longer than the tail on the right-hand side, and the mean is less than the mode. This situation is also called negative skewness.



The sign gives the direction of skewness. A zero means no skewness at all. A negative value means the distribution is negatively skewed, and a positive value means the distribution is positively skewed.

Kurtosis: Kurtosis is a measure of whether the data are heavy-tailed (profusion of outliers) or light-tailed (lack of outliers) relative to a normal distribution.



Types of Kurtosis

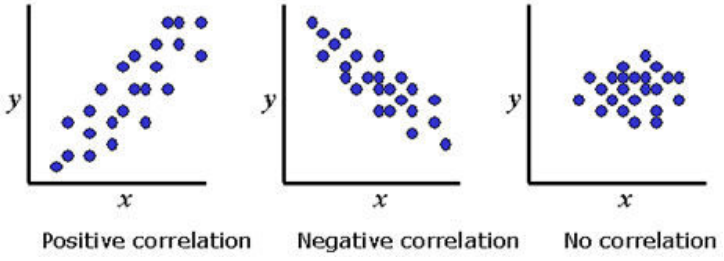
Mesokurtic: The distribution which has similar kurtosis as normal distribution kurtosis, which is zero.

Leptokurtic: distributions have kurtosis greater than a

Mesokurtic distribution. Tails of such distributions are thick and heavy. If the curve of a distribution is more peaked than the Mesokurtic curve, it is referred to as a Leptokurtic curve.

Platykurtic: distributions have kurtosis lesser than a Mesokurtic distribution. Tails of such distributions thinner. If a distribution curve is less peaked than a Mesokurtic curve, it is referred to as a Platykurtic curve.

Correlation: Correlation is a statistical technique that can show whether and how strongly pairs of variables are related.

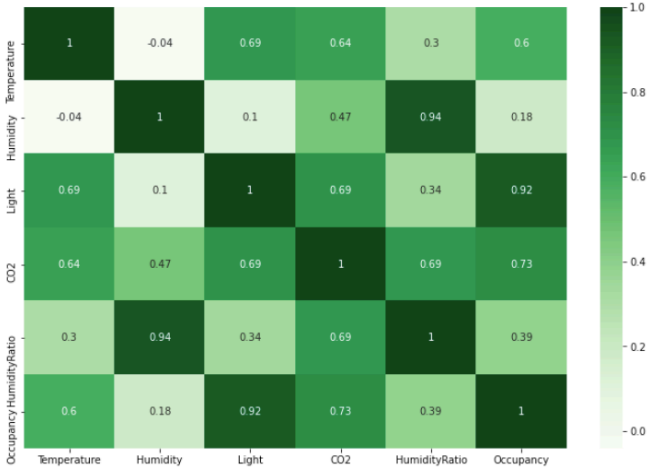


Examples of correlation

If you buy ads on Twitter to promote your personal blog and you notice as the more money you put behind the ads, the more website traffic you get, this is a positive correlation. The more ads you purchase (or as your ad spend increases), click-throughs to your site increase. If x (the horizontal) repres3ents the amount of Twitter ads and y represents your bank account balance, the relationship between the two would be a negative correlation.

The correlation coefficient measures the strength of the correlation. It ranges from -1.0 to +1.0. The closer our correlation coefficient is to +1 or -1, the more closely the two variables are related. If our coefficient was close to 0, then there is no relationship or correlation between the variables. We also have an inverse correlation as one variable gets larger, the other gets smaller. For me, the more time I spend planning my tasks, the less stress I have about getting them done.

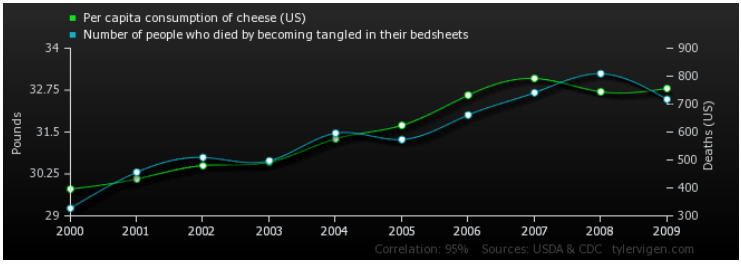
Often in Data Science, you'll see correlations represented with heat maps. This is a heatmap of room occupancy data retrieved from the University of California Irvine ML Repository.



This heatmap shows the relationship between the variables numerically, and the intensity of the color represents the strength of the relationship. AS you can see, the same variables are listed on the X and Y-axis. Each square represents an interaction of variables. The darkest or strongest correlations are between Humidity ratio and humidity, which has a positive correlation

of .94, which makes sense since Humidity ratio is a calculated variable that includes humidity. The next strongest relationship is between occupancy (the response variable) and light. This also makes logical sense as most often, when rooms are occupied, they have lights on.

Correlations are interesting because despite how the interactions may be meaningful, you'll hear this phrase in this book, and many times in industry, correlation doesn't mean causation. This just means that because certain features are correlated like the ones in the heatmap above, it doesn't mean that one feature causes another. A great resource to understand some interesting correlations but without causation is the Spurious Correlations website.



*Per capita consumption of cheese **correlates** with the number of people who died by becoming tangled in their bedsheets.*

Hypothesis Test: The main purpose is to test the results of surveys to see if the results were meaningful and not random. We use this method to decide whether we can accept or reject a claim using statistics.

Null Hypothesis: The null hypothesis (H0) is usually the hypothesis that observations happen purely because of chance.

Researchers work to reject it. The null hypothesis proposes that no statistical significance exists in our dataset.

Alternative Hypothesis: The alternative hypothesis (H1 or Ha) is the hypothesis that sample observations are influenced by something non-random; that's it! Cause. The idea is that the relationship in the sample reflects the real relationship in the population.

		Conclusion about null hypothesis from statistical test	
		Accept Null	Reject Null
Truth about null hypothesis in population	True	Correct	Type I error Observe difference when none exists
	False	Type II error Fail to observe difference when one exists	Correct

Types of Errors:

Type I Error means rejecting the True null hypothesis and accepting the alternate hypothesis. Type I errors are just a fancy way of saying false positives. An example of this is a smoke detector detecting smoke when there is none.

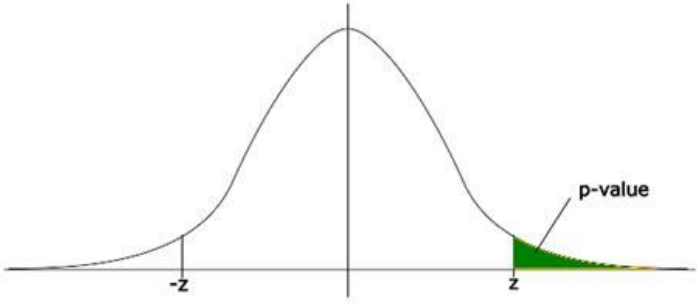
Type II Error means accepting the null hypothesis when an alternate hypothesis is true. It is a false negative like when a fire alarm fails to detect fire.

P-value: It is used in hypothesis testing to check the strength of

the evidence provided by the population. The p-value is always between 0 and 1:

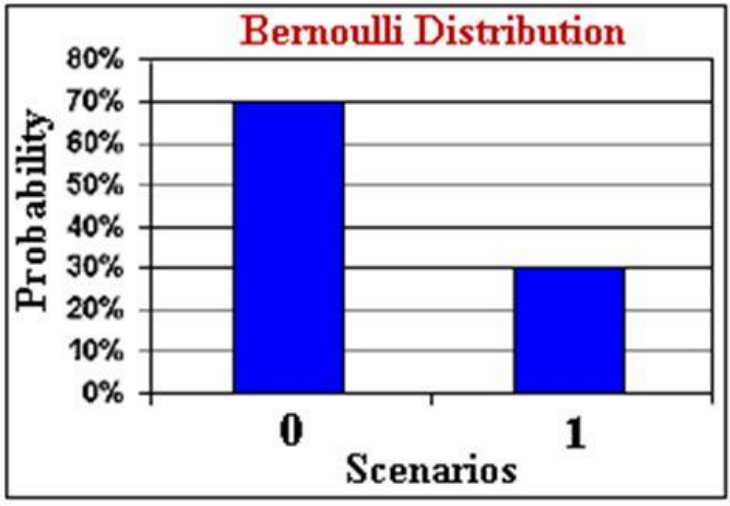
A small p-value (≤ 0.05) shows strong evidence against the null hypothesis; in other words, a statistically significant relationship, so you reject the null hypothesis. I

A large p-value (> 0.05) indicates weak evidence against the null hypothesis; therefore, you fail to reject the null hypothesis. It means that your data could have ended up the way it did randomly and not because of causation.

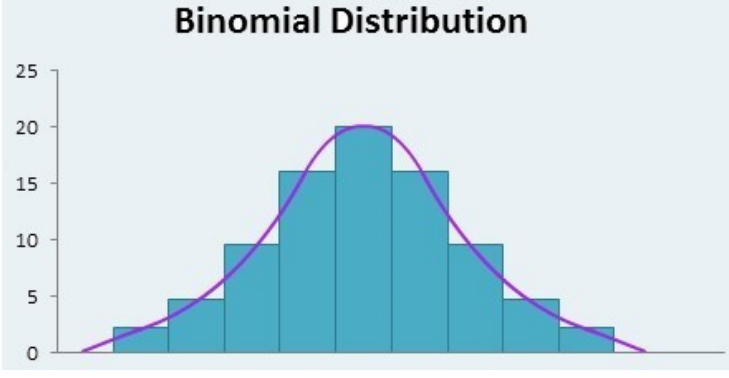


3.4 Probability

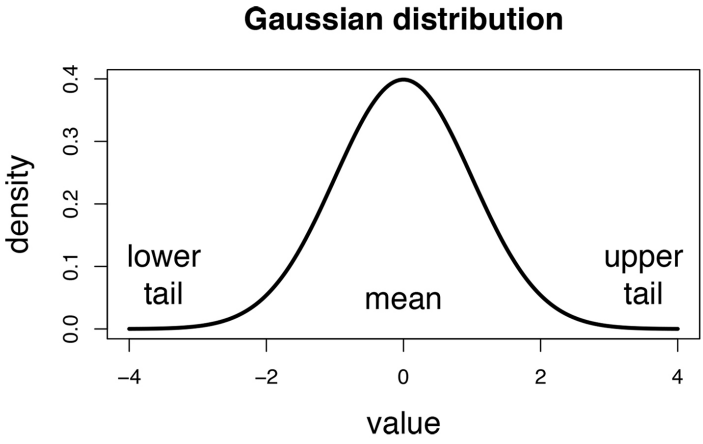
Probability distributions are a mathematical way of displaying the probabilities of events. In any given dataset, we have information about events, and we can plot the probability of them.



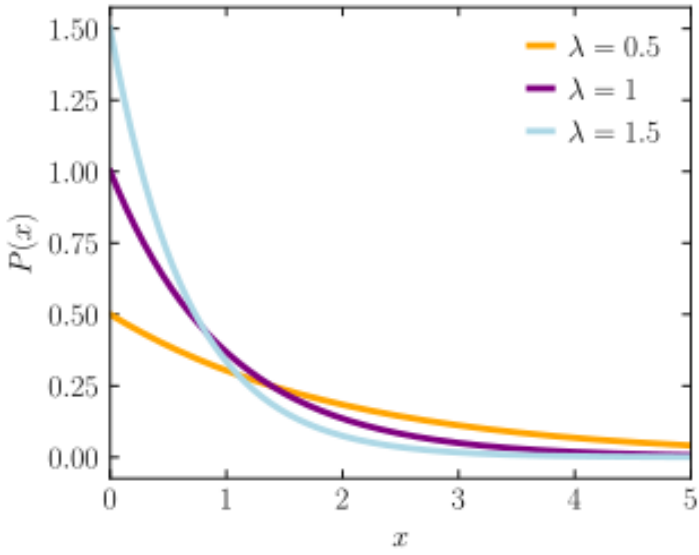
Bernoulli distribution is the probability distribution of a random variable. This is typically related to a True/False or a classification scenario.



Multiple Bernoulli trials create a **binomial distribution**. It's the probability distribution constituting True/False questions in x number of trials. Rules for binomial probability are that each event is independent, and there are only two possible outcomes.



The **Gaussian** or **Normal Distribution** is commonly used in machine learning and found in nature. This distribution requires minimum prior knowledge as it can solely be defined using the mean and variance of data, making our lives a lot easier.



The **Exponential distribution** is framed around time, more specifically, the relevant time until an event occurs. It actually looks less like our other curves, and it has a sharp point at *x = o*.

Probability: probability measures the extent of certainty about an uncertain event. Probability represents the certainty factor. Certainty is the rate that you would assign to an event to happen. Say you are rolling a die, and you say that the certainty that you roll a 6 is . That means you have a 16.67% chance of getting a 6 (on a fair die) every time you roll it. This is our frequentist probability. The frequentist probability denotes the frequency with which the event can happen amongst many trials/events. Not all scenarios are frequency related as in our previous assumption. If we consider a machine learning problem in which we estimate the probability of inflation or deflation of bitcoin's price, we wouldn't be thinking this from the perspective of repetition. For the latter, we use **Bayesian**

probability. Rather than considering the frequency with which an event repeats, we quantify our belief. If we think there's a 29% chance that a patient will recover from an illness because of a specific medication, we aren't' trying to repeat the problem by creating infinite clones of our patient. Instead, we say we've quantified our belief with a 29% certainty that a patient will recover. We should also keep in mind the eight rules or probability.

Eight Rules of Probability

Rule #1: For any event A, **0 ≤ P(A) ≤ 1**; *in other words, the probability of an event can range from o to 1.*

Rule #2: The sum of the probabilities of all possible outcomes always equals

Rule #3: **P(not A) = 1 — P(A)**; *This rule explains the relationship between an event's probability and its complement event. A complement event is one that includes all possible outcomes that aren't in A.*

Rule #4: If A and B are disjoint events (mutually exclusive), then **P(A or B) = P(A) + P(B)**; *this is called the addition rule for disjoint events*

Rule #5: **P(A or B) = P(A) + P(B) — P(A and B)**; *this is called the general addition rule.*

Rule #6: If A and B are two independent events, then **P(A and B) = P(A) * P(B)**; *this is called the multiplication rule for independent*

events.

Rule #7: The conditional probability of event B given event A is **P(B|A) = P(A and B) / P(A)**

Rule #8: For any two events A and B, **P(A and B) = P(A) * P(B|A)**; *this is called the general multiplication rule*

Bayes' Theorem: Bayes' theorem provides a way to revise existing predictions with new data. It's the foundation of Bayesian statistics and just helps us to determine conditional probability. This is the likelihood of an event occurring based on if a previous event occurs. Bayes' Theorem relies on two ideas. The first being your initial belief; we call this our prior probability (we have this **prior** to getting more data). The posterior probability is the revised probability after taking into account be information. Bayes' theorem gives a way to quantify an event's probability-based on new information related to that event.

Formula For Bayes' Theorem

$$P(A|B) = \frac{P(A \bigcap B)}{P(B)} = \frac{P(A) \cdot P(B|A)}{P(B)}$$

where:

$P(A) =$ The probability of A occurring

$P(B) =$ The probability of B occurring

$P(A|B) =$ The probability of A given B

$P(B|A) =$ The probability of B given A

$P\left(A \bigcap B\right)) =$ The probability of both A and B occurring

WHEW!

I know that was a long chapter filled with lots of new information. Take a break to digest it all or further your statistics knowledge with a course I enjoy that takes you further into some of the topics I was unable to cover.

4

Data Retrieval with SQL

One skill I wish I had before I started working in industry as a Data Scientist would be better SQL skills. When I got to my first role at a large company (2,000+ employees), I definitely realized my data retrieval skills weren't up to par. I wasn't only tasked with conducting analyses on data, but I also had to get the data I needed to extract the correct insights accurately. What got me over the SQL hump? I learned WHY gathering data correctly was a large part of Data Science work. The data I retrieved was pivotal for finding patterns and creating accurate models.

This chapter isn't exhaustive of everything you need to know but will take you from wondering what's the big deal with the all caps keywords in SQL to feeling comfortable getting the data you need for analysis.

I've seen early career data scientists struggle to locate the data they need for analysis; this is a situation where it's best to lean on senior team members and don't be afraid to ask questions. Rely heavily on seniors to provide common SQL code or scripts to get you started.

Something I wish I knew before starting a career in Data Science was how much time is spent on data cleaning, validation, and sampling. It's easy to get excited about the prospects of creating high-tech AI systems, but dealing with dirty data is one of the most difficult parts. While we're in a digital age that creates petabytes of data, it's easy to assume much of that is clean and ready for analysis, but that's rarely ever the case. Companies hire Data Engineers to create pipelines and systems that make sampling and modeling data seamless for Data Scientists. Despite this, it's critical that Data Scientists have the ability to access their data independently. As industry data is often disparate, this may mean complex SQL queries, pulling from APIs, using real-time database sources, or one of many other data options.

4.1 The What and Why of SQL

SQL is a prevalent coding language that helps users create and get data from databases. While it may be straightforward why Data Scientists use SQL, but lawyers, journalists, and doctors can leverage SQL to manage large amounts of data in their database. SQL isn't a new kid on the coding block, and since it's been around since 1974, it's incredibly helpful how ingrained it is in many systems.

One of the many reasons you'd use SQL is the limitations of many other pieces of tech like Microsoft Excel. With a maximum of 1,048,576, SQL databases let you work with terabytes of data. There is a lot of SQL involved that allows you to create your own database, but we'll focus on the common keywords you'd use to get data from or query a database. We'll focus on data gathering because in the vast majority of cases, people in Data Science

roles work less in creating new tables and more on getting what they need from data stored there.

4.2 Data Retrieval

So, let's say you work for an online cosmetics store and you want to know how many sales you made last month. With a single line of SQL, you can get every row from that table. Below July20_Orders is the name of the table that has all the data from orders placed in July 2020.

```
SELECT * FROM July20_Orders;
```

What does the asterisk mean?

The * between select and from is the wildcard; it's a standing that represents every value for the column name. This short line is a shorthanded way to get every column from our July orders table. In the real world, if you were to write a query like the one above to see what columns you have in your data, you'd write something like this:

```
SELECT * FROM July20_Orders
LIMIT 10;
```

Using limit just restricts the number of rows retrieved. One thing to be careful of with SQL is the dreaded big data; if you millions of rows in your dataset, or even thousands, it's important that you're consistently thinking of how to retrieve the smallest

amount of data you need. This is because many frameworks are operating under the constraints of memory and speed. It's important to consider how many of your colleagues work on the same severs and not create bottlenecks for your team. That being said, we can select just a few column names like so:

```
SELECT buyer_id, order_id, quantity FROM
July20_Orders;
```

The above query gets all rows from just those columns. Let's say we're interested in only unique values for quantities. As you can imagine, the order data for a website may have hundreds of orders with quantity 1 and fewer orders with more items purchased at once. To discover the range of values in a column, we can use the DISTINCT keyword, which can also be used on more than once column at once.

```
SELECT DISTINCT quantity
FROM July20_Orders;

SELECT DISTINCT buyer_id, quantity
FROM July20_Orders;
```

Now that we've gotten just unique values, we can sort how our data is displayed using ORDER BY. We say ORDER BY followed by the columns to sort. This doesn't change the order of the original table, but just how we view results. Order By defaults to ordering data by smallest to largest unless you specify using the DESC keyword at the end of the column list to get rows in largest to smallest order.

```
SELECT buyer_id, quantity
FROM July20_Orders
ORDER BY Quantity DESC
```

The query above would give us the following results. Sorting works well on numerical columns, but columns like buyer names don't always just give us names in reverse alphabetical order.

4.3 Filtering

To trim down the number of rows we retrieve, we can apply filters to our query that only gets data that meets certain criteria. Using orders as our examples, we can find all orders where they purchased more than 2 products by filtering on our quantity column.

```
SELECT buyer_id, quantity
FROM July20_Orders
WHERE quantity > 2
```

WHERE is one of those keywords that can use many types of comparison and matching operators to filter data.

```
SELECT order_id, quantity
FROM July20_Orders
WHERE quantity != 1
```

For more complex queries, we can combine logical operators with AND and OR keywords. These comparison operators exist to compare two values for filtering. For example, if we want to find exact values for someone's name or company name. We can also find customers who have loyalty accounts and have had a

minimum number of orders. Comparison operators make using filtering functions like WHERE easily. You can use comparison operators on both numerical and non-numerical data.

```
SELECT *
FROM customers
WHERE loyalty_acct = TRUE
    AND num_orders > 5;

SELECT *
FROM customers
WHERE last_name = 'Cole'
    OR last_name = 'Johnson';

SELECT *
FROM customers
WHERE street_name = 'Martin Luther King Jr Blvd'
    AND (num_orders > 5 OR loyalty_acct = TRUE);

SELECT *
FROM customers
WHERE order_date > 'January'
```

The challenge and the exciting part of SQL is what you can do when you put it all together.

```
SELECT first_name, last_name, loyalty_acct, order_date
FROM customers
WHERE loyalty_card = TRUE
AND num orders > 15 & num orders < 30
ORDER BY order_date DESC;
```

You can also perform arithmetic in SQL using the operators +, -, *, /. The main thing to note is that you can only do this arithmetic across columns on values in a given row in SQL. To

perform arithmetic across multiple forms, we can use aggregate functions!

```
SELECT year, Q1, Q2, Q3, Q4, Q1 + Q2 + Q3 + Q4 as
total
from sales
where year >= 2018
```

4.4 SQL Operators

A large part of how we filter data is by learning to use logical and comparison operators to limit the data we retrieve. The most basic way to filter data is by using these comparison operators to compare values.

Comparison Operators:

Equal to =
Not equal to <> or !=
Greater than >
Less than <
Greater than or equal to >=
Less than or equal to <=

We can apply these comparison operators to numerical columns or non-numerical columns where it makes sense. We can also leverage logical operators to create more advanced filters.

LIKE allows you to match similar values instead of exact values.
IN allows you to specify a list of values you'd like to include.
BETWEEN allows you to select only rows within a certain

range.

IS NULL allows you to select rows that contain no data in a given column.

AND allows you to select only rows that satisfy two conditions.

OR allows you to select rows that satisfy either of the two conditions.

NOT allows you to select rows that do not match a certain condition.

If you want to search for these patterns in strings, we can use the keywords LIKE and ILIKE alongside WHERE. In the table above, we didn't have any string data so; we'll have a further look at our imaginary table of customers. Let's keep thinking about our online makeup store. In our customers' table it'd be likely we'll see information about our customers like their name, their address, how many orders they've placed and when, if they have a loyalty account, and more. The % used below represents any character or set of characters. In this case, % is referred to as a "wildcard." LIKE is case sensitive, so B is not the same as b. If you want to find all instances of a letter regardless of capitalization, you'd use ILIKE.

```
SELECT first_name
FROM customers
WHERE first_name LIKE 'Breanna%';

SELECT first_name
FROM customers
WHERE first_name ILIKE 'Breanna%';
```

IN is a logical operator in SQL that allows you to specify a list of values that you'd like to include in the results. This can be used

for both numerical and non-numerical columns.

```
SELECT *
FROM customers
WHERE first_name IN ('Taylor', 'Sam', 'Rebecca')
```

BETWEEN is another logical operator that allows you to select only rows that are in a specific range. This is often used for dates.

```
SELECT *
FROM customers
WHERE order_date BETWEEN 2009 AND 2020
```

If you want to find rows with missing data, you can use the IS NULL operator to easily select rows with null values.

```
SELECT *
FROM customers
WHERE last_name IS NULL
```

Does your data need to satisfy two conditions? If so, you can use the AND operator to only pull records that meet two or more conditions.

```
SELECT first_name, last_name
FROM customers
WHERE order_date > 2018
AND num_orders > 10
AND loyalty_acct = TRUE
```

If you want the data, you retrieve to meet one of two conditions, you can use the OR operator. The OR operator returns true if only one condition is met. The query below will pull all customers

who have placed more than ten orders or who have a loyalty account. While both conditions don't have to be met to return true, but customers who have both more than 10 orders and loyalty accounts will also be filtered into your group if you use the OR operator.

```
SELECT first_name, last_name
FROM customers
WHERE num_orders > 10 OR loyalty_acct = TRUE
```

To invert the meaning of a statement, the NOT operator is quick to see the reverse results from your query. NOT is most commonly used with LIKE when you know what results you don't want to include.

```
SELECT first_name, last_name
FROM customers
WHERE first_name NOT LIKE 'Lewis'
```

We can also use NOT to find our non-null rows.

```
SELECT *
FROM customers
WHERE first_name IS NOT NULL
```

After we've filtered our data down a bit, we can sort how the data is displayed back to us. For that, we can use the ORDER BY clause to reorder our results. The following query returns all of our customer's names in alphabetical order based on their last name. This alphabetical way of ordering is called ascending order and is the default way of ordering data in SQL. Numerical columns that are in ascending order start with the smallest numbers (often 0 or negative numbers) and go higher as the rows continue.

```
SELECT *
FROM customers
ORDER BY last_name
```

If you want to order your data in the opposite way, for example, to see the customers who have placed the most orders, we'd use DESC to get our data into descending order.

```
SELECT *
FROM customers
ORDER BY num_orders DESC
```

Most "flavors" of SQL will automatically sort your data in ascending order or smallest to largest unless you specify otherwise. You can write a query like the one below, but it's a bit redundant.

```
SELECT *
FROM customers
ORDER BY num_orders ASC
```

What kinds of queries will I write on the job?

This depends slightly on the level of SQL proficiency the job requires, but the following concepts outlined by proficiency level should help you further your SQL practice. As you get started, SQL and Python will be your most valuable skills!

Basic SQL: Selecting data from somewhere to answer a question

1. SELECT and WHERE for filtering and selection
2. COUNT, SUM, MAX, GROUP BY, HAVING for aggregating data
3. DISTINCT, COUNT DISTINCT
4. OUTER and INNER JOINs and when/where to use them
5. SELF JOINS
6. Strings and time conversions
7. UNION and UNION ALL.

Intermediate SQL: Efficiently pulling various data views

1. Common Table Expressions (CTEs)
2. DML/DDL/DCL concepts
3. Handling NULLs creatively (COALESCE)
4. Subqueries and subqueries efficiency
5. Temporary tables
6. Self joins
7. Window functions like PARTITION, LEAD, LAG, NTILE
8. User-defined functions
9. Use of indexes in querying to make operations faster

Advanced SQL: Mastering moving, manipulating, and automatic SQL queries

1. Recursive CTEs
2. Dynamic SQL generation
3. Query optimization
4. Materialized views

What about soft skills?

Even retrieving data takes some tenacity and grit. Companies are looking for people who are comfortable dealing with bad data, analyzing and visualizing data, and teaching and listening.

5

Data Storytelling

One of the biggest skills often overlooked when Data Science is talked about is Data storytelling. I see data visualization as an important aspect of data storytelling; it just uses visual tools to get our point across. Data storytelling is so important because the bulk of our job as Data Scientists is to inform or help make decisions with our findings. There's a huge gap between how you convey information that determines if executives and those relying on your analysis listen to your recommendations or ignore them.

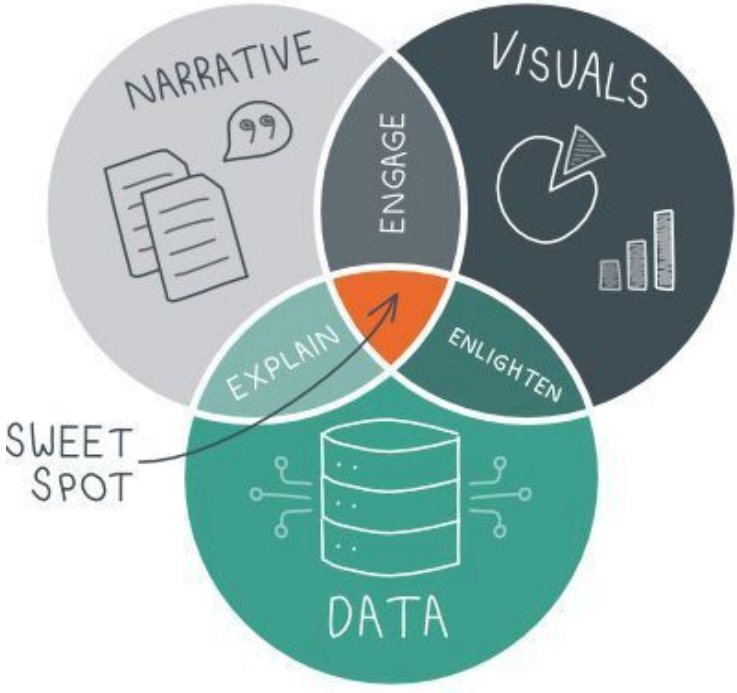
I've found what's most important is to know your audience. Sometimes we technical folks want to talk about what we think are cool to us, how we sampled our data, the features we engineered, and other nerdy details that execs rarely care about. What's important to them typically is the bottom line. What do our findings mean for them? This chapter will be about tailoring your data story to get the best feedback from stakeholders.

We have to think about data as a story and as a tool to tell stories about our data. In many cases, this data is about our software users, but it can also be census data, healthcare data,

or measurement data like from sensors. All good stories have the same components: a beginning, middle, and end. If we think about how we use business data, it helps us define what our story is. Depending on where you get a job, Data Storytelling might be part of the visualization portion, communication portion, or both. The same way we're taught to construct a story in grade school writing courses is the same way we do this in industry. Today's tools can offer unparalleled insights, but these tools require savvy operators who speak the language of data to extract them.

In this chapter, you'll hear me talk about providing insights to companies, but what are insights really? The word insight comes from Middle English for "inner sight" or "sight" with the eyes of the mind. Another way of thinking about insights is that, as psychologist Gary Klein described, "an unexpected shift in the way we understand things." Informing and communicating are different concepts. When you inform someone of something, you're simply passing along information. When you communicate, you take the additional step to make sure they understand it as well.

> *Numbers have an important story to tell. They rely on you to give them a clear and convincing voice.*
>
> *–Stephen Few, Data Visualization Expert*



Data storytelling relies on three main elements to effectively communicate insights: data, narrative, and visuals. Our visual elements are discussed the most, but we can leverage narratives and using the right kinds of data to get the point across

5.1 Stories > Statistics

While it may be tempting to "stick to the facts," a Stanford study found 63% could remember stories, but only 5% could remember a single statistic. Stories have the ability to both persuade and engage people. As mathematician John Allen Paulos observed, "In listening to stories we tend to suspend disbelief to be entertained, whereas in evaluating statistics we

generally have an opposite inclination to suspend belief in order not to be beguiled." When we look at statistics, we tend to be more critical of the number we see and nitpick details in plots and graphs. This has happened to me many times when simply trying to share my observations. If you're speaking to other data-savvy folks, they may get caught up in your charts' legend, units, or other details while focusing less on what the data is saying.

To craft our story, we need to understand a few things first. When you land an industry job, this is much easier to parse out, but we should have a clear idea of our objective.

1. What is the business goal? Are you trying to increase revenue, create a metric to compare against, or add value to the product?
2. What kind of data do we have access to? Does our data work as a good proxy for the problem we're solving?
3. What are the results of our analysis? Should we proceed with integrating a model into the product? Do we have the right kind of data to predict things about our users?
4. What is the context? Here we answer the Who, What, Where, Why, and How of our data.

Who's Listening?

To tell a good data science story, it's important to know who your audience is. Stakeholders in your company may expect the fine details while outside audiences might not. Understand who your audience is and what kind of tone your story or presentation should represent. It's important to know your audience's goals, so your narratives align with the information they; 're seeking.

It's also good to know their professional expectations and familiarity with the topic. You may be working with researchers who understand statistical concepts or business groups who are looking for data in terms of business metrics.

As Data Science grows, many are slowly diverging from the idea that its magic or a silver bullet is used to fix all types of problems. It's important to let your audience know why you're working on this (even if it was assigned to you), as well as what limitations you had. This can be the size of your data, if it's robust or filled with null values, or if you had issues accessing it. This may be too much detail for those outside your organization, but I've found it typically enlightening for audiences to learn about how I outlined and executed a data analysis project. Sometimes it's okay to let folks in on how hard our work can really be. Talk about removing outliers like ages defaulting to 0 or that you spent a lot of time discovering why there was a steep decline in samples from a specific age or racial group.

5.3 Data Visualization

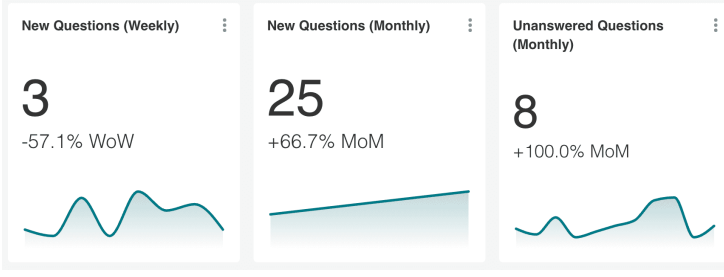
Data Visualization can be described in a nutshell as the process of creating visuals to get a better understanding of the data. One of the reasons visualization is so powerful is because we, as humans, can recognize patterns quickly and efficiently if they're displayed the right way. Before creating any charts, it's important to reflect on the kind of data we want to present. First, think of why we want to visualize the data at hand. Do we want to highlight a specific value that sticks out, or do we want to show

the comparison between data and a feature like time or other values? Are we trying to show our audience the relationships in our data, or are we more concerned with what our data is composed of?

Most data visualizations exist to help us understand the composition of our data set, the relationship between variables, the distribution of values, the trends over time, variables compared to each other, spatial area (maps), or the flow of data information. There are many different types of visualizations to use, and your selection should be based on the type of value you want to highlight.

Specific Value

Used to highlight specific values, we simply show the raw number prominently displayed. When doing so, we want to make sure the value is eye-catching and stands out. It should be big enough to be read at a glance.



Table

Show the exact values and compare pairs of related values. To make tables that make sense, arrange time-based data horizontally and show rankings vertically from most important at the top to least important at the bottom. Always include column headers at the top of the table.

Name of cereal	Amount of elemental iron from least to greatest
Coco Puffs	3
Total	5
Corn Pops	1
Cheerios	4
Fruit Loops	2

Highlight Table

Show off exact values and use color to convey the relative magnitude. Color is a core component of highlight tables. If you're showing ordered data, used a sequence of colors in the same hue where lighter colors correspond to smaller values and darker colors to larger values.

	January	February	March	April	May	June	July	August	September	October	November	December
Accessories	$5,478	$5,369	$8,735	$7,981	$9,615	$8,858	$17,135	$11,758	$25,400	$13,087	$25,477	$28,486
Appliances	$3,176	$4,933	$6,700	$6,075	$7,526	$7,479	$3,384	$12,862	$10,828	$9,155	$18,306	$17,107
Art	$966	$1,006	$1,413	$2,382	$2,256	$2,182	$2,102	$1,690	$3,660	$1,905	$3,816	$3,740
Binders	$12,412	$4,286	$13,728	$13,384	$9,245	$13,218	$7,755	$21,302	$37,337	$18,090	$20,789	$31,867
Bookcases	$5,062	$1,940	$7,147	$4,926	$6,290	$7,445	$10,292	$5,622	$12,849	$8,771	$23,561	$10,977
Chairs	$11,285	$7,768	$20,832	$18,855	$25,703	$21,145	$23,585	$17,770	$52,147	$21,905	$47,314	$60,141
Copiers	$3,960		$22,590	$6,880	$18,400	$900	$9,780	$5,730	$10,320	$37,020	$15,150	$18,800
Envelopes	$750	$669	$1,657	$852	$1,190	$514	$1,200	$701	$2,177	$1,393	$2,917	$2,458
Fasteners	$88	$159	$150	$258	$109	$116	$182	$235	$414	$326	$548	$441
Furnishings	$3,980	$2,316	$5,068	$7,185	$7,305	$5,900	$7,355	$4,343	$11,805	$5,447	$16,757	$14,244
Labels	$207	$300	$940	$408	$885	$1,207	$1,692	$876	$1,476	$1,269	$1,850	$1,376
Machines	$7,215	$8,990	$35,052	$18,190	$11,268	$12,183	$4,065	$6,262	$26,386	$10,613	$33,807	$15,210
Paper	$2,287	$2,805	$6,218	$3,865	$6,359	$6,546	$4,319	$6,360	$10,575	$5,309	$12,563	$11,274
Phones	$13,772	$9,000	$26,712	$18,647	$24,859	$25,492	$23,807	$28,046	$38,464	$25,963	$56,075	$39,169
Storage	$9,374	$6,125	$14,793	$15,806	$14,670	$17,272	$13,768	$17,421	$29,866	$15,822	$37,418	$31,510
Supplies	$4,403	$289	$10,607	$6,246	$1,154	$1,267	$8,816	$859	$6,442	$816	$1,372	$4,402
Tables	$10,952	$4,218	$16,913	$9,913	$9,288	$15,360	$10,344	$17,752	$19,626	$20,223	$31,401	$40,075

Heatmap

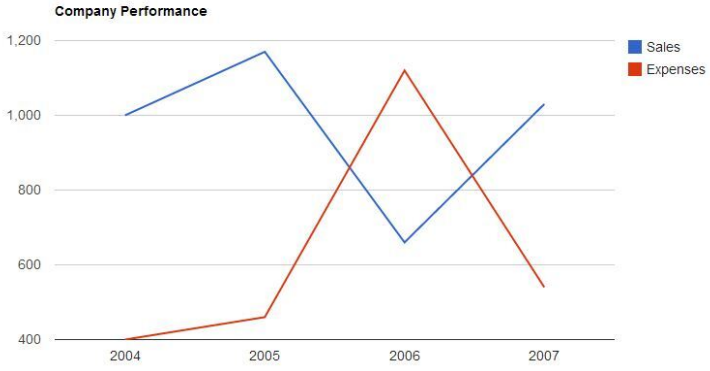
Heatmaps help us to compare values by encoding the marks with color and size. Sizing is crucial and is representative of the values so ensure marks are sized in scale. Ensure your heatmap is clearly labeled, so it's easy to understand the interaction between the variables.

Single Line Chart

We can use a single line chart to compare trends over a period of time for a single feature. To do this well, we want to select the right interval for our data's tick marks. In many cases, the y-axis will start at zero; if it doesn't in your chart, you'll need to clearly point this out to avoid confusing people. It's usually a good idea to get rid of gridlines and add data labels to the starting and ending points or min and max values in your view.

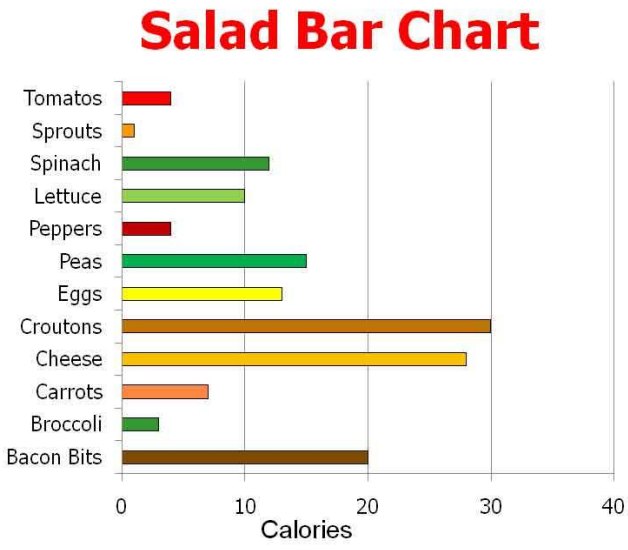
Multiple Lines Chart

When we attempt to display trends over a period of time for multiple categories, it can get messy. Try not to compare more than 5 lines at once. Each new line you add increases the time it takes a user to understand the chart. Color is a good way to differentiate the different lines, but don't use sequential colors as they can be confusing for these charts.



Bar Chart

Bar charts help us show comparisons among different categories. In order to make a good bar chart sort your data either from largest to smallest and keep each bar the same color. Suppose you have to display a lot of categorial flip it, and make a horizontal bar chart. Sometimes we need data labels to better see the bars' height, but you should never ever have a baseline that's something other than 0.

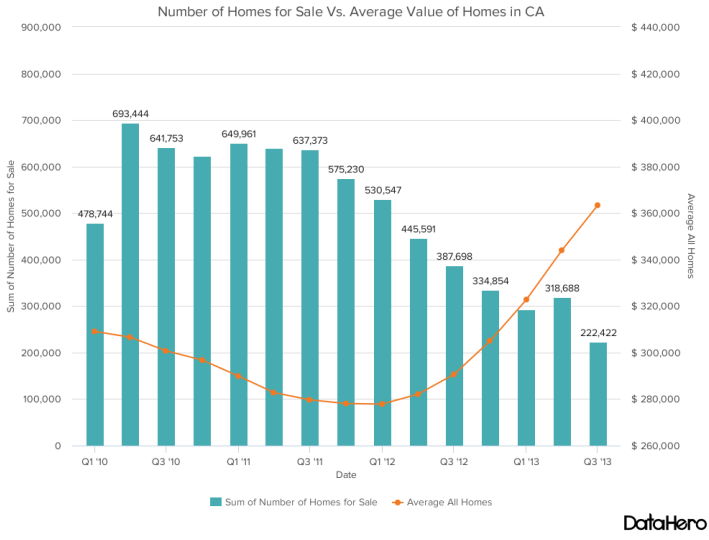


Histograms

While bar charts are helpful, they differ from histograms that show us only the underlying shape of a set of continuous data. Unlike bar charts, histograms can only be used for numerical data or counts of categorical data.

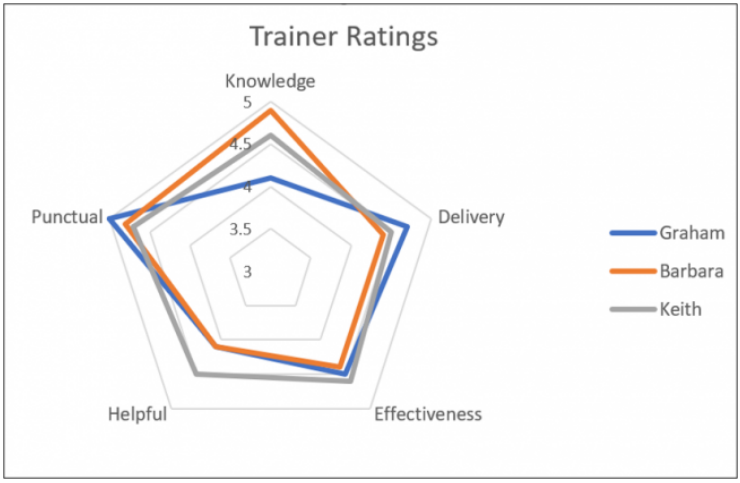
Dual Axis Chart

Show the relationship between two variables with different magnitudes and scales. By and large, these are not recommended. They are more often too confusing for audiences, and the same data can be represented better with a story or multiple charts.



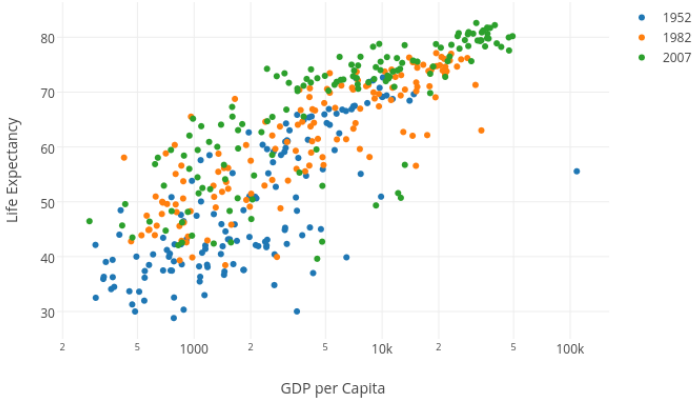
Radar Charts

We can use radar plots to see quantitative data in a radial plane. We can plot one or more series of values over multiple quantitative variables. These plots are good at helping us understand the collective attributes of a dataset like personality traits. Try not to plot more than 8 categories as these can become hard to read with each additional one. Radar plots are best used with other plots to compare values over time.



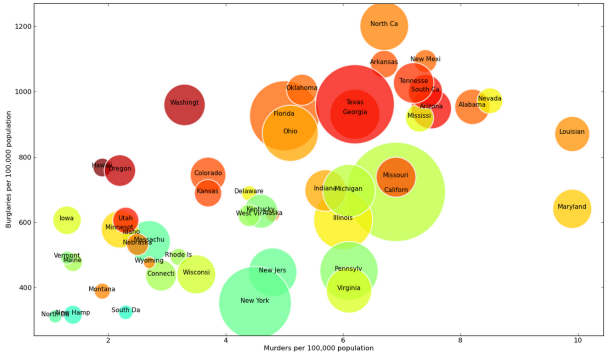
Scatterplots

Scatterplots help us show the relationship between two variables. A good best practice for scatterplots is to properly label both axes as well as include a legend. You can have scatterplots of all the same color, but using distinct colors for each class is preferable.



Bubble Charts

These display three dimensions of data, x,y location, and its size. The bubbles can be different colors or the same, but what matters most is that the size of each bubble represents how large its value is. Buble charts help us see relational values without regard to axes. Avoid using sequential colors as they can be confusing.



There are a whole lot of other charts I didn't cover, but what's important is that you know what you want to get across before creating a new visualization.

Visualization Best Practices

Decide what you want to visualize before you visualize it. Start this process by thinking about what you need the audience to understand. Do you want them to know that sales are good? Do you want them to see what the data is composed of? Is there a variable like total revenue you know they're interested in, and you can compare to other values? Do they need to make a decision based on what you show them? If so, what do you want them to decide? Close your eyes and imagine what they should learn from your visualization.

A good data visualization is made up of several components that have to be together. The first is the Data Component. An important first step in deciding how to visualize data is to know what type of data it is, e.g., categorical data, discrete

data, continuous data, time-series data, etc. Next, we should decide what kind of visualizations are suitable for your data, e.g., scatterplots, line graphs, bar plots, histograms, Q-Q plots, smooth densities, boxplots, pair plots, heatmaps, etc. After we know what kind of visualization to build, we need to decide which variables go where or, in other words, what variable to use as your x-variable and what to use as your y-variable. Don't forget that you also have power over the scale of your visualization. Is it on a linear scale or log scale? Lastly, consider if you should have a legend, axes labels, and other aesthetic components.

5.4 Data Storytelling

It may seem like Data Storytelling is important, but not so important in the grand scheme of things, and I want you to know that is **far** from the case. While strong data storytelling doesn't make up for lacking statistics foundations, knowing stats without being able to communicate them well is hardly an accomplishment. An example of bad or the lack of data storytelling is the account of Ignaz Semmelweis. He was a nineteenth-century obstetrician who discovered handwashing could save countless lives, but he failed at communicating his findings well.

In these times, Semmelweis noticed the doctors had a high mortality rate at 9.9%. Compared to midwives at another Vienna clinic whose midwives only had a 3.9% mortality rate. Semmelweis made a discovery after a colleague was accidentally poked with a student's scalpel then died shortly after. What was the cause?

When comparing both clinics, Semmelweis found that stu-

dents at the first clinic would start the day doing autopsies then spend the rest of the day treating patients without washing their hands afterward. Compared to the doctors, the midwives weren't performing autopsies nor were exposed to cadavers in the same way. So Semmelweis implemented a strict hand-washing policy for the doctors and saw a dramatic decrease. During the month he implemented the policy, the mortality rate for doctors was 12.2%. After the policy was implemented, the mortality rate dropped all the way to just 2.2%. Unfortunately, he couldn't prove why his policy worked. His data was ignored, his life-saving ideas were rejected, and his colleagues sadly discredited him. There can be incredible negative impacts when we're unable to communicate our data effectively.

> *It is dangerous to be right in matters on which the established authorities are wrong.*
>
> *-Voltaire*

Keys to a Good Data Story

1. Data is the Star: The building blocks of every data story is the data itself! Each story is formed from a collection of information that we've extracted using analysis.
2. Main Point: Clearly outline the point of your story. What's the main idea or insight you need to take away. All other supporting data should back up this point, but what do you want your audience to remember from your story?
3. Explanatory Focus: It's not enough to just show people data; often we misinterpret it anyway. We should explain what's happening in our data by giving the who, what, when, where, and why. What are the reasons (you think)

you're noticing whatever cool insight you found?

4. Linear Sequence: Tell your story in a linear sequence despite the fact that the data may not be linear. It's; important to describe the series of events and highlight notable events. Try to frame your story chronologically and expose your audience to these findings slowly.
5. Dramatic Elements: Did you find anything unexpected that would surprise your audience? Were there red herrings or twists you took along the way?
6. Visual Anchors: Show, don't tell. You want to compliment the narrative of your story with visuals that show what you're talking about. Skip the long bullets and put that in your speaker notes. Say it, show what will be memorable.

For more in-depth knowledge of how to build data stories that people actually want to listen to, I recommend reading Storytelling with Data, which focuses on the aspect of creating stories that make sense to real people! The book lends helpful tips and warns we shouldn't stick to your visualization builder's defaults, remove elements to make clear and concise charts.

6

Feature Engineering

Feature engineering is one most important parts of Data Science as it directly influences our data models' outcomes. Beginners should think of feature engineering as the task they **absolutely** have to be comfortable with if they want to land a job that uses machine learning. Feature engineering is the process of formulating the most appropriate features given the data, model, and task. Imagine you have a thermometer sensing people's body temperature. If we take only the raw sensor readings, the data won't look very informative. We'd just have a list like 99.0, 96.1, 98.5, and 97.9. Even with terabytes of this data, it'd be impossible to do anything with it. We can't predict the chance of having the flu due to increased temperature or get insights other than the average temperature of the people in our sample.

Feature engineering takes us from this type of data to rows and columns that are compatible with modeling. Most machine learning models take data as inputs and learn some function to predict an outcome event. In this chapter, we'll go through some popular steps to get your data ready for modeling.

Feature engineering, while one of the most important aspects of Machine Learning, is one part most impacted by our individual biases. For example, companies like Microsoft and IBM sold various facial recognition APIs that only use binary gender classifications. There is often much hype surrounding projects like these, but rarely covered is that 90% of these facial recognition algorithms exclude non-binary people. This is an area where bias may not be intentional, but the way our dominant culture has impacted even researcher thinking creates harmful models. Also, these systems work drastically worse for women and people with dark skin.

Data Scientists are masters of our own demise. We're able to choose our evaluation metrics and how we create features in our training data. It's common that Data Scientists under pressure result to **P-hacking** or reanalyzing data in many different ways to yield a specific result that usually confirms their prior beliefs or meets incentives. If we neglect to understand the correlations between training data features, we can unintentionally create models based on bad features like highly correlated proxies. Many researchers have assumed not including features like gender and race means their models are unbiased, but that's far from the truth. These sensitive features are often highly correlated to columns that arent protected classes like weight and zip code. You might be wondering how to decide if a feature is good or usable; this is a hard question to answer even for professionals. It depends on what you're trying to approximate with your model, the context of your data, and many other factors. What's important to know is how to create features and measure how well they encapsulate the signal in the noise.

6.1 Understand Your Features

First, let's check the magnitude of our data. For our analysis, is it important that some variables are positive or negative? Does the range of a feature matter? If we look at the scale of numerical features, it's important to know if they span several orders of magnitude.

Models that are smooth functions are susceptible to the scale of the input data. Any modeling method that uses Euclidean distance, such as K Nearest Neighbors or k-means clustering, should take normalized features as its inputs. Normalization ensures that our output stays on an expected scale and that we're accurately capturing relationships. However, logical functions like logistic regression and the step function aren't sensitive to input features' scale. The output of these models is binary regardless of the scale of inputs. Decision trees are step functions repeated over input features, so models based on partitioning data like decision trees, gradient boosting machines, and random forests aren't sensitive to scale.

Next, we can consider the spread or distribution of our numerical features. Are they distributed evenly around the mean or highly skewed? This matters for some models like linear regression that assume prediction errors have a Gaussian or normal distribution. The exception to this rule is if the feature distribution spans several orders of magnitude. In this case, the normal error assumption no longer applies. This is a fairly common problem to face as many Data Scientists rely heavily on basic statistical methods like linear regression. If you have a feature that spreads several orders of magnitude, one solution is to transform our target variable to reduce the spread of a certain feature.

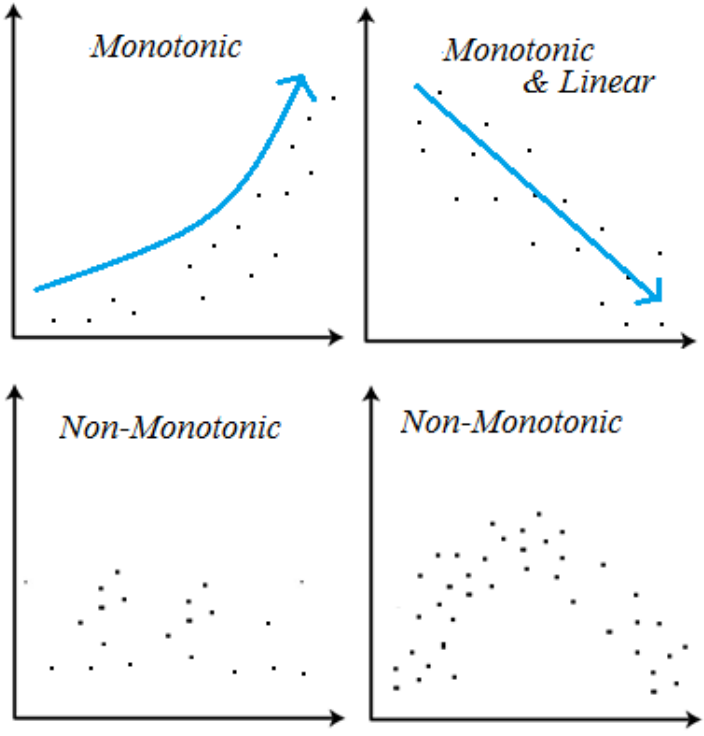
In the image below, we can see the magnitude of Cryrotherpay data and its results on treating warts.

	sex	age	Time	Number_of_Warts	Type	Area	Result_of_Treatment
count	90.000000	90.000000	90.000000	90.000000	90.000000	90.000000	90.000000
mean	1.477778	28.600000	7.666667	5.511111	1.700000	85.833333	0.533333
std	0.502304	13.360852	3.406661	3.567155	0.905042	131.733153	0.501683
min	1.000000	15.000000	0.250000	1.000000	1.000000	4.000000	0.000000
25%	1.000000	18.000000	4.562500	2.000000	1.000000	20.000000	0.000000
50%	1.000000	25.500000	8.500000	5.000000	1.000000	70.000000	1.000000
75%	2.000000	35.000000	10.687500	8.000000	3.000000	100.000000	1.000000
max	2.000000	67.000000	12.000000	12.000000	3.000000	750.000000	1.000000

Numeric Variables

In feature engineering, we use power transforms, like log transforms, to distribute our data closer to the normal distribution. In an ideal world, we'd simply turn our original data into a dataset ready to train models, but in reality, we take lots of disparate data, engineer features, and create an ML-ready dataset to train models on. When we create features of numerical variables, it's a lot easier to deal with missing data and for our models to interpret them better.

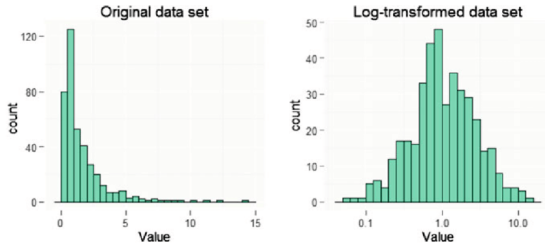
Power transforms a statistical family of methods used to preserve the monotonic relationship in data. A monotonic function is one that preserves the ordered relationship in a dataset. Sometimes we want to reverse this relationship but keep the order. Below you can gain an intuition of what a monotonic set of data points looks like.



Log Transformation

In Data Science, we use the log transformation to compress the range of large numbers and expand the range of small numbers. What our log transformation does is replaces each variable x with log(x). The larger x is, the slower log(x) increases. This transformation is useful for dealing with positive numbers in a heavy left or right-tailed distribution. There is a larger likelihood of an event in the tail than in normally distributed data in these distributions. Log transforms help us deal with data that is very spread out. You can think of this as "zooming"

in to see the detail in the distribution better. This is shown in the figure below.

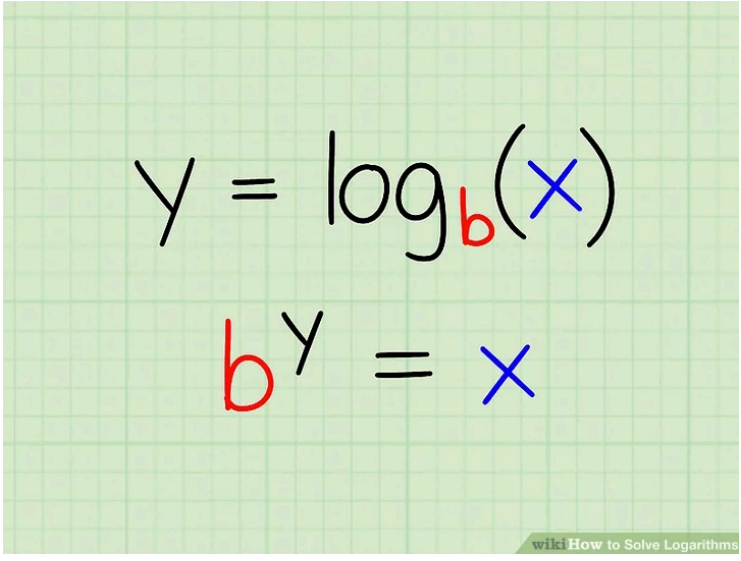


When performing data transforms, it's important to leverage

probability plots to visually compare real data distribution against theoretical distributions.

A bit about Logarithms

Logarithms are tools in statistical modeling and statistical analysis. A logarithm can be defined with respect to a base (b), which can be any positive number. The formula is below.

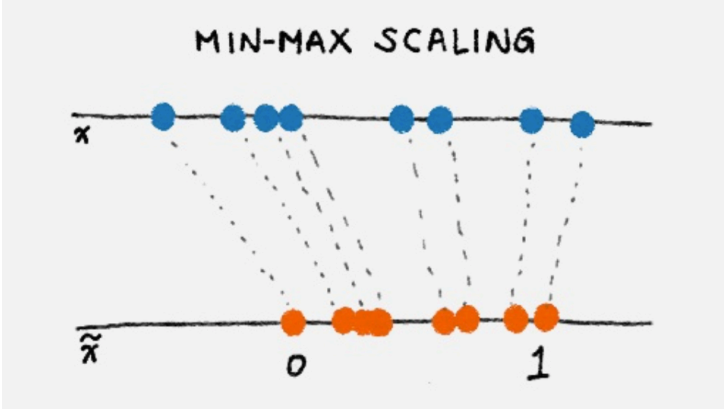


The logarithm of any number X is equal to **y** because X equals to the b to the power of y. The most commonly used bases used for logarithms are Base 2, Base 10, and the Natural Log, which you may see referenced as the mathematical constant "e" or Euler's number. When you see "e" floating around, just remember it's about 2.7 (2.718282).

Feature Scaling

As mentioned earlier, models that are sensitive to the scale of the input need to be normalized. It's important to recognize algorithms that are smooth functions of the input like linear and logistic regression are affected by the scale of the input, whereas this isn't an important step for models like decision trees. There are various methods to scale our features, and we do this on an individual feature basis.

Min-Max Scaling: This is one method that looks at the minimum values of our individual feature value. This scaling method squishes or stretches all values to b within the range of 0 to 1.



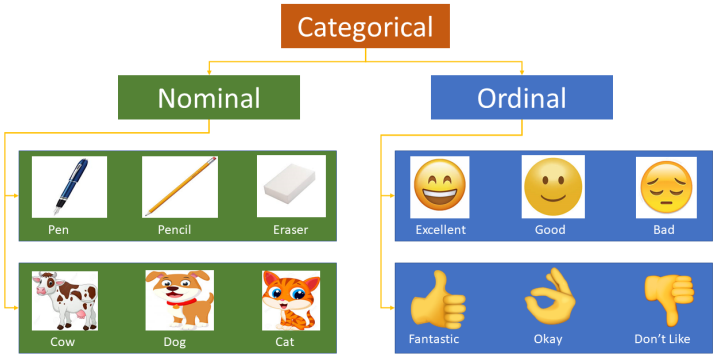
Standardization (Variance Scaling): This method of transforming data ensures a numerical feature has a mean of 0 and variance of 1, meeting many machine learning algorithms' assumptions.

Euclidean or e^2 Normalization: Scales individual rows and divides those by values by the vectors' L2 norm aka Eucliendan norm. Normalizing our feature vectors helps us meet the assumptions of vector space models that envision models in 2D or 3D planes. Vector space models are used frequently for classifying text and data clustering.

Categorical Variables

There are a few things I can generalize about all the different kinds of data you may interact with in your career, but you'll almost always have to manipulate categorial variables before they are ready for modeling. If we have high cardinality or it can create very sparse datasets. It's also really hard to deal with missing data when we can't calculate an "average" category or guess based on other nearby data points. There are a few context clues that allow us to easily predict what a missing value should be. Categorical variables can either be nominal where we aren't concerned with the relationships between each value, or ordinal

where excellent is definitely better than good and good is better than bad.

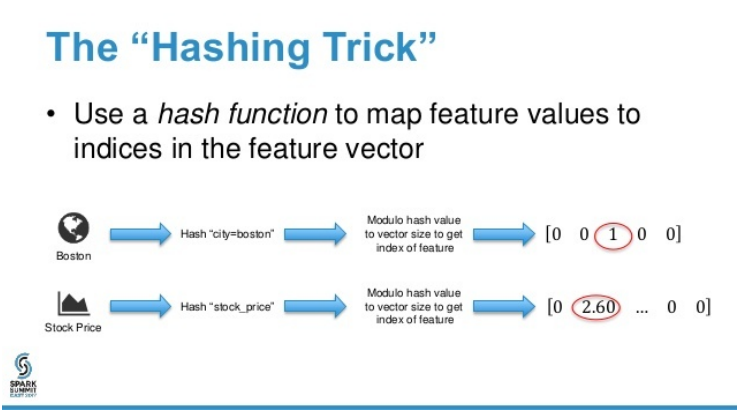


One-Hot Encoding: Transforming a categorical feature with x number of total features into x binary features. If the variable can't belong to more than one category, then only one bit in our encoded category can be "on." This sparse format we create with one-hot encoding is helpful when working for in-memory data or data that we're working with on our computer and not the cloud. As you can see below, we can one-hot encode categorical data like colors.

color	color_red	color_blue	color_green
red	1	0	0
green	0	0	1
blue	0	1	0
red	1	0	0

Feature hashing: This process hashes categorical values into vectors that have a fixed length. Most ML algorithms require you to represent input data as a real number matrix. Feature hashing has lower sparsity than one-hot encoding but offers higher data compression. This deals with new and rare categorical variables well but has the potential to introduce data collisions. According to the R FeatureHashing package's introduction:

> *"Feature hashing, also called the hashing trick, is a method to transform features to vector. Without looking up the indices in an associative array, it applies a hash function to the features and uses their hash values as indices directly."*



Category embedding: We can also leverage neural networks to create dense embeddings from categorial variables. This allows us to essentially map variables to a function approximation problem. This allows us to train models drastically faster and uses less memory. In my experience, this has given better accuracy than the one-hot encoded method.

Okay, so we've covered numbers and categories and can now explore other common data types you may have to deal with.

Temporal Features

Time Zones can often be difficult to work with because it's common we have multiple time zones in some countries. We also need to be aware of the start and end dates for Daylight Savings Time.

Time binning: By binning our time feature, we're able to make them categorical and more general. This means we break our

time into hours or periods of the day. We can also extract time periods like the weekdays, weeks, and quarters.

Trendlines: We can encode variables over time to understand the amount of ad spend in the last week, month, quarter, or year. This gives our algorithm a trend to learn that's useful in predicting future events. Maybe if we're dealing with time-sensitive data, we want to hard code categorical features from dates.

One way of thinking about this is recording a date's closeness to major events like a product launch or a new competitor entering the market. This could aslsop be holidays or closeness to a customer's payday. We and also consider the difference in time between important dates. Maybe there's a delay in users downloading our app and actually registering for it. This might be important information for our product team to know.

Spatial Features

Spatial Features encode a location in space like GPS coordinates, addresses, and zip codes. We can derive features like the distance between a user and one of our stores or detect travel speeds that are highly unlikely to find anomalies and fraud. Spatial data can be beneficial but's most helpful to data modelers when enriched with external geographic data like statistics about population and weather or traffic data. There is relatively less cleaning necessary for spatial features as what you should be most interested in is finding outliers, nonsensical values, and impossibilities. Having a look at the most geo-tagged place on earth, Null Island, should give you an intuition for what to look

for errors in spatial data. Be aware that many bias problems can arise in spatial data, mostly in which areas have good coverage collecting spatial data and defects in measurement software like speedometers.

Text Features

Text data of all types of data you might encounter has the most potential for cleaning. Commons steps to prepare text data for machine learning are cleaning, tokenizing, removing words, reaching root words, and enriching our text data.

Text data needs to be pre-processed by cleaning and sometimes transforming the data. It's important to also analyze the length of text data by breaking text into n-grams like the ones you can see below. After this understanding, it's typically a good idea to understand the sentiment or if a phrase is positive or negative in meaning. We also tag certain pieces of text with pre-defined categories that help draw relationships between people, places, and things. For instance, when we have a text that includes "Tim Cook," we can ensure his name has a tag for a job title that equals CEO and a company tag that equals Apple. We can also extract the main topics from our text using topic modeling and transform our words into vectors of numbers so algorithms can process them,

6.2 Input Features

Sparse categorical features that have very few total observations can prove problematic for certain machine learning algorithms. If you have multiple classes with less than 50 observations in a class, we want to **combine sparse features**. We can group some

of these sparse classes into "other" categories or re-label class combinations into their umbrella term. For example, if we're doing with car data, we may have a lot of types, but we can group "short_sleeve", "long-sleeve", and "tank_tops" into the category "shirts".

Since most machine learning algorithms can't handle categorical features directly, we need to create dummy variables for categorical features. **Dummy variables** are binary (0 or 1) variables that each represent each class of a categorical feature. This representation allows us to make predictions on categorical data. As part of creating new input features, we should also discover which features we should exclude from our model. It's common you'll have a dataset with columns that don't make sense to expose our models to. This can be column descriptions and, importantly, features we don't have available when we're predicting. Historical data can often be manipulated before landing in your lap for modeling. Ensure each feature you feed into your model is something you'll actually have available when it comes to making new predictions.

After you've cleaned your data and completed Feature Engineering, you'll save your transformed dataset as your analytical base table. From here, you can save your table in whichever database manager you use and reuse values from this table to experiment with. This is a crucial step in Data Science many ignore, but you don't want to skip creating this table and documenting it well. Remember, not all of the features you engineer need to be extremely predictive. In fact, most of them won't actually improve your model. You can select models that can automatically select the best features for you, making this process much easier and faster. This way, you can more easily avoid overfitting.

Once your analytical base table is saved, you can move on to **Model Prototyping**. This is one of the most exciting aspects of data science as you'll work to see what kind of machine learning models fit your given task the best. During this stage, you'll finally find out how exactly you'll turn your data into value. You've done all this work, and prototyping will help you find out if it's really feasible to extract data from your insights. This step is about figuring out what will and won't work, not making the perfect model. We want to find these things out as quickly as possible, to we work to prototype "rapidly" or build and fail (or succeed) quickly. You can choose the tools you're most comfortable with, whether that's Jupyter Notebooks, SQL, or even in the command line with bash. When this is done, the last step of your project depends on your findings. Are you working on an ad-hoc analysis? If so, you might be spending time visualizing and presenting your findings. If this is a machine learning product, you'll need to further investigate how to implement it. Is it a model that can be run quickly on-board a wearable device, or does it need a lot of memory, like K-Nearest Neighbors?

6.2 Interaction Features

A common method in feature engineering is creating more complex features that are mathematical combinations of already existing features. The hope is that the more complex features better capture the relationships in the source data and prove to be more predictive than using source features alone. Many consider this part of the "art" of data science and machine learning. This requires a deep knowledge of the data and examinations of feature correlations. By having more complex

input features, it's easy to simplify our model and without sacrificing the quality of our predictions.

A simple pairwise interaction feature can be thought of as the logical AND. Let's say we're trying to predict a car's resale value, and we have a feature called "years_old" that has the car age. We might also have another feature, "luxury" that's a binary feature that places more resale value on certain manufacturers. We might suspect that even if a car is a bit older, it may lose value slightly slower if it's a luxury car. In this case, we can try to capture that interaction by creating a new feature called "luxury_depreciation" which equals years_old x 0.5 if the example is a luxury car (0.5 is just a made up constant we can set to denote that a luxury car depreciates half as fast as other cars). We can also do this by creating new polynomial features with sklearn. A best practice is to leverage the Vowpal Wabbit sparse format to delineate features in your namespace.

6.3 Dimensionality Reduction

Dimensionality reduction is the process of transforming data that's in high-dimensional space to one that's in a low dimensional space. When we represent data in a low-dimensional space, it's far easier to visualize while retaining some of the meaningful properties of the original data. Our training sets can be hundreds of columns, and by applying dimensionality reduction, we can reduce the number of columns so we can create a 2D or 3D rendering of the data. The curse of dimensionality is a phenomenon that arises when you are working with high-dimensional data. The higher the number of variables, the more difficult it is to visualize your dataset. Many times when

we have a lot of variables, they're highly correlated, so while it may initially seem better to have as much data as possible, we can run into the curse of dimensionality. Dimensionality reduction usually helps us improve model; performance as well as eliminating noise. We can use it to compress our data, which can also reduce computing time. While used often in feature engineering, many dimensionality reduction algorithms are unsupervised learning algorithms that can be used on a stand-alone basis.

Principal Component Analysis is one of the leading linear techniques of dimensionality reduction. This method maps the data to a lesser dimensional space so. PCA converts the data into a new set of data known as principal components from our original dataset. This step makes sure that the first principal component creates the largest variance. This method requires that data is normalized as it's sensitive to scale, but can help us extract new, lower-dimension features from our data.

Other Dimensionality Reduction Techniques:

- Non-negative matrix factorization (NMF)
- Linear discriminant analysis (LDA)
- Generalized discriminant analysis (GDA)
- Missing Values Ratio
- High/Low Variance Filter
- Backward Feature Elimination

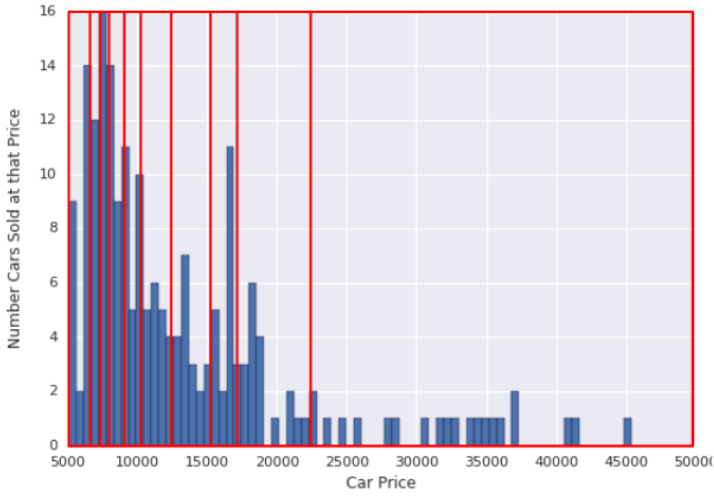
6.4 Data Pre-Processing

This section will briefly cover different pre-processing methods that you can leverage to transform your data into an easier format for machine learning modeling.

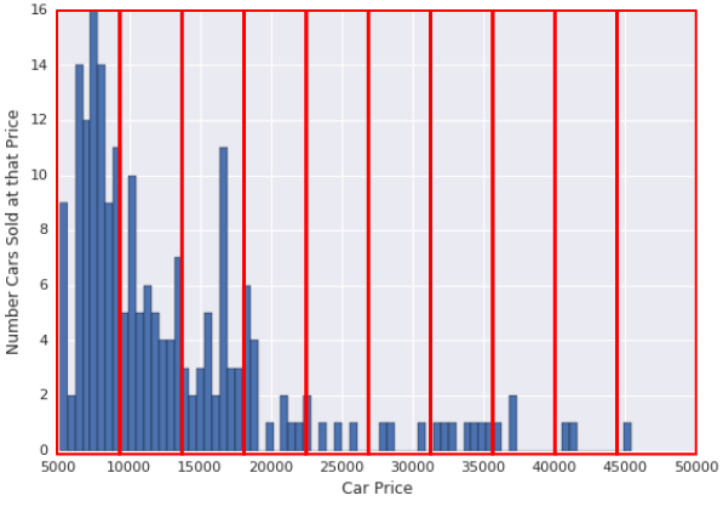
Discretizing: Discretizing our features means we transform continuous variables into categorical variables. For example, if we have a database with our web page and individual page views, it would be simpler for our model to understand this if we use 1 to denote if a page was viewed and 0 if it wasn't viewed. This can be done with one line of code from the Python sklearn package!

Rounding: Another transform we can apply is rounding; this is a form of lossy compression, where we maintain the most significant features of the data. Sometimes too much precision or detail is just noise. If we have a dataset where tiny differences in metrics aren't relevant to predictions, we can round variables and treat them as categorical variables. Some models, like Association Rules, only work with categorical features, so we can convert numerical data like percentages into categorical features with rounding.

Binning: Quantile binning allows us to divide data into portions like median, quartiles, and deciles. In quantile binning, each bucket has an equal number of data points. If we have equally spaced bins, then the bins have the same range. We are likely to see drastically more points in some bins than others.



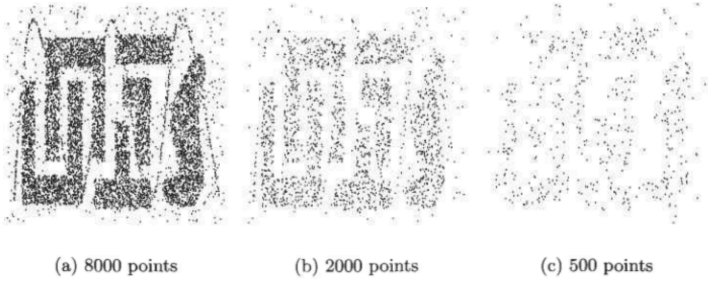
Buckets with quantile boundaries



Buckets with equally spaced boundaries

Aggregation - Combining objects into a single object. We can leverage aggregation when data is somewhat sparse or care more about feature statistics than their raw values. This can be counts, sums, or averages. This results in slight data reduction, but it's easier to process fewer rows of statistical values.

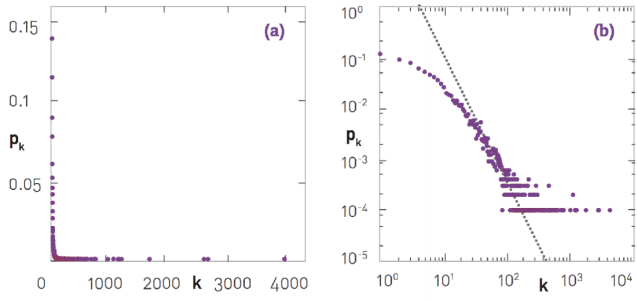
Exclusion Sampling - This is just a way to reduce the amount of complexity or training data by excluding some records. The biggest mistake practitioners make is creating samples that don't preserve the original data set's properties.



(a) 8000 points (b) 2000 points (c) 500 points

Exclusion Sampling

Variable transformation - Apply a function to all values variable transformation is common in skewed data, such as the Log Transform. If you have normally distributed data, standardization is a method to ensure your mean is 0 and a standard deviation is 1



Model Selection

Wrapper methods - Wrapper methods are a way to help us optimize the best subset of features to use for training. An example of this is stepwise regression, and while they can work well, they're computationally expensive.

Embedded methods - Feature selection as part of the model training process is only possible with some modeling methods. We're able to use this method to find the best features simply by setting model hyperparameters.

> *"More data beats clever algorithms; but better data beats more data"*
>
> *-Peter Norvig*

Unfortunately, machine learning is subjective in what a "good" model is considering our data and content. We can get drastically different model results as we chance our feature and modeling architectures. Since this is the case, we have to focus on choosing the best methods in our toolbox to meet our goals.

7

Machine Learning Fundamentals

Now that we're at the core of what most people would consider to be "Data Science" I want to reinforce that machine learning is not simply about the algorithms outlined in this chapter, but matching learning is the comprehensive approach to solving s problem. We leverage large amounts of data and statistical methods to solve problems. Algorithms are tools created by people, and they're only one piece of the puzzle; the rest is applying them in contexts that make sense, with the right purpose and using the ideal architecture.

In Machine Learning, we train a machine to make predictions based on historical examples. What machine learning really is pattern recognition. What makes ML novel is the speed at which we can develop and test new models on data and easily scale decision-making instantly.

For companies, what this means is if they had a manual system of processing applications for a college, healthcare program, or deciding which transactions are fraudulent. We can use the historical data on all the past decisions to make future decisions.

This gets tricky because we are often unsure that human-made decisions in the past were the best decision. What criteria did they use to make that decision? Can we quantify which decisions were right and train our models only on those? For most teams, this aspect isn't truly considered, and groups move forward with modeling efforts regardless of how irrelevant past decisions may be in predicting future events.

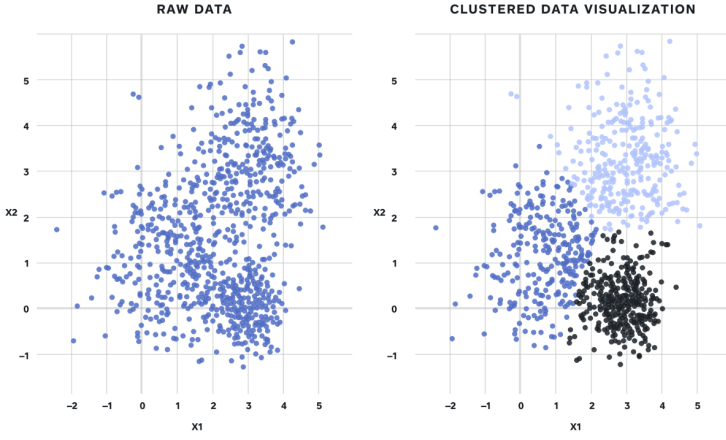
We've employed so many novel algorithms in high-risk fields like housing and healthcare that we've inadvertently given machine learning models the power to decide life and death for large populations of people. We just had humans making those decisions in the past, and many of our attempts to fix human bias by automating decisions have backfired. In fact, we've made this bias drastically larger In this chapter, I'll set the groundwork for how you can approach building machine learning algorithms with an ethical mindset. We'll discover both how these algorithms work, as well as how we can evaluate them taking into consideration the impact these models will have when used in production or "in the real world."

7.1 Flavors of Machine Learning

Machine learning can be thought of as having four major subgroups, supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. We derive these flavors by the type of data used to train the models. In supervised learning, the model is given a set of labeled input data. This allows the model to learn from the expected outputs and creates new predictions based on the learned relationships. Having a large amount of labeled data is critical, as subtle differences in patterns are more easily spotted when we have

many examples. This is one of the biggest problems in machine learning; in many cases, we just don't have enough data to make good predictions. In this case, you can work with others at your organization to gather more data or frame your problem differently.

When models perform unsupervised learning, they attempt to make predictions on data that doesn't have a definitive class or response variable values. What unsupervised learning attempts to do is understand the relationships within a dataset based on the distance of data points between each other. When plot on a 2D map, it's simpler to imagine how unsupervised learning is used to create logical groupings or clusters of data. We can use unsupervised learning for anomaly detection or identifying strange cases in fraud systems, for example.



Data Clusters

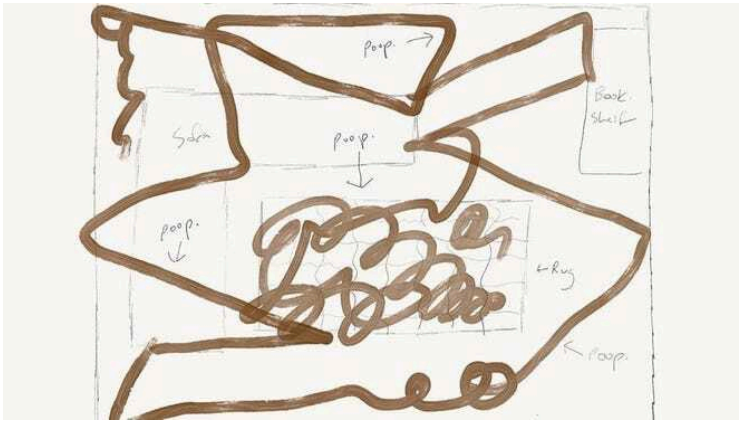
Semi-supervised learning is an approach to machine learning that combines a small amount of labeled data with a large

amount of unlabeled data. Semi-supervised learning has proved to be incredibly powerful as unsupervised learning methods are leveraged after labeled training data was fed into the model. This is one method to reduce the costs of human annotation of vast amounts of data so experts can be used to label a fraction of data necessary compared to supervised learning methods. As we'll detail later in this chapter, Semi-supervised learning relies on some assumptions to build accurate representations of data.

There are a few assumptions important in semi-supervised learning. The first is continuity; this means that points that are close to each other are more likely to share a class. This is assumed for a lot of algorithms in supervised learning as well as those that measure distances between data points like K nearest neighbors. For semi-supervised learning to work, the data should tend to form clusters of points, and the points in the center of the cluster are almost always the same class. Data should also lie on a manifold lower dimension than the input space. We can think of a manifold as space; we can plot math formulas like the Euclidean space. This assumption is helpful when working with sensor data or other high-dimensional data that is hard to model directly. For instance, the human voice is controlled by a few vocal folds, and a few muscles control images of various facial expressions.

Lastly, reinforcement learning leverages the use of digital "agents" taking actions based on the reward of their previous actions. Most people think of this kind of learning as "AI" the agents learn from the feedback they get from their environment, which can be positive or negative. The biggest drawback of reinforcement learning is that agents almost always act in ways that maximize the positive reward regardless of the other consequences. In one internet-famous example, the smart

vacuum machine has been criticized for spreading certain kinds of dirt in an attempt to clean all of the dirty particles, resulting in the below "pooptastrophe."



Jesse Newton's drawing of what his house looked like after his Roomba ran over dog poop

What are we trying to solve?

You will likely be tasked with using Machine Learning to solve some kind of business problem. This may be predicting product metrics for an app company or deciding which patients need the most urgent care. The very first thing you want to ask is what you're really trying to measure. This is one of the hardest answers to find truly because often, those who suggest ML projects are unaware of the solution they really want. In addition, it's rare that the data you have available and the problem you're attempting to solve align perfectly. This is why it's important to keep in mind that models are just guesses based

on statistics. Once we have an idea of what kind of outcomes we're looking for, you can decide on what kind of deliverable or outputs of a project are relevant. For Data Scientists, project deliverables can range from data visualizations to stored model predictions.

It's good practice to start thinking about which deliverables you'll pass off to project stakeholders when you're done. After understanding what you're attempting to solve, you want to classify your main project output in one of the following categories.

1. Analysis - Sometimes, you don't need to create a model with the purpose of creating new predictions. Often when we're asked to find "something interesting" in data, our colleagues want an analysis of some kind.
2. Predictive Modeling- In these cases, you want to either speed up a decision typically made by humans or create "value" by automating decisions. This can add value to customers, speed up internal processes, or discover new ways to solve problems.
3. Visualizations - Images that represent statistics from your EDA or modeling. This can be charts, dashboards, presentations, or interactive products.

Should you use Machine Learning?

When receiving a new ML task, not only do you need to ask what you're trying to solve and what you can deliver, but you also need to question of truly using machine learning is the right solution. Unfortunately, I can't give you any hard and fast rules that will always apply, but I can give you a method to guide the decision

to work on a project or not. We must demand and push back against the notions of scaling solutions as quickly as possible. This is one of the most damaging behaviors and incentivized habits that have led to many of the past AI incidents, like people being falsely accused and arrested because of faulty algorithms.

What are our business metrics?

If you're working in industry it's important to always think about your work scientifically, but you'll also be tasked with solving problems that reduce cost or increase revenue. You should know if you're trying to reduce costs or improve site loading speed. Figure out how you can measure the impact of the solution. Take the step that few people in industry do by thinking about the end before the beginning.

What are our incentives?

Are you working on solutions at the suggestion of investors? Are you being pushed to classify people for marketing efforts? Discovering the reasons why certain projects arise is crucial to understanding the underlying goals. If we're working on projects with the purpose of surviving people because we're incentivized to, we will, no matter our best-laid plans.

Are we in a high-risk industry?

Depending on the kind of company your working for, your code can literally be life or death. There have been AI incidents in high-stakes industries like finance, healthcare, and recruiting that it's easy to spot the potential harm. Either someone doesn't get approved for a loan or delayed treatment when they're sick.

Are we trying to solve a social problem?

This is one of the biggest issues in framing data science project; it's complicated to solve social problems well with technological solutions. It's more likely the case where the solutions work for a few, but not for all. We have judges who are tasked with deciding someone's jail sentence using tools they can't interpret to predict what sentence someone should get. We should first work with social scientists and other domain experts to better understand the root causes of the symptoms we see before attempting to solve it with automated tools.

Are there inequities we can change?

Depending on your industry, if you have the power to create radically new algorithms with the purpose of addressing the inequities that are existing, you're in a unique position. I firmly believe addressing some existing problems in our various fields is one of the best ways to leverage machine learning.

What kind of data is available?

This question is crucial to analyzing data and building data models. Discover what kind of data you have available and its potential flaws. One of the best ways to determine what kinds of analysis and potentially modeling methods will work is to have a good understanding of what data is on hand.

It's important to fill out a Datasheet for **EACH** dataset you use in the modeling process. Getting used to doing this step now will save you a lot of headaches and pressure later. By truly understanding the source of your data, the reasons it was collected, by whom, and for what, you can start to analyze if the data is a good enough proxy for the problem you're aiming to solve. It's not uncommon to not have the data you need to complete an analysis or modeling project. In this case, you can

petition to get more data by collecting or buying it, pivot your modeling goals by redirecting to questions you can answer with your data is another method.

How will I evaluate this model?

Define Success: Knowing if you are meeting a certain level of success with your model is crucial to evaluate if it should be developed further. It's common that Data Scientists are asked to complete analysis and try our hand at creating a predictive model with new data, new methods, and in the time frame of a week or two. Work with your team, those close to the data, and domain experts to understand what your model should be able to do.

Set a Baseline: Are you just trying to beat random chance guesses with your model, or can you compare your model to humans' performance like radiologists or doctors? What is an acceptable failure rate given your industry and the potential damage your efforts can do? Set a threshold for the maximum amount of errors you're okay with a live model having.

Quantify Risk: What is the worst that can happen? How will this model be used in the real world, and have I seen what this real-world data look like? Can consumers see financial or physical damages in the case our model fails?

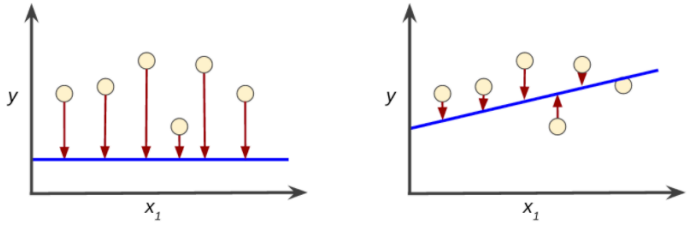
Measure Performance: What is the metric that matters the most for this project? Is my model accounting for how the future may not be exactly the same as the past? my accuracy

Measure Fairness: Do we have data on protected classes to measure this in the first place? Does my model predict the same across each class? Does individual or group fairness matter more? Are we able to track how our fairness metrics drift over time?

7.2 Types of Algorithms

Supervised Learning Methods

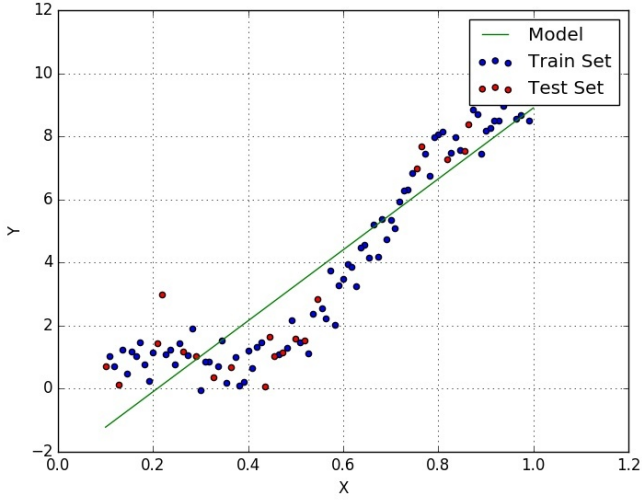
Each supervised learning algorithm is trying to do the same thing at its core: learn from training data and build a model that minimizes loss. Loss is the penalty for a bad prediction, so if the loss of a model is 0, then the model is perfect and predicts the right answer every time. The goal of training a machine learning model is to find a set of weights and biases that have, on average low loss.



As you can see, we have the same points of data, and the blue line represents our mode. The difference between how long our red lines or loss are between each model is pretty drastic. To encapsulate this concept in a mathematical term, we use a loss function called squared loss (L2 loss). All this does is subtracts how far our predictions were from the actual values then squares

that number. To average this up over an entire dataset, we use the **Mean square error** (**MSE**).

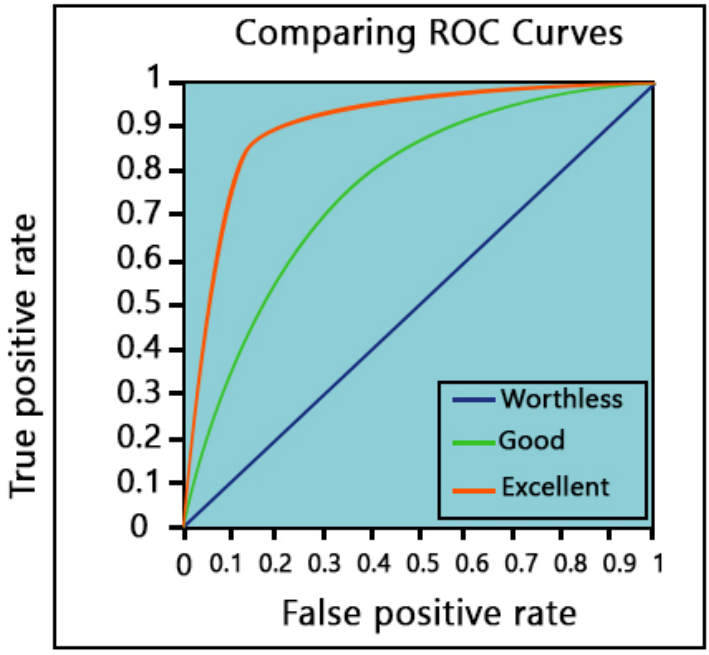
Regression is a type of supervised learning technique that predicts a number or continuous variable. This can be variables like gas prices, product downloads, or temperature. Regression is used when we want to estimate an exact value for something. We use regression as a way to fit a line to various data points. This line is important as if we can find the "best" position for it, we can reduce the error or distance between our straight line and the data. When we initially train a regression model, it can learn from only the training data and draw a line that best fits the training data. Then the model can compare its line to each example in the testing set. This tells us how good or bad our model is at crossing through each point as best as it can.



There are various different kinds of regression we can use, and the simplest is **Linear Regression**. As the name suggests, it can be used in cases where there is a continuous (numerical) target variable that has some linear relationship with at least one of the dependent variables. The formula for linear regression is y = a*x + b + e. In this formula, y is the target variable we are trying to predict; a is the intercept and b is the slope, x is our dependent variable used to make the prediction. Since we only have one dependent variable to help us present the target, this is an example of simple linear regression. The formula gets a little more complicated for multiple linear regression, but the main idea is that the formula is the same, just the a * x part gets repeated for each dependent variable we have. y = a1*x1 + a2*x2 + ... + a(n)*x(n) + b + e.

We use one method to determine if a linear regression model is "good" by assessing the square root of the mean of the sum of the difference between the actual and the predicted values. This is our Root Mean Square Error (RMSE).

For **Logistic Regression,** the target variable in question is categorial. It can be binary, multinomial, or even ordinal. For logistic regression, it's important which activation function we choose. The most common way to assess a Logistic regression model for a binary classification problem is by using the sigmoid activation functions, which returns probability values between 0 and 1. When we have more than two classes to predict, using the softmax function as a sigmoid for multinomial numbers can be extremely time-consuming. Common methods used to evaluate classification problems are ROC curves and accuracy. The more the area under the ROC, the better is the model.

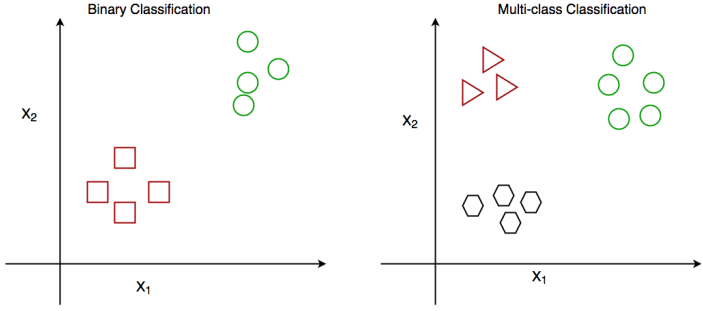


Kinds of Regression Models:

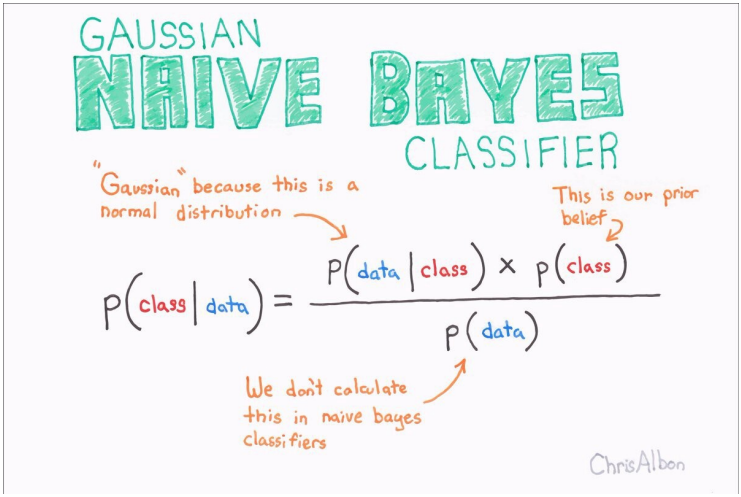
- Ordinary Least Squares Regression
- Stepwise Regression
- Multivariate Adaptive Regression Splines (MARS)
- Locally Estimated Scatterplot Smoothing (LOESS)

Classification is a method that attempts to groups new data into classes the model has been trained to recognize. Some popular examples of fo data classification are email spam detection and predicting whether an image is a class or dog. Classification often fails, it's when new data is exposed to a model, and the model has no reference class data to classify the new example.

Classification, as a type of supervised learning, often requires a lot of data in order to train models that can generalize to new data well.



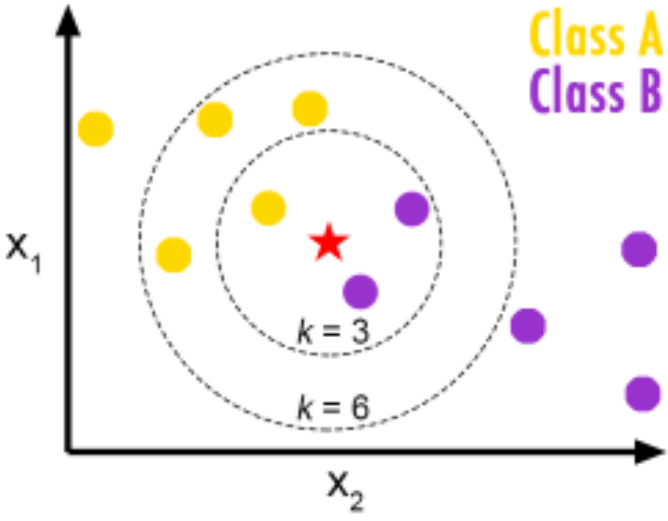
Naïve Bayes is a classification algorithm based on Bayes' theorem. These algorithms work on the same principle that none features are dependent on each other. Let us see how the learning model for it will look like. Naïve Bayes is naïve because it assumes one variable's presence has no relation to the other variables with is never really the case in real life. These models can be fast as they don't require a lot of fo storage, but it suffers when features are highly dependent. A good case for using Naïve Bayes models is spam detection and detecting other anomalies.



Other Bayesian Models:

- Averaged One-Dependence Estimators (ADDE)
- Bayesian Belief Networks (BBN)
- GaussianNaive Bayes
- Multinomial Naive Bayes
- BayessianNetwork (BN)

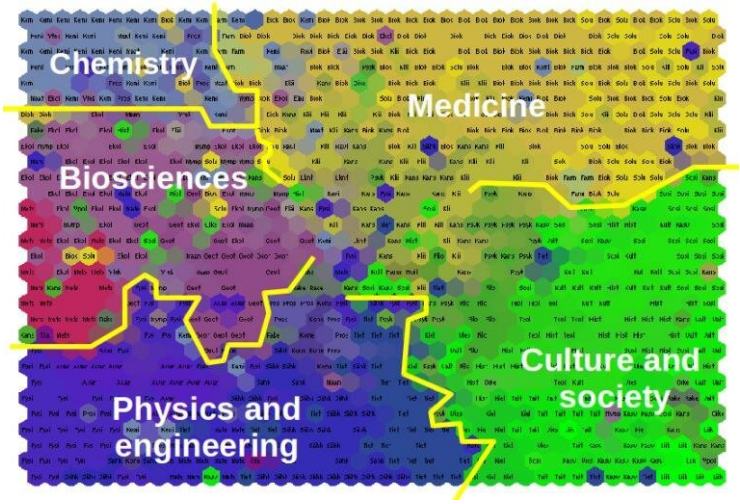
Instance-Based Learning is a family of algorithms that look at an entire dataset and compare new instances to training instances. It's also known as **memory-based learning** or **lazy-learning.** These models can be quite expensive. Unfortunately, with these models, our hypothesis complexity grows as the size of our data grows. One advantage of these methods is that they are able to adapt to previously unseen data meaning it's easy to update new instances of training data.



K-Nearest Neighbors

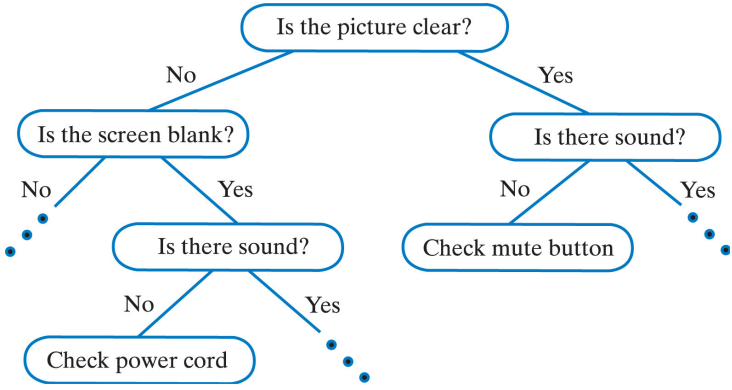
Instance-Based Models:

- K-Nearest Neighbors (KNN)
- Learning Vector Optimization (LVQ)
- Self-Organizing Maps
- Locally Weighted Learning (LWL)



Self-Organizing Maps

We can build **Decision Trees** for both classification and regression problems. This is the simplest algorithm to interpret because decision trees simply tell you why they made a decision. They are unaffected by outliers or missing values in the data and can capture non-linear relationships well. When we build tree models, all features are initially considered the first major predictor, but the feature with the maximum information gain is taken as the root node. The root node allows for the largest split of data points and is the first of many decision noes. One of the biggest challenges in creating Decision Trees is overfitting. Overfitting is the biggest practical challenge in supervised learning and occurs when the model memorizes the training data and has difficulty predicting well on the test data. For Decision Trees, there are two ways to avoid overfitting. First, we can set constraints on tree size, and second, we can prune our decision trees.

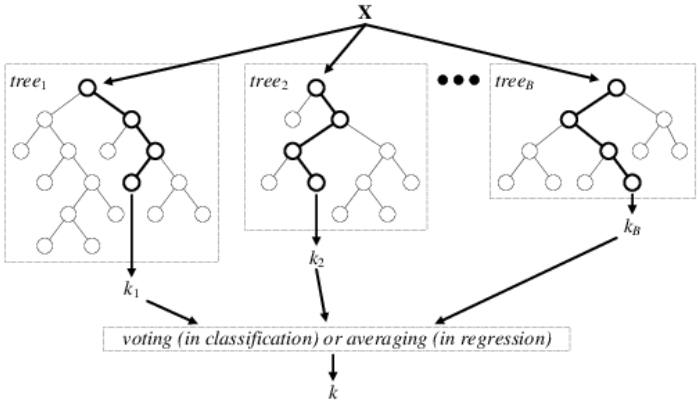


Other Tree Models:

- Classification and Regression Tree (CART)
- Iterative Dichotomiser 3 (ID3)
- C4.5
- Chi-Squared Automatic Interaction Detection (CHAID)
- Decision Stump
- Conditional Decision Trees

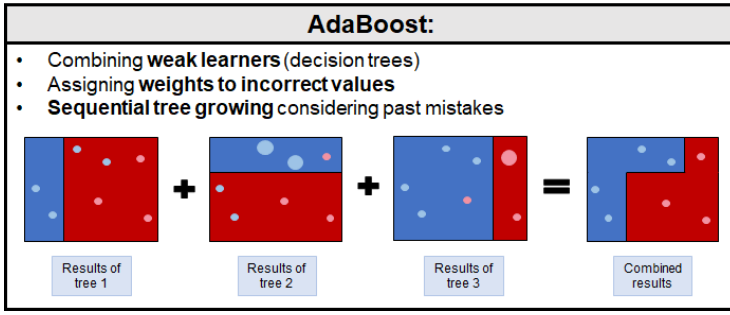
Ensemble methods like **Random Forests** are a way to reduce the amount of overfitting we see when building decision trees. Two types of randomnesses are built into the trees; first, we randomly sample data into multiple datasets. Second, the features are selected at random for each set. For each tree, not all features are used to predict the target. This helps us find relationships in the data without relying too heavily on a few features. This process is applied to build multiple trees. When assessing random forests, we should have a good idea of the feature's importance or the class's impurity reduction due to the feature. The partial dependency plot can visualize how a feature and class are correlated. We create these plots to see

the marginal effect of a feature on the class probability. To evaluate random forests, you can use correlation matrixes and classification reports (built into Python) to see your model's evaluation metrics easily.

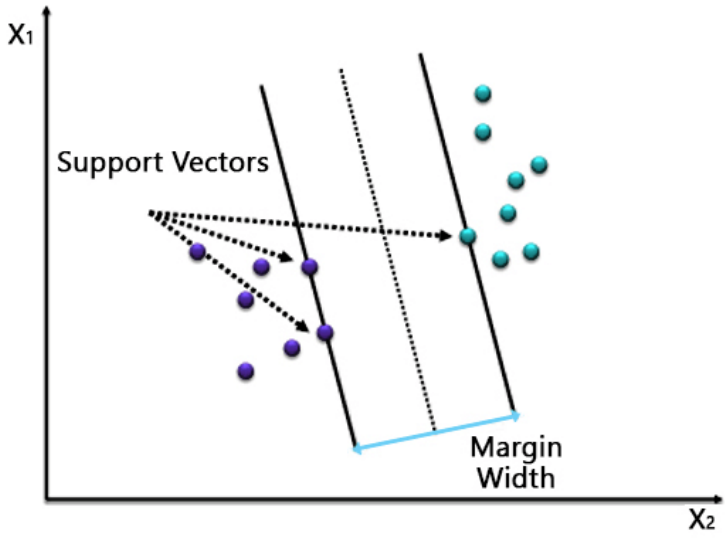


Other Ensemble Models:

- Gradient Boosting Machines (GBM)
- Bootstrapped Aggretagion (Bagging)
- AdaBoost
- Stacked Generalization (Blending)
- Gradient Boosted Regression Trees (GBRT)



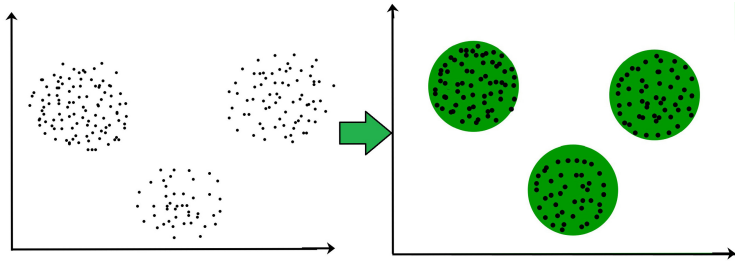
Support Vector Machines are a classification algorithm where a hyperplane separates the two classes in binary classification. Two vectors from each class are known as the support vectors. They guide where we draw the hyperplane to separate the classes. In most cases, the data isn't so easily separable in a 2D graph. We have to tune the parameters of support vector machines in order to distinguish between the classes accurately. We can set parameter values such as what type of regularization (L1 or L2), kernel (linear or polynomial), and gamma to use. Gamma defines the impact of training examples. Points close to the line are considered in high gamma, and those far from the support vectors have low gamma.



Unsupervised Learning Methods

Clustering is the most popular type of unsupervised learning, and as you saw in FIGURE SOMETHING, we can group data easily by clustering them together. Clustering works under the assumption that "similar birds flock together" or that they will be nearby each other distance-wise if objects truly are similar. One type of unsupervised machine learning we covered in the last chapter is dimensionality reduction. While we can use this to pre-process data, we can also use dimensionality reduction algorithms like Principal Component Analysis and Partial Least Squares Regression on their own. Unsupervised methods are simply those where there is no labeled data. Instead of trying to predict a specific target variable, these models are used more to

group data points that we think are similar.



Finding Data Clusters

K-Means a clustering method that finds similar groups in unlabeled data. The idea of clustering is to find groups that are "far" apart from each other, while the distance between points in a single cluster should be at a minimum. K is a variable that can be any integer. It refers to the number of clusters needed to maintaining maximum variance in the dataset. K-Means has evolved from signal processing, and it aims to partition observations based on how close each example is to the nearest cluster mean.

- K-medians
- Expectation Maximization
- Hierarchical Clustering

7.3 Model Evaluation

Methods for evaluating a model's performance are divided into 2 categories holdout or cross-validation. With each technique, we leverage a training set of data that the model hasn't yet

been exposed to in order to evaluate performance. When we use holdout, we divide our data into three random subsets: training, validation, and test sets. Our validation set helps us test the performance of the model in the training phase; we can fine-tune a model's parameters before testing. (Not every algorithm needs to use a validation set). Using holdout is fast as we can sort data in these subsets quickly; however, this approach suffers from high variability. Cross-validation is a method of partitioning the original dataset into a training set and testing set, most commonly with the k-fold method. This partitions the original data into k equal subsets that we call folds. We do this k times, and in each iteration, we use one of the k subsets as the test set, and the other subsets create the training set. The error is then averaged over all our trials to get the effectiveness of our model.

Classification Metrics

There are various different metrics you can use to get even more insight into model performance.

Accuracy is commonly mentioned, but it rarely gives us an idea of the full picture. We can start by understanding that accuracy is the total number of correct predictions divided by the total number of predictions.

$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

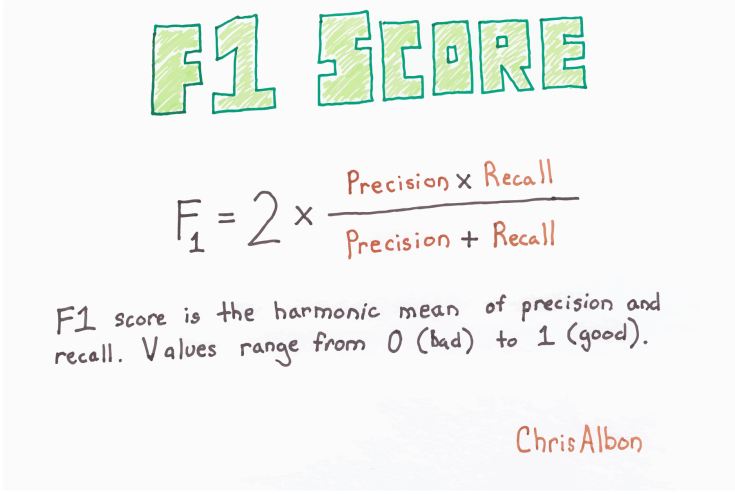
$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$\textbf{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}}$$

Precision is the number of correctly-identified objects of a class divided by all the times the model predicted that class.

Recall is the number of objects of a class that we've identified correctly divided by the total number of members in that class.

F1 score is handy because it combines the best parts of both precision and recall in a single metric. If precision and recall are both high, F1 will be high too! And vice versa. The F1 score gets cool if one is high and the other low, the total F1 score will be low. F1 is a quick way to tell whether the classifier is actually good at identifying members of a class.



Log Loss is the performance of a classification model where the prediction input is a probability value. We see these kinds of inputs somewhat less often, but when assessing these models, it's important to know that log loss increases as the predicted probability gets further from the actual value. This means that we want a small log loss, and a model that predicts correctly every time would have a log loss of 0.

Regression Metrics

In addition to these metrics for classification, some of the most common metrics for evaluating regression problems are Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). The first is the sum of the absolute differences between predictions and actual values. In contrast, RMSE measures the error's average magnitude by taking the square root of the average of squared differences between prediction and actual observation.

Sometimes we go through all of this to create a model that fails. I want you to resist the urge to abandon the project when your initial models aren't performing well. Instead of giving up on the project (unless it's a bad use case), you can try making changes to your features or model architecture and experiment with other types of models. At the very least, you should be able to have insight into why a model didn't work as intended. Sometimes this is about bad data, and sometimes it can be a bad match between our data and the model we selected. Every time you build a model that "fails," seek to understand the potential reasons why. This will allow you to speak to what you'd change, looking back.

No.	'WHAT TO TRY NEXT?'	Results	Fixes
1.	Try Smaller Set of Features	Decreases Model Complexity	*High Variance*
2.	Add New Features	Increases Model Complexity	*High Bias*
3.	Add Polynomial Features	Increases Model Complexity	*High Bias*
4.	Decrease Regularization Parameter (λ)	Decreases Penalty	*High Bias*
5.	Increase Regularization Parameter (λ)	Increases Penalty	*High Variance*
6.	Get More Training Examples	Increases Sample Size	*High Variance*

In order to have effective machine learning systems, we have to have a skilled human guide. You! No matter how awesome you are, successful projects also require relevant data, the selection of the right algorithms, and tuning them correctly on unseen testing data. Making predictions on future data is often the main problem we want to solve, and it's important to understand the context before choosing a metric.

8

Bias, Fairnes, and Accountability

One of the most challenging problems to deal with in Data Science is that of algorithmic bias. This chapter will outline both our technical and societal meanings of bias and ways we can address them. I'll outline some initial definitions and then highlight some interpretable models that often come at the trade-off of explainability but are always a good starting point for modeling.

White supremacy is a global phenomenon that persists in the technology we create. There are aspects of how this impacts our tech because they impact nearly every cultural group. Nearly every culture globally exhibits colorism, which is the idea the lighter-skinned people are better than and given more opportunities than those with dark skin. This is an unfortunate effect of white supremacy in that the closer a relationship to whiteness, the more superior they are. White supremacy, as well as other bigoted ideas of homophobia, sexism, and ableism, are exacerbated by products that use machine learning. The root of the problem lies within individuals; however, biased individuals

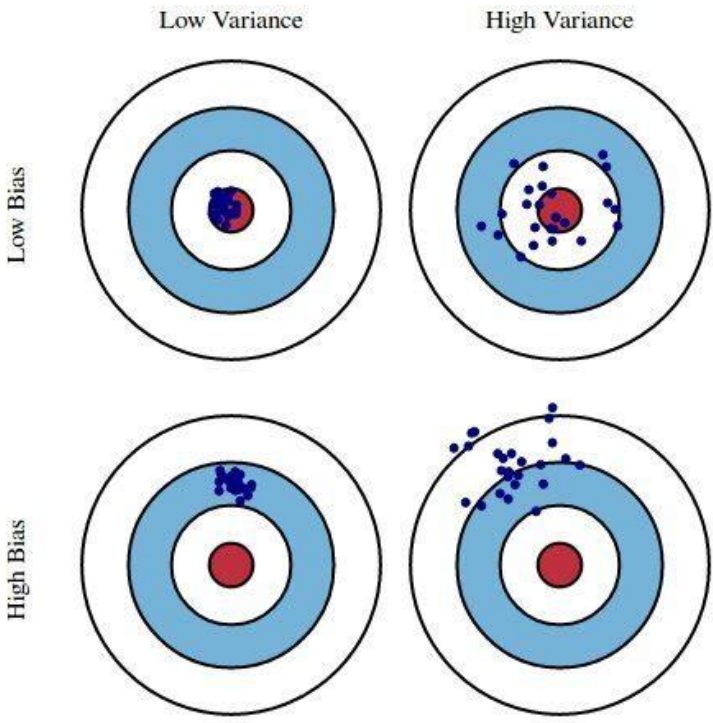
in the past have made decisions we then train models on. Then, in addition, white supremacy has impacted who is admitted to these top-tier computer science programs and the culture of Silicon Valley that promotes hiring almost exclusively from top-tier schools.

> *Every data set involving people implies subjects and objects, those who collect and those who make up the collection. It is imperative to remember that on both sides, we have human beings.*
>
> *-Mimi Onuoha, Data & Society.*

If you take away only one thing from this chapter, and ultimately this book, treat data in the ways in which you'd treat real people. You have to remember, in many cases, each row is a human life. There are families, stories, trauma, and joy behind what we see condensed into an ID and a few columns of data. While in many cases, bias is perpetuated by machine learning unintentionally, there are cases where insidious uses of science sell for millions and billions of fo dollars. Unfortunately, there may be people who benefit from selling this expensive, biased technology. In nearly all cases, they're part of the dominant culture and arent the targets of surveillance and other harmful tech.

8.1 Societal Bias

Bias has multiple meanings depending on the context. When you typically hear about bias in Data Science, it's often to describe the difference between our model's predicted values and actual values. The Bias-Variance tradeoff is a common concept when discussing the outcomes of a machine learning model.



In society, bias is a tendency, inclination, or prejudice toward or against someone or groups of people. Biases are often based on stereotypes rather than actual knowledge of an individual or circumstance. Whether positive or negative, such cognitive shortcuts can result in prejudgments that lead to rash decisions or discriminatory practices.

Biases like gender, race, and ethnic bias are encoded in algorithms through skewed data, the bias in researcher feature selection, and the use cases in which data modeling is used. As technologists, we should assume every model we create will encode racist, sexist, and biased norms unless you address them

specifically.

Bias also exists in our data partly because people are biased, and many surveys are biased to groups who are more likely to respond and those who have easily accessible contact information. Another reason bias is so prevalent is because there are ways many scientists and researchers are unaware that these biases exist. For some, it's a new concept that policing data is biased by anti-Blackness.

Bias isn't the only aspect for us to consider, but we must also define and consider fairness. If we're looking at large-scale data practices like how the Baltimore police used social media images of protestors to match them to social media profiles, target them for other crimes. Machine learning aside, this isn't a fair process. Facing issues of bias in ML doesn't just mean making the processes we use to create them fairly, but we need to address if the systems these models operate in are fair.

The process of applying for housing in the United States is biased based on credit score and many times zip code as a proxy for race. As I've covered in various talks, even this small bias can play a huge role in who gets a credit card or loan. Financial institutes are also historically extremely biased against women and racial minorities. When proxies for a race like zip code or gender like weight are used to create algorithms, our complex pattern recognition machines also tend to find the underlying patterns in human behavior as well. These are often the structural norms that impact the data at hand. An important aspect to consider when doing any work with data on real people is that the data you're analyzing represents real people. This can be seemingly harmless events like app data or website clicks, but often we're privy to medical diagnoses, criminal records, and other personally identifiable information. While we might

have a large amount of data, we must understand this data represents only a tiny sliver of all the information about a person. Medical diagnosis data doesn't include aspects like a person's neighborhood, access to healthcare, income, support system, and many of the other features that impact a patient's outcomes.

How does bias live in our Data?

It's a drastic reduction to excuse bias in our data and models because of the idea that bias exists everywhere. While in a sense this is true, not all of these biases are encoded into our tech and propagated in life-changing ways. Facial recognition tech is some of the most pervasive, biased, and harmful technology that's currently used to police and surveil populations of people.

Bias also persists because it's human to be biased. This means that when we use historical data about people and decisions made by people, we have to consider the cultural structures in which those decisions were made. Housing data is incredibly biased in the United States as historical practices of redlining and denying access to equal housing opportunities to different races has been common and impacts all housing data to some extent. There is no way to de-bias or separated the bias from our data. The bias is baked in, and while awareness of this bias is the first step, we must examine if we continue to use this data at all to make predictions about people currently. Would we subject them to historical standards of decision-making? As it's been well-documented, women of color and Black women specifically make less than white men. Based on these standards, how can we create fair outcomes if we compare black women and white men on the same scale, taking into account that white men on average earn 1.6x what Black women do for the same work? We

need to push our companies to consider this more often when we are structuring our projects.

Computer Vision Sees Color

We must address the fallacy that the problems with biased algorithms are only biased because of the data. There are many use cases that outright continue to demonize marginalized communities. Facial recognition is used by law enforcement as surveillance has biased applications in addition to the data facial recognition uses. The way companies collect this data has been by scraping the internet for images, but many times these large-scale datasets don't ask for consent before someone's image is used to train facial recognition models. I find this aspect of being relatable for everyone, as I wouldn't want images, I posted on social media from 15-20 years ago to be used to create these algorithms.

As highlighted by Gender Shades paper, there is a high disparity of error rates between light-skinned people and dark-skinned people as well as men and women in facial recognition in APIs created by Microsoft, IBM, and Face++. These are API products that other companies can pay for and use in their products. Authors Joy Boulamwini and Timnit Gebru found drastic differences in how accurately these commercial facial recognition systems worked based on skin type and gender. Not only was the largest difference in error rates between white men and dark-skinned women, but there are structural issues within organizations that enable white supremacy as well.

Data Annotation

Building neural networks and, as an extension, deep learning problems need millions of rows to train a model well. The way most big companies using ML do that by using pre-trained models or buying labeled data from sources like Appen or Mechanical Turk. One of the worries around this is how these labels are created and how an individual human, data labeler can impose their regional, cultural, and racial biases as they label data used in far-reaching ML models.

This is a major problem as most common ML algorithms are used for supervised learning, where we have labeled examples. It's incredibly hard to train models like neural networks without a lot of this data. By "a lot," I usually mean more than 5,000 or 10,000 examples. While it's ideal to have more like 50,000 training examples, getting this labeled data is both tedious and expensive. anywhere between 5,000 and

Researcher Bias

One of the major hurdles we have to overcome in ML/AI is the god-complex researchers can have when it comes to their work. This often leads to the Dunning-Kruger effect, where they may hold a lot of specialized knowledge but overestimate their knowledge in other fields, such as how tech intersects with society. Machine Learning and AI are part of technology as a whole but are one of the most academically-focused and elite subsets of tech. This is because most of the advancements in ML are closely tied to publishing graduate-level publications. There is a lot of bias in AI communities that stem from mostly homogenous research groups. As demonstrated in this wonderful paper, the

team creating these solutions need to be composed of people likely to be harmed most by these technologies. Researcher bias also leads many to believe what they see in a dataset if it confirms their beliefs. Many of the changes to weed out bias that seeps in because of researchers come from mindset shifts. Many white male researchers must confront their privilege and assess how their biases impact how they see others from a data and colleague perspective.

8.2 Statistical Bias

Let's gain an understanding of bias in a statistical context.

Selection Bias: Selection bias is widespread and happens when researchers select our sample dataset incorrectly. This means working accidentally with a specific subset of your population that isn't representative of the entire population. This is incredibly common when we're working on data that's easy to access. A popular example of this is survey results. If you send out a survey to newsletter subscribers asking them what products they'd pay for, you'll only get a subset of your audience population that's a member of your email list. Let's say you have 20K followers, and less than 50 of them are subscribed to your newsletter. You can easily access their emails (because they gave them to you), so you send them your survey without taking into account all of your company's audience is in your population.

Self-selection Bias: this is a subcategory of selection bias that lets subjects of your population select themselves. This means people who tend to be less proactive will be less represented in our data. This behavior can correlate with other specific

behaviors, so any analysis or modeling based on this type of biased input data will not be representative of the entire population. A real-life example of this is comparing users who read support articles to those who don't. You may find that users that read support articles are more active in the product, but we don't know this for sure. Just because this disparity exists doesn't mean our support guides are effective or have a higher commitment to your product simply because they "selected themselves" into the reader group. When we create new project architectures, we have to identify and many times limit the number of training examples where users have self-selected into testing groups.

Recall Bias: In interviews and surveys, recall bias is another common error. This happens simply when the respondent doesn't remember things correctly. Our minds are amazing, but honestly, overall human memory is subpar. By default, we have selective memory, so this is less about attempting to interview respondents who have a good memory. As time goes on, we tend to recall less and less detail from events. For example, many companies that those live events notice a stark difference in responses when they look at event surveys sent one day after and one week after an event. When we had more in-person events, a day after, people are more likely to remember if there wasn't enough cream cheese for the bagels or if the audio was too low. A week after, people are more likely to just remember and comment on the positive aspects of what they remember.

Survivorship Bias: This happens when researchers focus on the part of a dataset that went through some pre-selection process. This means we easily miss data pooi9njts that "fall off" during

this process. A common example of this is

Omitted Variable Bias: This occurs when we leave out one or more important variables from our data model. This is most common during predicting business metrics like user churn. It's rare that we're able to capture events like competitors in our data, so when we make predictions that most users will no churn (or cancel their subscription), our model is omitting variables like strong competitors and outside events like a global pandemic.

Cause-effect Bias: Humans are great at spotting patterns, but this nature can hurt us by assuming causation. We're wired to see it everywhere correlation shows. up. This is less of a "classical" statistical vias, but many data scientists and machine learning engineers are unaware of it. Cause-effect bias leads us to think the impact of a program is vital when we don't know for sure. The only way to truly understand if there is causation is to run experiments. You can easily run an A/B test for user data, but it's difficult to parse this out when dealing with historical human data.

Funding Bias: This is sometimes referred to as sponsorship bias, but it happens when a scientific study results from biases that support the research's financial sponsor. Recent works in data science have criticized the brain drain caused by silicon valley poaching top tenure and tenure-track professors. To attempt to "fix" this problem, many large tech companies have sponsored endowments for professors at universities. There are great articles like "You cannot serve two masters" that critiques how one can expect to get unbiased results from

researchers sponsored by corporations with their own goals and incentives. It;'s important to consider that individuals working and companies do this as well. If you're working for a company as a Data Analyst or Scientist, you're essentially being funded by the company for your work. While you may feel you can reasonably challenge

Observer Bias: This happens when researchers subconsciously project their expectations into their work. This is very common in the industry due to the incentive structures that lead many of these projects. This can involve cherry-picking statistics that support our hypothesis, influencing participants during surveys, or asking unbiased questions.

Cognitive Bias:

1. **Hindsight bias:** Looking back in time, we tend to be more critical of our work even when we discover pretty big findings. This bias happens when things that were unclear at the start of your analysis become known later.
2. **Confirmation bias:** Confirmation bias happens when stakeholders have strong pre-conceptions and only interpret the parts of the data that confirm their beliefs.
3. **Belief bias:** This happens when someone is so incredibly sure about their feeling they may ignore the results of an analysis altogether.
4. **Curse of knowledge:** This one is both tricky and common in industry. This assumes someone has the same background knowledge that you do. It can lead to skipping important steps when presenting non-data-savvy people.

8.3 Fairness

There are 21 different definitions of fairness; not only is it challenging to define broadly, but people across various roles in Data Science have a unique perspective and can have different definitions for words like values and fairness. There are three main ways fairness methods can be applied to Machine Learning, through data preprocessing, when training machine learning models, or post-processing results.

> *"Big Data Processes codify the past. They do not invent the future. Doing that requires moral imagination, and that's something only humans can provide. We have to explicitly embed better values into our algorithms, creating Big Data models that follow our ethical lead. Sometimes that will mean putting fairness ahead of profit."*
>
> -Cathy O'Neil

Group vs. Individual Fairness

To answer the question of group fairness, we ask if outcomes systematically differ between demographic groups. Group fairness requires certain constraints to be satisfied at the population level. Group fairness is about statistical parity, which means there is an equal chance people in a protected group get a benefit, like a scholarship or admittance into a program, as the group of people not in that protected class. Fairness can also be at the individual level where consistency is most important even though this can lead to equally bad outcomes and group fairness, which focuses on statistical parity between members of a group and other groups. Individual fairness about consistency, similar

people experience similar outcomes; they may all be treated equally well or equally badly. In terms of solo sports like singles tennis, fairness would be if all players on Serena Williams' level experience similar outcomes like compensation and exposure.

In resource allocation, envy-freeness is a criterion of fair resource division. In 1958 George Gamow and Marvin Stern introduced the problem of fair cake cutting. If everyone feels like they've gotten a good share and aren't envious of the other's portions, then the decision or cake cutting was fair.

Data Ethics Checklist

Inspired by the article Of Oaths and Checklists by DJ Patil, Hilary Mason, and Mike Loukides, this checklist is helpful for each stage of the data modeling processes. It's also available as part of the DEON ethics checklist for Data Scientists hosted on Github.

1. Data Collection

- If there are human subjects, have they given informed consent, where subjects affirmatively opt-in and have a clear understanding of the data uses to which they consent?
- Have we considered sources of bias that could be introduced during data collection and survey design and taken steps to mitigate those?
- Have we considered ways to minimize exposure of personally identifiable information (PII), for example, through anonymization or not collecting information that isn't relevant for analysis?

2. Data Storage

- Do we have a plan to protect and secure data (e.g., encryption at rest and in transit, access controls on internal users and third parties, access logs, and up-to-date software)?
- Do we have a mechanism through which an individual can request their personal information be removed?
- Is there a schedule or plan to delete the data after it is no longer needed?

3. Analysis

- Have we sought to address blind spots in the analysis through engagement with relevant stakeholders (e.g., checking assumptions and discussing implications with affected communities and subject matter experts)?
- Have we examined the data for possible sources of bias and taken steps to mitigate or address these biases (e.g., stereotype perpetuation, confirmation bias, imbalanced classes, or omitted confounding variables)?
- Are our visualizations, summary statistics, and reports designed to honestly represent the underlying data?
- Have we ensured that data with PII are not used or displayed unless necessary for the analysis?
- Is the process of generating the analysis well documented and reproducible if we discover issues in the future?

4. Modeling

- Have we ensured that the model does not rely on variables or proxies for variables that are unfairly discriminatory?
- Have we tested model results for fairness with respect to different affected groups (e.g., tested for disparate error

rates)?
- Have we considered the effects of optimizing for our defined metrics and considered additional metrics?
- Can we explain in understandable terms a decision the model made in cases where a justification is needed?
- Have we communicated the shortcomings, limitations, and biases of the model to relevant stakeholders in ways that can be generally understood?

5. Deployment

- Have we discussed with our organization a plan for a response if the results harm users (e.g., how does the data science team evaluate these cases and update analysis and models to prevent future harm)?
- Is there a way to turn off or roll back the model in production if necessary?
- Do we test and monitor for concept drift to ensure the model remains fair over time?
- Have we taken steps to identify and prevent unintended uses and abuse of the model, and do we have a plan to monitor these once the model is deployed?

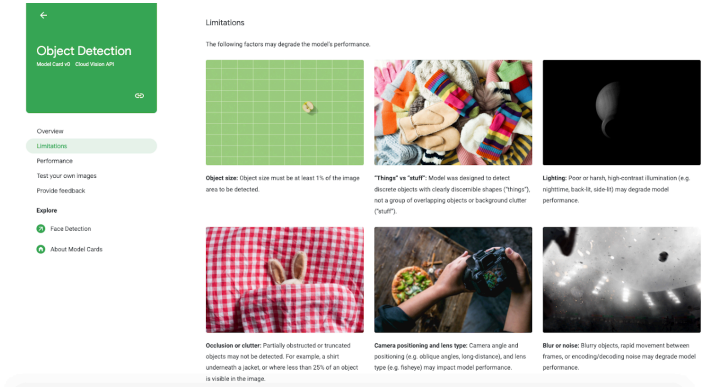
8.4 Practitioner Responsibility

You may be wondering what you can do to deal with this problem. The first step will always be documentation. Our datasets and models are far too underdocumented in the real world to allow smooth handoffs to new researchers. We have to outline in detail our data and the methods used to create machine learning models as well as the kinds of experiments we ran and why we

made certain modeling or data cleaning decisions (like imputing the average for missing values).

Documentation

One of the biggest barriers to having reproducible systems is the lack of good documentation. Data Scientists can take two methods right now to move towards responsible AI by implementing Datasheets for datasets and Model Cards.



Google's Object Detection Model Card

Model documentation should contain the who, what, when, where, and how for any personnel, hardware, data, or algorithms used in an ML system. We should be frank when communicating to stakeholders and users about the limitations of our models and cases that are correlated to biased outcomes. Many times, production machine learning systems don't have the capabilities to diagnose or respond effectively when something bad happens with a model, let alone reproduce the same results. ML systems must also be monitored. Typically monitoring is

used to watch the inputs or outputs of an ML system change over time, particularly for impact on model accuracy. However, it should be the default to monitor for drift in fairness or security characteristics.

Risk Identification

Those working closest to building new machine learning models are not the best suited to point out the potential risks. Many times, we as individuals have blind spots and are unable to clearly see the harm we can cause. By working with other teams, social scientists, and domain experts, we can identify how our models can harm us. Of course, there are unknown unknowns, but having risk mitigation plans in place allows us to try new models and either "kill" them when they're misbehaving or allows users to appeal decisions. Human augmentation is human review and assessment of risks. These scientists can help technical teams outline how to capture user feedback, set fairness standards, and architect new methods. We can assess risk, but we have to keep in mind the risk needs context. To evaluate the bias in the data, systems, and people creating data models requires us to understand, document, and monitor sociological discrimination.

Harm Mitigation

There are many ways humans can be harmed by the models we create, and one of those is job displacement. We hear a lot about this with very doom-and-gloom tones speaking of a weak outlook for what the future of work is for people. As data professionals, we have to weigh the chances of the models

we create replacing human workers. Beyond that, we should consider how we can create alternative work options for those who have had their roles automated. Many of these workers have first-hand experience with the phasing in of automated systems and can, in some cases, provide data annotation or decision monitoring given on the job training. We should identify and document relevant information so that business change processes can be developed to mitigate workers' impact being automated.

8.5 Accountability

Many areas of the Data Science Development Lifecycle took too many cues from Engineering instead of Science. Unfortunately, most companies develop data products don't allow you to take the special time and consideration needed to make equitable products. Equitable products, unline equal products, don't aim to just level the playing field for everyone, but they provide specific amounts of support to communities where they need it. We need to make the biggest change as Data Scientists, Engineers, and Researchers is that our workflow needs to incorporate reproducibility and fairness techniques.

A key to the successful mitigation of ML risks is real accountability. We have to build this into the culture of our data teams and tech companies at large. Build the muscle memory to ask, "Who tracks how ML is developed and used at my company? Who is responsible for auditing our ML systems? Do we have AI incident response plans?".

In order to be accountable, we must first take organizational responsibility for what happens to users after we put models into production. This gives us the incentive to take care and deeply

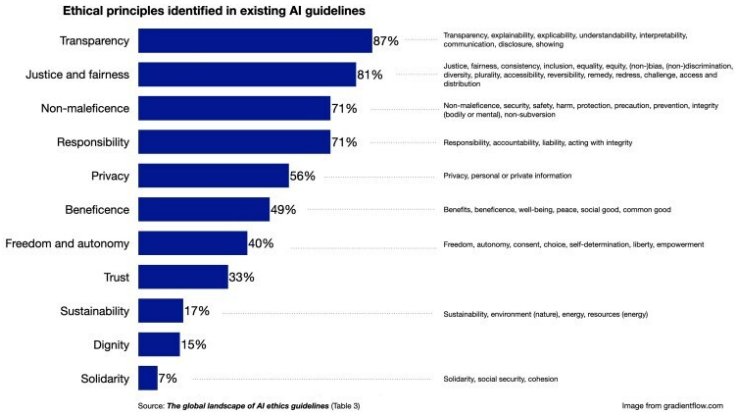
assess what kinds of tools we make. We also have to create open and transparent models and allow users, stakeholders, and the most vulnerable groups to assess our work. They need to be able to know we used an ML model to make a decision about them and test it for themselves. We can't pretend to be serious bout accountability if we offer nothing to users we've harmed. In some cases, this is financial compensation; in other cases, we can use alternative methods like discounted pricing offers for paying users who have been impacted by erroneous algorithmic bias.

Source: GradientFlow.

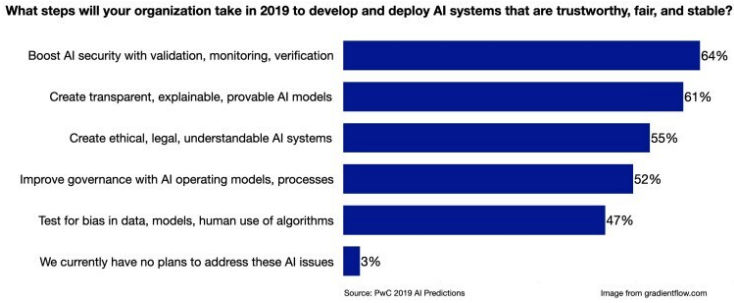
Current Practices

While machine learning fairness is still a relatively new field, a recent study for ETH Zurich attempted to find a consensus emerging around AI principles. While they were unable to find a

single principle that occurred in all 84 source guidelines, many prevalent themes are below.

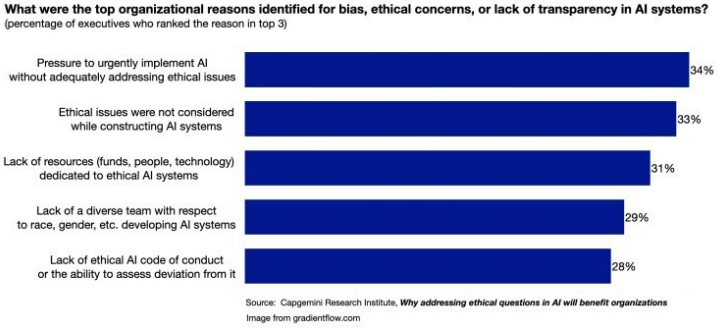


It's important to have an understanding of the trends companies are taking to address bias. Thankfully, according to a PwC survey, responsible AI principles are being worked on in the near term, but as you can see, many companies have focused on security and monitoring while some of the most pressing issues for users can be discovered fastest by testing for bias and maintaining model accuracy and fairness over time.

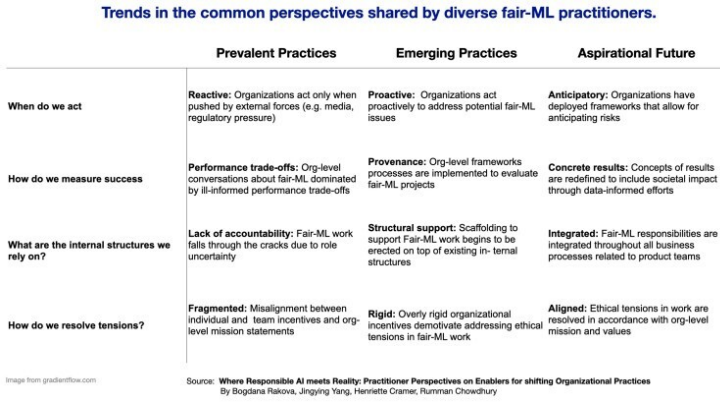


As I can speak from personal experience, the pressure to weave

AI into products for innovation's sake is large and mounting. Unfortunately, as we've marketed AI as a technology that can complete "superhuman" tasks, we've failed to educate the limitations. According to this Capgemini survey of executives, many felt the need to urgently implement AI without spending the time to address ethical issues. In an alarming amount of cases, these ethical issues aren't considered whatsoever.



We have a long way to go to get to our fair, utopian future. Unfortunately, now we're still tasked with translating fairness work into business metrics like clicks and churn rate even when it doesn't apply. In some organizations, they've prioritized losses to their reputation as the biggest factor pushing them towards fair ML instead of truly reducing harm to users.



Trends in the common perspectives shared by diverse fair-ML practitioners.

	Prevalent Practices	Emerging Practices	Aspirational Future
When do we act	**Reactive:** Organizations act only when pushed by external forces (e.g. media, regulatory pressure)	**Proactive:** Organizations act proactively to address potential fair-ML issues	**Anticipatory:** Organizations have deployed frameworks that allow for anticipating risks
How do we measure success	**Performance trade-offs:** Org-level conversations about fair-ML dominated by ill-informed performance trade-offs	**Provenance:** Org-level frameworks processes are implemented to evaluate fair-ML projects	**Concrete results:** Concepts of results are redefined to include societal impact through data-informed efforts
What are the internal structures we rely on?	**Lack of accountability:** Fair-ML work falls through the cracks due to role uncertainty	**Structural support:** Scaffolding to support Fair-ML work begins to be erected on top of existing in- ternal structures	**Integrated:** Fair-ML responsibilities are integrated throughout all business processes related to product teams
How do we resolve tensions?	**Fragmented:** Misalignment between individual and team incentives and org-level mission statements	**Rigid:** Overly rigid organizational incentives demotivate addressing ethical tensions in fair-ML work	**Aligned:** Ethical tensions in work are resolved in accordance with org-level mission and values

Image from gradientflow.com

Source: **Where Responsible AI meets Reality: Practitioner Perspectives on Enablers for shifting Organizational Practices**
By Bogdana Rakova, Jingying Yang, Henriette Cramer, Rumman Chowdhury

"Fair" ML trends

How to Avoid Bias

- **Don't underestimate it.** Bias is almost always there, and it will be present in the models you create until you take steps to address it.
- **Just because we have the data on something doesn't necessarily make it true.** Just like for inaccurate sensor readings and spurious correlations, don't accept "facts" without critiquing them.
- **When you get new data, ask about the collection methods.** All data is not equal, nor was it all collected properly.
- **Be proactive and consider this before you *have* to.** As you can see from the chart above, most companies are reactive when it comes.

I hope this chapter helped to set a baseline of knowledge in bias and accountability. If you want to stay up to date with my

favorite resources on Data Bias and Ethics, please check out my company's Ethics Resource Database.

9

Interview Questions

Continuing with the theme of active learning, I want to provide you with a list of common Data Science interview questions without providing the answers. The reason for this is that if you take the step to learn the answer, you'll remember it. If you want to memorize and regurgitate an answer to interview questions, books can be helpful. These interview questions don't always match what you'll be doing on the job, and I don't want to just you to recite canned answers. You might get the answer correct without truly understanding the context around the topic you memorized. What I will provide you with is a framework for answering these types of questions and what interviewers are looking for in your answers.

9.1 Behavioral Questions

When you're faced with answering behavioral questions, they typically ask you about how you've worked with colleagues, other teams, and your traits to assess if you're a good match for

their team. When answering these questions, the best framework for answering this is by leveraging the STAR interview response technique. The framework stands for the four steps in answering a question that leads your interviewer to your answer.

1. Describe the **Situation**. In this step, you want to set the stage for the scenario you'll be talking about.
2. Detail what your **Tasks** were. You can mention what steps you took and how that differs from what your team was working on.
3. Okay, so you had tasks; what **Action** did you take in this project?

9.2 Technical Questions

When you're being asked a technical question, the interviewer is looking for two things a) do you know the answer, and b) can you explain it to a layperson. You don't need to apply the STAR method here; be sure you answer the question being asked and simplify your explanation as much as possible.

If you don't know the answer to a question, it's always better to be honest about that than to flub through a response. Say something like, "You know, I have heard of the Central Limit Theorem, but I don't encounter it much in my day-to-day, so I can't explain it well." Your interviewer will likely nod and move on without deducting too many points for stumbling or getting the answer wrong.

What is the difference between supervised and unsupervised machine learning?

What is the difference between a Type I and Type II error?

Does gradient descent always converge to the same point?

What is an A/B Test?

What is overfitting?

Why is dimension reduction important?

What is boosting?

Why would you use Hypothesis Testing?

What's the difference between bagging and boosting?

What is a p-value?

What is better, good data or good models?

Is there a universal good model? Are there any models that are definitely not so good?

Is it better to spend 5 days developing a 90% accurate solution or 10 days for 100% accuracy? Depends on the context?

Do you know data reduction techniques other than Principal Component Analysis?

When would you use step-wise regression?

How do you handle missing data?

How would you turn unstructured data into structured data?

What are hash table collisions, and how is it avoided?

Do you think 50 small decision trees are better than a large one? Why?

When is it better to write your own code than using a data science software package?

What is root cause analysis? How to identify a cause vs. a correlation?

Give examples of data that does not have a Gaussian distribution.

What is a recommendation engine?

How to check if the regression model fits the data well?

What is collinearity, and what to do with it? How to remove multicollinearity?

Explain what precision and recall are.

How do you select performance metrics?

What is cross-validation?

When would you use random forests vs. SVMs?

Describe different regularization methods, such as L1 and L2 regularization.

How do you deal with unbalanced binary classification?

What is Gradient descent means?

What is data normalization?

What is the curse of dimensionality?

What are the drawbacks of a linear model?

What are the assumptions required for linear regression? What if some of these assumptions are violated?

Explain the kernel trick.

What is the difference between convex and non-convex cost function; what does it mean when a cost function is non-convex?

How do you assess the statistical significance of an insight?

Explain what a long-tailed distribution is.

What is the Central Limit Theorem?

What is the statistical power?

What is an outlier? Explain how you might screen for outliers

and what would you do if you found them in your dataset.

10

Career Insight

This chapter will drop some of the biggest gems I've gained from working in Data Science. There's so much mystery around the roles in data because there are few clear definitions of what a Data Scientist does and so many people view it as "magic" before trying to understand what we do. What this has led to is a culture of gatekeepers and ego-driven engineers with a god-complex. That doesn't mean you can't thrive in these roles, but navigating the industry people can be just as tricky as managing your data. I'll show you what skills you should focus on and how to develop your niche in Data Science.

If there's one thing I want readers to take away, hands-on practice is worth gold for your career. It's more important than your education and pedigree. When recruiters ask for "experience," at least for Data Science, it doesn't always mean full-time work experience. Many recruiters know there are various ways to get started in Data Science and ultimately want to be able to communicate the types of things you've worked on with hiring managers. Many people in industry

don't have degrees in related fields, but through visible projects, personal marketing, and networking, they have landed roles they wouldn't otherwise consider.

When I worked in my first industry roles, I was too overwhelmed by how much I used daily and what it felt like I needed to know. We were starting to use tools like Alteryx, Snowflake, DataGrip, Jira, and so many more, but it just seemed like foreign words to me. Chances are, by the time your land a role in Data Science, you'll be working with new tools and frameworks. This is totally normal. Think of this as part of the job where you leverage your quick learning skills. In fact, if you're in an interview and being asked about your experience with a tool you've never used, one of the best ways to answer is by giving an example of how you learned to use a leveraged a new software tool in the past. For almost all big data roles, you can learn things like command-line tools and virtual environments on the job.

10.1 The Bare Minimum

- Learn Basic Tools Well

It's really easy to be intimidated by the amount it seems like you have to know to land a job in Data Science, but that's not the case. Learn a few tools really well and transfer that knowledge to similar tools to get out of not having experience with one particular software.

1. Programming: Python or R
2. Data Retrieval: SQL / MySQL / PostgreSQL
3. Business Intelligence: PowerBI, Domo, SAS

4. Machine Learning & Related Math
5. Model Governance and Fairness Testing

- Understand project deliverables

As someone who experienced a formal Data Science education, I can say first hand that few programs cover data ethics, much less what it's actually like to do Data Science "in the real world." It's imperative when you land the job to understand what is expected of you. In a technical yet, often a free-flowing world of data, get clarity on what your team expects you to turn in for homework. Is it an analysis summarized in Powerpoint or trained models ready to be used by engineering? Are you creating an application that performs a task or interpretation to gain insight from? Discover what resources you have available in terms of data sources (Can you get new data? Does your team have a Data Engineer to create a data source for you? Do you have to conduct a survey or collect data from scratch?).

After you've discovered what you have access to, you can start to scope a project. I urge you to formalize a hypothesis related to the problem you want to solve. Keep in mind that the data we have available is just a proxy for something we want to solve. AFTER you have a hypothesis, do some light EDA to understand the spread and variability of the dataset at hand and identify areas where your data may encode bias. Discover if your data has PII or information about people in protected classes. Even if that's not the case, develop a reference for the kind of harm that can because if your model went "haywire" and consistently made poor decisions.

Once you have this baseline understanding, ask stakeholders about the kind of harm your org can cause and your work context.

I encourage data technologists to be close to the domain to truly understand the use cases and point out the weak spots in projects often directed by a part of an organization outside of Data Science. This is where you can solidify what they want from you. Be explicit and clear in communicating what you can and cannot do. For example, in a past role, I was asked to create a model that would classify people by gender, based on their names. While to me, this seemed like an obviously absurd idea, I had to understand the use case and incentives driving our marketing department to request a model like this. The company's product saw drastically different trends between male and female customers on social media, and they wanted to create a way to customize their notifications and recommended content. This, from a marketing perspective, may seem trivial; however, it poses a lot of legal and ethical quandaries. First, should we even be attempting to predict a person's gender when they have not given it to us willingly? Second, should we have marketing campaigns that are simplified to just identify customers as male and female? This excludes transgender and non-binary people. In addition, it assumes both genders are a monolith and can simply be seen as having male or female traits.

- Domain knowledge is crucial

Use what you know to increase the value you can add. It's always easier to change your role in the same industry or change your industry but has the same role. Rarely both. With that knowledge, start thinking about not only the hottest fields using AI but ones you may already have some experience in. This can be retail, marketing, logistics, or even the music industry.

Nearly every industry is using machine learning, so target companies where your knowledge in their domain would make you specifically perfect for the role.

You're in a unique position to be able to point out the bias more easily as you may have seen similar data in past roles. Many of us can impact our companies' decisions regarding modeling architecture and deployment methods in many cases. Many of the p[oepe who thrive in data roles enjoy learning and implementing new strategies. This is a common theme in nearly every data science department. It's not uncommon to be expected to research a new technique or method and implement it on real data in a matter of weeks. This moves close to the speed of software engineering, but we hardly have enough time to truly research new methods, our data caveats, and the worst-case scenarios for our models. This is where you can lean on your domain knowledge to advocate for the users. Remember, companies aren't altruistic, but almost all of us want to do the right thing as individuals. It's tough to do this when a lot of our incentive structures don't match up with what we know we "should" do. Use your domain expertise as a way to be even more valuable than other candidates.

- Do NOT fear the Math

Yes, some math is necessary for work in Data Science (more for ML and deep learning) but don't be afraid of it. It's relevant, and understanding the funky squiggles that make up statistical formulas is crucial. Work on learning the Greek letter notations as it will make looking at a formula go from nonsensical hieroglyphics to understandable commands. Stop seeing math as something you are bad at and as something you can improve.

Probability skills aren't something we're born with. I struggled incredibly with this when I first got started, and the two things that helped me were a mindset shift and some practice. Math really is one of those subjects where practice makes perfect. You can practice reading formulas or computing probabilities for your favorite team, winning. If you struggle with the subject, you'll just have to spend more time working on it, and that's okay! You don't have to know everything, and nobody expects you to, but you have to learn the core concepts to be proficient at your job.

- Impostor Syndrome is **normal**

Data Science as a whole moves incredibly fast, and everyone feels like they aren't up to date with everything. There are seasoned practitioners who are still googling the same errors that you are. There are data leaders who still struggle with tricky SQL queries. What matters is that you practice every single day. Even if you just do some interview questions or a video on YouTube or Udemy, practice every single day. One of the methods that helped me when I was in grad school was to limit my consumption of non-data news and social media sources. Being so deeply embedded in the relevant content made it easier to learn at an exponential rate. While I have a love/hate relationship with Medium, I have used my paid subscription to stay up to date with Data Science news. You can also do this with RSS feeds like Feedly. Leverage Twitter lists to find more content from Data Scientists and those working in ML/AI.

Sometimes we underestimate the labor that goes into our work. Data Science is fucking hard! I've been at several companies in the industry where we had PhDs and ex-FAANG

folks cramming their brains together and were unable to find solutions to these data problems. This isn't an easy line of work despite the perceived sexiness. Data Science is tough work because it's political. There are very few industries in which you don't have the chance to hurt people. With that in mind, we want to make sure we're "good" at what we do so the outcomes of our work are fair.

Some of the best ways to work on your overall Data Science skills are to do end-to-end data projects, test for fairness, document them, and present them. This process repeated over various types of data and use cases is one of the fastest ways to build your data science portfolio and true skills. Once you are able to create data pipelines, train models, and evaluate ss metrics, you can start going deeper into topics like data sampling, entity recognition, and deep learning.

- Practice is more valuable than credentials

As you get started, SQL and Python will be your most valuable skills! When you step into your next big role, it's important to keep in mind why you're there. Your company wants to use Data Science to guide decisions like optimizing routes, serving ads, or personalizing their product. It's important for you to prioritize your work and make sure you're accountable for the quality of your code. You'll likely be working with a project manager who will help guide your team in developing new models and putting them into production. Their job is to make sure these projects come to life and that each person working on it is making progress. That being said, don't take frequent check-ins from PMs personally. They're doing their job and reporting up the chain how work is progressing. This leads to an area

of friction for new-to-industry Data Scientists; how do you estimate how long a project will take. To do this, we have to acknowledge the difference in how long we want something to take, how long it "should" take, and how long it actually takes you.

In the case you aren't working with a PM, you'll likely have a data /analytics team you check in with on a weekly or daily basis. In these meetings, it's typical to briefly cover the work you're doing, any issues you've had along the way, and any needs you have. It's likely you'll also have a manager to check in with on a one-on-one basis. For many juniors in the field, outlined tasks and strong infrastructure are the most important aspects of helping projects get completed on time.

Below is a checklist to help guide the new project that's landed in your job at work.

- What are your deliverables, and what is being asked of you technically?
- Frame the model: Are you trying to predict a risk score, predict product metrics, or give customers recommendations?
- Does the right data exist, and how do I get to it?
- What are we *really* trying to measure, and what are they asking me to measure?
- Why did they (stakeholders) want this model/analysis?

Keep these things in mind when you're working on your data projects, and be ready to explain how you can provide value to a company. Working within our capitalist systems' constraints, we have specific goals we have to achieve as data professionals. Companies often perceive Data Science and AI as something

they invest in but don't always value. When you structure your portfolio project, you'll want to consider how a company could see your work as useful...or not.

10.2 Be the Squeaky Wheel

One of the biggest tips I wish someone had told me would be to stand by my values. This means not working on projects that cause harm. I also mean taking the initiative to to speak up when I see product issues I know impact marginalized groups disproportionately. Even greater than my altruism, one should understand standing idly by while companies create harmful technology that can be damaging to your career.

In June 2020, the Association for Computing Machinery recommended that governments ban facial recognition as there are too many ways in which facial recognition hurts dark-skinned people and women especially. This means for many engineers whose life's work is computer vision for IBM or Microsoft, then the past decade or so of their careers will be valued less than their peers.

While many people who aren't directly impacted by the bias machine learning can perpetuate are able to put their heads down and ignore these problems in the workplace, it's imperative you don't if only to secure your career. While the subfield of AI Ethics is new, it's growing quickly and is the direction we are headed towards in the future.

Believe it or not, the future of AI is not in automated bi-pedal robots, but explainable AI and human in loop systems. Despite the number of research articles published or money made from working in these roles, taking on a technology that's actively harmful can be just as devastating to your career.

10.3 Tools of the Trade

There's a difference between understanding the theoretical knowledge and knowing how to use the tools. I encourage you to become familiar with the following tools and how they're used in Data Science. Some of these can quickly be learned by playing with the software; others may need a few tutorials and practice. Don't be intimidated by this list; it's not like you're supposed to know all of these things on day 1. I show this to you, so you aren't surprised when you see something new mentioned. Being at least somewhat familiar with what something is used for or having used something similar is greatly valuable.

Analyzing Data - You absolutely need to master Excel, and as you develop, you can use the tools below if you have the opportunity or do this EDA in R or Python. You don't have to be familiar with these other tools, but you may have access to them in some roles.

- Excel - the simplest and most common spreadsheet and EDA tool
- Alteryx - an integrated platform to discover, prep, and analyze data
- KNIME - design data science workflows and reusable components
- RapidMiner - a tool for working on each step of prediction modeling
- Matlab - another tool to analyze data and develop algorithms and for creating models.

Coding Languages - You can learn all three, but learn and

master one first!

- R - a statistical programming language made to deal with numerical data and statistical methods
- Python - a high-level programming language with a large Data Science library
- Scala - an extension of the Java language that combines object-oriented and functional programming

Coding Environments - Where you write your code.

- Jupyter Notebooks - for experimental work in R or Python
- R Studio - IDE for R programming
- Anaconda - Python/R group of programs made for Data Science
- DataGrip - Popular IDE for working with SQL databases

Retrieving Data -You can just learn SQL and learn the ins and outs of the other tools.

- SQL - open-source Relational Database Management System (RDBMS)
- Snowflake - fully relational SQL data warehouse
- Amazon Redshift - another commonly used cloud data warehouse
- Google BigQuery - a scalable, serverless data warehouse
- Microsoft Azure - a set of cloud services

Distributed Computing - A popular method to deal with the vast amounts of data is to run processes on data on various different computers at once.

- Hadoop - allows you to distribute processing across computer clusters.
- Spark - a Hadoop extension that handles batch (offline) processing and stream (online) data processing

Cloud Computing - Doing work on other peoples' computers!

- AWS - allows you to run models on Amazon's computers
- Docker - OS-level virtualization that delivers software in packages called containers.
- Kubernetes- open-source automation, deployment, and management for containerized applications

Data Visualization - There are many tools to translate data stories into engaging visuals.

- Tableau - a drag and drop tool to visualize data
- Microsoft PowerBI -
- SAS - proprietary software for doing statistical operations on data generally used by large companies

Version Control - This helps us track changes in code and models. It's common that people interact with Git to version their code using one of the following interfaces.

- Github
- Gitlab
- Bitbucket
- Git Kracken

Project Management - These tools are used widely in engineer-

ing teams in industry to collaborate and track technical projects.

- Azure DevOps
- Jira
- Trello
- GanttPRO

What do Data Scientists really do?

1. **Coding** - There is a lot of coding involved in Data Science, but I like to see it merely as a tool to do our work faster or at scale. We mostly leverage coding to collect and clean data, conduct data analysis, and build machine learning models.
2. **Automating Tasks** - A core belief in Data Science is that we shouldn't do receptive tasks manually. A good mindset to adopt now is that it's worth learning a new technique to automate our work rather than spend the time doing it ourselves. Leverage meaning the computer do the task for you, and you'll also come away with a new automation skill.
3. **Thinking** - A lot of our work is mentally understanding what projects to start and how to do so. Data Science is a mentally difficult profession. We are constantly answering difficult questions and having deep scientific and philosophical conversations. You spend a lot of time thinking and formulating a plan to approach problems.
4. **Talking to colleagues** - This includes standup meetings, collecting insights, collaborating on projects, and presenting findings.
5. **Learning new Techniques** - Data Science is still a relatively new and rapidly growing field; this means we

have to stay great by learning new things on a daily basis. Even senior Data Scientists are learning new tools and techniques frequently. Fortuynatltey, if you like to learn, there is no lack of papers, articles, and tutorials that can help you learn new techniques and how to apply them. I lump research into this category because as 2we're often tasked to work with new kinds of data or methods we aren't well versed in, we spend a lot of time researching methods in fields like NLP and how it's been applied to our industries before scoping out a project.

10.4 Projects Fail, People Don't

You should be ready to distance your perceptions of yourself as a professional from your work. You will have data projects that fail and data projects that succeed. I find it helpful to talk about the reasons why a project might fail and what you can do about it. The most common experience I've had is that projects fail because I don't have the right data. This can be because I don't have the right access or permissions, or I need access to hard-to-get-data to properly answer the question. Other common reasons that data projects fail is because our team doesn't have time to use our analysis or models, or what I like to call the "What do we do now?" problem. Often without a Data leader on the teams, it's hard to collaborate with others to structure a data analytics and ML product plan. It's also frequent that new scientists excited to work on cutting-edge work create far too complex models for the data. It's better to create interpretable models first and then test against a baseline to create models that match your data's level of complexity. The most dangerous factor that can cause data projects to fail is the

lack of reproducibility. Unlike some of the many other reasons project can fail, this is what you can control the most (along with starting with simple statistical models). One way to ensure model reproducibility is to use Git/GitHub to version control your code and models. It's also crucial to leverage automated testing creating data pipelines to pre-process data quickly at scale.

10.5 I got the job! Now what?

Data science jobs require a lot of different skills and juggling many kinds of tasks. Use these tips to help you adjust to your new data filled life.

Structuring Your Day

- What data projects are you working on, and which has priority?
- Do I have the data I need?
- Does this data match the problem properly (prototyping)?
- Is my data clean enough for modeling?
- What models would work on this data?
- What's the worst harm I can inflict by doing this?
- Do you have relevant data meetings?
- Do you have less-relevant organizational meetings?

The End-to-End Machine Learning Project

In Data Science, you may hear about the "end-to-end" project a lot. This really means that you can complete many of the different types of tasks required to get a machine learning project

going. Though different specialists in some organizations do these tasks, a generalist should be able to do all of these things well.

Data Collection - Data collection, also described as Data Wrangling, is the ability to get our hands on data and accurately aggregate or work with it. This can be using SQL queries, scraping the internet, or writing custom code that interacts directly with APIs you are getting data from.

Data Analysis - Remember the main points of chapter two and know the analysis is one of the most important aspects of the end-to-end project. Document the things you find during this step so you can guide your ML model selection and recall why you made certain data cleansing and pre-processing decisions.

Visualization - Being able to accurately visualize data that tells a story is critical to sharing the results of your project. Ask yourself what you want the audience to gain before creating visualizations. Decide what you should highlight (e.g., data composition, trends over time, and variable relationships) and choose the right kinds of graphs that do so.

Machine Learning - The Machine learning step has been over-hyped and under-scrutinized. In this step, please go further than most tutorials and test for bias and fairness. In addition, you should be able to list the pros and cons of using certain methods and why you made the decisions you did. In this step, you can train, build, and test new models as well as evaluate their performance and iterate on this process. It's important to take into consideration the model architecture you're using, the

kinds of data you have access to, how it can be flawed, and the data you wish you had.

Documentation & Communication - A good end-to-end project has a clear documentation and communication piece. This can look like a simple Github README file or a PowerPoint presentation. Ensure to take note of your audience and their needs when communicating results. If you're preparing for job interviews, be prepared to succinctly talk about each step you completed in this process while demonstrating your technical knowledge. It's likely you'll be asked if you know the difference between bagging and boosting if you say you've worked on decision trees.

10.6 To Specialize or Nah?

One of the best ways to set yourself apart in a very competitive field like Data Science is to develop a specialization or niche. For those transitioning into Data Science from other roles, it's expected you can transfer the domain knowledge from your old work in a Data Science role. One of the exciting things about ML is that it's being implemented in nearly every industry. I found it's much easier to change your job title or company, but rarely both at the same time. When I was moving from marketing to Data Science, I started working on analytics and ML problems within marketing so I could leverage my domain knowledge. As my technical skills grew, I could comfortably move into dealing with data in a different domain. This isn't to understate that a deep knowledge of a dataset's context is important to building good models.

My undergrad degree was in Media & Professional Com-

munications, so that became my superpower. Within three months of my highest paying job as a Data Scientist, I had presented in 4 webinars hosted by my company. For comparison, my coworkers stated our Principal Data Scientist hadn't been forthcoming about wanting to present frequently. If you're outgoing and outspoken, being a communicator is a valid and high paying niche to have.

Before you can seriously start thinking about how to build this, you have to know what kind of specialties are out there. I'll outline two kinds of niches, that of industry and those that are about technical skills. If you've worked in any industry prior to focusing your tasks on predictive molding, you'll want to leverage that as your domain expertise. Some of the popular verticals non-technical people have transitioned to data science from include Finance, Healthcare, Marketing, Business, and Creative work. Each of those industries has a set of jargon and norms that having prior knowledge of proves useful.

You may be wondering what the areas you can specialize in as a Data Scientists are. There are two ways to look at this; you can have a deep understanding of a technical niche like NLP, Computer Vision, or time series modeling, OR you can have a "soft skill" superpower you can leverage regardless of the subject matter. For me, my niche leverages my communications degree and strong social skills. In many past roles, I was relied upon to deliver news to stakeholders or meet with new clients. My ability to do this comfortably was unique compared to those I was on teams with. Even the math PhDs volunteered me to speak for the group. Make yourself needed either by the expert knowledge you have or your ability to deliver it well. This can be being a rockstar at creating understandable visualizations or explaining statistical concepts.

NLP: If you're interested in language, linguistics, or how people communicate with each other and computers, I suggest learning about Natural Language Processing and the types of industries and roles that use NLP.

Computer Vision: For those who find computational photography, self-driving cars, or extracting meaning from images, there are many companies utilizing computer vision to better find manufacturing defects in products or for detecting different types of diseases.

Time-Series Modeling: There are some professionals who focus their efforts on time series modeling or trying to predict the future. Time series modeling is a little bit less of a niche as there are many use cases for time series data in various industries, but some professionals have built careers around this as their focus.

Deep Learning: Some professionals are intimately acquainted with neural networks and work on deep learning projects that typically require a lot of data and processing power to train. For this niche, you'll want to know the ins and outs of neural networks, specifically deep neural networks.

What I find most interesting about Data Science is how many people are able to transition from non-traditional backgrounds. There are a few tried and true ways of getting into Data Science for those new to Data Science. Some people with Statistics degrees have had to level up their coding and development skills to work in Data Science specific roles. While this can be seen as the traditional way, along with studying Robotics or Machine

Learning at top-tier universities, there is no right way to carve a career in Data Science. The industry has so many applications across nearly every industry that your past experience in one of these industries can be your differentiator. If you're a new college grad, you have the opportunity to choose industries that interest you.

When I was in grad school, I was told it's easier to change your job title in the same industry or change industries, keep your job title, and nearly impossible to do both simultaneously. Since my background was in marketing, I started my career as a "Marketing Data Scientist," where I worked on metrics that applied to SaaS companies, like predicting customer churn.

As a newcomer, I urge you to collect, sample, and clean data well in addition to bringing your expertise from another field. Data collection and documentation regarding data governance are few and far between in most organizations. If you're completely new to data science, this is some of the "low hanging fruit" you can use to your advantage. If you have extensive knowledge or experience with data governance as a junior in the industry, you've separated yourself from the pack. This is a skillset many data professionals, even from traditional backgrounds, have little experience. Don't let Data Science's age fool you; the industry is highly competitive, with big-name companies pioneering the use of ML and AI in products that reach millions. Test for and document the bias in your data and make sure it's explicit in your storytelling.

In closing, I want to encourage you to learn actively. Online courses and tutorials tend to handhold you too much or not give you enough context. My goal with this book has been to give you the context of what doing data science is like in industry,

as well as encouraging you to think about the impacts of your work before you dive into it. I suggest you practice your analysis, feature building, data storytelling, and model evaluation skills. Data science is truly hard work, but it is the most rewarding work I've done. I have had the chance to be on life and death projects as well as work on fun AI projects like a Megan Thee Stallion lyric generator. Don't be intimidated, as it's a great field for those who are interested in learning continually. Data Science allows us to answer hard questions with statistical tools. The Data Science journey is continual, and it's easy to be demotivated after a string of failed interviews or technical screenings. After 5 years in the field, I'm still learning new topics every day. I want you to be successful, and I want to see you thrive! Don't compare your journey to others, instead work on improving your skills a little by little each day. The tools are just the means to arrive at the same task, don't get caught up in specifics of tools and products as you can't always transfer that, but knowing the underlying concepts will always be beneficial.

* * *

I'm happy to help you along on this journey, and as a small gift to you, my readers, I'm offering 50% of Ascend Data Science Group Coaching for 3 months with code: GSDS. More information about group coaching will be available on my website. If you are looking for one-on-one mentorship you can connect with me on MentorCruise.

Move slower and empower people
Ruha Benjamin

About the Author

Ayodele Odubela is a Data Scientist working on driver risk mitigation at SambaSafety in Denver, CO. She earned her Master's degree in Data Science after transitioning to tech from social media marketing. She's created algorithms that predict consumer segment movement, goals in hockey, and the location of firearms using radio frequency sensors. Ayodele is passionate about using tech to improve the lives of marginalized people.

You can connect with me on:

https://www.ayodeleodubela.com
https://twitter.com/DataSciBae
https://www.linkedin.com/in/ayodeleodubela

Subscribe to my newsletter:

https://ayodeleodubela.substack.com