

THE COMPLETE VISUAL GUIDE TO

# MACHINE LEARNING



★★★★★ With Best-Selling Instructors **Josh MacCarty & Chris Dutton**



# SETTING EXPECTATIONS

---



This is **NOT** a coding/programming course; it's about introducing and demystifying essential machine learning topics

- *Our goal is to break down complex techniques using simple and intuitive explanations and demos*



We'll focus on tools commonly applied to **business intelligence** use cases

- *We'll focus on techniques like data profiling, linear/logistic regression, forecasting, and unsupervised learning, but will not cover some more advanced or specialized techniques (deep learning, NLP, etc.)*



We'll use **Microsoft Excel** as a tool to help explain key concepts

- *Excel's intuitive, visual interface allows us to expose the nuts and bolts of each technique to understand HOW and WHY these algorithms work (rather than simply running lines of code)*



You do **NOT** need a math or stats background to take this course

- *We'll cover the basics as needed, but won't dive deep into statistics or econometric theory*

# SETTING EXPECTATIONS

---



## ***Who this is for:***

- Analysts or BI professionals looking to transition into a ML/data science role
- Students looking to develop a deep conceptual understanding of core machine learning topics
- Anyone who wants to understand WHEN, WHY, and HOW to deploy machine learning tools & techniques



## ***Who this is NOT for:***

- Senior data professionals looking to master advanced topics in ML/AI
- Students looking for a hands-on coding course or bootcamp (i.e. python/R)
- Anyone who would rather copy and paste code than become fluent in the underlying algorithms

PART 1:

# DATA PROFILING



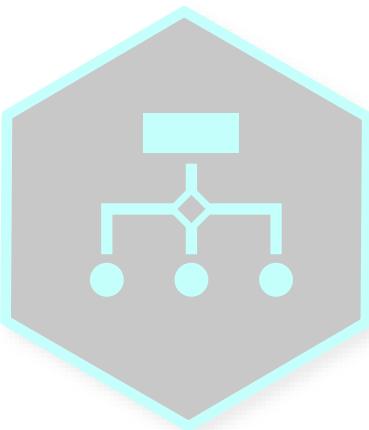
# ABOUT THIS SERIES

This is **Part 1** of a **4-Part series** designed to help you build a deep, foundational understanding of machine learning, including data QA & profiling, classification, forecasting and unsupervised learning



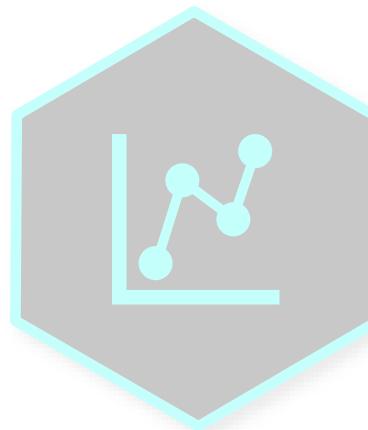
## PART 1

QA & Data Profiling



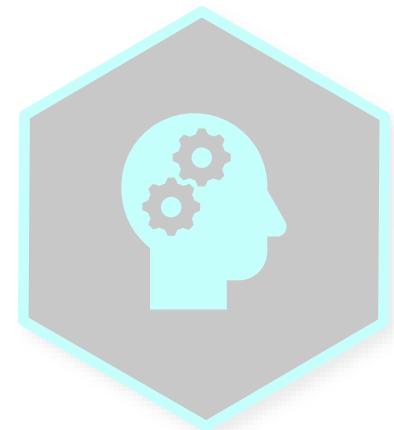
## PART 2

Classification



## PART 3

Regression & Forecasting



## PART 4

Unsupervised Learning

# COURSE OUTLINE

---

1

## ML Intro & Landscape

Machine Learning introduction, definition, process & landscape

2

## Preliminary Data QA

Tools to explore *data quality* (variable types, empty values, range & count calculations, table structure, left/right censored data, etc.)

3

## Univariate Profiling

Tools to understand *individual variables* (distribution, histograms & kernel densities, data profiling metrics, etc.)

4

## Multivariate Profiling

Tools to understand *multiple variables* (kernel densities, violin & box plots, correlation, variance, etc.)

# ML INTRO & LANDSCAPE

# INTRO TO MACHINE LEARNING (ML)

---



## MACHINE LEARNING

[ muh-sheen-lur-ning ]

---

**noun**

1. The capacity of a computer to process and evaluate data beyond programmed algorithms, through contextualized inference\*



Using statistical models to find patterns and make predictions

# COMMON ML QUESTIONS

---



*Which customers are most likely to churn next month?*



*What will sales look like for the next 12 months?*



*What patterns do we see in terms of product cross-selling?*



*How can we use online customer reviews to monitor changes in sentiment?*

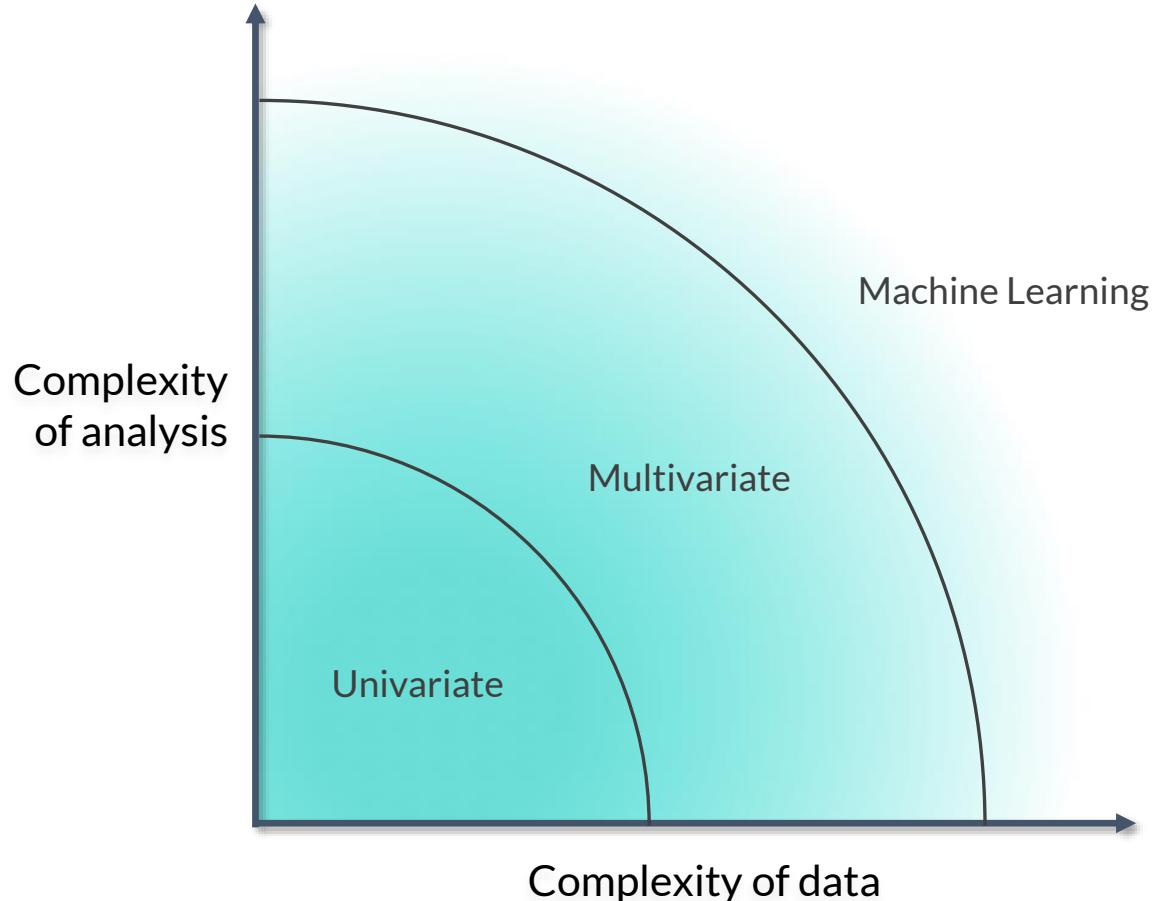


*When we adjusted tactics last month, did we drive any incremental revenue?*



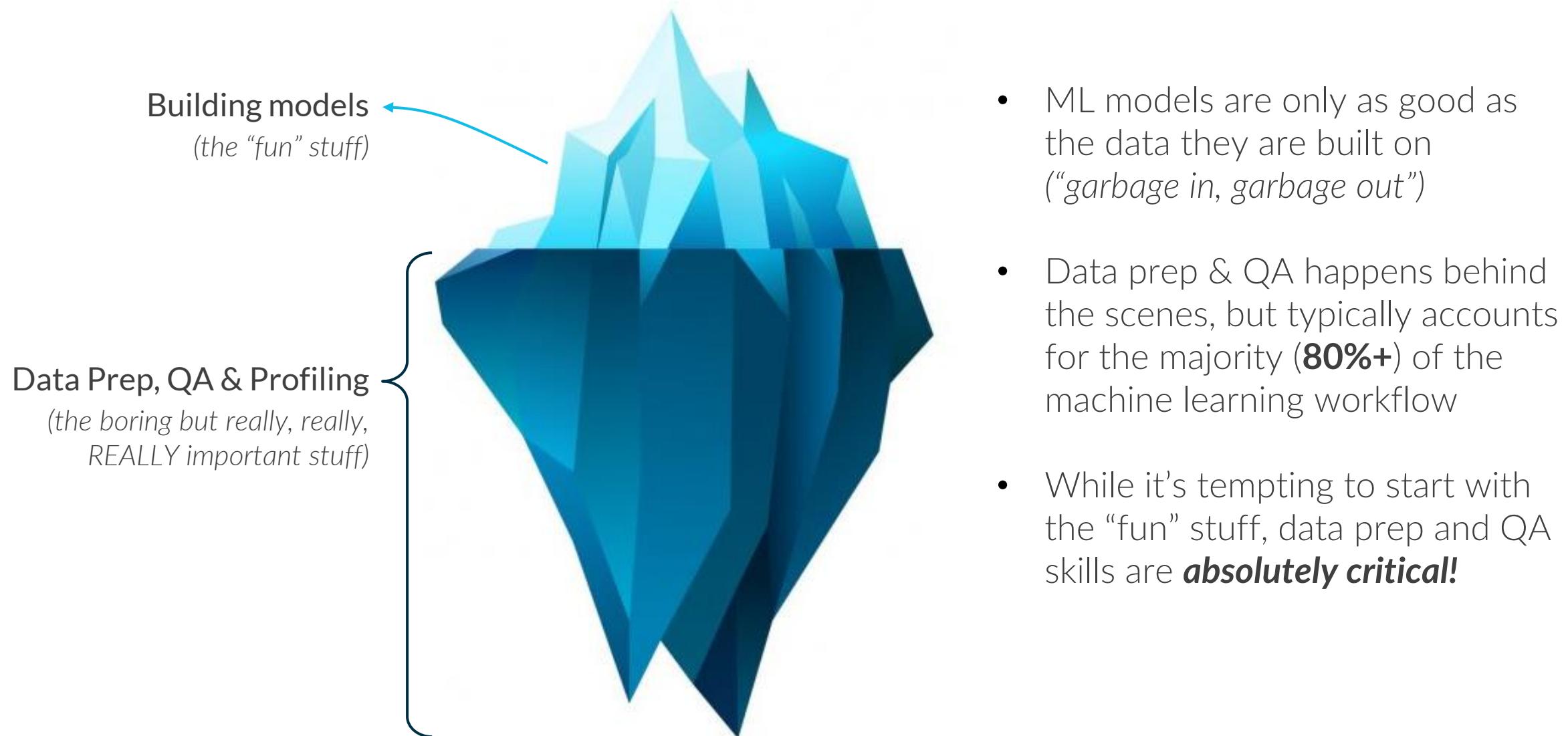
*Which product is customer X most likely to purchase next?*

# WHEN IS ML THE RIGHT FIT?

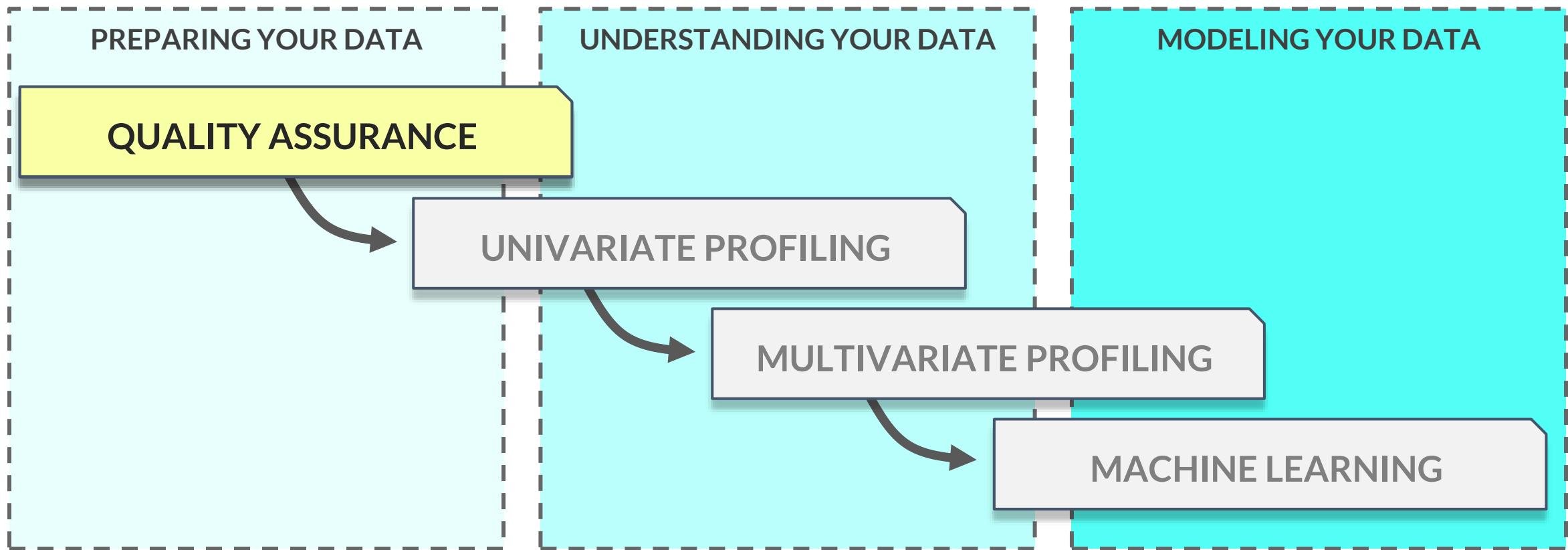


- Machine Learning is a natural extension of data profiling and basic visual analysis
- ML is required when the underlying data or analysis is **too complex for basic data profiling** (*i.e.* visualizing relationships between 3+ variables)
- Machine Learning is ideal for finding optimal solutions that would be **impossible or impractical to derive through human trial-and-error**

# THE MACHINE LEARNING PROCESS

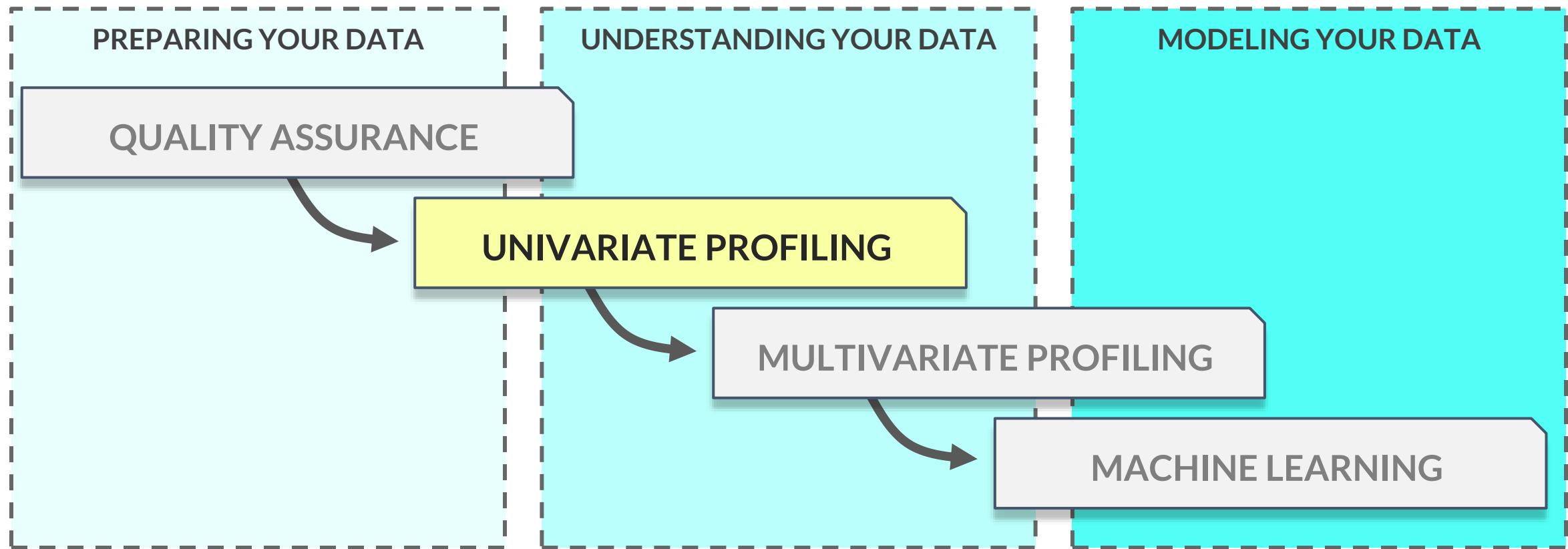


# MACHINE LEARNING PROCESS



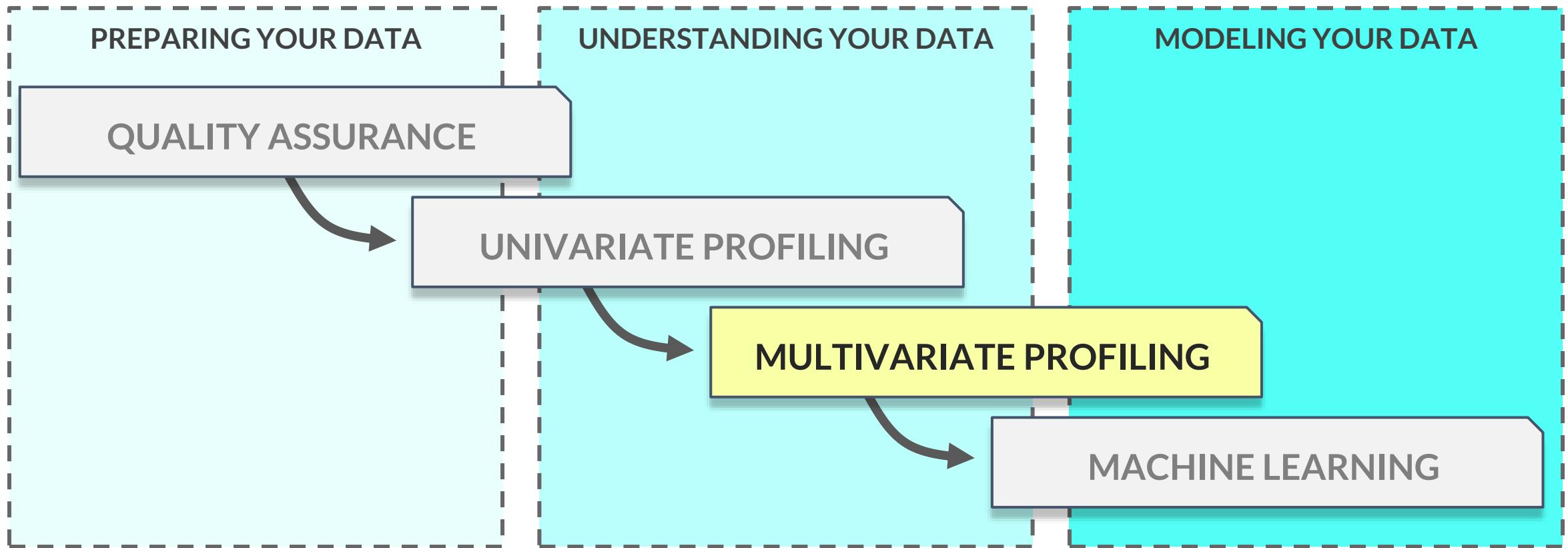
Quality Assurance (QA) is about **preparing & cleaning data prior to analysis**. We'll cover common QA topics including variable types, empty/missing values, range & count calculations, censored data, etc.

# MACHINE LEARNING PROCESS



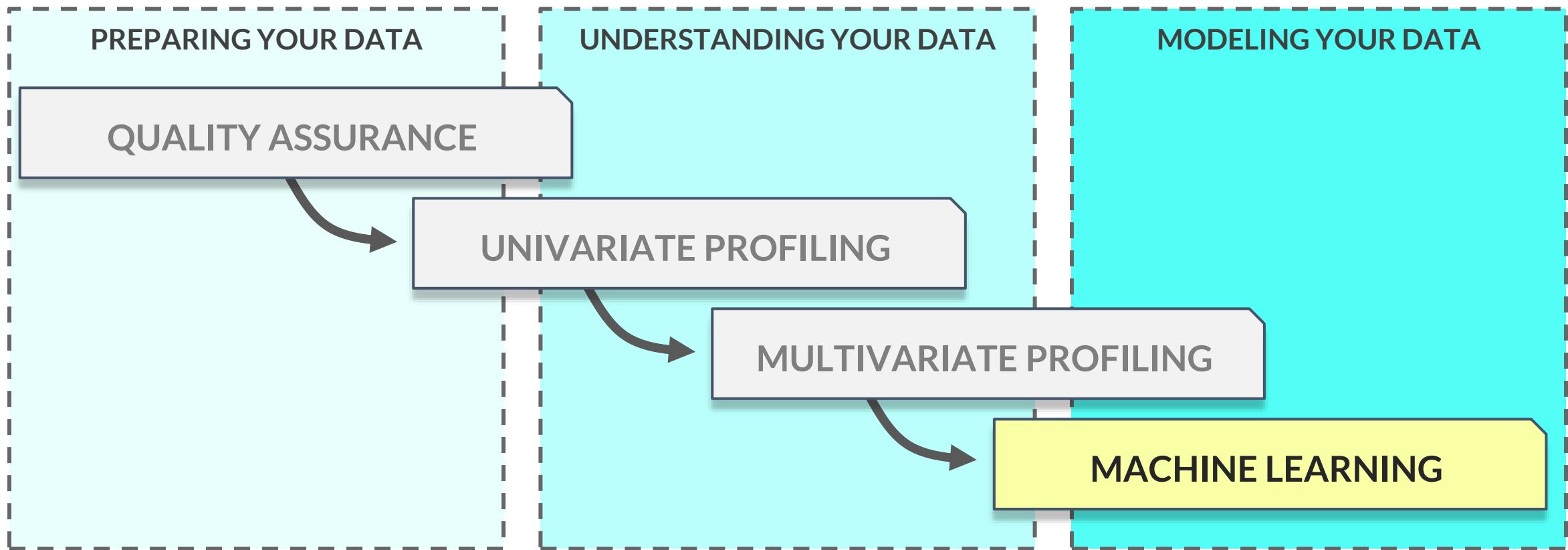
Univariate profiling is about **exploring individual variables** to build an understanding of your data. We'll cover common topics like normal distributions, frequency tables, histograms, etc.

# MACHINE LEARNING PROCESS



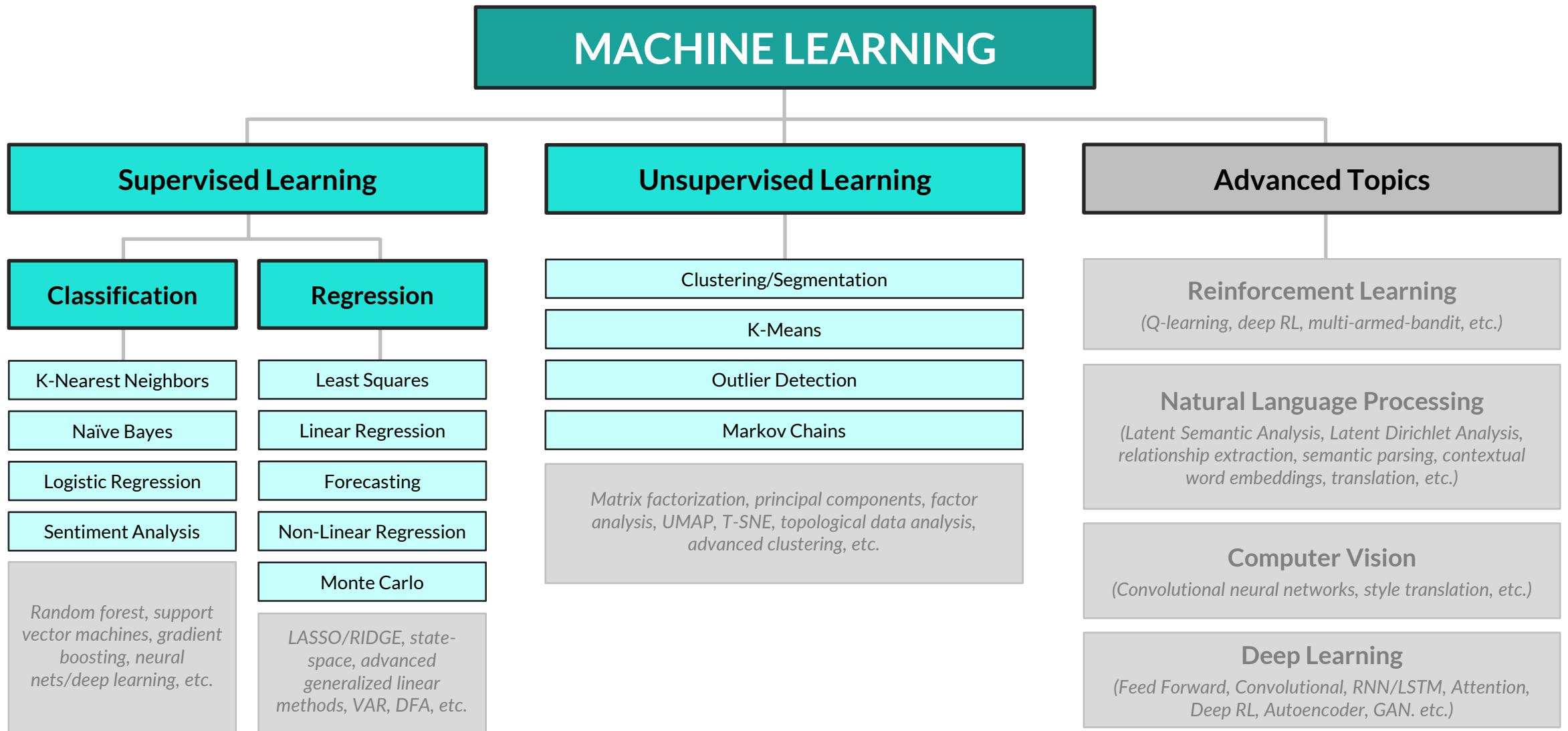
Multivariate profiling is about **understanding relationships between *multiple* variables**. We'll cover common tools for exploring categorical & numerical data, including kernel densities, violin & box plots, scatterplots, etc.

# MACHINE LEARNING PROCESS



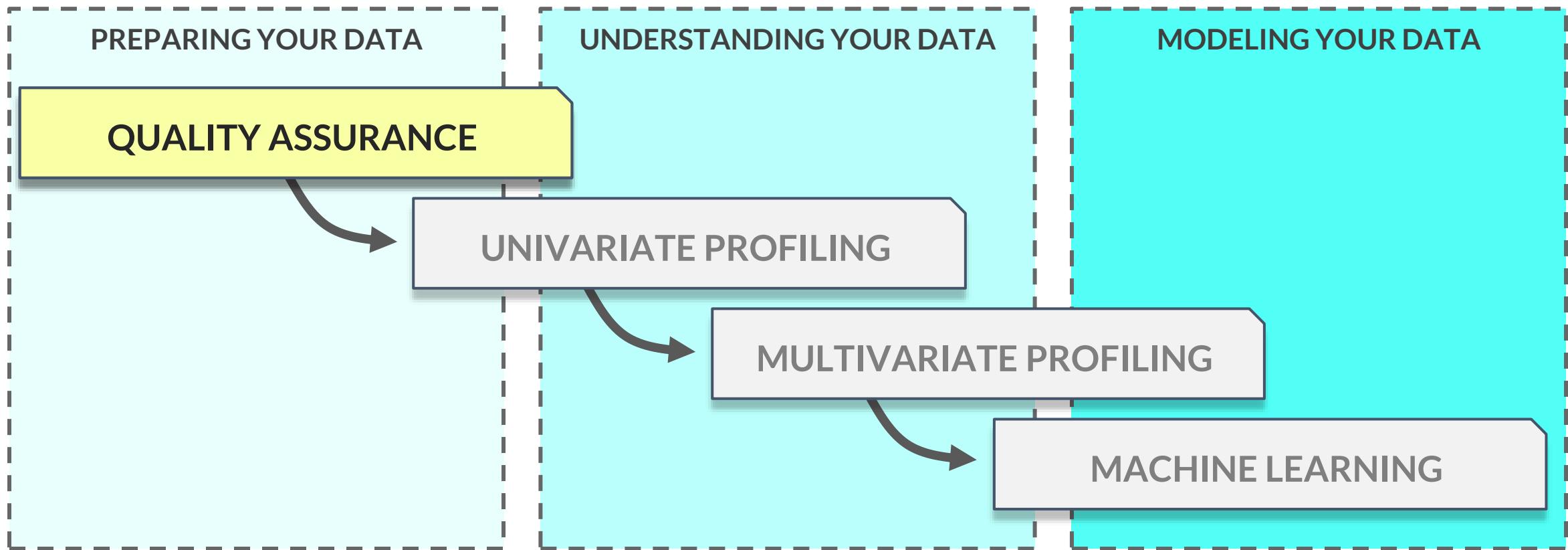
Machine learning is a natural extension of multivariate profiling, and uses **statistical models and methods** to answer questions which are too complex to solve using simple visual analysis or trial-and-error

# MACHINE LEARNING LANDSCAPE



# PRELIMINARY DATA QA

# MACHINE LEARNING PROCESS



Quality Assurance (QA) is about **preparing & cleaning data prior to analysis**. We'll cover common QA topics including variable types, empty/missing values, range & count calculations, censored data, etc.

# PRELIMINARY DATA QA



**Data QA** (otherwise known as **Quality Assurance** or **Quality Control**) is the first step in the analytics and machine learning process; QA allows you to identify and correct underlying data issues (*blanks, errors, incorrect formats, etc.*) prior to analysis

## TOPICS WE'LL COVER:

Variable Types

Empty Values

Range Calculations

Count Calculations

Table Structures

Left/Right Censors

## COMMON USE CASES:

- Minimizing the risk of drawing false conclusions or misunderstanding your data
- Confirming that dates are properly formatted for analysis (as *date values, not text strings*)
- Replacing blanks or errors with appropriate values to prevent summarization errors
- Calculating ranges (*max & min*) to spot-check for outliers or unexpected values

# PRELIMINARY DATA QA

---



**POP QUIZ:** When should you QA your data?

---

**EVERY. SINGLE. TIME.**

(no exceptions!)

# WHY IS QA IMPORTANT?

As an analyst, **Preliminary Data QA** will help you answer questions like:



*Are there any missing or empty values in the data shared by the HR team?*



*Was the data our client captured from the online survey encoded properly?*



*Is there any risk that the data capture process was biased in some way?*



*Are there any outliers that might skew the results of our analysis?*

# VARIABLE TYPES

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

**Variable types** give us information about our variables

- January 1, 2000 as a **number** simply displays a date
- January 1, 2000 as a **date** implies a set of information we can use (*first day of the year, winter, Saturday, weekend, etc.*) all of which can be used to build strong machine learning models

Common **variable types** include:

## Numeric

- Customer count

## Ordinal

- Small, Medium, Large

## Interval

- Temperature

## Ratio

- Weight

## Discrete

- Count

## Categorical

- Gender

## Nominal

- Nationality

## Binary

- Yes/No

## Date

- 1/1/2020

## Complex

- $4 + 2i$

## Logical

- True/False

## Monetary

- \$4.50

# VARIABLE TYPES

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

Understanding **variable types** is fundamental to the machine learning process, and can help avoid common QA issues, including:

- Numeric variables formatted as characters or strings, preventing proper aggregation or analysis
- String/character variables formatted as numeric, preventing proper text-based operations
- Values that Less obvious cases like variables that are re-coded from raw values to buckets (surveys)

ZIP CODES	
Numeric	String
90210	90210
32992	32992
1730	01730
2215	02215
92120	92120

Here we're formatting *zip codes* (which will never be analyzed as values), as a *numeric* rather than *string*



When we talk about **variables**, we might also refer to them as metrics, KPIs, dimensions, or columns. Similarly, we may use rows, observations, and records interchangeably

# EMPTY VALUES

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

Investigating **empty values**, and how they are recorded in your data, is a prerequisite for every *single* analysis

Empty values can be recorded in many ways (NA, N/A, #N/A, NaN, Null, "-", "Invalid", blank, etc.), but the most common mistake is **turning empty numerical values into zeros (0)**

Sales Leaders		
Name	Age	Sales
John	33	278
Sally	30	173
Mark	38	210
Gina	0	122
Phil	42	246

Average Age = 28.6 ✗

Sales Leaders		
Name	Age	Sales
John	33	278
Sally	30	173
Mark	38	210
Gina		122
Phil	42	246

Average Age = 35.8 ✓

# EMPTY VALUES

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

Empty values can be handled in 3 ways: **keep**, **remove**, or **impute**

- **Keep** empty or zero values if you are certain that they are accurate and meaningful (*i.e. no sales of a product on a specific date*)
- **Remove** empty values if you have a large volume of data and can confirm that there is no pattern or bias to the missing values (*i.e. all sales from a specific product category are missing*)
- **Impute** (substitute) empty values if you can accurately populate the data or if you are working with limited data and can use statistical methods (mean, conditional mean, linear interpolation, etc.) without introducing bias

PRODUCT SALES				
Date	Product Name	Product ID	Retail Price	Units Sold
5-Jan-20	Club Chocolate Bar	10660	\$1.35	4
5-Jan-20	Big Time Waffles	62736	\$3.69	
6-Jan-20	Excellent Apple Drink	30389	\$2.98	7
7-Jan-20	High Top Oranges	98772		11
7-Jan-20	Jeffers Oatmeal	60396	\$1.54	3

For a missing Units Sold value, you would likely **remove** the row unless you are certain that an empty value represents 0 sales

For a missing Retail Price, you would likely be able to **impute** the value since you know the product name/ID

# RANGE CALCULATIONS

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

One of the simplest QA tools for numerical variables is to calculate the **range of values** in a column (*minimum and maximum values*)

**Range calculation** is a helpful tool to understand your variables, confirm that ranges are realistic, and identify potential outliers

$\text{min}(\text{Age}) = 18$

$\text{max}(\text{Age}) = 65$

Is there a clear **lower or upper limit** (i.e. 18+ or capped at 65)?

$\text{min}(\text{Income}) = 0$

$\text{max}(\text{Income}) = 100$

Is your variable transformed to a **standard max/min scale** (i.e. 1-10, 0-100)?

$\text{min}(\text{height}) = -10$

$\text{max}(\text{height}) = 10$

Is your variable **normalized around a central value** (i.e. 0)?

# COUNT CALCULATIONS

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

**Count calculations** help you understand the number of records or observations that fall within specific categories, and can be used to:

- Identify categories you were not expecting to see
- Begin understanding how your data is distributed
- Gather knowledge for building accurate ML models (*more on this later*)

**Distinct counts** can be particularly useful for QA, and help to:

- Understand the granularity or “grain” of your data
- Identify how many unique values a field contains
- Ensure consistency by identifying misspellings or categorization errors which might otherwise be difficult to catch (*i.e. leading or trailing spaces*)



**PRO TIP:** For numerical variables with many unique values (*i.e. long decimals*), use a histogram to plot frequency based on custom ranges or “**bins**” (*more on that soon!*)

# LEFT/RIGHT CENSORED

Variable Types

Empty Values

Range Calculations

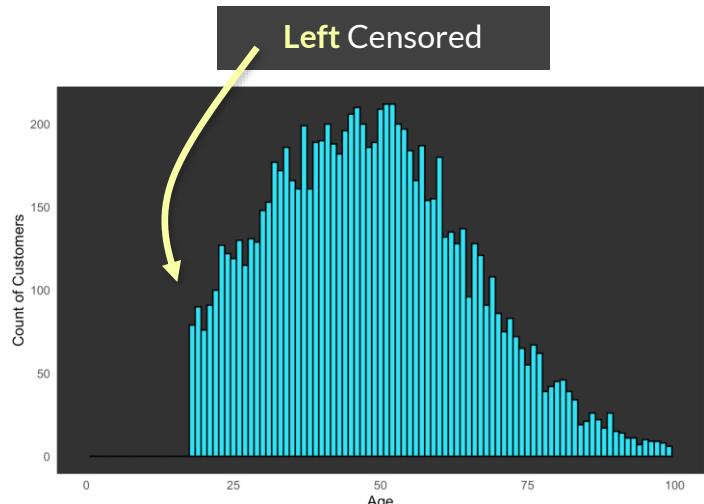
Count Calculations

Left/Right Censored

Table Structure

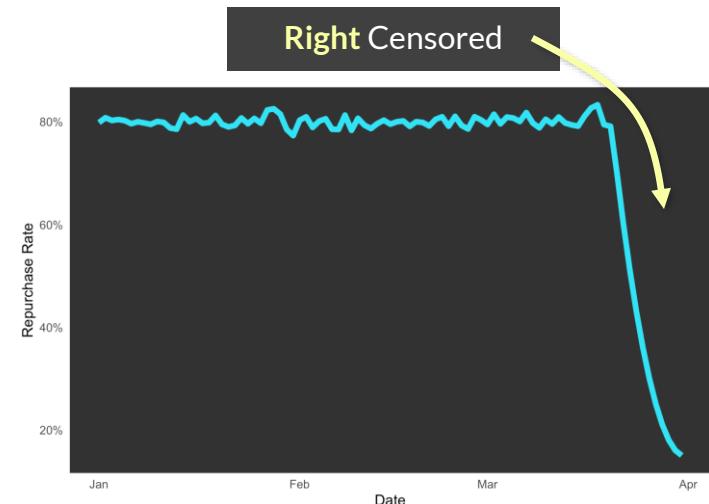
When data is **left or right censored**, it means that due to some circumstance the min or max value observed is **not the natural minimum or maximum** of that metric

- This can be difficult to spot unless you are aware of how the data is being recorded (which means it's a particularly dangerous issue to watch out for!)



Mall Shopper Survey Results

Only tracks shoppers over the age of 18 due to legal reasons, so anyone under 18 is excluded (even though there are plenty of mall shoppers under 18)



Ecommerce Repeat Purchase Rate

Sharp drop as you approach the current date has nothing to do with customer behavior, but the fact that recent customers haven't have the opportunity or need to repurchase yet

# TABLE STRUCTURE

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

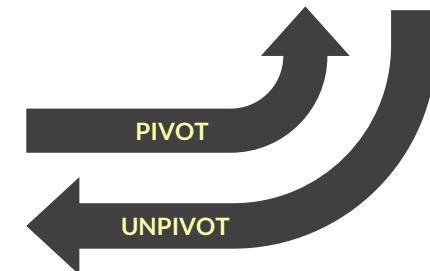
**Table structures** generally come in two flavors: **long** or **wide**

Long Table

Product Details				
ID	Size	Section	Price	Quantity
1	L	Camping	5	2000
2	L	Camping	10	1600
3	L	Camping	30	1100
4	L	Camping	100	500
5	L	Camping	200	20
6	L	Camping	15	1700
7	L	Camping	20	1600
8	L	Camping	25	1500
9	L	Biking	600	100
10	L	Biking	50	200
11	S	Camping	4	1000
12	S	Camping	8	1000
13	S	Camping	26	600
14	S	Camping	80	900
15	S	Camping	190	10
16	S	Camping	12	500
17	S	Biking	600	150
18	S	Biking	50	1000
19	S	Biking	20	300
20	S	Biking	30	700

Wide Table

Product Details				
Section	Large Price	Small Price	Large Quantity	Small Quantity
Camping	400	300	10	20
Biking	200	150	100	110



Pivoting is the process of adjusting a table from long to wide by transforming rows into columns, and Unpivoting is the opposite (wide to long)

# TABLE STRUCTURE

Variable Types

Empty Values

Range Calculations

Count Calculations

Left/Right Censored

Table Structure

**Long** tables typically contain a **single, distinct column** for each field (Date, Product, Category, Quantity, Profit, etc.)

- Easy to see all available fields and variable types
- Great for exploratory data analysis and aggregation (i.e. PivotTables)

**Wide** tables typically split the same metric into **multiple columns** or categories (i.e. 2018 Sales, 2019 Sales, 2020 Sales, etc.)

- Typically not ideal for human readability, since wide tables may contain thousands of columns (vs. only a handful if pivoted to a long format)
- Often (but not always) the best format for machine learning model input
- Great format for visualizing categorical data (i.e. sales by product category)



There's no **right** or **wrong** table structure; each type has strengths & weaknesses!

# CASE STUDY: PRELIMINARY QA



## THE SITUATION

You've just been hired as a Data Analyst for **Maven Market**, a local grocery store looking for help with basic data management and analysis.



## THE ASSIGNMENT

The store manager would like you to conduct some analyses on product inventory and sales, but the data is a mess.

You'll need to **explore the data, conduct a preliminary QA, and help clean it up** to prepare the data for further analysis.



## THE OBJECTIVES

1. Look at common data profiling metrics to identify potential issues
2. Take note of any issues you find in the product inventory sample
3. Correct the issues to prepare the data for further analysis

# BEST PRACTICES: PRELIMINARY QA

---



Review all fields to ensure that **variable types** are configured for proper analysis (i.e. no dates formatted as strings, text formatted as values, etc.)



Remember that **NA** and **0** do not mean the same thing! Think carefully about how to handle missing data and the impact it may have on your analysis



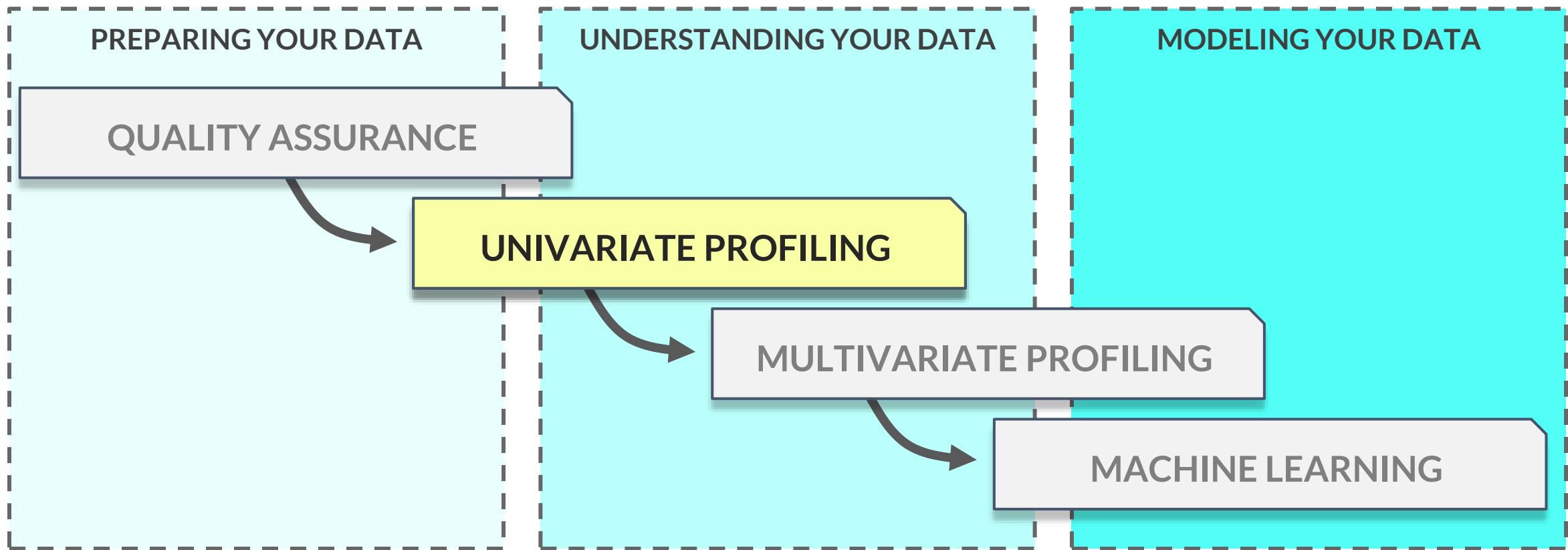
Run basic diagnostics like **Range**, **Count**, and **Left/Right Censored** checks against all columns in your data set...every time



Understand your **table structure** before conducting any analysis to reduce the risk of double counting, inaccurate calculations, omitted data, etc.

# UNIVARIATE PROFILING

# MACHINE LEARNING PROCESS



Univariate profiling is about **exploring individual variables** to build an understanding of your data. We'll cover common topics like normal distributions, frequency tables, histograms, etc.

# UNIVARIATE PROFILING



**Univariate profiling** is the next step after preliminary QA; think of univariate profiling as conducting a descriptive analysis of each variable *by itself*

## TOPICS WE'LL COVER:

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

## COMMON USE CASES:

- Developing a deeper understanding of the fields you're working with (*outliers, distribution, etc.*)
- Preparing to build ML models (*necessary for selecting the correct model type*)
- Exploring individual variables before conducting a deeper multivariate analysis

# CATEGORICAL VARIABLES

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

## VARIABLE TYPES

CATEGORICAL

NUMERICAL

**Categorical** variables contain categories as values, instead of numbers (i.e. Product Type, Customer Name, Country, Month, etc.)

- Categorical fields are exceptionally important for data analysis, and are typically used as dimensions by which we **filter** or “**cut**” numerical values (i.e. sales by store)
- Many types of predictive models are used to predict categorical dependent variables, or “classes” (more on that later!)



Terms like **discrete**, **categorical**, **multinomial**, and **classes** may all be used interchangeably. **Binary** is a special type of categorical variable which takes only 1 of 2 cases: *true* for *false* (or 1 or 0) and is also known as a **logical** variable or a **binary flag**

# DISCRETIZATION

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Discretization** is the process of creating a new *categorical* variable from an existing *numerical* variable, based on the values of the numerical variable

Product Details					
ID	Size	Section	Price	Quantity	Discretized Price
1	L	Camping	5	2000	Low
2	L	Camping	10	1600	Low
3	L	Camping	30	1100	Low
4	L	Camping	100	500	Med
5	L	Camping	200	20	Med
6	L	Camping	15	1700	Low
7	L	Camping	20	1600	Low
8	L	Camping	25	1500	Low
9	L	Biking	600	100	High
10	S	Biking	50	200	Low
11	S	Camping	4	1000	Low
12	S	Camping	8	1000	Low
13	S	Camping	26	600	Low
14	S	Camping	80	900	Low
15	S	Camping	190	10	Med
16	S	Camping	12	500	Low
17	S	Biking	600	150	High
18	S	Biking	50	1000	Low
19	S	Biking	20	300	Low
20	S	Biking	30	700	Low

Discretization Rules:

If Price <100 then Price Level = Low

If Price >=100 & Price <500 then Price Level = Med

If Price >=500 then Price Level = High

# NOMINAL VS. ORDINAL VARIABLES

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

## VARIABLE TYPES

CATEGORICAL

NUMERICAL

NOMINAL

ORDINAL

There are two types of categorical variables: **nominal** and **ordinal**

- **Nominal variables** contain categories with no inherent logical rank, which can be re-ordered with no consequence (i.e. *Product Type* = Camping, Biking or Hiking)
- **Ordinal variables** contain categories with a logical order (i.e. *Size* = Small, Medium, Large), but the interval between those categories has no logical interpretation

# CATEGORICAL DISTRIBUTIONS

Categorical Variables

Categorical  
Distributions

Numerical Variables

Histograms &  
Kernel Densities

Normal Distribution

Data Profiling

**Categorical distributions** are visual and/or numeric representations of the unique values a variable contains, and how often each occurs

Common categorical distributions include:

- **Frequency tables:** Show the count (or frequency) of each distinct value
- **Proportions tables:** Show the count of each value as a % of the total
- **Heat maps:** Formatted to visualize patterns (*typically used for multiple variables*)

Understanding categorical distributions will help us gather knowledge for building accurate machine learning models (*more on this later!*)

# CATEGORICAL DISTRIBUTIONS

Categorical Variables

Categorical Distributions

Numerical Variables

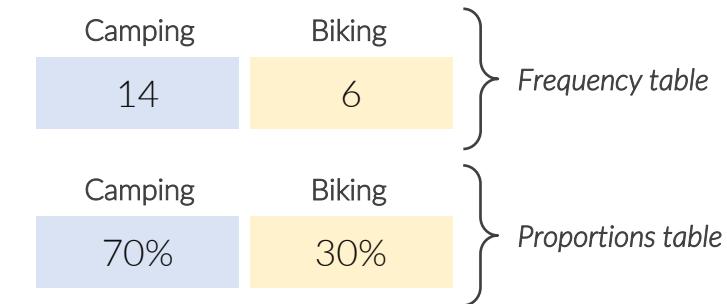
Histograms & Kernel Densities

Normal Distribution

Data Profiling

Product Details				
ID	Size	Section	Price	Quantity
1	L	Camping	5	2000
2	L	Camping	10	1600
3	L	Camping	30	1100
4	L	Camping	100	500
5	L	Camping	200	20
6	L	Camping	15	1700
7	L	Camping	20	1600
8	L	Camping	25	1500
9	L	Biking	600	100
10	L	Biking	50	200
11	S	Camping	4	1000
12	S	Camping	8	1000
13	S	Camping	26	600
14	S	Camping	80	900
15	S	Camping	190	10
16	S	Camping	12	500
17	S	Biking	600	150
18	S	Biking	50	1000
19	S	Biking	20	300
20	S	Biking	30	700

Section Distribution:



Size & Section Distribution:



# NUMERICAL VARIABLES

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

## VARIABLE TYPES

CATEGORICAL

NUMERICAL

**Numerical** variables contain numbers as values, instead of categories (i.e. Gross Revenue, Pageviews, Quantity, Retail Price, etc.)

- Numerical fields are typically **aggregated** (as a sum, count, average, max, min, etc.) and broken down by different dimensions or categories
- Numerical data is also known as **quantitative** data, while categorical data is often referred to as “qualitative”



You may hear numeric variables described further as **interval** and **ratio**, but the distinction is trivial and rarely makes a difference in common use cases

# HISTOGRAMS

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

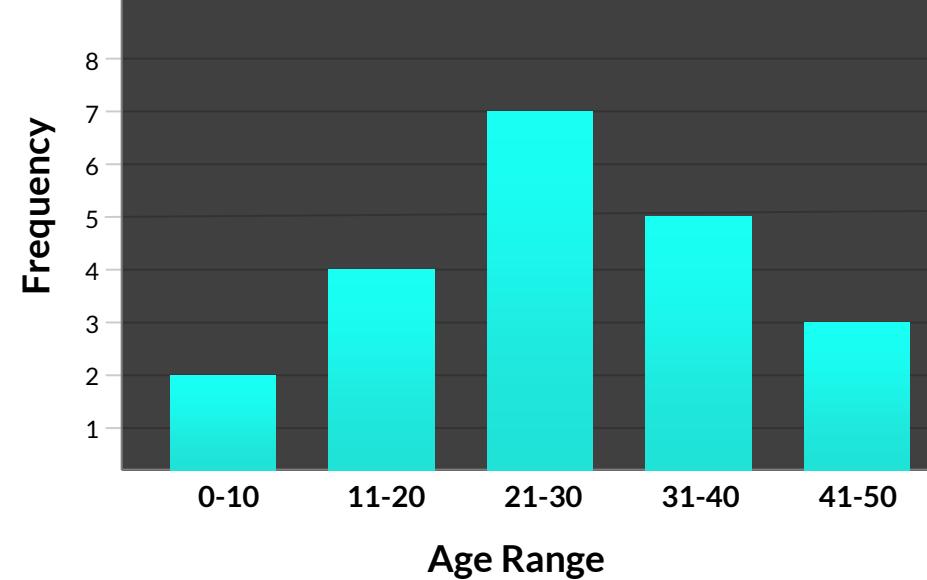
Data Profiling

**Histograms** are used to plot a single, discretized numerical variable

Imagine taking a numerical variable (*like age*), defining ranges or “bins” (1-5, 6-10, etc.), and counting the number of observations which fall into each bin; ***this is exactly what histograms are designed to do!***

Age Values:

8	29	45
25	33	37
19	43	21
28	32	40
24	17	28
5	22	39
15	47	12



# KERNEL DENSITIES

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

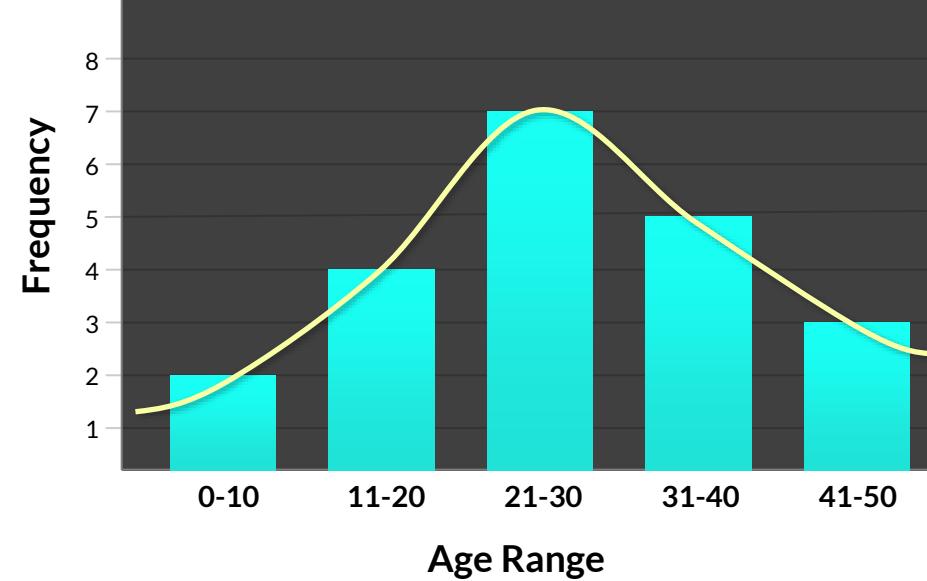
Data Profiling

**Kernel densities** are “smooth” versions of histograms, which can help to prevent users from over-interpreting breaks between bins

- **Technical definition:** “Non-parametric density estimation via smoothing”
- **Intuitive definition:** “Wet noodle laying on a histogram”

Age Values:

8	29	45
25	33	37
19	43	21
28	32	40
24	17	28
5	22	39
15	47	12



# HISTOGRAMS & KERNEL DENSITIES

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

**When to use** histograms and kernel density charts:

- Visualizing how a given variable is distributed
- Providing a visual glimpse of profiling metrics like mean, mode, and skewness

**Things to watch out for:**

- **Bin sensitivity:** Bin size can significantly change the shape and "smoothness" of a histogram, so select a bin width that accurately shows the data distribution
- **Outliers:** Histograms can be used to identify outliers in your data set, but you may need to remove them to avoid skewing the distribution
- **Sample size:** Histograms are best suited for variables with many observations, to reflect the true population distribution



**PRO TIP:** If your data is relatively symmetrical (not skewed), you can use **Sturge's Rule** as a quick "rule of thumb" to determine an appropriate number of bins:  $K = 1 + 3.322 \log(N)$  (where **K** = number of bins, **N** = number of observations)

# CASE STUDY: HISTOGRAMS



## THE SITUATION

You've just been promoted as the new Pit Boss at **The Lucky Roll Casino**. Your mission? Use data to help expose cheats on the casino floor.



## THE ASSIGNMENT

Profits at the craps tables have been unusually low, and you've been asked to investigate the possible use of loaded die (weighted towards specific numbers). Your plan is to **track the outcome of each roll**, then **compare your results against the expected probability distribution** to see how closely they match.



## THE OBJECTIVES

1. Record the outcomes for a series of individual dice rolls
2. Plot the frequency of each result (1-12) using a histogram
3. Compare your plot against the expected frequency distribution

# NORMAL DISTRIBUTION

Categorical Variables

Categorical Distributions

Numerical Variables

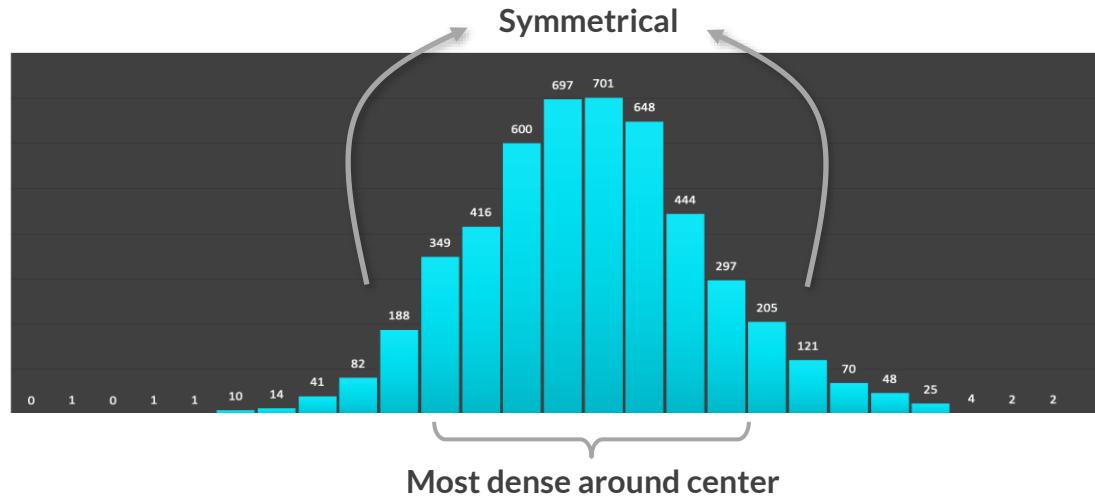
Histograms & Kernel Densities

Normal Distribution

Data Profiling

Many numerical variables naturally follow a **normal distribution**, also known as a Gaussian distribution or “bell curve”

- Normal distributions are symmetrical and dense around the center, with flared out “tails” on both ends (*like a bell!*)
- Normal distributions are essential to many underlying assumptions in ML, and a helpful tool for comparing distributions or testing differences between them



You can find normal distributions in many real-world examples: heights, weights, test scores, etc.

# NORMAL DISTRIBUTION

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

$$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Parabola centered around the mean

Turn the parabola upside down

Make the tails flare out

The diagram illustrates the components of the normal distribution formula. The formula is shown as a fraction with the numerator being 1 and the denominator being  $\sigma\sqrt{2\pi}$ . The numerator is highlighted in purple. The denominator is split into two parts:  $e^{-\frac{1}{2}}$  (highlighted in yellow) and  $(x-\mu)^2 / \sigma^2$  (highlighted in teal). A blue arrow points from the teal box to the text "Parabola centered around the mean". A yellow arrow points from the yellow box to the text "Turn the parabola upside down". A purple arrow points from the purple box to the text "Make the tails flare out".

# CASE STUDY: NORMAL DISTRIBUTION



## THE SITUATION

It's August 2016, and you've been invited to Rio de Janeiro as a Data Analyst for the **Global Olympic Committee**.



## THE ASSIGNMENT

Your job is to collect demographic data for all female athletes competing in the Summer Games and determine how the **distribution of Olympic athlete heights compares against the general public**.



## THE OBJECTIVES

1. Gather heights for all female athletes competing in the 2016 Games
2. Plot height frequencies using a Histogram, and test various bin widths
3. Determine if athlete heights follow a normal distribution, or “bell curve”
4. Compare the distributions for athletes vs. the general public

# DATA PROFILING

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Data profiling** describes the process of using statistics to describe or summarize information about a particular variable

- Data profiling is critical for understanding variable characteristics which cannot be seen or easily visualized using tools like histograms
- Data profiling communicates **meaning**, using simple, concise, and universally understood metrics

Common data profiling metrics include:

- **Mode**
- **Mean**
- **Median**
- **Percentile**
- **Variance**
- **Standard Deviation**
- **Skewness**



**PRO TIP:** Combining visual distributions with data profiling metrics is a powerful way to communicate meaning in advanced analytics

# MODE

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

The **mode** is the most frequently observed value

- With a *numerical* variable (most common), it's the general range around the highest "peak" in the distribution
- With a *categorical* variable, it's simply the value that appears most often

Customer ID	First Name	Last Name	Gender	City	Sessions
1	Jennifer	Mcgrath	F	Seattle	24
2	Chad	Lewis	M	Seattle	12
3	Susan	Rodriguez	F	Houston	9
4	Wayne	Nielson	M	Miami	31
5	John	Depaul	M	Houston	24
6	Joseph	Martinez	M	Houston	11
7	Diane	Henry	F	Miami	3
8	Veronica	Comerford	F	Miami	24
9	Beverly	Nixon	F	Houston	15
10	Ivan	Layton	M	Houston	27

Mode of **City** = “Houston”

Mode of **Sessions** = 24

Mode of **Gender** = F, M  
(this is a **bimodal** field!)

Common uses:

- Understanding the most common values within a dataset
- Diagnosing if one variable is influenced by another

# MODE

Categorical Variables

Categorical Distributions

Numerical Variables

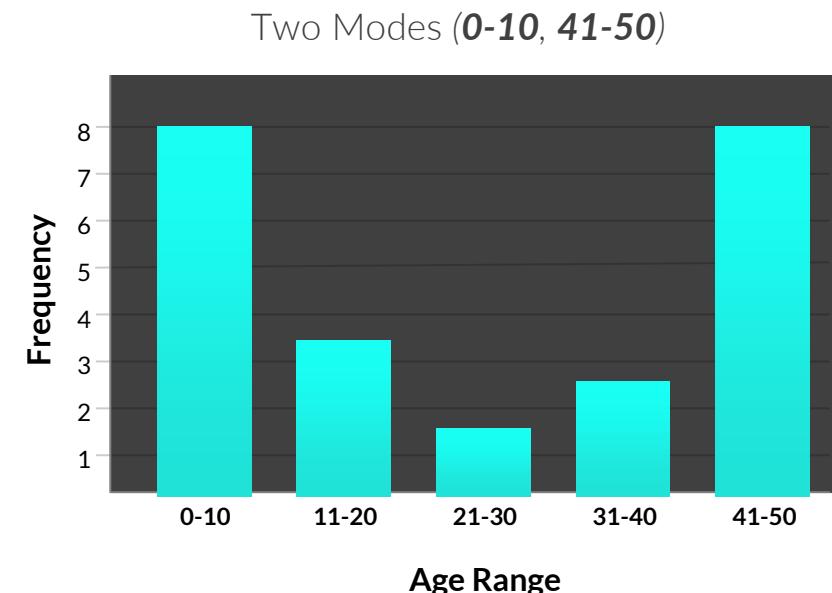
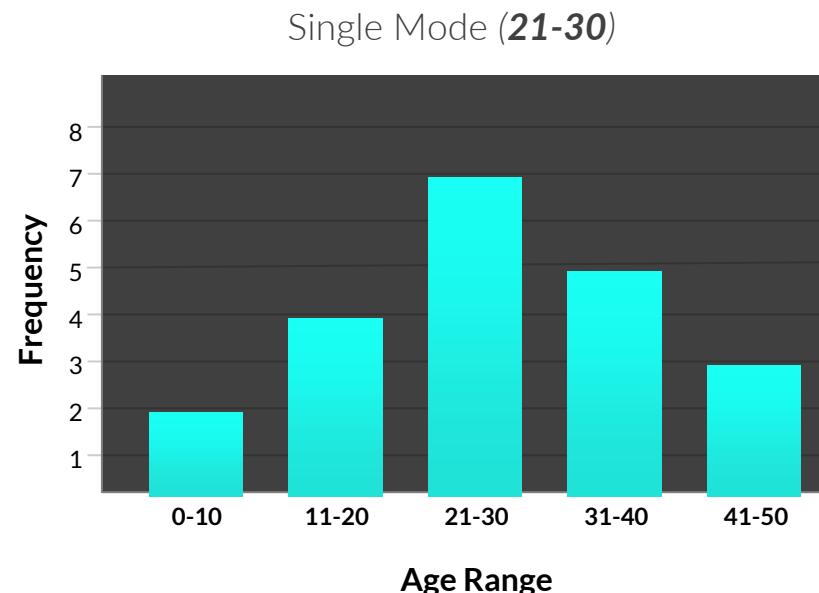
Histograms & Kernel Densities

Normal Distribution

Data Profiling

While **modes** typically aren't very useful on their own, they can provide helpful hints for deeper data exploration

- For example, the right histogram below shows a *multi-modal* distribution, which indicates that there may be another variable impacting the age distribution



# MEAN

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

The **mean** is the calculated “central” value in a discrete set on numbers

- Mean is what most people think of when they hear the word “average”, and is calculated by dividing the **sum of all values** by the **count of all observations**
- Means can only be applied to *numerical* variables (not categorical)

Product Details				
ID	Size	Section	Price	Quantity
1	L	Camping	5	2000
2	L	Camping	10	1600
3	L	Camping	30	1100
4	L	Camping	100	500
5	L	Camping	200	20

$$\left. \begin{aligned} mean &= \frac{\text{sum of all values}}{\text{count of observations}} \\ &= \frac{5,220}{5} = 1,044 \end{aligned} \right\}$$

Common uses:

- Making a “best-guess” estimate of a value
- Calculating a central value when outliers are not present

# MEDIAN

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

The **median** is the *middle value* in a list of values sorted from highest to lowest (or vice versa)

- When there are two middle-ranked values, the median is the *average* of the two
- Medians can only be applied to *numerical* variables (not categorical)

Customer ID	First Name	Last Name	Sessions
4	Wayne	Nielson	31
10	Ivan	Layton	27
1	Jennifer	Mcgrath	24
5	John	Depaul	24
8	Veronica	Comerford	24
9	Beverly	Nixon	15
2	Chad	Lewis	12
6	Joseph	Martinez	11
3	Susan	Rodriguez	9
7	Diane	Henry	3

} Median = **19.5**  
(average of 15 and 24)

## Common uses:

- Identifying the “center” of a distribution
- Calculating a central value when outliers may be present

# PERCENTILE

Categorical Variables

Categorical Distributions

Numerical Variables

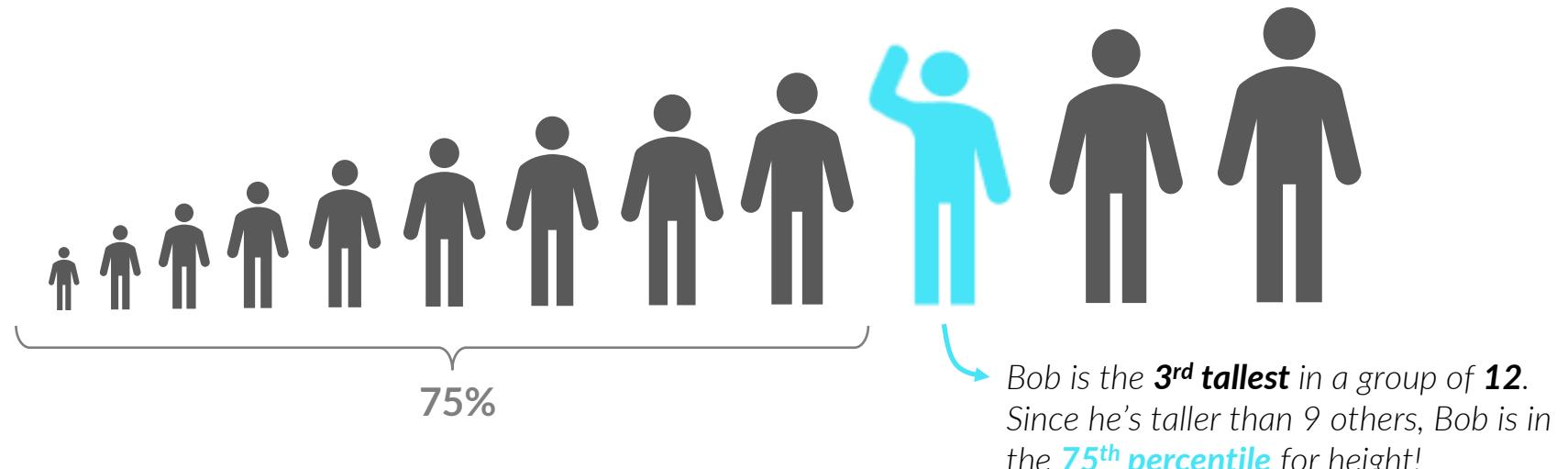
Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Percentiles** are used to describe the *percent of values* in a column which fall below a particular number

- If you are in the **90<sup>th</sup>** percentile for height, you're taller than **90%** of the population
- If the **50<sup>th</sup>** percentile for test scores is **86**, half of the students scored lower (which means **86** is also the *median*!)



## Common uses:

- Providing context and intuitive benchmarks for how values “rank” within a sample (test scores, height/weight, blood pressure, etc.)

# VARIANCE

Categorical Variables

Categorical Distributions

Numerical Variables

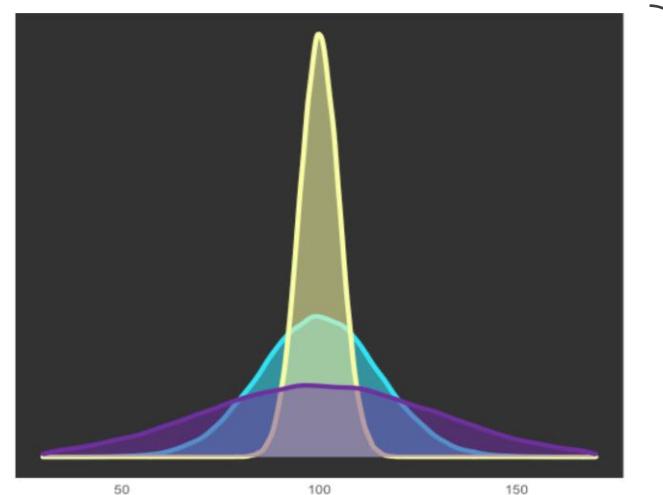
Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Variance**, in simple terms, describes how *thin* or *wide* a distribution is

- Variance measures how far the observations are from the mean, on average, and help us concisely describe a variable's distribution
- The *wider* a distribution, the *higher* the variance (and vice versa)



Variance = 5

Variance = 15

Variance = 30

**Common uses:**

- Comparing the numerical distributions of two different groups (i.e. prices of products ordered online vs. in store)

# VARIANCE

Categorical Variables

Categorical Distributions

Numerical Variables

Histograms & Kernel Densities

Normal Distribution

Data Profiling

$$\frac{\sum_{i=1}^n (x_i - \mu)^2}{n - 1}$$

## Calculation Steps:

- 1) Calculate the average of the variable
- 2) Subtract that average from the first row
- 3) Square that difference
- 4) Do steps 1-3 for every row and sum it up
- 5) Divide by the number of observations (-1)

*Average squared distance from the mean*

# STANDARD DEVIATION

Categorical Variables

Categorical Distributions

Numerical Variables

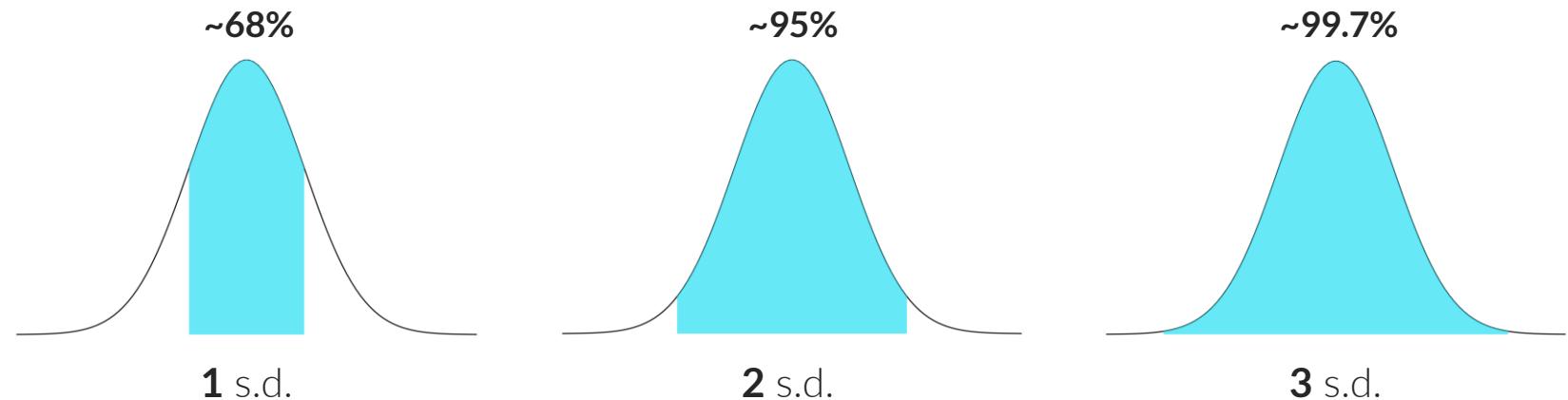
Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Standard Deviation** is the square root of the variance

- This converts the variance back to the scale of the variable itself
- In a normal distribution, **~68%** of values fall within 1 standard deviation of the mean, **~95%** fall within 2, and **~99.7%** fall within 3 (known as the “**empirical rule**”)



**Common uses:**

- Comparing segments for a given metric (i.e. time on site for mobile users vs. desktop)
- Understanding how likely certain values are bound to occur

# SKEWNESS

Categorical Variables

Categorical Distributions

Numerical Variables

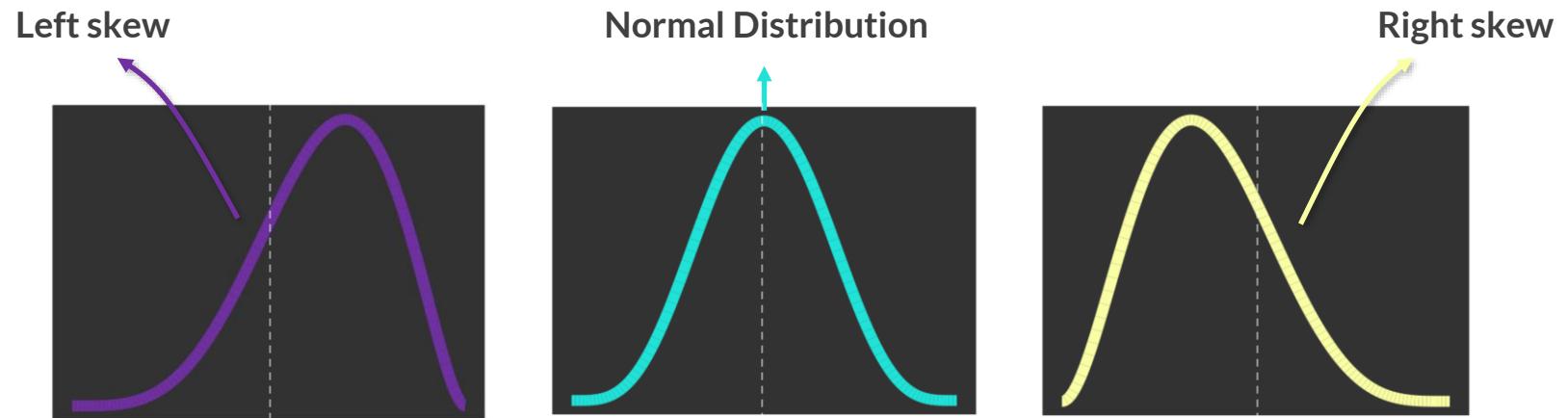
Histograms & Kernel Densities

Normal Distribution

Data Profiling

**Skewness** tells us how a distribution varies from a normal distribution

- This is commonly used to mathematically describe skew to the left or right



**Common uses:**

- Identifying non-normal distributions, and describing them mathematically

# BEST PRACTICES: UNIVARIATE PROFILING

---



Make sure you are using the appropriate tools for profiling **categorical** variables vs. **numerical** variables



**Distributions** are a great way to quickly and visually explore variables



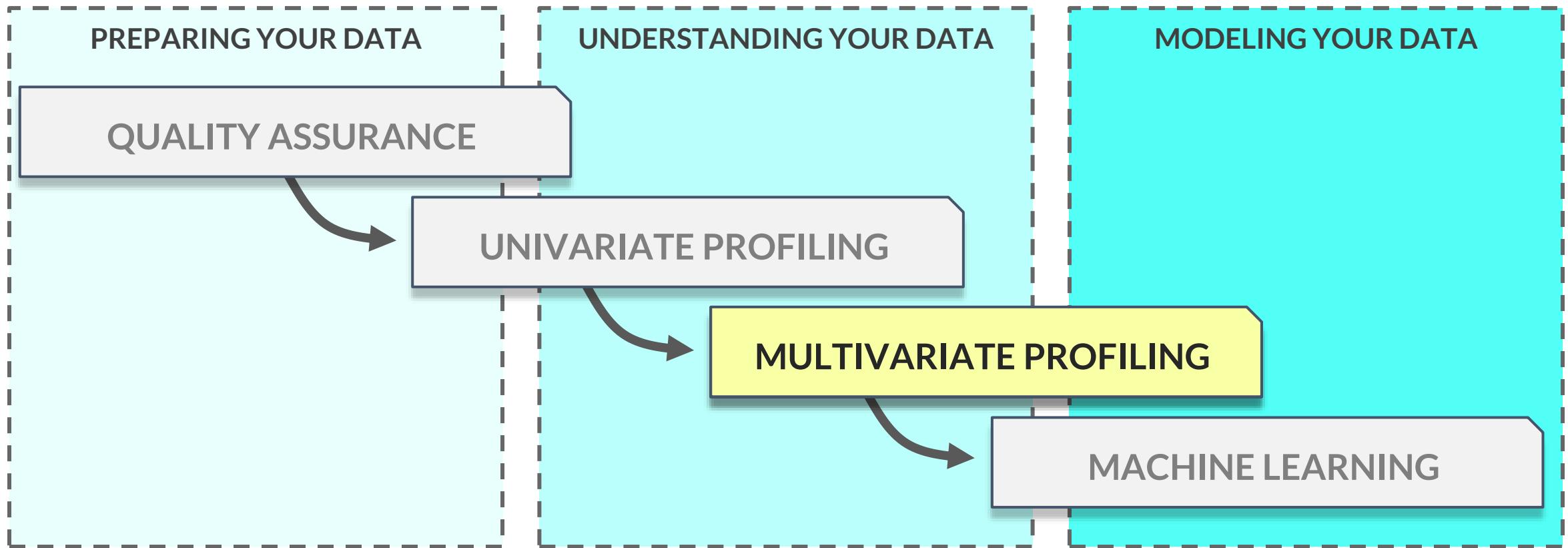
**QA still comes first!** Profiling metrics are important, but can lead to misleading results without proper QA (i.e. handling outliers or missing values)



One single distribution, visualization, or metric isn't enough to fully understand a variable; **always explore your data from multiple angles**

# MULTIVARIATE DISTRIBUTIONS

# MACHINE LEARNING PROCESS



Multivariate profiling is about **understanding relationships between *multiple* variables**. We'll cover common tools for exploring categorical & numerical data, including kernel densities, violin & box plots, scatterplots, etc.

# MULTIVARIATE PROFILING



**Multivariate profiling** is the next step after univariate profiling, since single-metric distributions are rarely enough to draw meaningful insights or conclusions

## TOPICS WE'LL COVER:

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

## COMMON USE CASES:

- Exploring relationships between multiple categorical and/or numerical variables
- Informing which specific machine learning models or techniques to use



Multivariate distributions are generally called “**joint distributions**”, and it’s clear why—it’s joining two variables into the same distribution!

# CATEGORICAL-CATEGORICAL DISTRIBUTIONS

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

**Categorical/categorical distributions** represent the frequency of unique combinations between two or more categorical variables

This is one of the simplest forms of multivariate profiling, and leverages the same tools we used to analyze univariate distributions:

- **Frequency tables:** Show the count (or frequency) of each distinct combination
- **Proportions tables:** Show the count of each combination as a % of the total
- **Heat maps:** Frequency or proportions table formatted to visualize patterns

## Common uses:

- Understanding product mix based on multiple characteristics (i.e. size & category)
- Exploring customer demographics based on attributes like location, gender, etc.

# CATEGORICAL-CATEGORICAL DISTRIBUTIONS

Categorical-Categorical Distributions

Categorical-Numerical Distributions

Multivariate Kernel Densities

Violin & Box Plots

Numerical-Numerical Distributions

Scatter Plots & Correlation

In this example we're looking at product inventory using a distribution of two categorical fields: **Design** and **Size**

We can show this joint distribution as a simple count (*Frequency Table*), as a percentage of the total (*Proportions Table*), or as a conditionally formatted table (*Heat Map*)

Product Inventory					
	Design 1	Design 2	Design 3	Design 4	Design 5
X-Small	2	31	44	48	3
Small	5	33	39	33	19
Medium	5	48	28	36	37
Large	16	27	6	38	42
X-Large	35	28	1	31	3

Frequency Table

Product Inventory					
	Design 1	Design 2	Design 3	Design 4	Design 5
X-Small	0.3%	4.9%	6.9%	7.5%	0.5%
Small	0.8%	5.2%	6.1%	5.2%	3.0%
Medium	0.8%	7.5%	4.4%	5.6%	5.8%
Large	2.5%	4.2%	0.9%	6.0%	6.6%
X-Large	5.5%	4.4%	0.2%	4.9%	0.5%

Proportions Table

Product Inventory					
	Design 1	Design 2	Design 3	Design 4	Design 5
X-Small	2	31	44	48	3
Small	5	33	39	33	19
Medium	5	48	28	36	37
Large	16	27	6	38	42
X-Large	35	28	1	31	3

Heat Map

# CASE STUDY: HEAT MAPS



## THE SITUATION

You've just been hired by the **New York Department of Transportation** (DOT) to help analyze traffic accidents in New York City from 2019-2020



## THE ASSIGNMENT

The DOT commissioner would like to understand accident frequency by **time of day** and **day of week**, in order to support a public service campaign promoting safe driving habits.

Your role is to provide the data that she needs to understand **when traffic accidents are most likely to occur**.



## THE OBJECTIVES

1. Create a table to plot accident frequency by time of day and day of week
2. Apply conditional formatting to the table to create a heatmap showing the days and times with the fewest (green) and most (red) accidents in the sample

# CATEGORICAL-NUMERICAL DISTRIBUTIONS

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

**Categorical-numerical distributions** are used for comparing numerical distributions across classes in a category (i.e. *age distribution by gender*)

These are typically visualized using variations of familiar univariate numerical distributions, including:

- **Histograms & Kernel Densities:** Show the count (or frequency) of values
- **Violin Plots:** Kernel density “glued” to its mirror image, and tilted on its side
- **Box Plots:** Like a kernel density, but formatted to visualize key statistical values (min/max, median, quartiles) and outliers

## Common uses:

- Comparing key business metrics (i.e. *customer lifetime value, average order size, purchase frequency, etc.*) by customer class (gender, loyalty status, location, etc.)
- Comparing sales performance by day of week or hour of day

# KERNEL DENSITIES

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

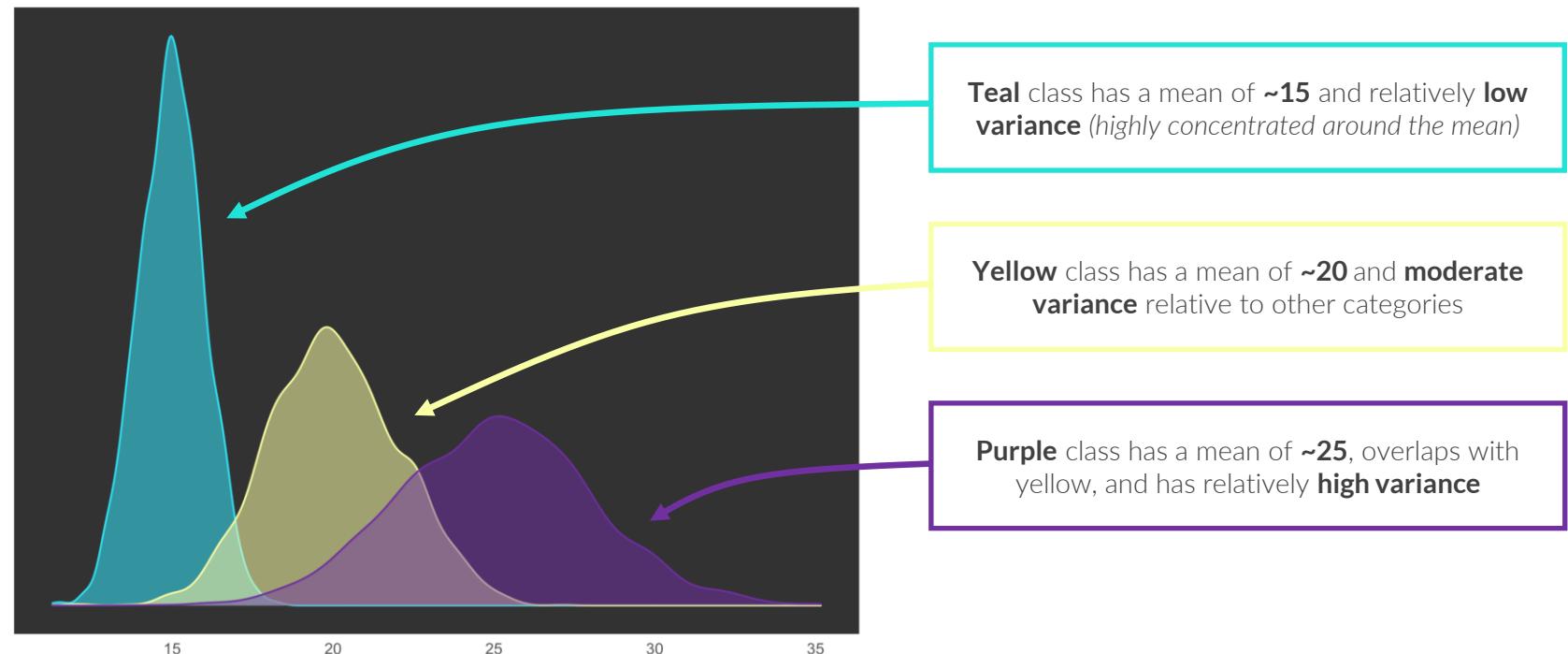
Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

Remember: **kernel densities** are just smooth versions of histograms

- To visualize a categorical-numerical distribution, kernel densities can be repeated to represent each class within a particular category



# VIOLIN PLOTS

Categorical-Categorical Distributions

Categorical-Numerical Distributions

Multivariate Kernel Densities

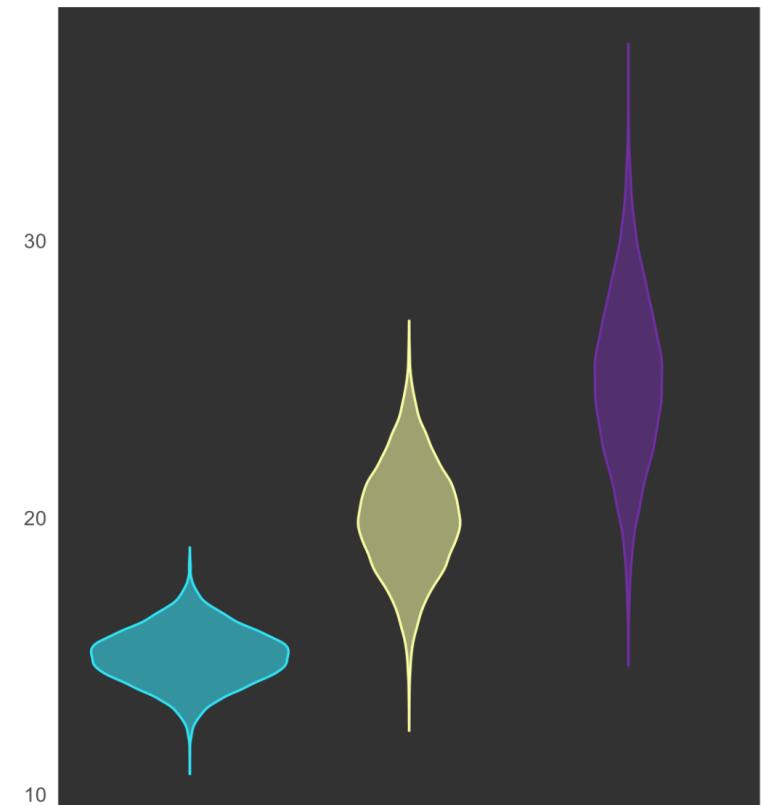
Violin & Box Plots

Numerical-Numerical Distributions

Scatter Plots & Correlation

A **violin plot** is essentially a kernel density flipped vertically and combined with its mirror image

- Violin plots use the exact same data as kernel densities, just visualized slightly differently
- These can help you visualize the shape of each distribution, and compare them across classes more clearly



# BOX PLOTS

Categorical-Categorical Distributions

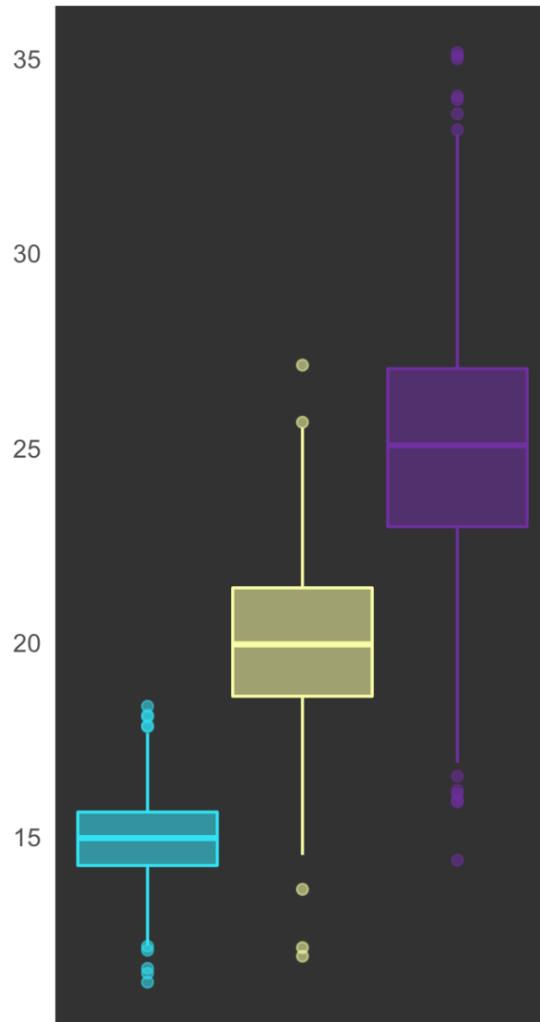
Categorical-Numerical Distributions

Multivariate Kernel Densities

Violin & Box Plots

Numerical-Numerical Distributions

Scatter Plots & Correlation



**Box plots** are like violin plots, but designed to show **key statistical attributes** rather than smooth distributions, including:

- **Median**
- **Min & Max** (*excluding outliers*)
- **25<sup>th</sup> & 75<sup>th</sup> Percentiles**
- **Outliers**

Box plots provide a ton of information in a single visual, and can be used to quickly compare statistical characteristics between classes

# LIMITATIONS OF CATEGORICAL DISTRIBUTIONS

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

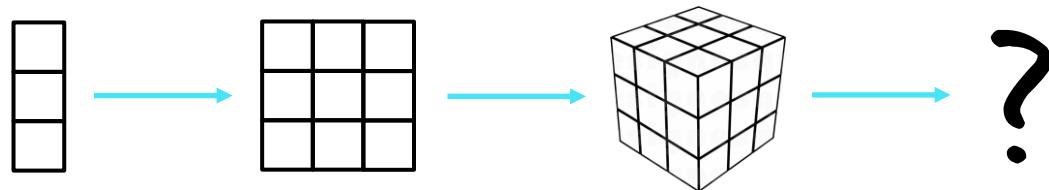
Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

Categorical profiling works for simple cases, but breaks down quickly

- Humans are pretty good at visualizing 1, 2, or maybe even 3 variables, but how would you visualize a joint distribution for **10** variables? **100**?



Categorical profiling can't answer **prescriptive** or **predictive** questions

- Suppose you randomized several elements on your sales page (font, image, layout, button, copy, etc.) to understand which ones drive conversions
- You could count conversions for individual elements, or some combinations of elements, but categorical distribution alone can't measure *causation*



***This is when you need machine learning!***

# NUMERICAL-NUMERICAL DISTRIBUTIONS

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

**Numerical-Numerical distributions** are common and intuitive, but the most complex mathematically

They are typically visualized using **scatter plots**, which plot points along the X and Y axis to show the relationship between two variables

- Scatter plots allow for simple, visual intuition: *when one variable increases or decreases, how does the other variable change?*
- There are many possibilities: no relationship, positive, negative, linear, non-linear, cubic, exponential, etc.

## Common uses:

- Quickly visualizing how two numerical variables relate
- Predicting how a change in one variable will impact another (*i.e. square footage and house price, marketing spend and sales, etc.*)

# CORRELATION

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

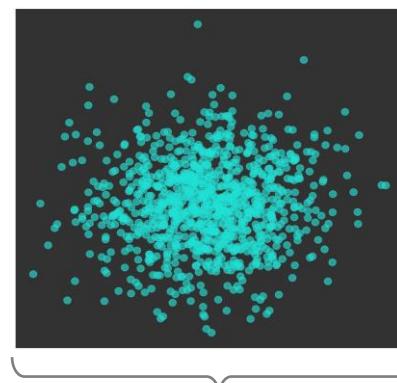
Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

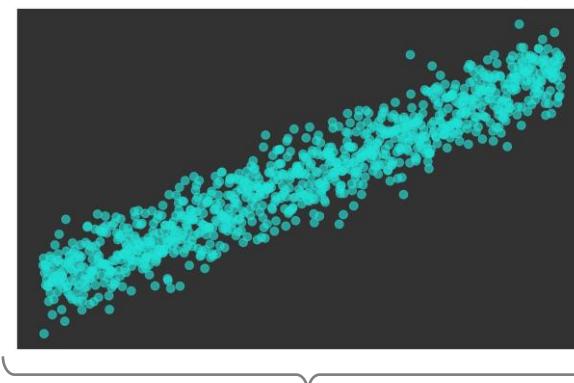
Univariate profiling metrics aren't much help in the multivariate world; we now need a way to describe *relationships* between variables

**Correlation** is the most common multivariate profiling metric, and is used to describe how a pair of variables are **linearly related**

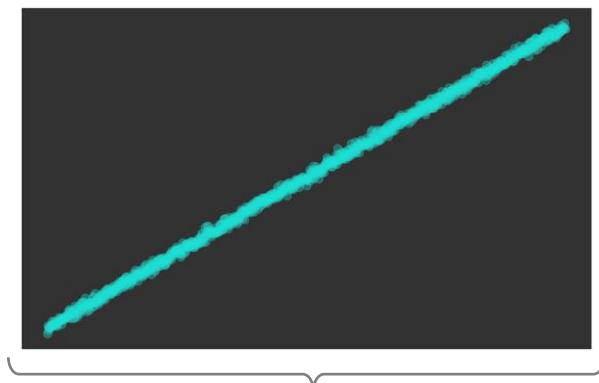
- In other words: for a given row, when one variable's observation goes above its mean, does the other variable's observation also go above its mean (and vice versa)?



No correlation



Positive correlation



Strong positive correlation

# CORRELATION

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

Correlation is an **extension of variance**

- Think of correlation as a way to measure the **variance** of both variables at one time (called “*co-variance*”), while controlling for the scales of each variable

**Variance** formula

$$\frac{\sum_{i=1}^n (x_i - \mu)^2}{n - 1}$$

**Correlation** formula

$$\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{(n - 1)S_x S_y}$$

- Here we multiply **variable X's difference from its mean** with **variable Y's difference from its mean**, instead of squaring a single variable (like we do with variance)
- $S_x$  and  $S_y$  are the **standard deviations** of X and Y, which puts them on the same scale

# CORRELATION VS. CAUSATION

Categorical-Categorical  
Distributions

Categorical-Numerical  
Distributions

Multivariate Kernel  
Densities

Violin & Box Plots

Numerical-Numerical  
Distributions

Scatter Plots &  
Correlation

## CORRELATION

 DOES **NOT** IMPLY 

## CAUSATION

# CORRELATION VS. CAUSATION

Categorical-Categorical Distributions

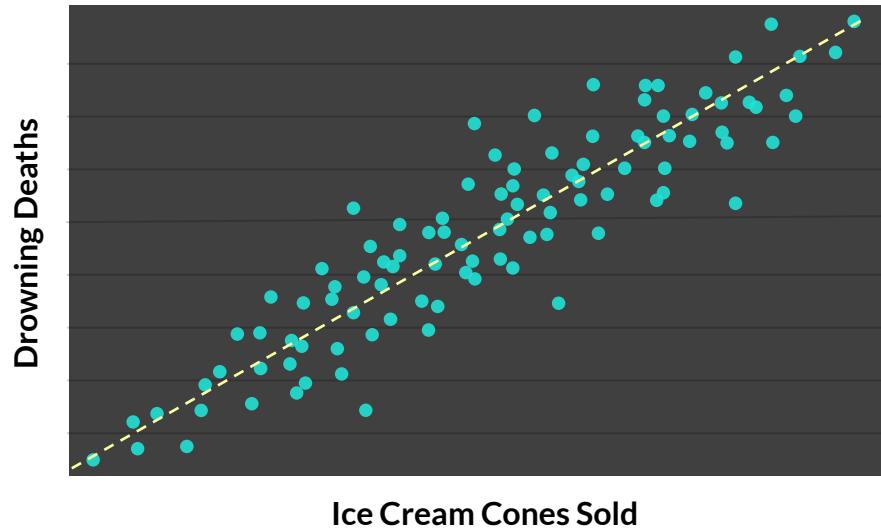
Categorical-Numerical Distributions

Multivariate Kernel Densities

Violin & Box Plots

Numerical-Numerical Distributions

Scatter Plots & Correlation



Consider the scatter plot above, showing daily **ice cream sales** and **drowning deaths** in a popular New England vacation town

- These two variables are clearly **correlated**, but do ice cream cones CAUSE people to drown? Do drowning deaths CAUSE a surge in ice cream sales?

***Of course not, because correlation does NOT imply causation!***

*So what do you think is really going on here?*

# PRO TIP: VISUALIZING A THIRD DIMENSION

Categorical-Categorical Distributions

Categorical-Numerical Distributions

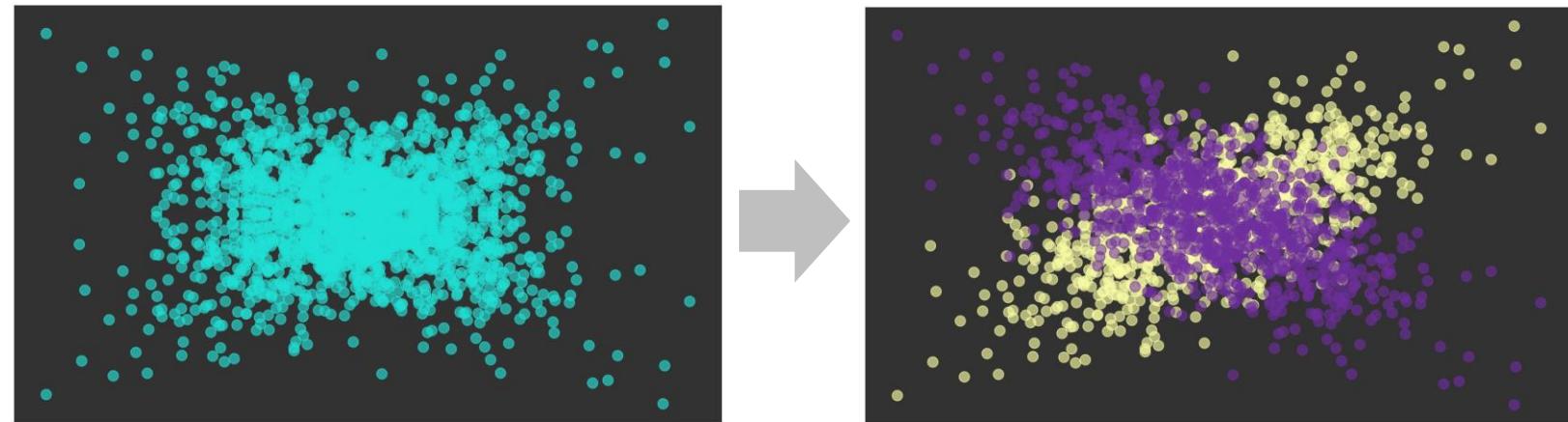
Multivariate Kernel Densities

Violin & Box Plots

Numerical-Numerical Distributions

Scatter Plots & Correlation

Scatter plots show two dimensions by default (X and Y), but using **symbols** or **color** allows you to visualize additional variables and expose otherwise hidden patterns or trends



Visualizing more than 3-4 dimensions is beyond human capability.  
**This is where you need machine learning!**

# CASE STUDY: CORRELATION



## THE SITUATION

You've just landed your dream job as a Marketing Analyst at **Loud & Clear**, the hottest ad agency in San Diego.



## THE ASSIGNMENT

Your client would like to understand the **impact of their digital media spend**, and how it relates to website traffic, offline spend, site load time, and sales.

Your role is to **collect and visualize these metrics** at the weekly-level in order to begin exploring the relationships between them.



## THE OBJECTIVES

1. Gather weekly spend, traffic, website, and sales data
2. Create a scatterplot to visualize any two given variables
3. Compare the relationship between Digital Media Spend and each variable
4. What patterns do you see, and how would you interpret them?

# BEST PRACTICES: MULTIVARIATE PROFILING

---



Univariate profiling is a great start, but multivariate profiling is necessary when working with **more than one variable** (pretty much *always*)



Understand the **types of variables** you're working with (categorical vs. numerical) to determine which profiling and visualization techniques to use



Use categorical variables to **filter or “cut”** your data and quickly compare profiling metrics or distributions across classes



Remember that **correlation does not imply causation**, and that variables can be related without one causing a change in the other

PART 2:

# CLASSIFICATION



# ABOUT THIS SERIES

This is **Part 2** of a **4-Part series** designed to help you build a deep, foundational understanding of machine learning, including data QA & profiling, classification, forecasting and unsupervised learning



## PART 1

QA & Data Profiling



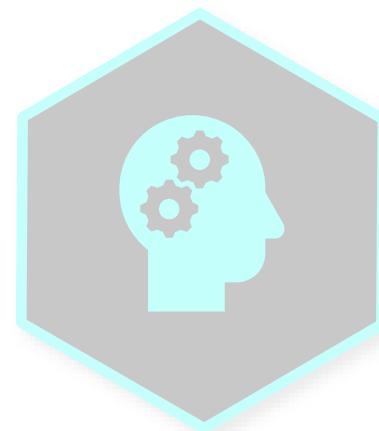
## PART 2

Classification



## PART 3

Regression & Forecasting



## PART 4

Unsupervised Learning

# COURSE OUTLINE

---

1

## Intro to Classification

Machine Learning landscape & key concepts, classification process, feature engineering, data splitting, overfitting, etc.

2

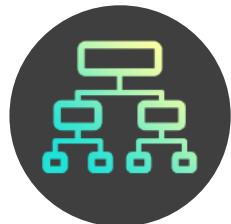
## Classification Models



K-Nearest  
Neighbors



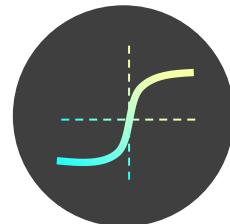
Naïve  
Bayes



Decision  
Trees



Random  
Forests



Logistic  
Regression



Sentiment  
Analysis

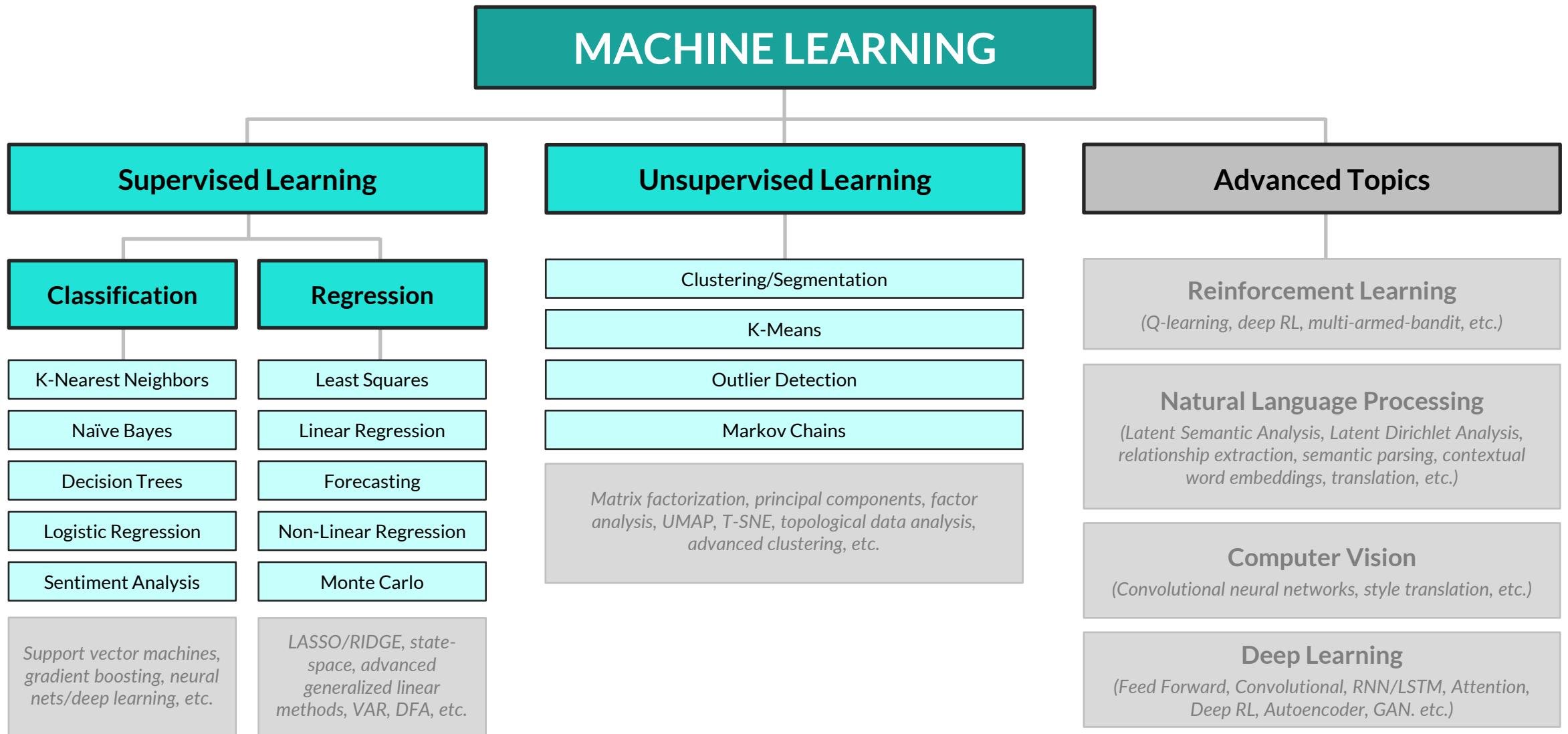
3

## Model Selection & Tuning

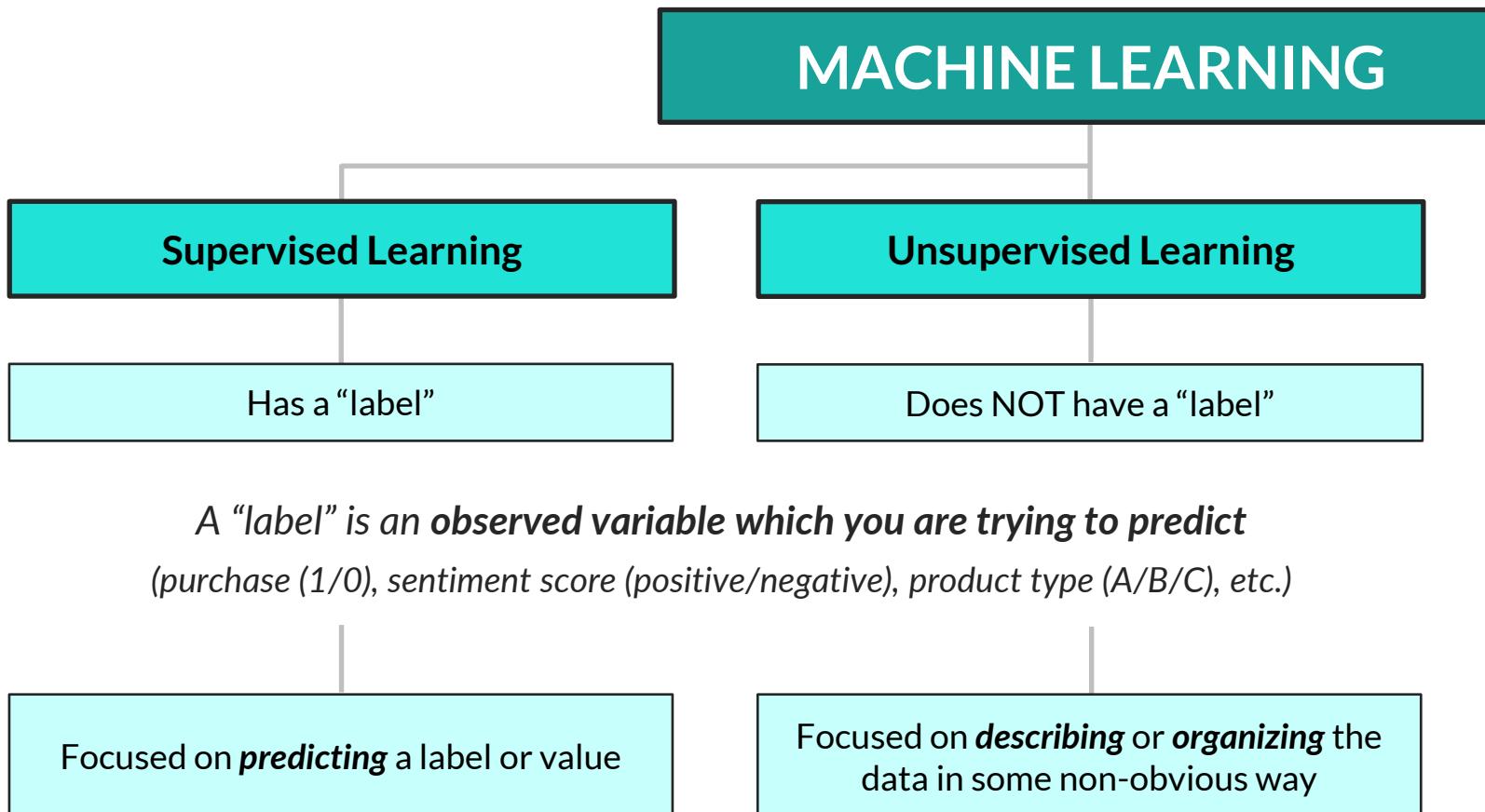
Model performance, hyperparameter tuning, confusion matrices, imbalanced classes, model drift, etc.

# INTRO TO CLASSIFICATION

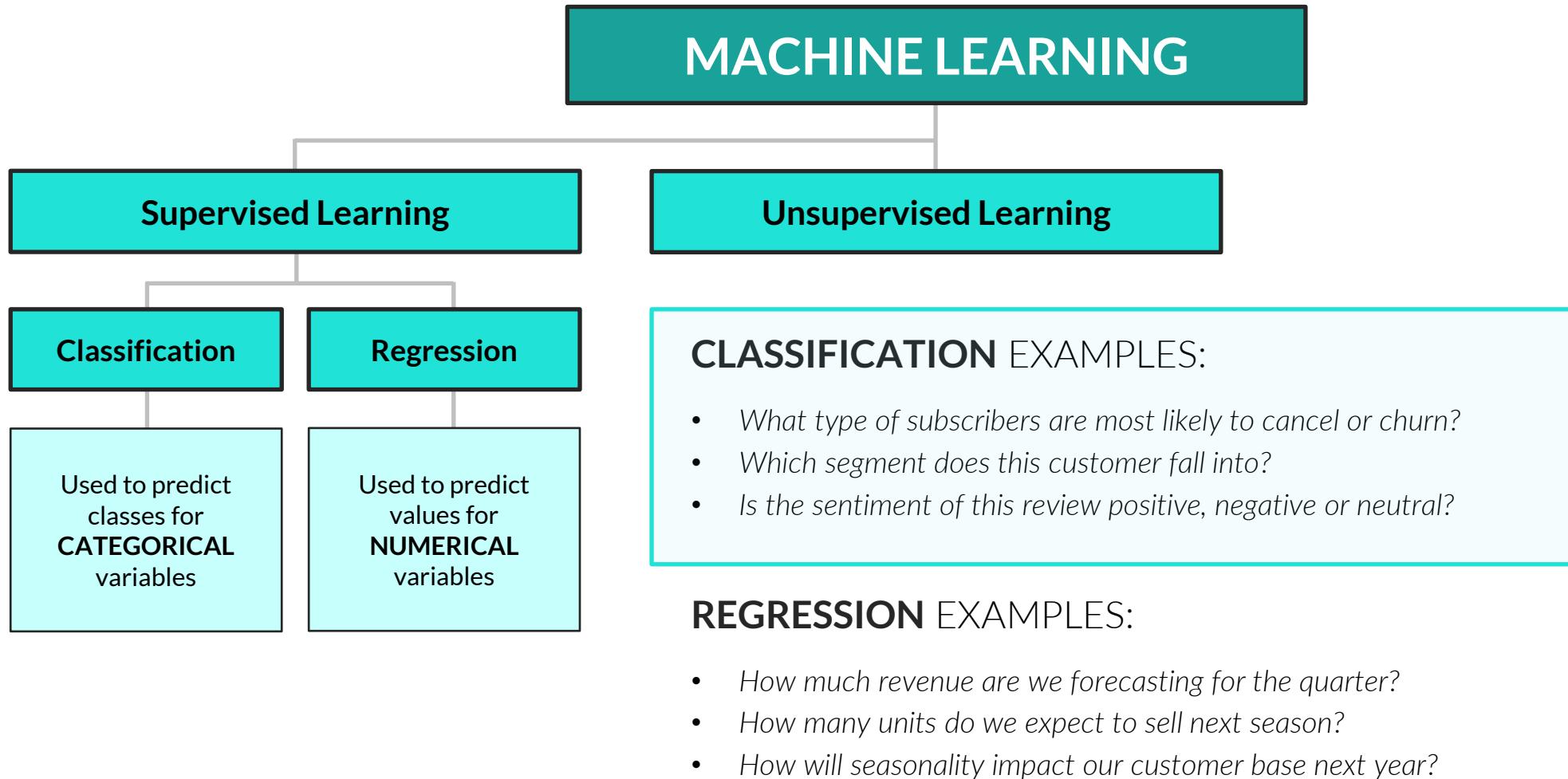
# MACHINE LEARNING LANDSCAPE



# MACHINE LEARNING LANDSCAPE



# MACHINE LEARNING LANDSCAPE



# RECAP: KEY CONCEPTS



## All tables contain rows and columns

- Each row represents an individual **record/observation**
- Each columns represent an individual **variable**



## Variables can be categorical or numerical

- Categorical variables contain **classes** or **categories** (used for filtering)
- Numerical variables contain **numbers** (used for aggregation)



## QA and data profiling comes first, every time

- Without quality data, you can't build quality models ("garbage in, garbage out")
- Profiling is the first step towards **conditioning** (or filtering) on key variables to understand their impact



## Machine Learning is needed when visual analysis falls short

- Humans aren't capable of thinking beyond 3+ dimensions, but machines are built for it
- ML is a natural extension of visual analysis and univariate/multivariate profiling

Session ID	Browser	Time (Sec)	Pageviews	Purchase
1	Chrome	354	11	1
2	Safari	94	4	1
3	Safari	36	2	0
4	IE	17	1	0
5	Chrome	229	9	1

# CLASSIFICATION 101

---

The goal of any classification model is to predict a **dependent variable** using **independent variables**

## **Dependent variable (DV)**

- This is the variable **you're trying to predict**
- The dependent variable is commonly referred to as "Y", "predicted", "output", or "target" variable
- Classification is about understanding how the DV is impacted by, or *dependent* on, other variables in the model

## **Independent variables (IVs)**

- These are the variables **which help you predict the dependent variable**
- Independent variables are commonly referred to as "X's", "predictors", "features", "dimensions", "explanatory variables", or "covariates"
- Classification is about understanding how the IVs impact, or *predict*, the DV

# CLASSIFICATION 101

**EXAMPLE:** Using data from a CRM database (*sample below*) to predict if a customer will churn next month

Cust. ID	Gender	Status	HH Income	Age	Sign-Up Date	Newsletter	Churn
1	M	Bronze	\$30,000	29	Jan 17, 2019	1	0
2	F	Gold	\$60,000	37	Mar 18, 2017	1	1
3	F	Bronze	\$15,000	24	Oct 1, 2020	0	0
4	F	Silver	\$75,000	41	Apr 12, 2019	1	0
5	M	Bronze	\$40,000	36	Jul 23, 2020	1	1
6	M	Gold	\$35,000	31	Oct 22, 2017	0	1
7	F	Gold	\$80,000	46	May 2, 2019	0	0
8	F	Bronze	\$20,000	33	Feb 20, 2020	0	0
9	M	Silver	\$100,000	51	Aug 5, 2020	1	1
10	M	Silver	\$45,000	29	Sep 15, 2019	1	0

**Churn** is our **dependent variable**, since it's what we want to predict

Gender, Status, HH Income, Age, Sign-Up Date and Newsletter are all **independent variables**, since they can help us explain, or predict, Churn

# CLASSIFICATION 101

**EXAMPLE:** Using data from a CRM database (*sample below*) to predict if a customer will churn next month

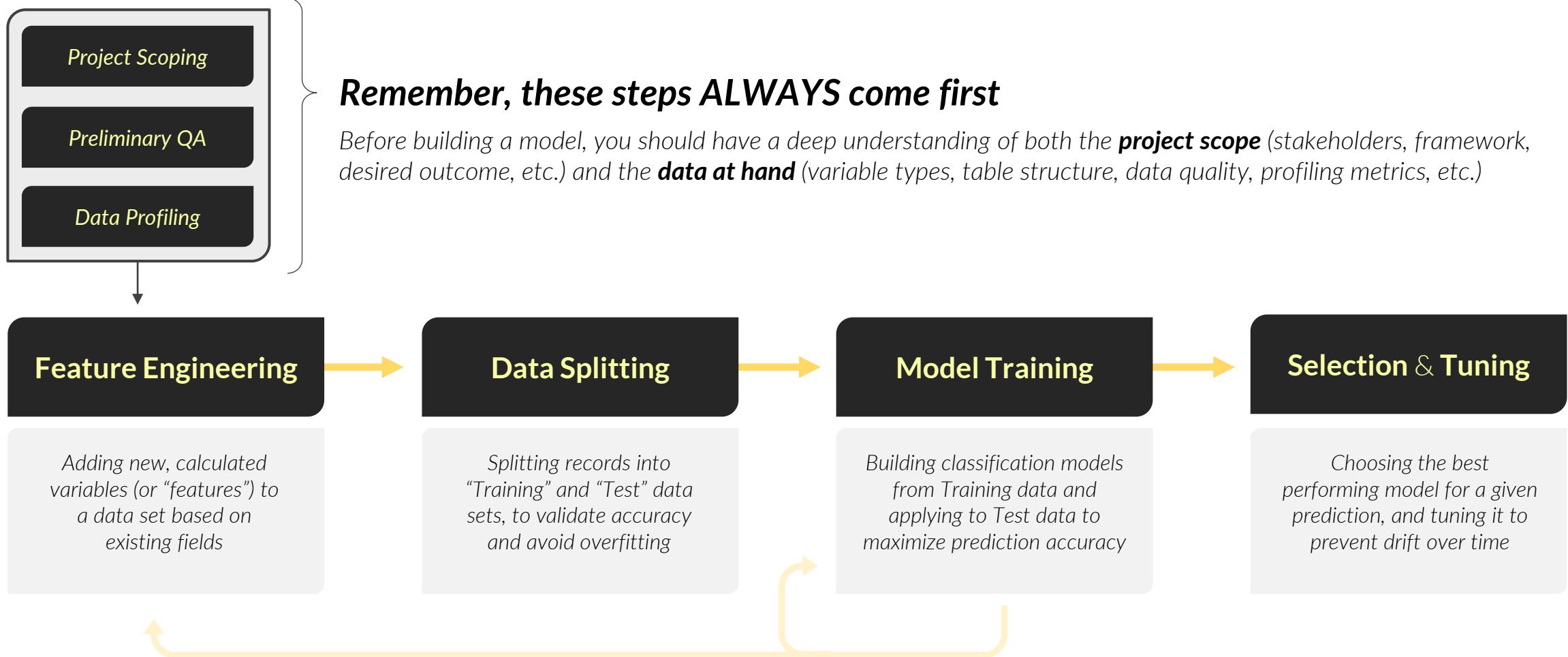
Cust. ID	Gender	Status	HH Income	Age	Sign-Up Date	Newsletter	Churn
1	M	Bronze	\$30,000	29	Jan 17, 2019	1	0
2	F	Gold	\$60,000	37	Mar 18, 2017	1	1
3	F	Bronze	\$15,000	24	Oct 1, 2020	0	0
4	F	Silver	\$75,000	41	Apr 12, 2019	1	0
5	M	Bronze	\$40,000	36	Jul 23, 2020	1	1
6	M	Gold	\$35,000	31	Oct 22, 2017	0	1
7	F	Gold	\$80,000	46	May 2, 2019	0	0
8	F	Bronze	\$20,000	33	Feb 20, 2020	0	0
9	M	Silver	\$100,000	51	Aug 5, 2020	1	1
10	M	Silver	\$45,000	29	Sep 15, 2019	1	0
11	F	Bronze	\$40,000	40	Dec 12, 2020	0	???

We use records with **observed values** for both independent and dependent variables to “train” our classification model...

...then apply that model to new, **unobserved values** containing IVs but no DV

***This is what our model will predict!***

# CLASSIFICATION WORKFLOW



# FEATURE ENGINEERING



**Feature engineering** is the process of enriching a data set by creating additional independent variables based on existing fields

- New features can help improve the accuracy and predictive power of your ML models
- Feature engineering is often used to convert fields into “model-friendly” formats; for example, **one-hot encoding** transforms categorical variables into binary (1/0) fields

Original features						Engineered features							
Cust. ID	Status	HH Income	Age	Sign-Up Date	Newsletter	Gold	Silver	Bronze	Scaled Income	Log Income	Age Group	Sign-Up Year	Priority
1	Bronze	\$30,000	29	Jan 17, 2019	1	0	0	1	.25	10.3	20-30	2019	1
2	Gold	\$60,000	37	Mar 18, 2017	1	1	0	0	.75	11.0	30-40	2017	1
3	Bronze	\$15,000	24	Oct 1, 2020	0	0	0	1	0	9.6	20-30	2020	0
4	Silver	\$75,000	41	Apr 12, 2019	1	0	1	0	1	11.2	40-50	2019	0
5	Bronze	\$40,000	36	Jul 23, 2020	1	0	0	1	.416	10.6	30-40	2020	1

# FEATURE ENGINEERING TECHNIQUES

Common **feature engineering** techniques:

Cust. ID	Status	HH Income	Age	Sign-Up Date	Newsletter	Gold	Silver	Bronze	Scaled Income	Log Income	Age Group	Sign-Up Year	Priority
1	Bronze	\$30,000	29	Jan 17, 2019	1	0	0	1	.25	10.3	20-30	2019	1
2	Gold	\$60,000	37	Mar 18, 2017	1	1	0	0	.75	11.0	30-40	2017	1
3	Bronze	\$15,000	24	Oct 1, 2020	0	0	0	1	0	9.6	20-30	2020	0
4	Silver	\$75,000	41	Apr 12, 2019	1	0	1	0	1	11.2	40-50	2019	0
5	Bronze	\$40,000	36	Jul 23, 2020	1	0	0	1	.416	10.6	30-40	2020	1

## One-Hot Encoding

Splitting categorical fields into separate binary features

## Scaling

Standardizing numerical ranges common scales (i.e. 1-10, 0-100%)

## Log Transformation

Converting a range of values into a compressed, "less-skewed" distribution

## Discretization

Grouping continuous values into discrete segments or bins

## Date Extraction

Transforming a date or datetime value into individual components

## Boolean Logic

Using and/or logic to encode "interactions" between variables

# DATA SPLITTING



**Splitting** is the process of partitioning data into separate sets of records for the purpose of training and testing machine learning models

- As a rule of thumb, ~70-80% of your data will be used for **Training** (which is what your model learns from), and ~20-30% will be reserved for **Testing** (to validate the model's accuracy)

Cust. ID	Gender	Status	HH Income	Age	Sign-Up Date	Newsletter	Churn
1	M	Bronze	\$30,000	29	Jan 17, 2019	1	0
2	F	Gold	\$60,000	37	Mar 18, 2017	1	1
3	F	Bronze	\$15,000	24	Oct 1, 2020	0	0
4	F	Silver	\$75,000	41	Apr 12, 2019	1	0
5	M	Bronze	\$40,000	36	Jul 23, 2020	1	1
6	M	Gold	\$35,000	31	Oct 22, 2017	0	1
7	F	Gold	\$80,000	46	May 2, 2019	0	0
8	F	Bronze	\$20,000	33	Feb 20, 2020	0	0
9	M	Silver	\$100,000	51	Aug 5, 2020	1	1
10	M	Silver	\$45,000	29	Sep 15, 2019	1	0

*Training*  
data

*Test*  
data



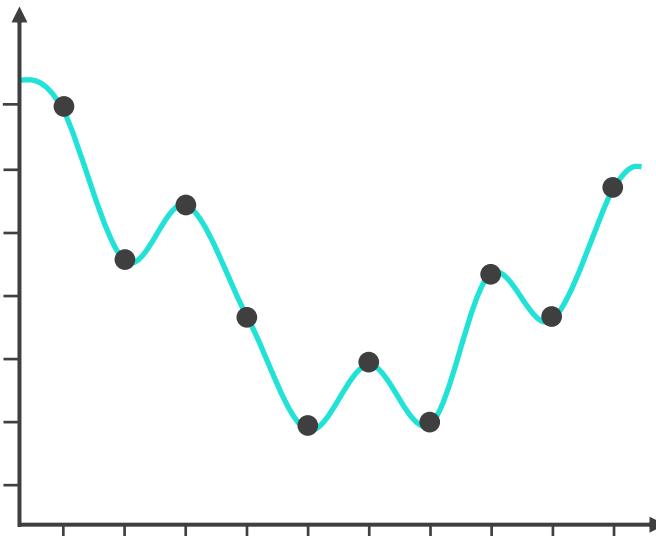
**Test data is NOT used to optimize models**

Using **Training** data for optimization and **Test** data for validation ensures that your model can accurately predict both known and *unknown* values, which helps to prevent **overfitting**

# OVERFITTING

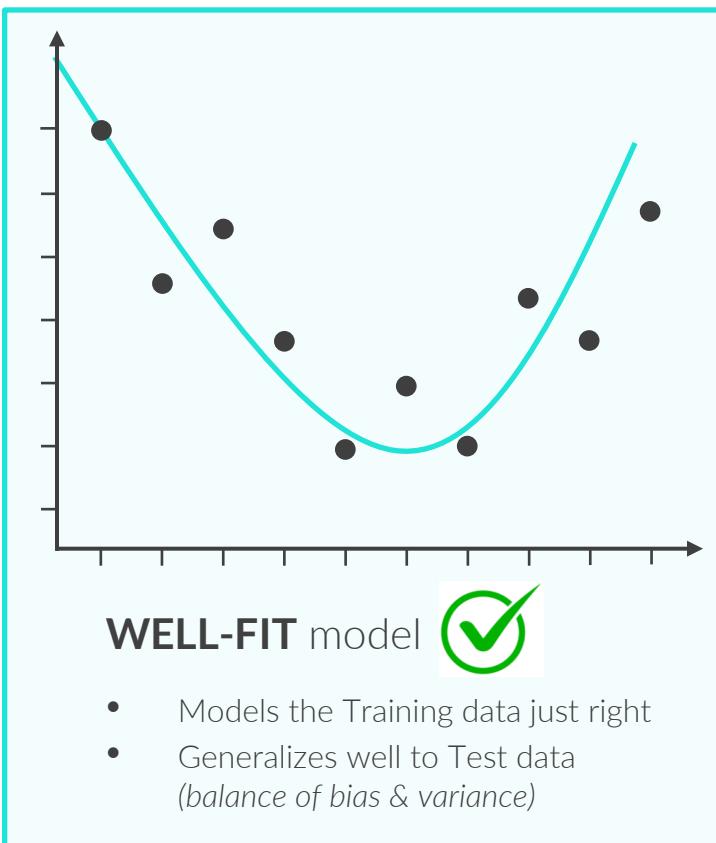
Splitting is primarily used to avoid **overfitting**, which is when a model predicts known (*Training*) data very well but unknown (*Test*) data poorly

- Think of overfitting like memorizing the answers to a test instead of actually *learning* the material; you'll ace the test, but lack the ability to **generalize** and apply your knowledge to unfamiliar questions



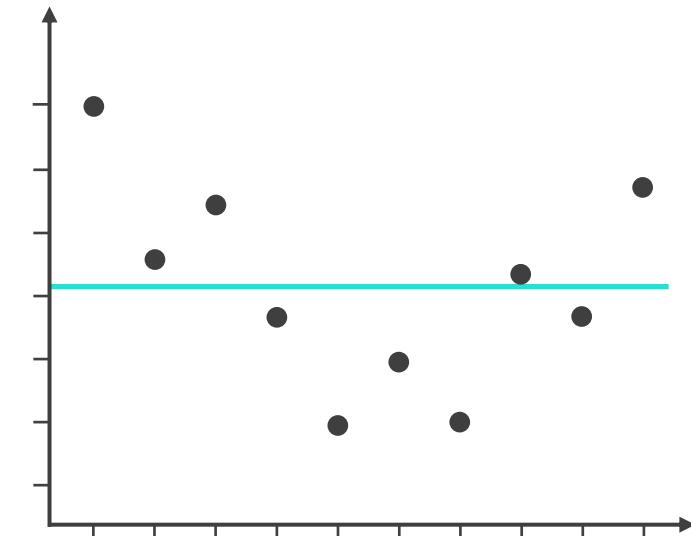
**OVERFIT** model

- Models the Training data too well
- Doesn't generalize well to Test data (*high variance, low bias*)



**WELL-FIT** model

- Models the Training data just right
- Generalizes well to Test data (*balance of bias & variance*)



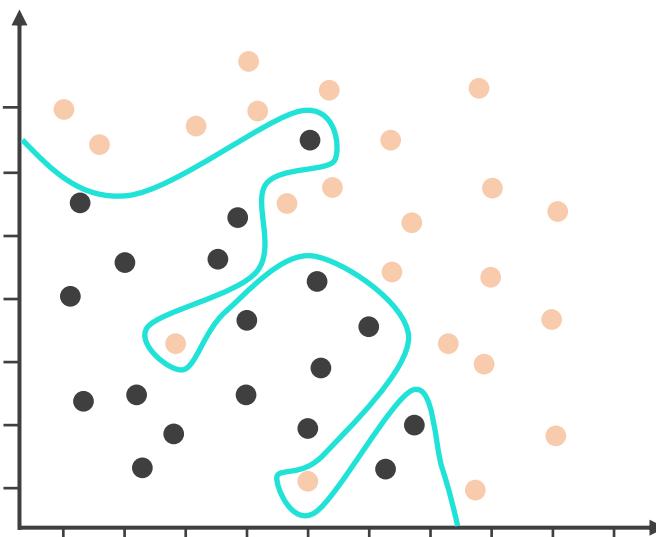
**UNDERFIT** model

- Doesn't model Training data well enough
- Doesn't generalize well to Test data (*high bias, low variance*)

# OVERFITTING

Splitting is primarily used to avoid **overfitting**, which is when a model predicts known (*Training*) data very well but unknown (*Test*) data poorly

- Think of overfitting like memorizing the answers to a test instead of actually *learning* the material; you'll ace the test, but lack the ability to **generalize** and apply your knowledge to unfamiliar questions



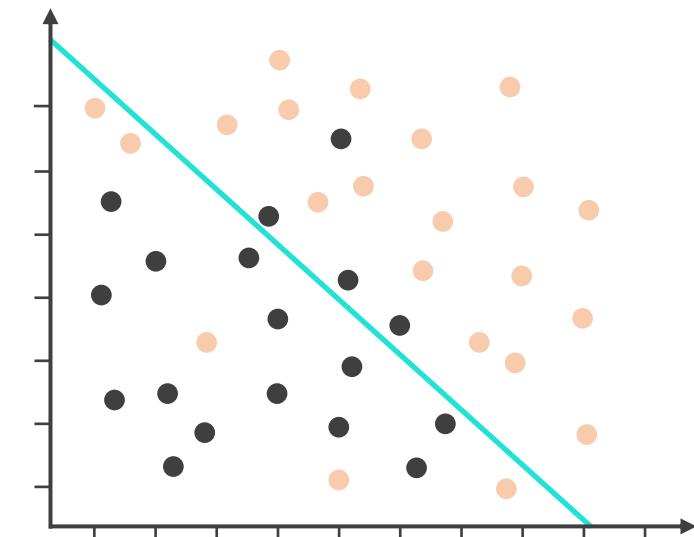
**OVERFIT** model

- Models the Training data too well
- Doesn't generalize well to Test data (*high variance, low bias*)



**WELL-FIT** model

- Models the Training data just right
- Generalizes well to Test data (*balance of bias & variance*)

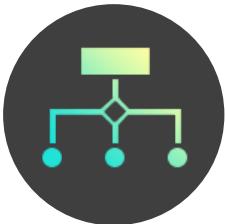


**UNDERFIT** model

- Doesn't model Training data well enough
- Doesn't generalize well to Test data (*high bias, low variance*)

# CLASSIFICATION MODELS

# CLASSIFICATION MODELS



In this section we'll introduce common **classification models** used to predict categorical outcomes, including KNN, naïve bayes, decision trees, logistic regression, and more

## TOPICS WE'LL COVER:

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

## COMMON USE CASES:

- Predicting if a subscriber will stay or churn
- Predicting which product a customer will purchase
- Predicting tone or sentiment from raw text (product reviews, survey responses, tweets, etc.)
- Predicting if an email is spam, or a bank transaction is fraudulent

# K-NEAREST NEIGHBORS



# K-NEAREST NEIGHBORS (KNN)

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

**K-Nearest Neighbors (KNN)** is a classification technique designed to predict outcomes based on similar observed values

- In its simplest form, KNN creates a scatter plot with training data, plots a new unobserved value, and assigns a class (DV) based on the classes of nearby points
- **K** represents the number of nearby points (or “neighbors”) the model will consider when making a prediction

**Example use cases:**

- Generating product recommendations on a website
- Classifying customers into pre-existing segments



# K-NEAREST NEIGHBORS (KNN)

**K-Nearest Neighbors**

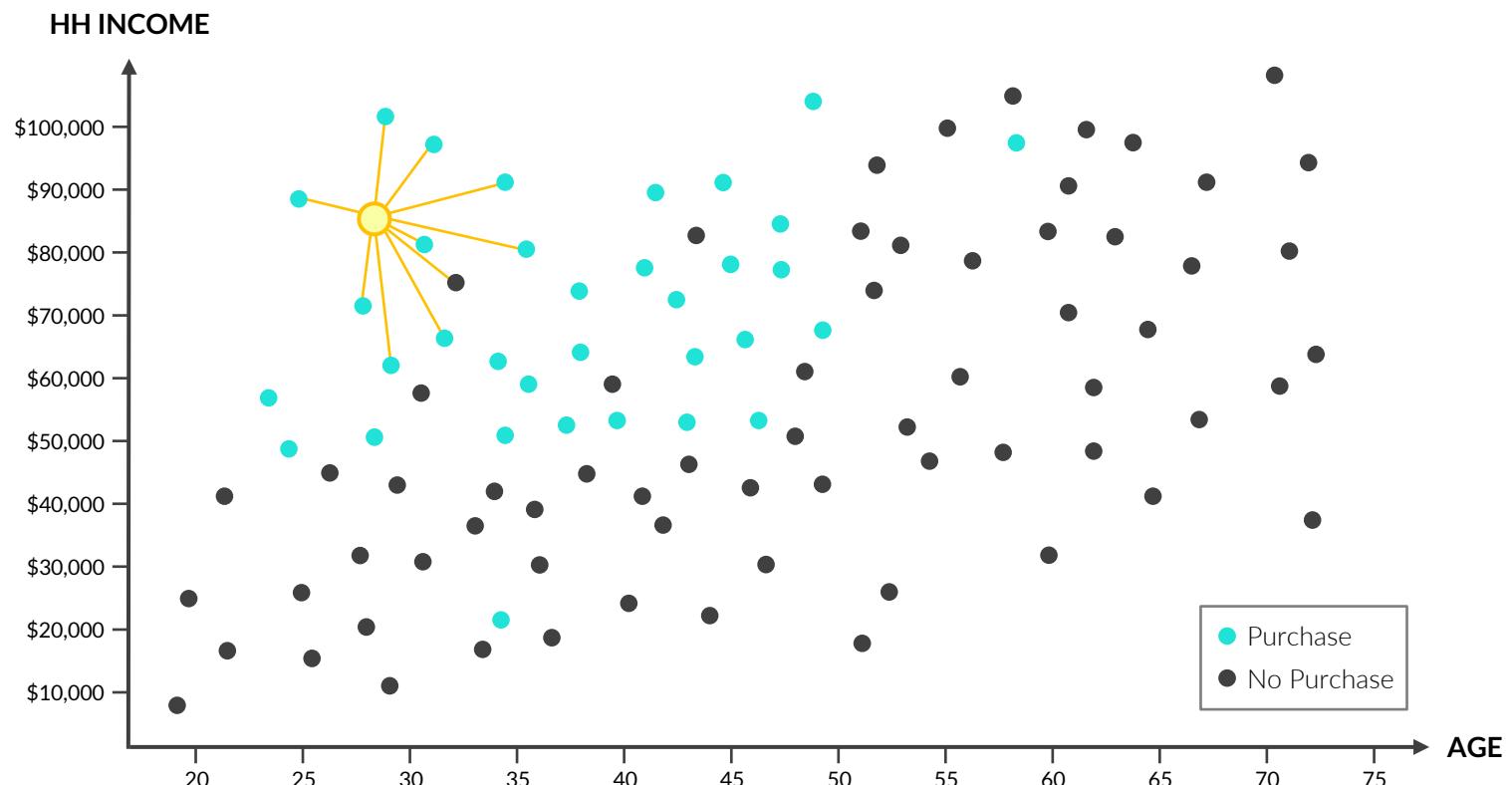
Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



X  
28

Y  
\$85,000

K  
10

UNOBSERVED VALUE → NEIGHBORS → PREDICTION

• 9 Purchase  
• 1 No Purchase

**PURCHASE**  
(90% Confidence)



# K-NEAREST NEIGHBORS (KNN)

**K-Nearest Neighbors**

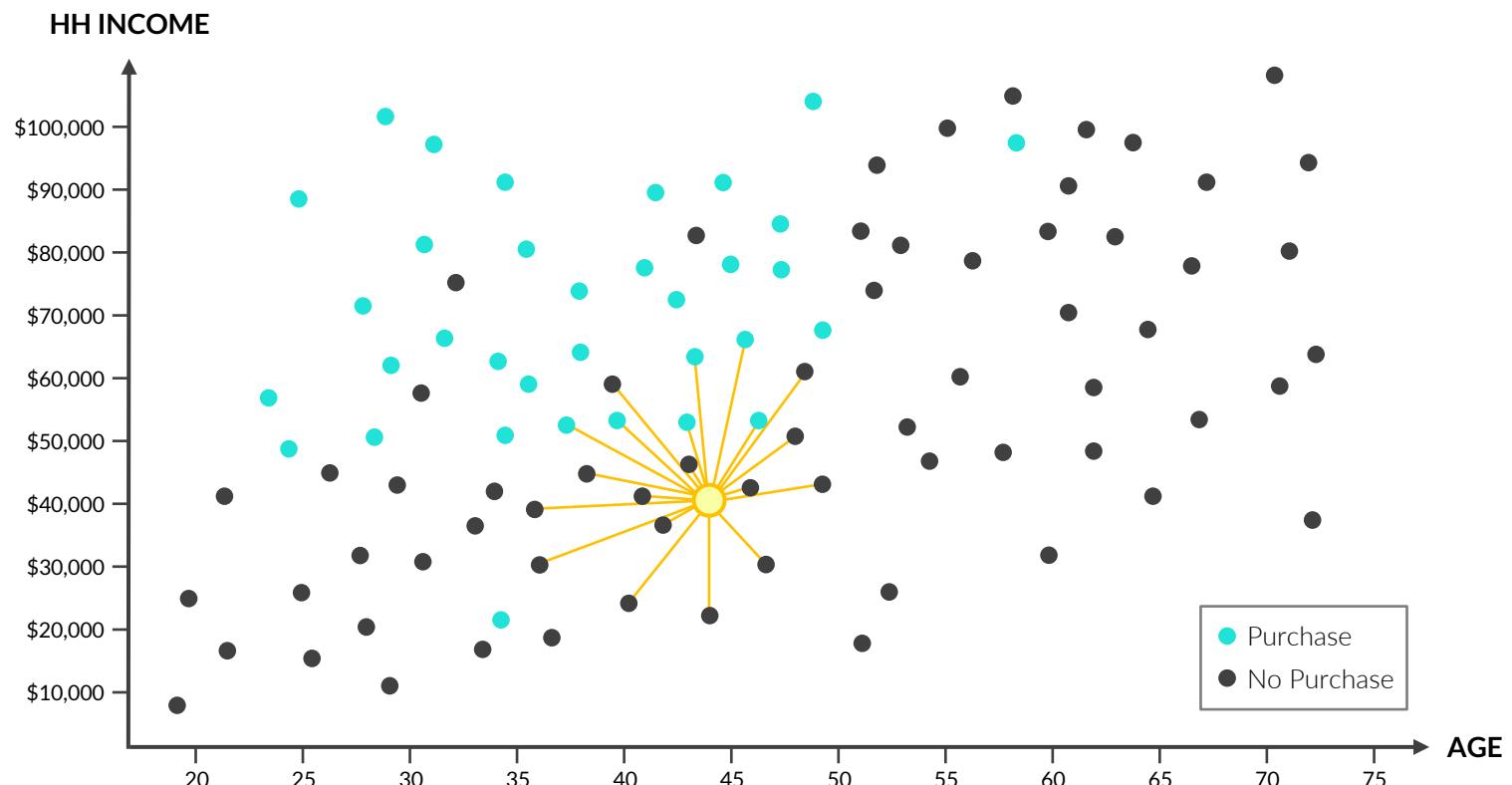
Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



X  
44

Y  
\$40,000

K  
20

6 Purchase  
14 No Purchase

**NO PURCHASE**  
(70% Confidence)

UNOBSERVED VALUE

NEIGHBORS

PREDICTION



# K-NEAREST NEIGHBORS (KNN)

**K-Nearest Neighbors**

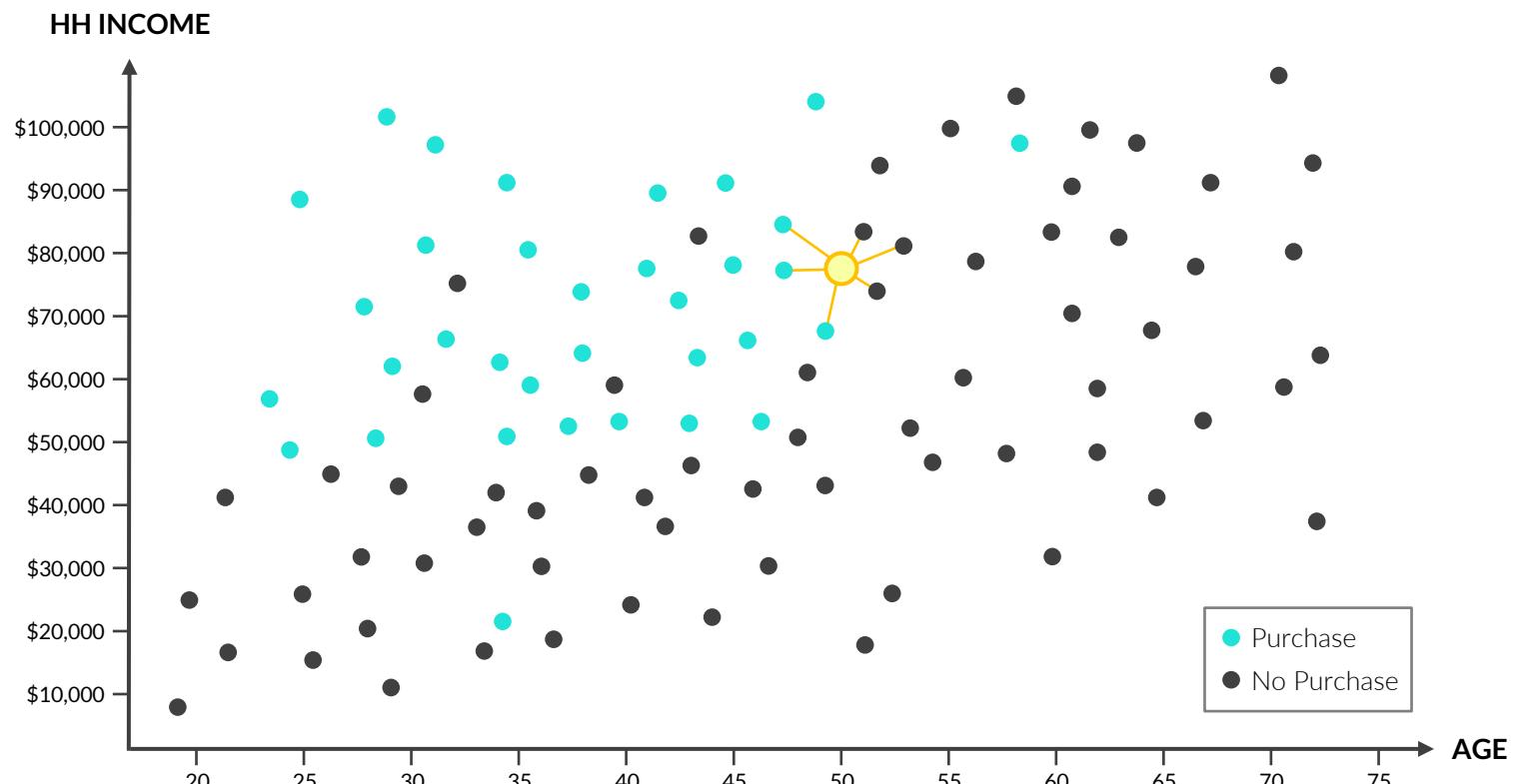
Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



X  
**50**

Y  
**\$80,000**

K  
**6**

UNOBSERVED VALUE → NEIGHBORS → PREDICTION

3 Purchase  
3 No Purchase

???



# K-NEAREST NEIGHBORS (KNN)

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



What if there's a tie?

- In practice, KNN also factors *distance* between neighbors; if there are multiple modes, the class with the **shortest total distance** to the observation wins



How do we figure out the “right” value for K?

- This is where **Machine Learning** comes in! The model tests multiple K values to see which one is most accurate when applied to your Test data



What if we have more than 2 IVs?

- Scatter plots are great for demonstrating how KNN works for 2 independent variables, but humans can't visualize beyond 3 or 4 dimensions
- Computers can easily apply the same logic and calculations to many IVs

# CASE STUDY: K-NEAREST NEIGHBORS



## THE SITUATION

Congratulations! You just landed your dream job as a Machine Learning Engineer at **Spotify**, one of the world's largest music streaming platforms



## THE ASSIGNMENT

Your first assignment is to **analyze user listening behavior to help improve Spotify's recommendation engine**.

You'll need to use a KNN model to predict if a user will **listen**, **skip** or **favorite** a given track, based on various attributes.



## THE OBJECTIVES

1. Collect sample data containing listener behaviors and song attributes
2. Visualize outcomes for a given pair of attributes using a Scatter Plot
3. Calculate the predicted outcome (*listen*, *skip* or *favorite*) and level of confidence for any unobserved value

# NAÏVE BAYES



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

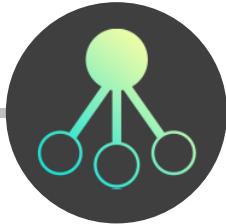
Sentiment Analysis

**Naïve Bayes** is a classification technique which uses conditional probabilities to predict multi-class or binary outcomes

- Naïve Bayes essentially creates **frequency tables** for each combination of variables, then calculates the **conditional probability** of each IV value for a given outcome
- For new observations, the model looks at the probability that each IV value would be observed *given each outcome*, and compares the results to make a prediction

## Example use cases:

- Predicting purchase probability for prospects in a marketing database
- Credit risk scoring in banking or financial industries



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

Each record represents a *customer*

These are the *independent variables* (IV's) which will help us make a prediction

This is the *dependent variable* (DV) we are predicting

These are our *observed values*, which we use to train the model

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0

11	0	1	1	???
----	---	---	---	-----

→ This is an *unobserved value*; which our model will predict



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???

Frequency tables between each IV and the DV

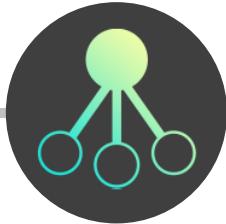


PURCHASE

	1	0
1	■	■
0	■	■

	1	0
1	■	■
0	■	■

	1	0
1	■	■
0	■	■



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

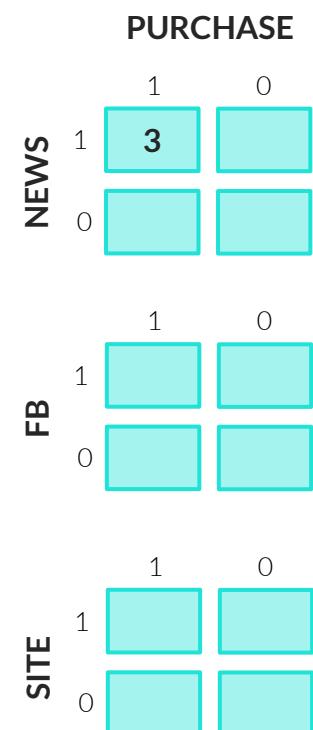
Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

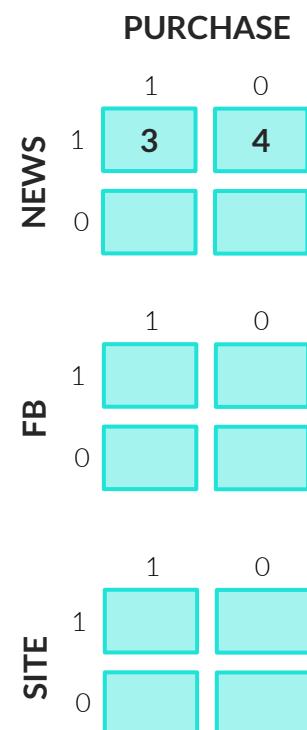
Decision Trees

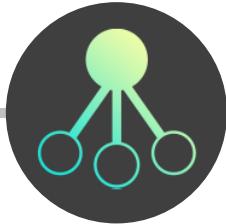
Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

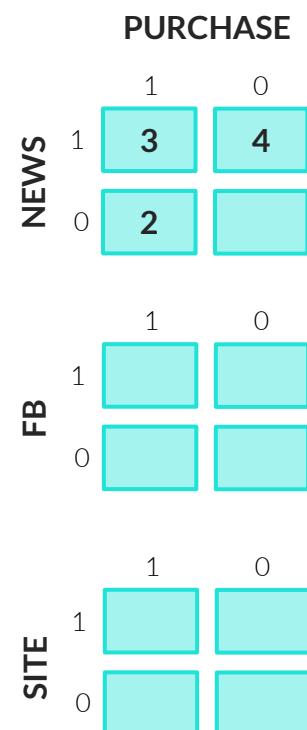
Decision Trees

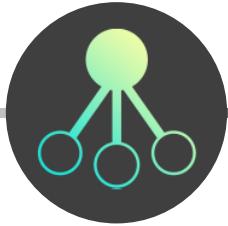
Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

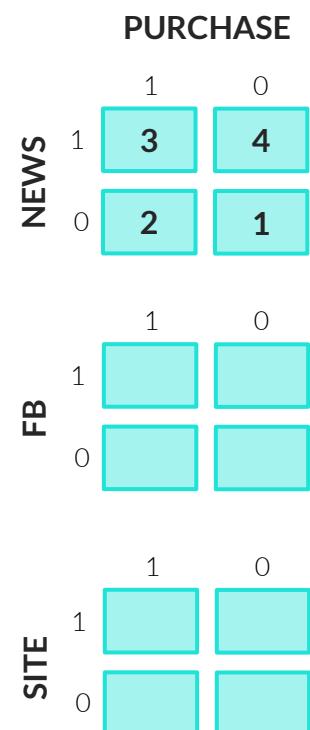
Decision Trees

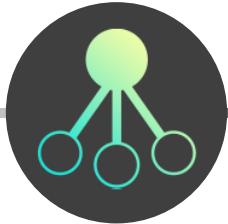
Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

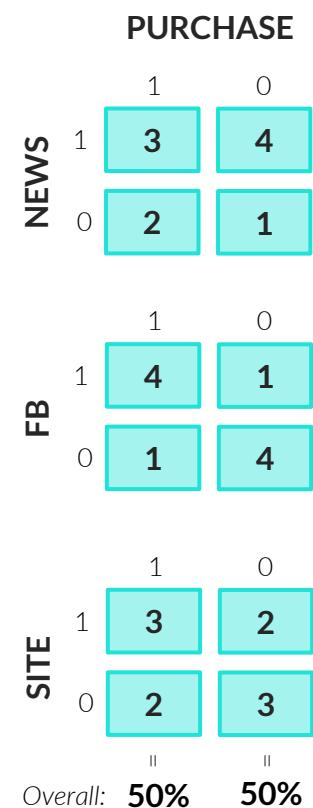
Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

Cust. ID	Subscribed to Newsletter	Followed FB Page	Visited Website	Purchase?
1	0	1	0	1
2	1	0	1	0
3	1	1	0	1
4	0	0	0	0
5	1	0	0	0
6	1	1	1	1
7	1	0	1	0
8	0	1	1	1
9	1	0	1	1
10	1	1	0	0
11	0	1	1	???





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

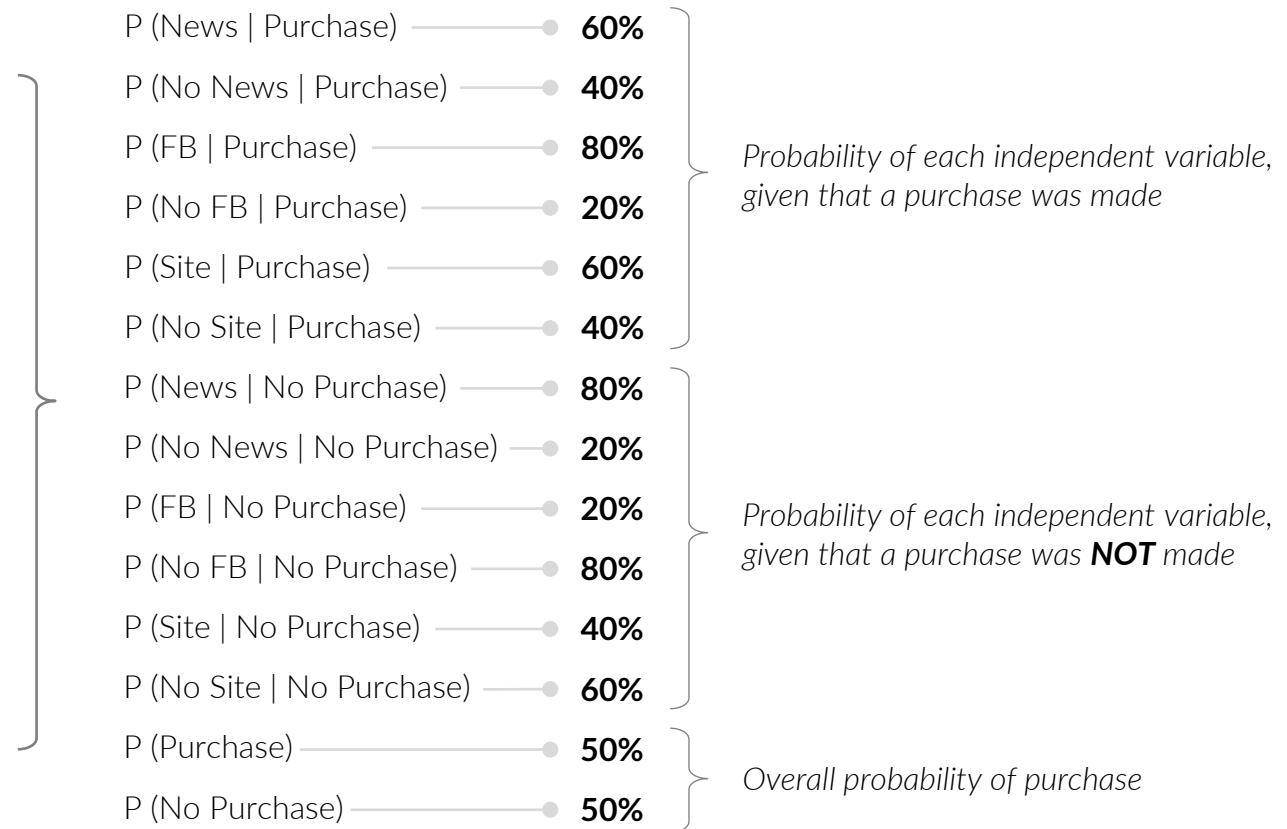
Logistic Regression

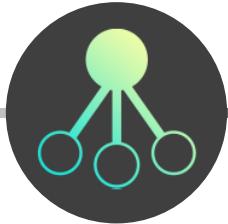
Sentiment Analysis

Frequency tables give us the **conditional probability** of each outcome

- For example, **P (News | Purchase)** tells us the probability that a customer subscribed to the newsletter, given (or conditioned on) the fact that they purchased

		PURCHASE	
		1	0
NEWS	1	3	4
	0	2	1
		1	0
FB	1	4	1
	0	1	4
		1	0
SITE	1	3	2
	0	2	3





# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

- P (News | Purchase) ————— ● **60%**
- P (No News | Purchase) ————— ● **40%**
- P (FB | Purchase) ————— ● **80%**
- P (No FB | Purchase) ————— ● **20%**
- P (Site | Purchase) ————— ● **60%**
- P (No Site | Purchase) ————— ● **40%**
- P (News | No Purchase) ————— ● **80%**
- P (No News | No Purchase) ————— ● **20%**
- P (FB | No Purchase) ————— ● **20%**
- P (No FB | No Purchase) ————— ● **80%**
- P (Site | No Purchase) ————— ● **40%**
- P (No Site | No Purchase) ————— ● **60%**
- P (Purchase) ————— ● **50%**
- P (No Purchase) ————— ● **50%**

**Unobserved value:**

NEWS	FB	SITE
0	1	1

Prob. given Purchase = **1**

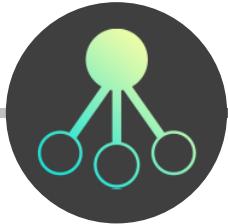
||

P (No News | Purchase)  
x  
P (FB | Purchase)  
x  
P (Site | Purchase)  
x  
P (Purchase)

Prob. given Purchase = **0**

||

P (No News | No Purchase)  
x  
P (FB | No Purchase)  
x  
P (Site | No Purchase)  
x  
P (No Purchase)



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

- P (News | Purchase) —————● 60%
- P (No News | Purchase) —————● 40%
- P (FB | Purchase) —————● 80%
- P (No FB | Purchase) —————● 20%
- P (Site | Purchase) —————● 60%
- P (No Site | Purchase) —————● 40%
- P (News | No Purchase) —————● 80%
- P (No News | No Purchase) —————● 20%
- P (FB | No Purchase) —————● 20%
- P (No FB | No Purchase) —————● 80%
- P (Site | No Purchase) —————● 40%
- P (No Site | No Purchase) —————● 60%
- P (Purchase) —————● 50%
- P (No Purchase) —————● 50%

**Unobserved value:**

NEWS	FB	SITE
0	1	1

Prob. given Purchase = 1

||

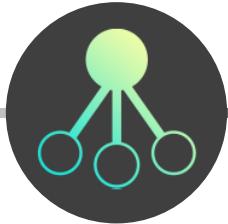
40%
x
80%
x
60%
x
50%

Prob. given Purchase = 0

||

P (No News   No Purchase)
x
P (FB   No Purchase)
x
P (Site   No Purchase)
x
P (No Purchase)

**9.6%**



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

- P (News | Purchase) ——● 60%
- P (No News | Purchase) ——● 40%
- P (FB | Purchase) ——● 80%
- P (No FB | Purchase) ——● 20%
- P (Site | Purchase) ——● 60%
- P (No Site | Purchase) ——● 40%
- P (News | No Purchase) ——● 80%
- P (No News | No Purchase) ——● 20% **(highlighted)**
- P (FB | No Purchase) ——● 20% **(highlighted)**
- P (No FB | No Purchase) ——● 80%
- P (Site | No Purchase) ——● 40% **(highlighted)**
- P (No Site | No Purchase) ——● 60%
- P (Purchase) ——● 50%
- P (No Purchase) ——● 50% **(highlighted)**

**Unobserved value:**

NEWS	FB	SITE
0	1	1

Prob. given Purchase = **1**

40%
x
80%
x
60%
x
50%

Prob. given Purchase = **0**

20%
x
20%
x
40%
x
50%

**9.6%**

**0.8%**



# NAÏVE BAYES

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

- P (News | Purchase) ————— ● 60%
- P (No News | Purchase) ————— ● 40%
- P (FB | Purchase) ————— ● 80%
- P (No FB | Purchase) ————— ● 20%
- P (Site | Purchase) ————— ● 60%
- P (No Site | Purchase) ————— ● 40%
- P (News | No Purchase) ————— ● 80%
- P (No News | No Purchase) ————— ● 20%
- P (FB | No Purchase) ————— ● 20%
- P (No FB | No Purchase) ————— ● 80%
- P (Site | No Purchase) ————— ● 40%
- P (No Site | No Purchase) ————— ● 60%
- P (Purchase) ————— ● 50%
- P (No Purchase) ————— ● 50%

**Unobserved value:**

NEWS	FB	SITE
0	1	1

Prob. given Purchase = 1

||

40%
x
80%
x
60%
x
50%

Prob. given Purchase = 0

||

20%
x
20%
x
40%
x
50%

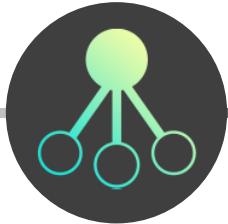
9.6%

0.8%

**Overall Purchase Probability:**  $\frac{9.6\%}{(9.6\% + 0.8\%)} = 92.3\%$  →

**PURCHASE**

PREDICTION



# NAÏVE BAYES

K-Nearest Neighbors

**Naïve Bayes**

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



Who has time to work through all this math?

- No one! These probabilities are all calculated automatically by the model (*no manual effort required!*)



Doesn't multiplying many probabilities lead to very small values?

- Yes, which is why the *relative* probability is so important; even so, Naïve Bayes can struggle with a very large number of IVs

# CASE STUDY: NAÏVE BAYES



## THE SITUATION

You've just been promoted to Marketing Manager at **Cat Slacks**, a global retail powerhouse specializing in high-quality pants for cats



## THE ASSIGNMENT

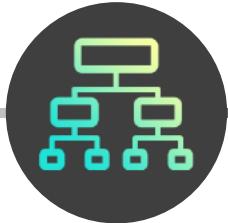
Your boss wants to understand the impact of key customer interactions on purchase behavior, including **subscribing to the newsletter**, **following the Facebook page**, and **visiting the Cat Slacks website**



## THE OBJECTIVES

1. Collect sample data containing customer interactions and purchase events
2. Compare the purchase probability for specific interactions
3. Calculate the overall purchase probability for any given set of interactions

# DECISION TREES



# DECISION TREES

K-Nearest Neighbors

Naïve Bayes

**Decision Trees**

Random Forests

Logistic Regression

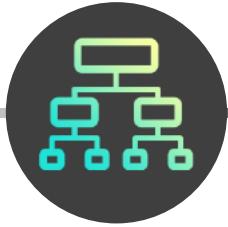
Sentiment Analysis

**Decision Trees** use a series of binary (true/false) rules to predict multi-class or binary outcomes

- For each rule, a decision tree selects a single IV and uses it to split observations into two groups, where each group (*ideally*) skews towards one class
- The goal is to determine which rules and IVs do the best job splitting up the classes, which we can measure using a metric known as **entropy**
- This is NOT a manual process; decision trees test *many* variables and select rules based on the change in entropy after the split (known as **information gain**)

## Example use cases:

- Predicting if a customer will cancel/churn next month
- Identifying fraudulent bank transactions or insurance claims



# DECISION TREES

K-Nearest Neighbors

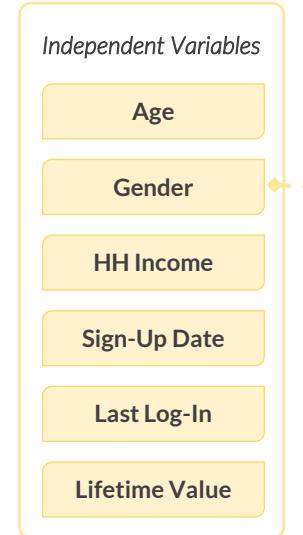
Naïve Bayes

**Decision Trees**

Random Forests

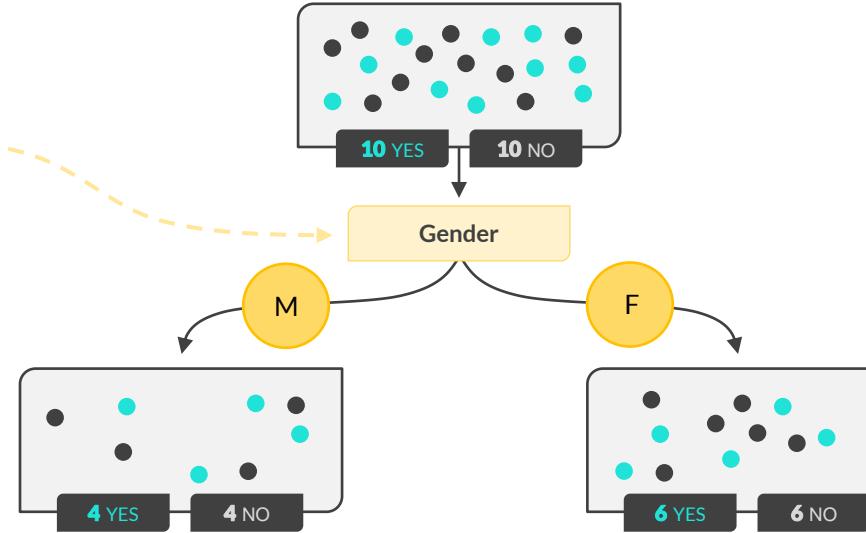
Logistic Regression

Sentiment Analysis

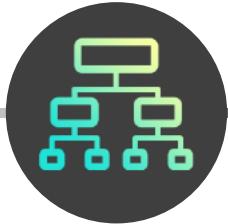


Dependent variable: **CHURN**

Q: Will this subscriber churn next month?



Splitting on **Gender** isn't effective, since our classes are still evenly distributed after the split  
(in other words, Gender isn't a good predictor for churn)



# DECISION TREES

K-Nearest Neighbors

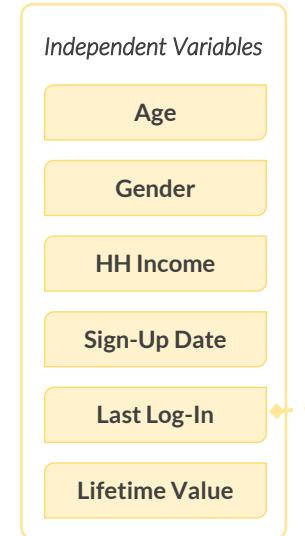
Naïve Bayes

**Decision Trees**

Random Forests

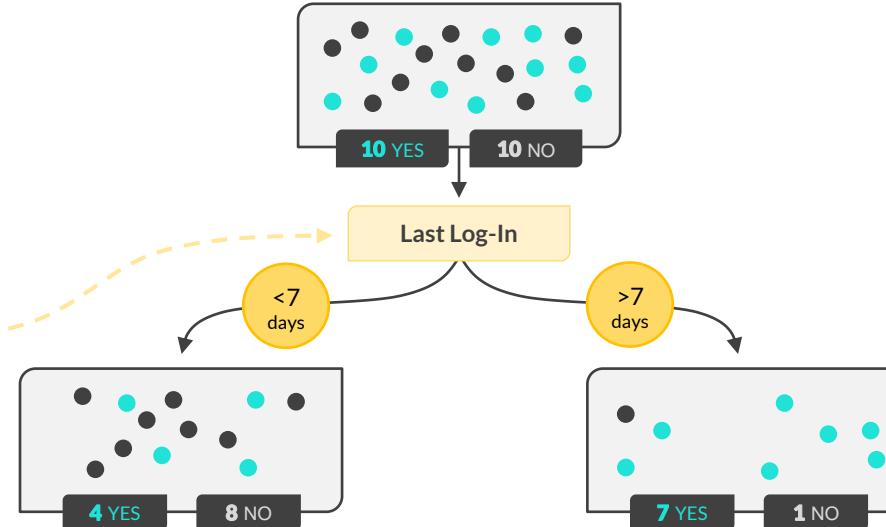
Logistic Regression

Sentiment Analysis



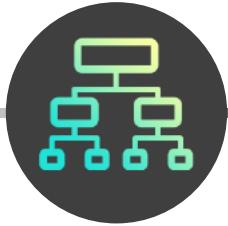
Dependent variable: **CHURN**

Q: Will this subscriber churn next month?



How do we know which independent variables to split?

- **Entropy** measures the “evenness” or “uncertainty” between classes, and can be compared before/after each split to measure information gain
- This is where **Machine Learning** comes in; the model tries *all* IVs to determine which splits reduce entropy the most



# DECISION TREES

K-Nearest Neighbors

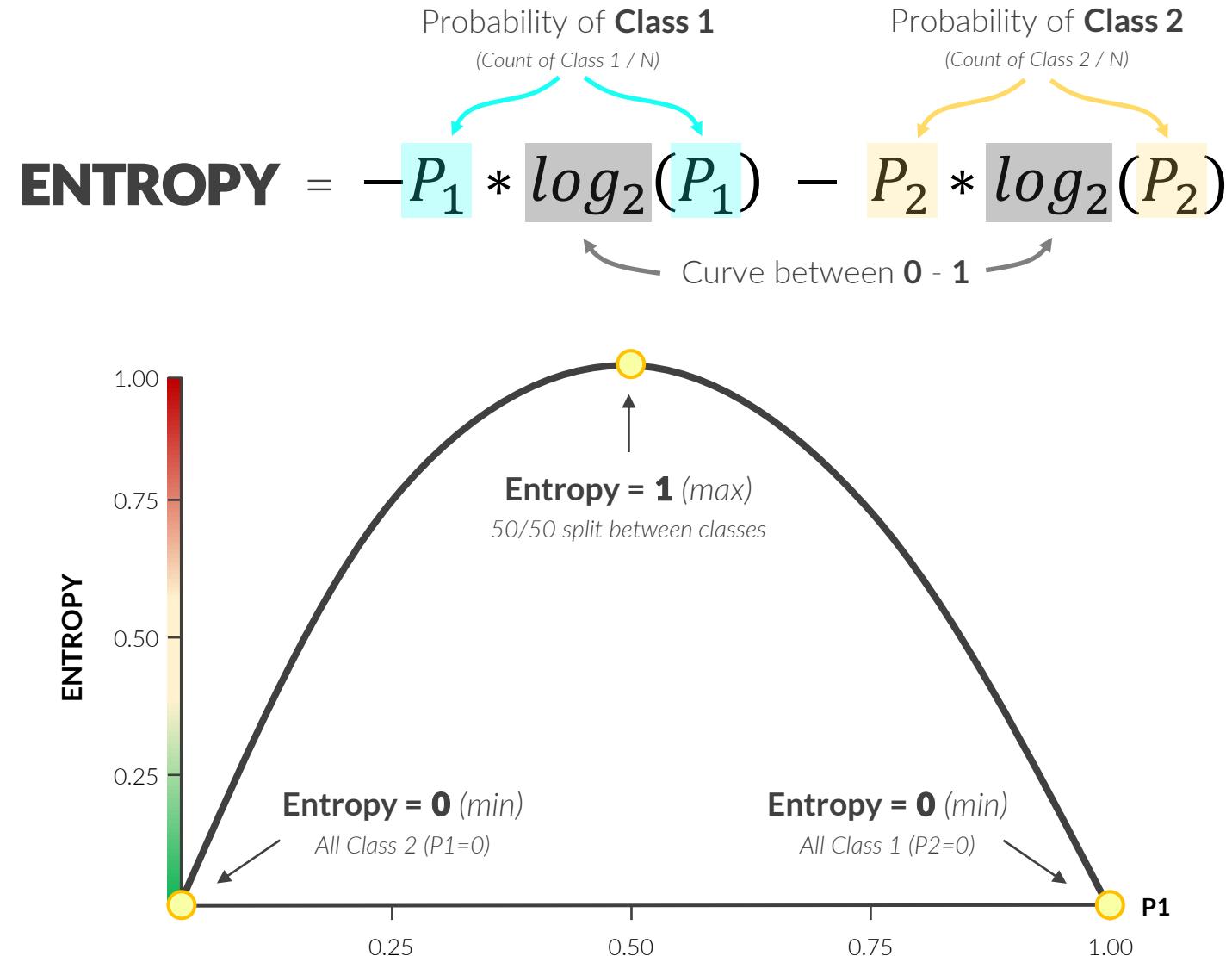
Naïve Bayes

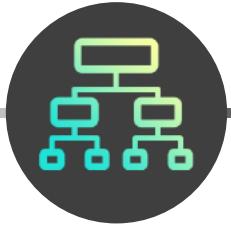
**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis





# DECISION TREES

K-Nearest Neighbors

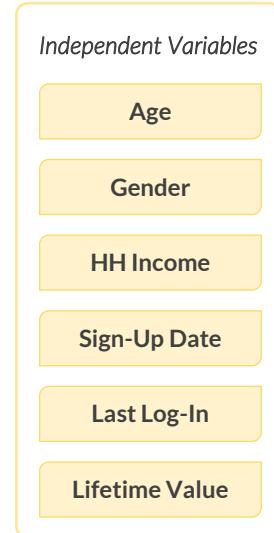
Naïve Bayes

**Decision Trees**

Random Forests

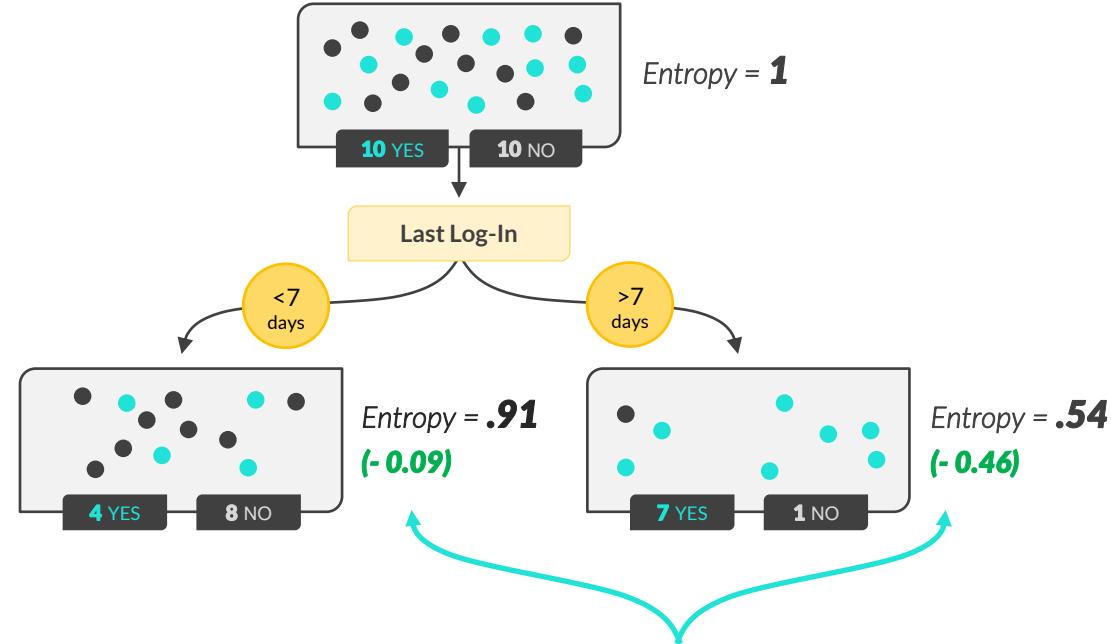
Logistic Regression

Sentiment Analysis

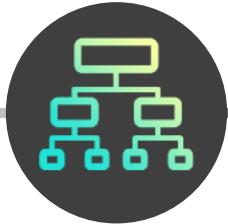


Dependent variable: **CHURN**

Q: Will this subscriber churn next month?



The reduction in entropy after the split tells us that we're **gaining information**, and teasing out differences between those who churn and those who do not



# DECISION TREES

K-Nearest Neighbors

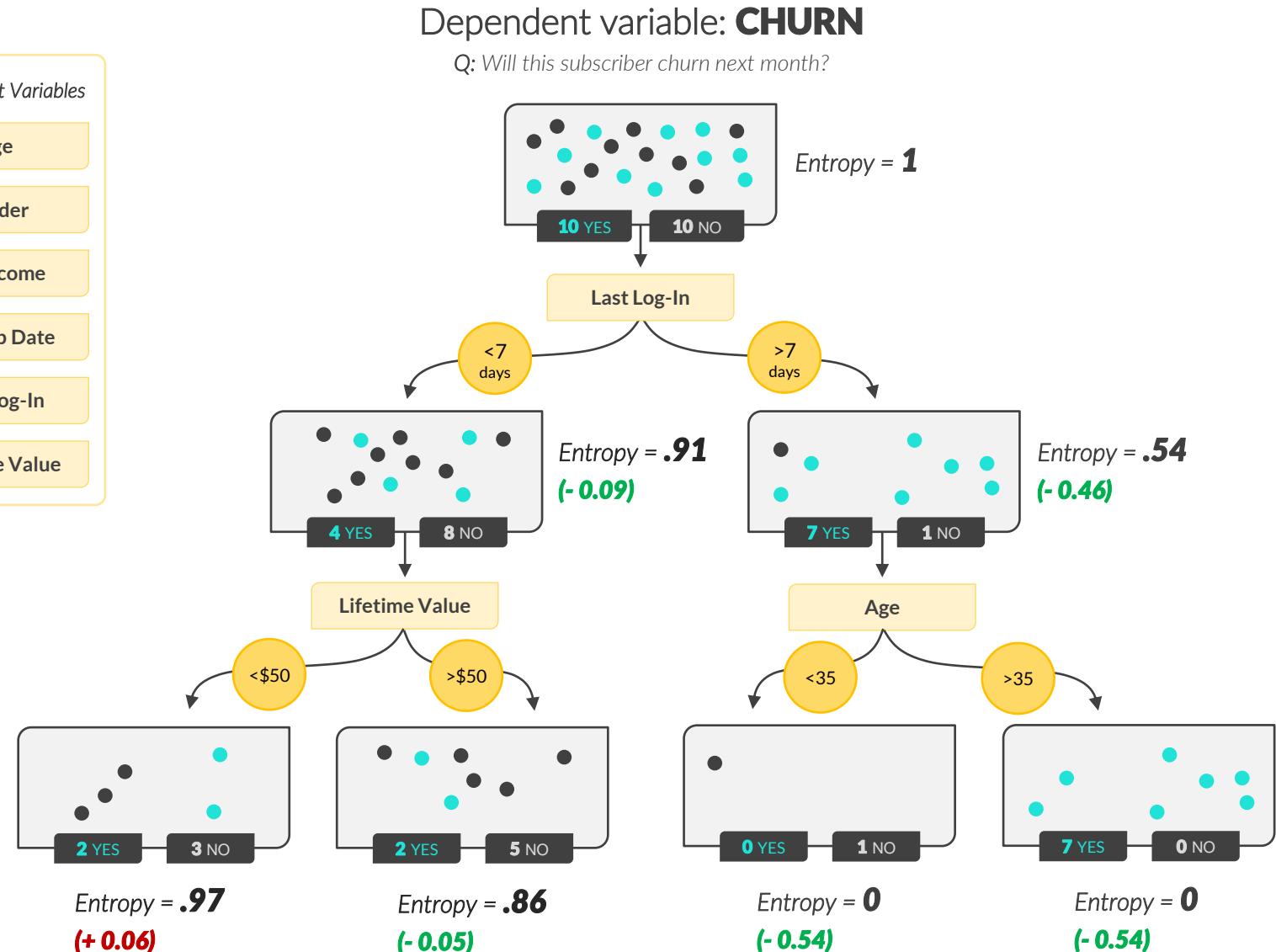
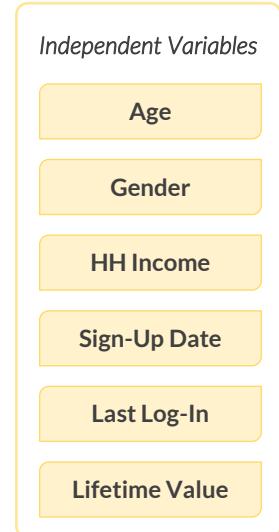
Naïve Bayes

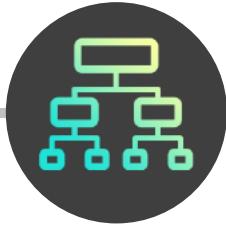
**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis





# DECISION TREES

K-Nearest Neighbors

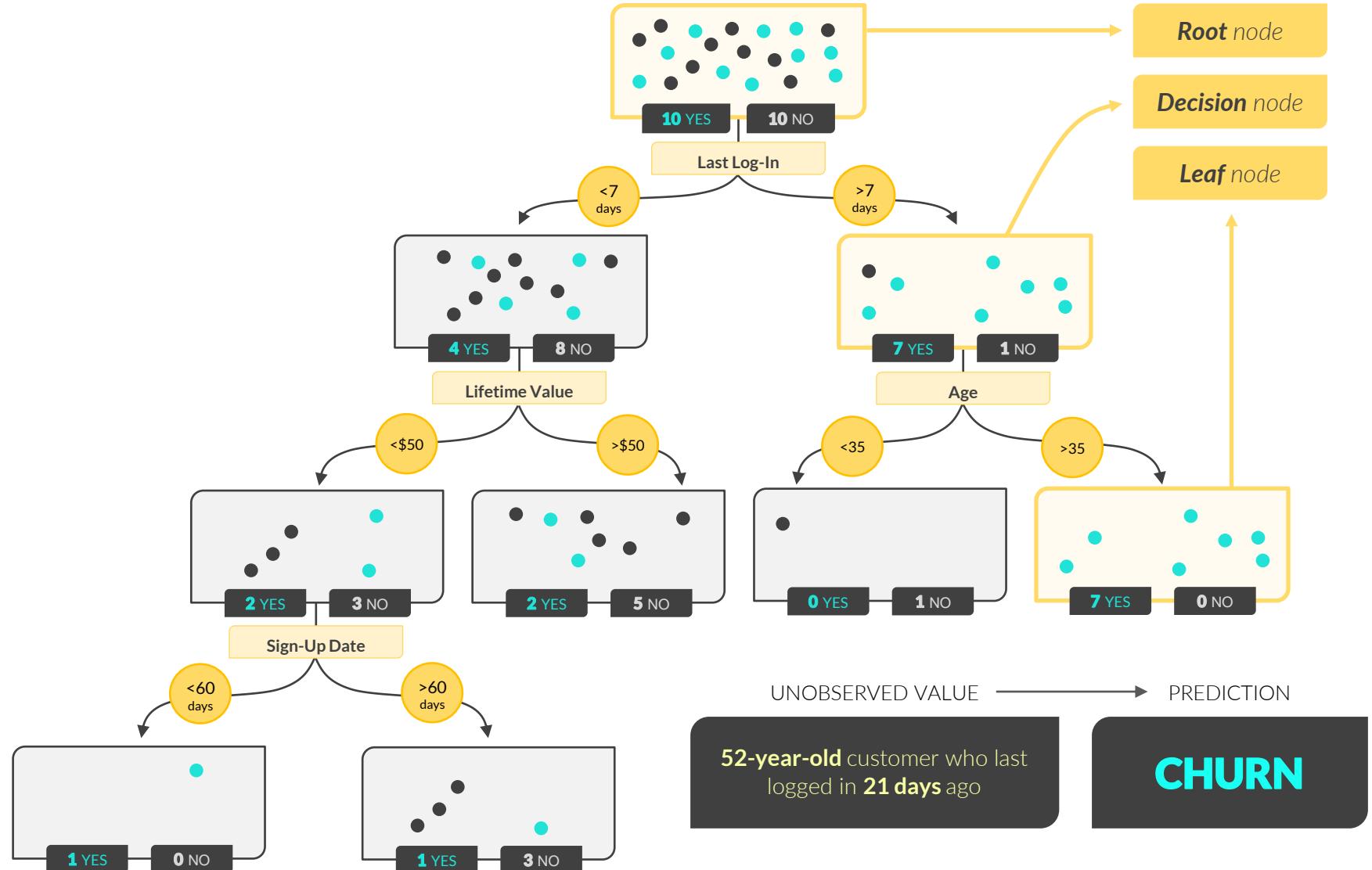
Naïve Bayes

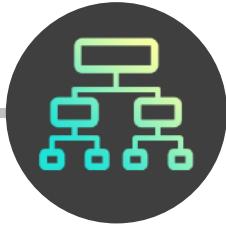
**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis





# DECISION TREES

K-Nearest Neighbors

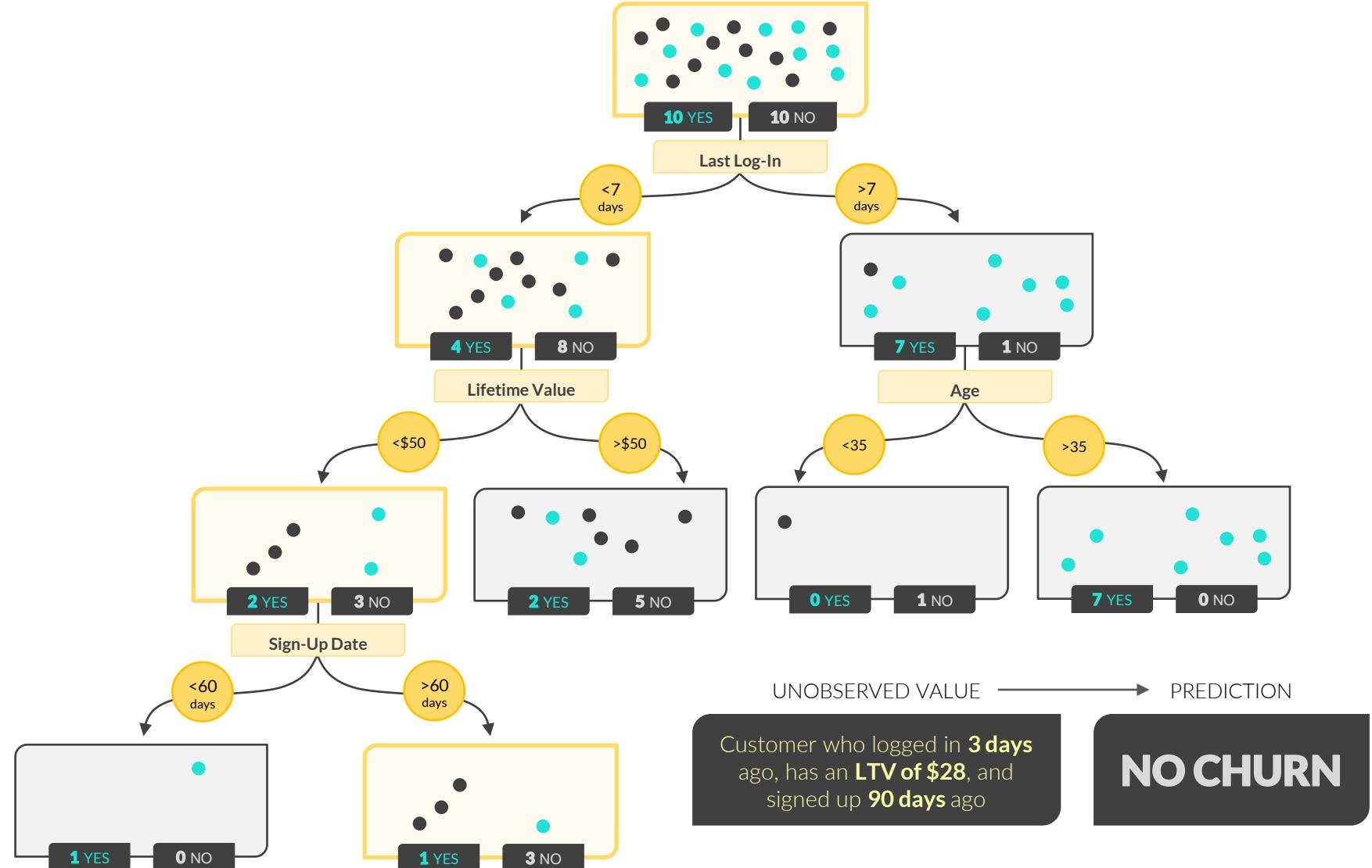
Naïve Bayes

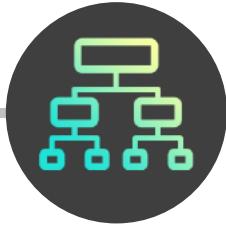
**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis





# DECISION TREES

K-Nearest Neighbors

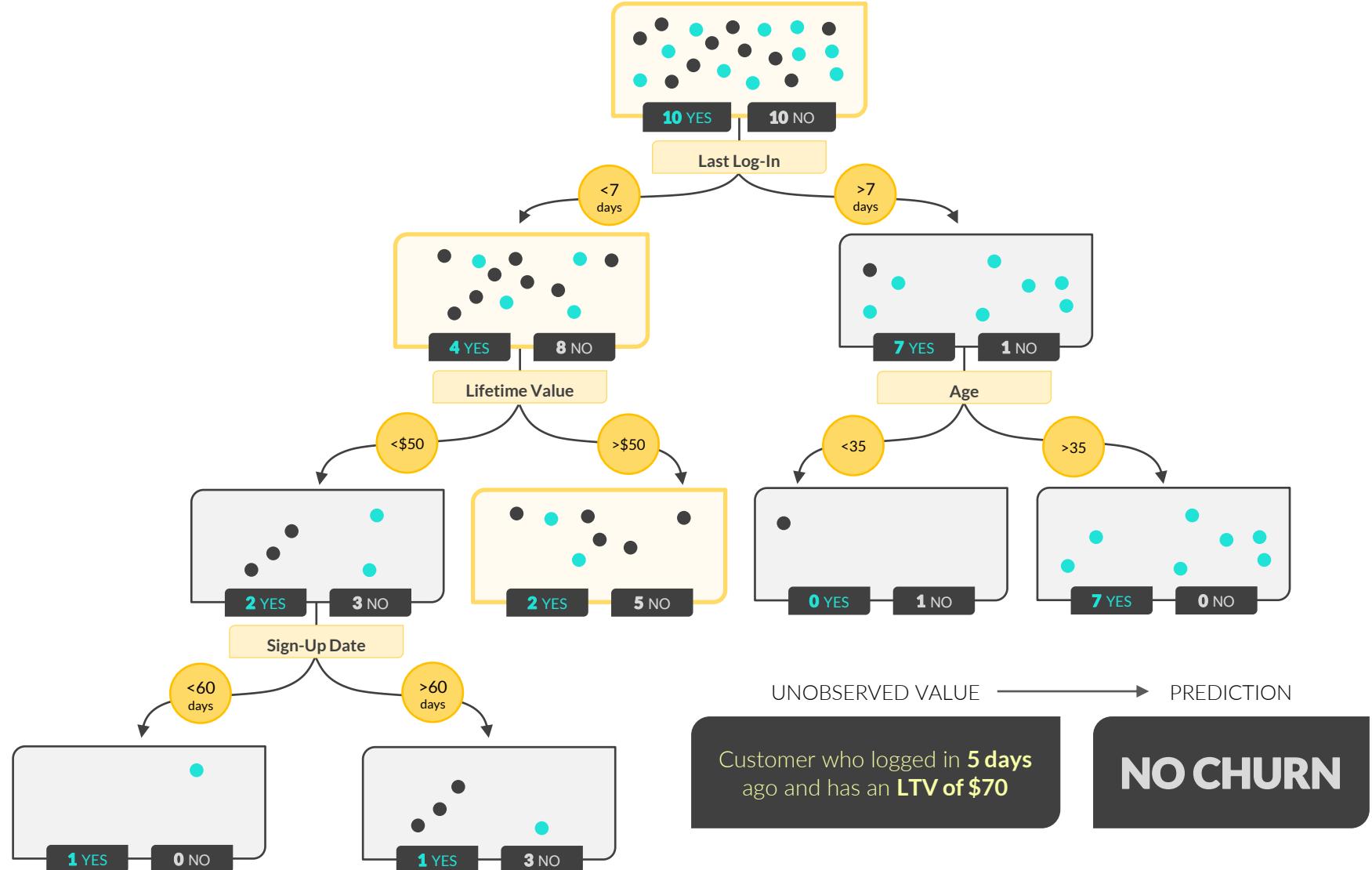
Naïve Bayes

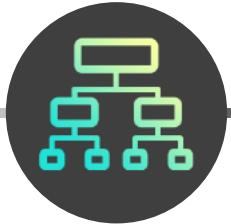
**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis





# DECISION TREES

K-Nearest Neighbors

Naïve Bayes

**Decision Trees**

Random Forests

Logistic Regression

Sentiment Analysis



How do we know when to stop splitting?

- Splitting too much can lead to overfitting, so you can adjust model inputs (known as “hyperparameters”) to limit things like **tree depth** or **leaf size**



How do we know where to split numerical IVs?

- In practice, decision trees will test different numerical splits and optimize based on information gain (just like testing individual IV splits)



Does the best first split *always* lead to the most accurate model?

- Not necessarily! That's why we often use a collection of *multiple* decision trees, known as a **random forest**, to maximize accuracy



# RANDOM FORESTS

K-Nearest Neighbors

Naïve Bayes

Decision Trees

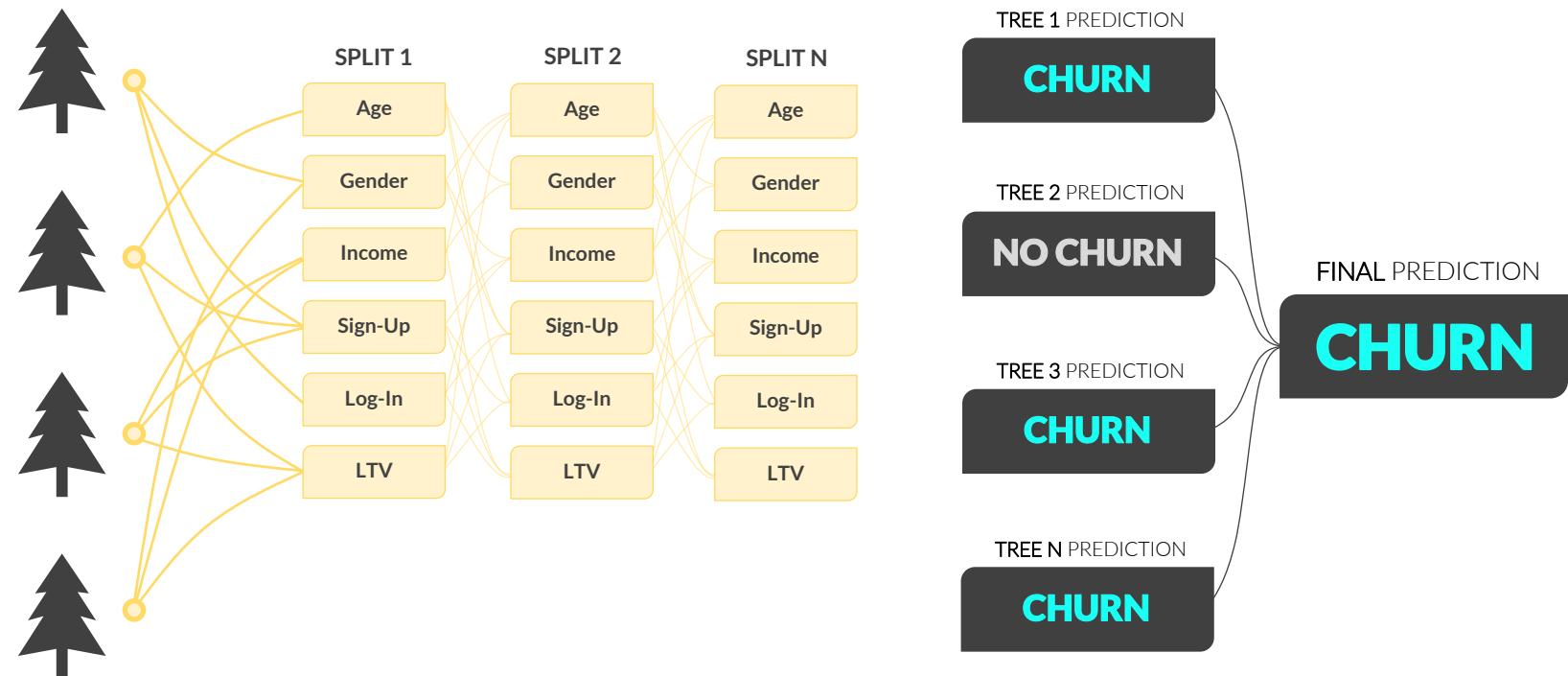
Random Forests

Logistic Regression

Sentiment Analysis

A **Random Forest** is a collection of individual decision trees, each built using a random subset of observations

- Each decision tree randomly selects variables to evaluate at each split
- Each tree produces a prediction, and the **mode**, or most frequent prediction, wins



# CASE STUDY: DECISION TREES



## THE SITUATION

You are the founder and CEO of **Trip Genie**, an online subscription service designed to connect global travelers with local guides.



## THE ASSIGNMENT

You'd like to better understand your customers and identify which types of behaviors can be used to help predict paid subscriptions.

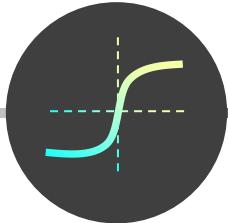
Your goal is to **build a decision tree** to help you predict subscriptions based on multiple factors (*newsletter, Facebook follow, time on site, sessions, etc.*).



## THE OBJECTIVES

1. Collect Training data at the customer-level
2. Split on each independent variable and compare changes in entropy
3. Build a decision tree to best predict subscriptions based on available IVs

# LOGISTIC REGRESSION



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

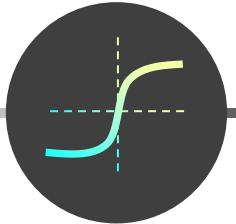
Sentiment Analysis

**Logistic Regression** is a classification technique used to predict the probability of a binary (true/false) outcome

- In its simplest form, logistic regression forms an **S-shaped curve between 0 - 1**, which represents the probability of a TRUE outcome for any given value of X
- The **likelihood function** measures how accurately a model predicts outcomes, and is used to optimize the “shape” of the curve
- Although it has the word “regression” in its name, logistic regression is not used for predicting numeric variables

## Example use cases:

- Flagging spam emails or fraudulent credit card transactions
- Determining whether to serve a particular ad to a website visitor



# LOGISTIC REGRESSION

K-Nearest Neighbors

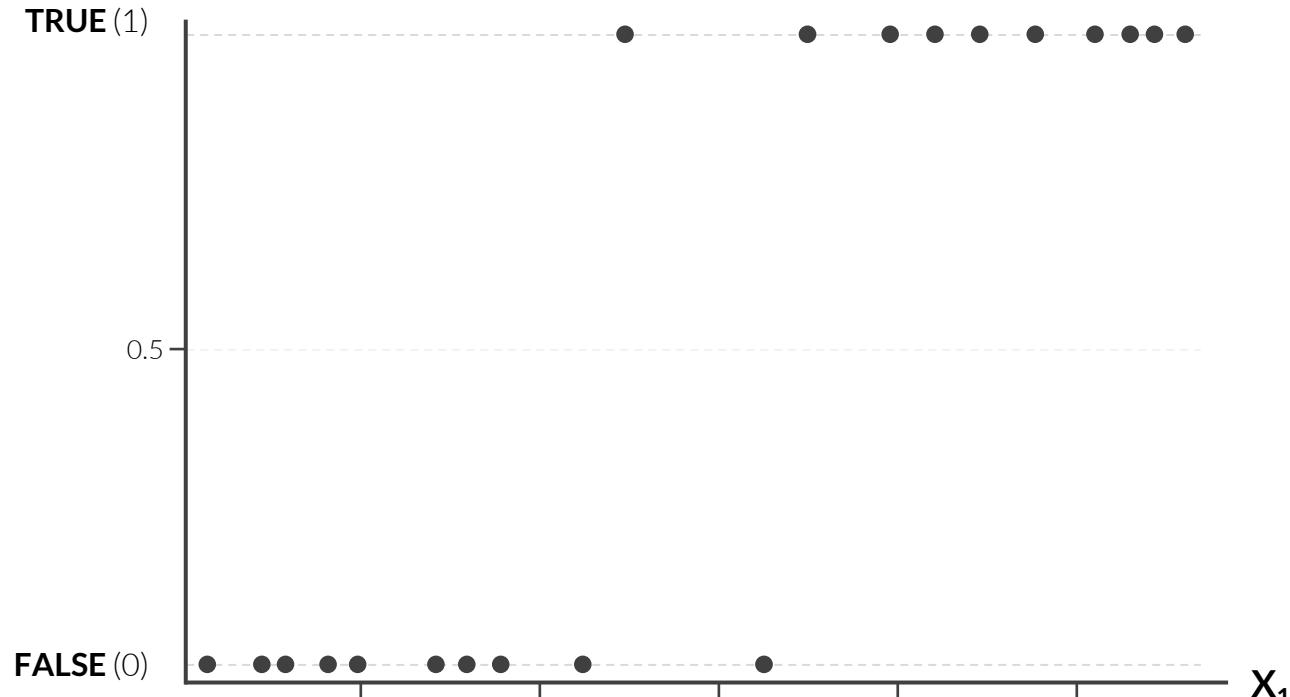
Naïve Bayes

Decision Trees

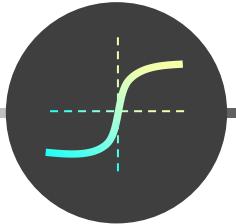
Random Forests

**Logistic Regression**

Sentiment Analysis



- Each dot represents an observed value, where **X is a numerical independent variable** and **Y is the binary outcome** (true/false) that we want to predict



# LOGISTIC REGRESSION

K-Nearest Neighbors

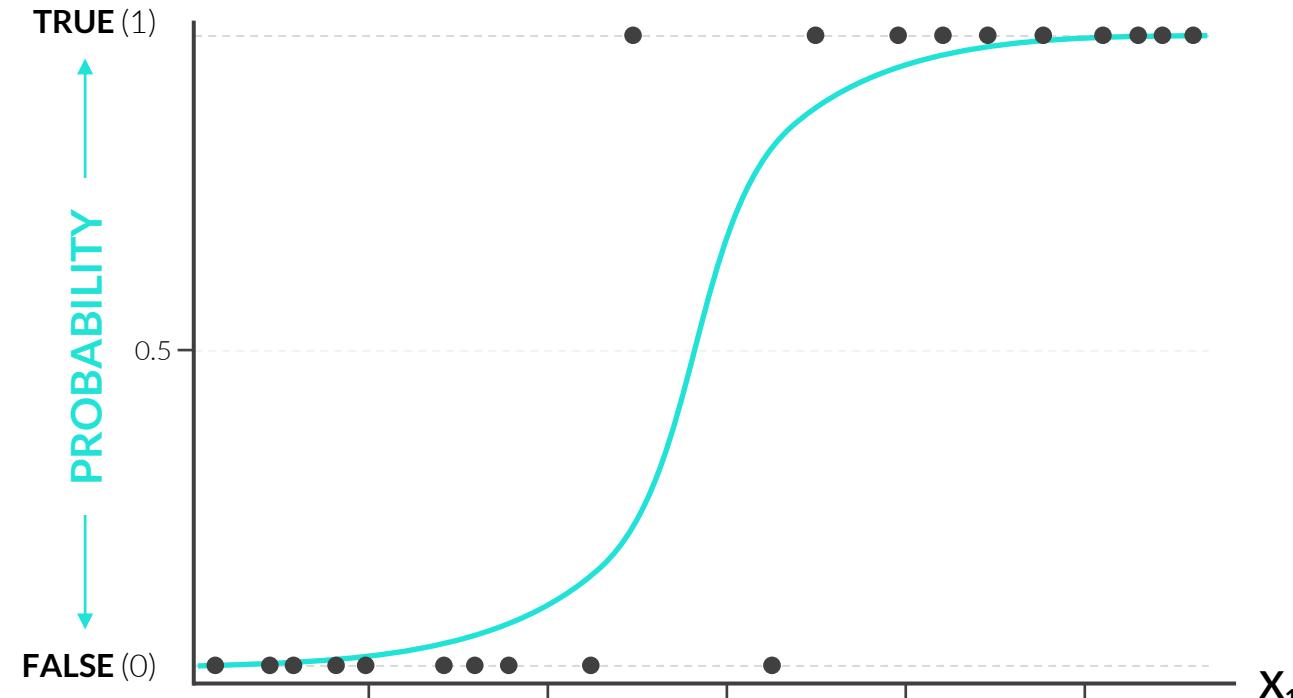
Naïve Bayes

Decision Trees

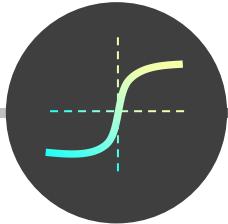
Random Forests

**Logistic Regression**

Sentiment Analysis



- Logistic regression plots the **best-fitting curve between 0 and 1**, which tells us the probability of Y being TRUE for any given value of  $X_1$



# LOGISTIC REGRESSION

K-Nearest Neighbors

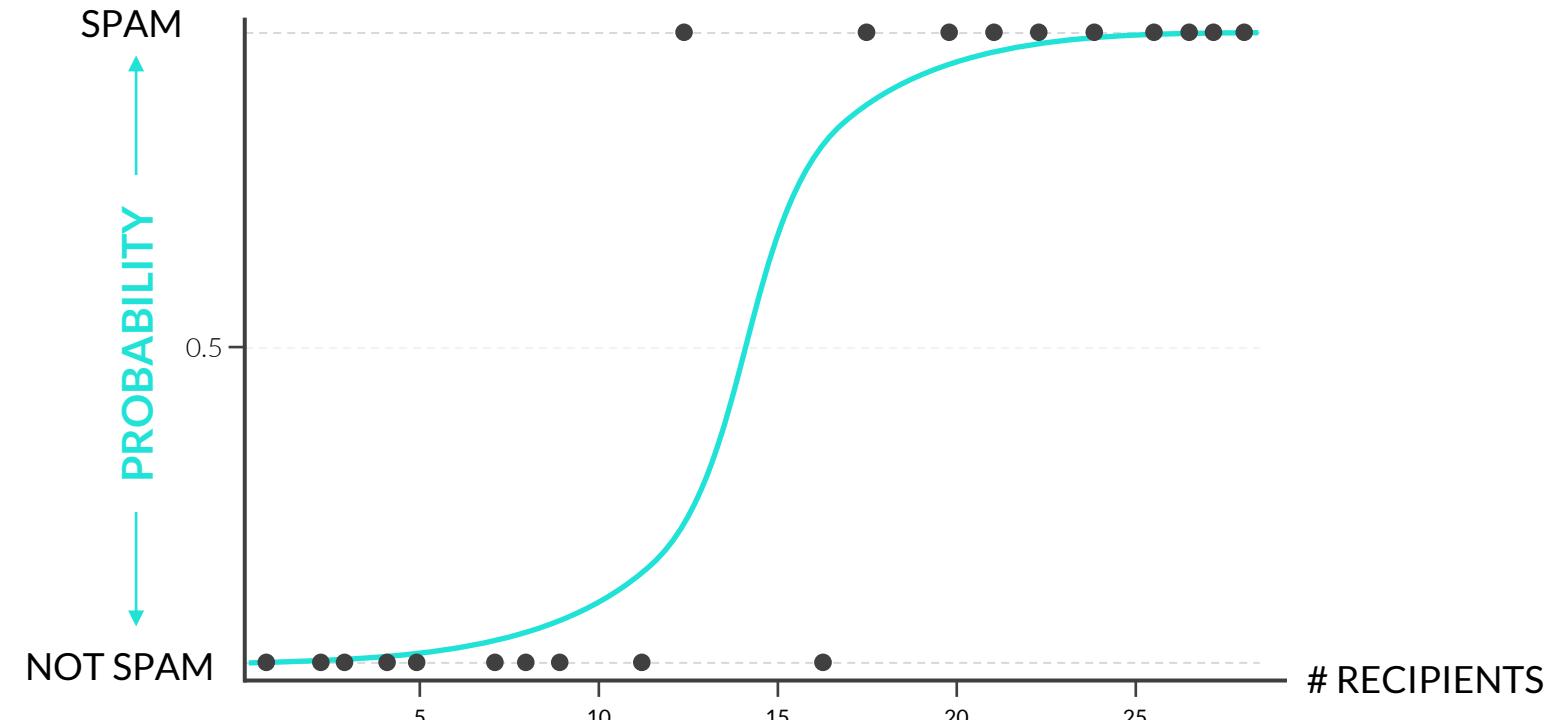
Naïve Bayes

Decision Trees

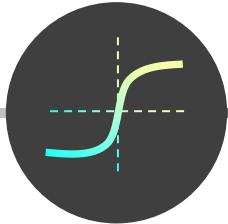
Random Forests

**Logistic Regression**

Sentiment Analysis



- Here we're using logistic regression to **predict if an email will be marked as spam**, based on the number of email recipients ( $X_1$ )



# LOGISTIC REGRESSION

K-Nearest Neighbors

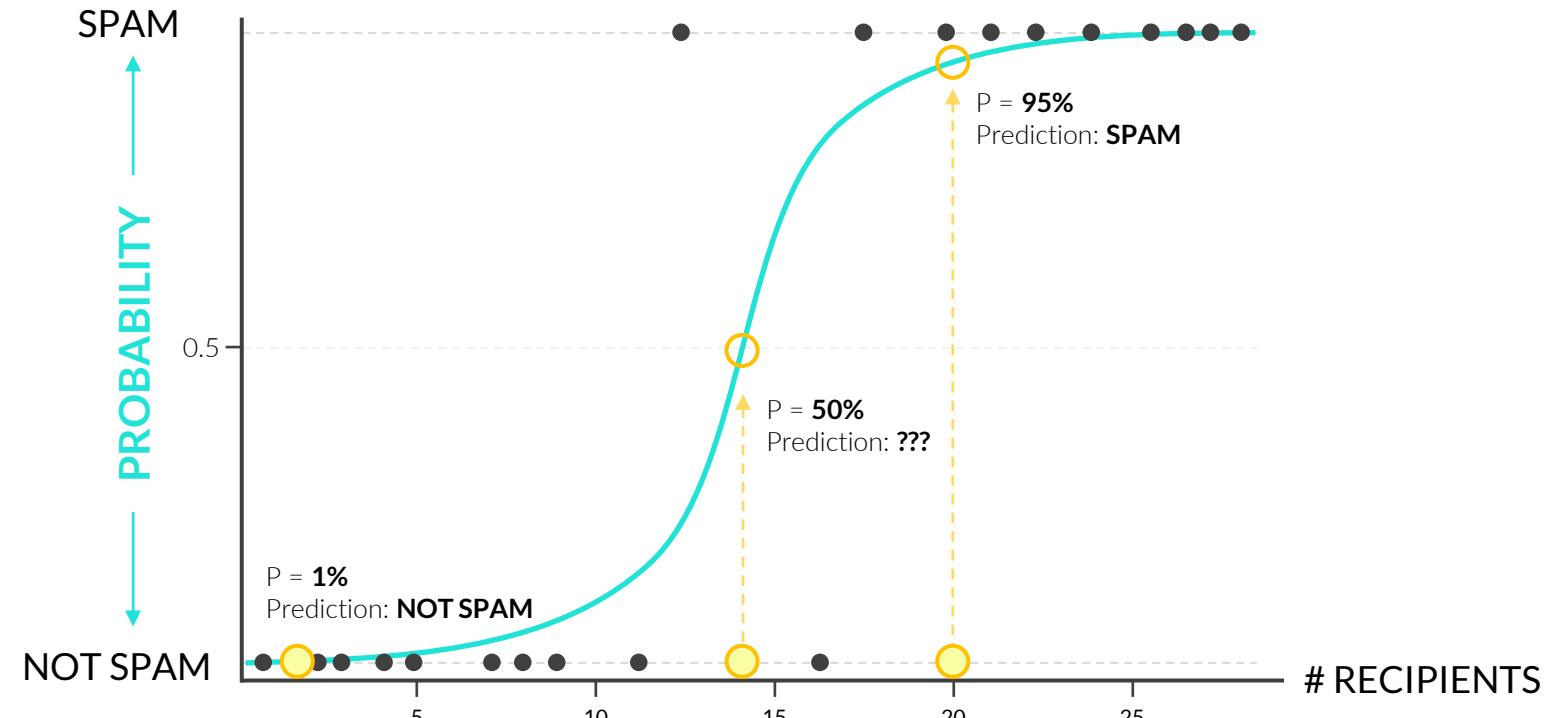
Naïve Bayes

Decision Trees

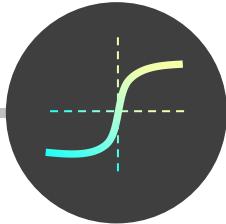
Random Forests

**Logistic Regression**

Sentiment Analysis



- Using this model, we can test unobserved values of  $X_1$  to predict the probability that  $Y$  is true or false (in this case the probability that an email is marked as spam)



# LOGISTIC REGRESSION

K-Nearest Neighbors

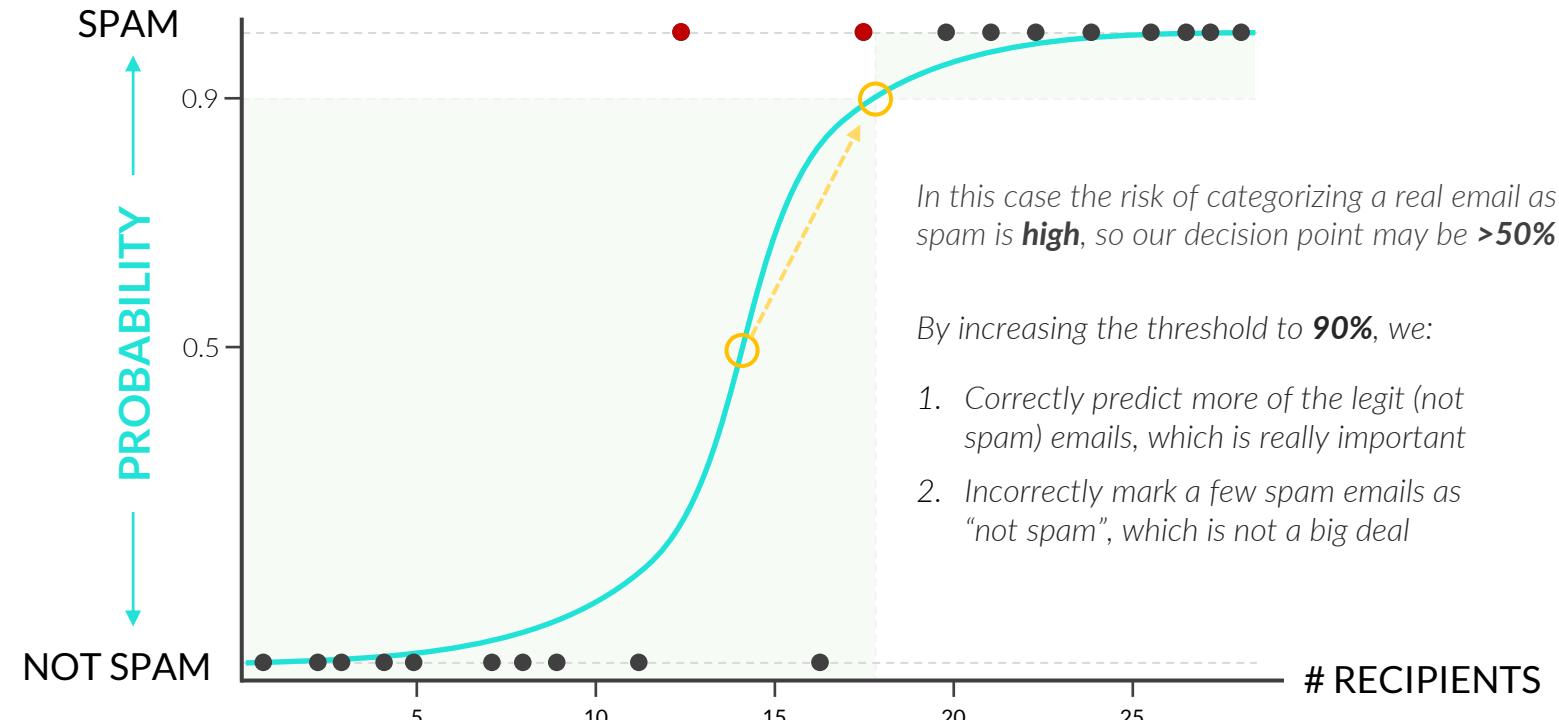
Naïve Bayes

Decision Trees

Random Forests

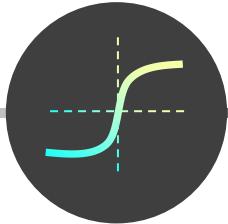
Logistic Regression

Sentiment Analysis



Is **50%** always the right decision point for logistic models?

- **No.** It depends on the relative risk of a **false positive** (incorrectly predicting a TRUE outcome) or **false negative** (incorrectly predicting a FALSE outcome)



# LOGISTIC REGRESSION

K-Nearest Neighbors

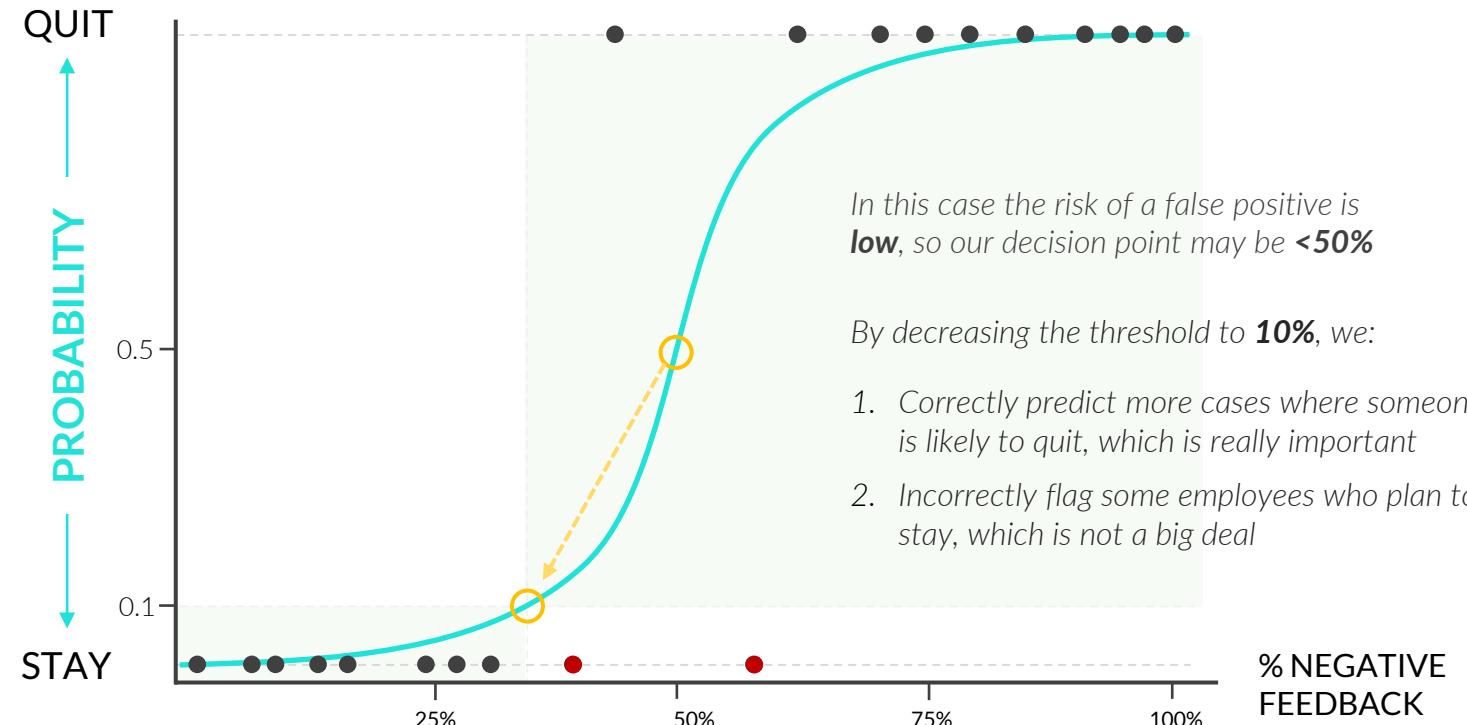
Naïve Bayes

Decision Trees

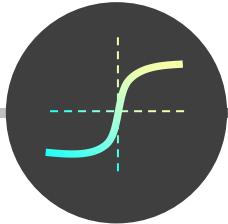
Random Forests

**Logistic Regression**

Sentiment Analysis



- Now consider a case where we're **predicting if an employee will quit** based on negative feedback from HR
- It's easier to train an employee than hire a new one, so the risk of a false positive is **low** but the risk of a false negative (**incorrectly predicting someone will stay**) is **high**



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

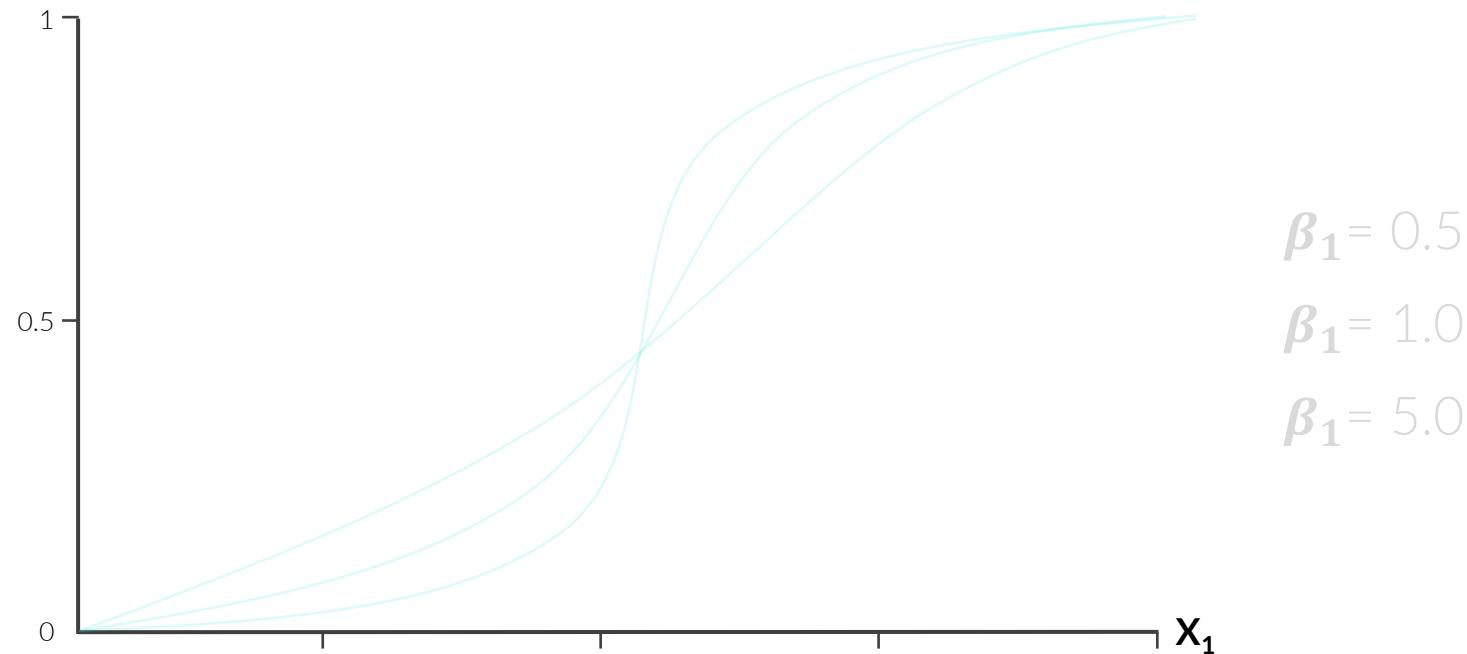
**Logistic Regression**

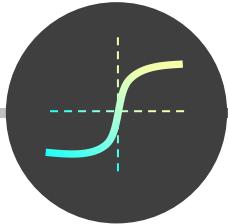
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

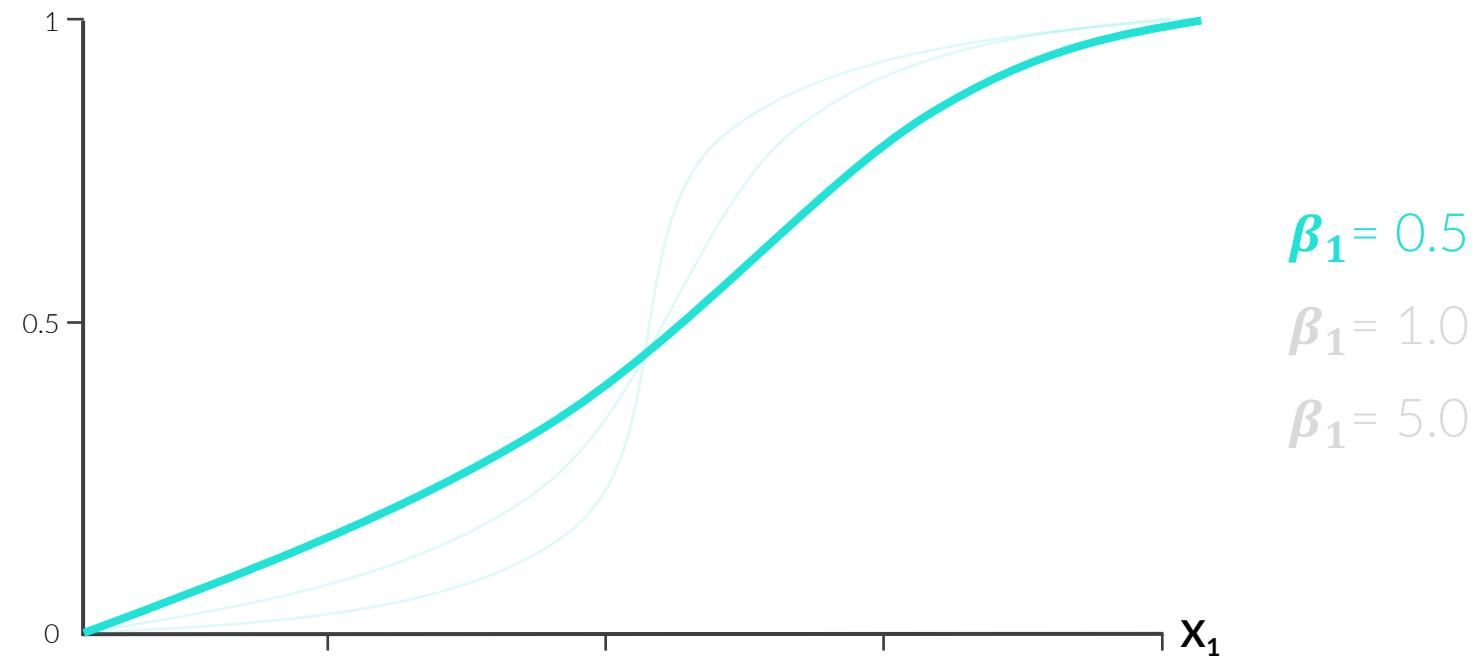
**Logistic Regression**

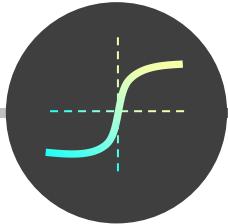
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

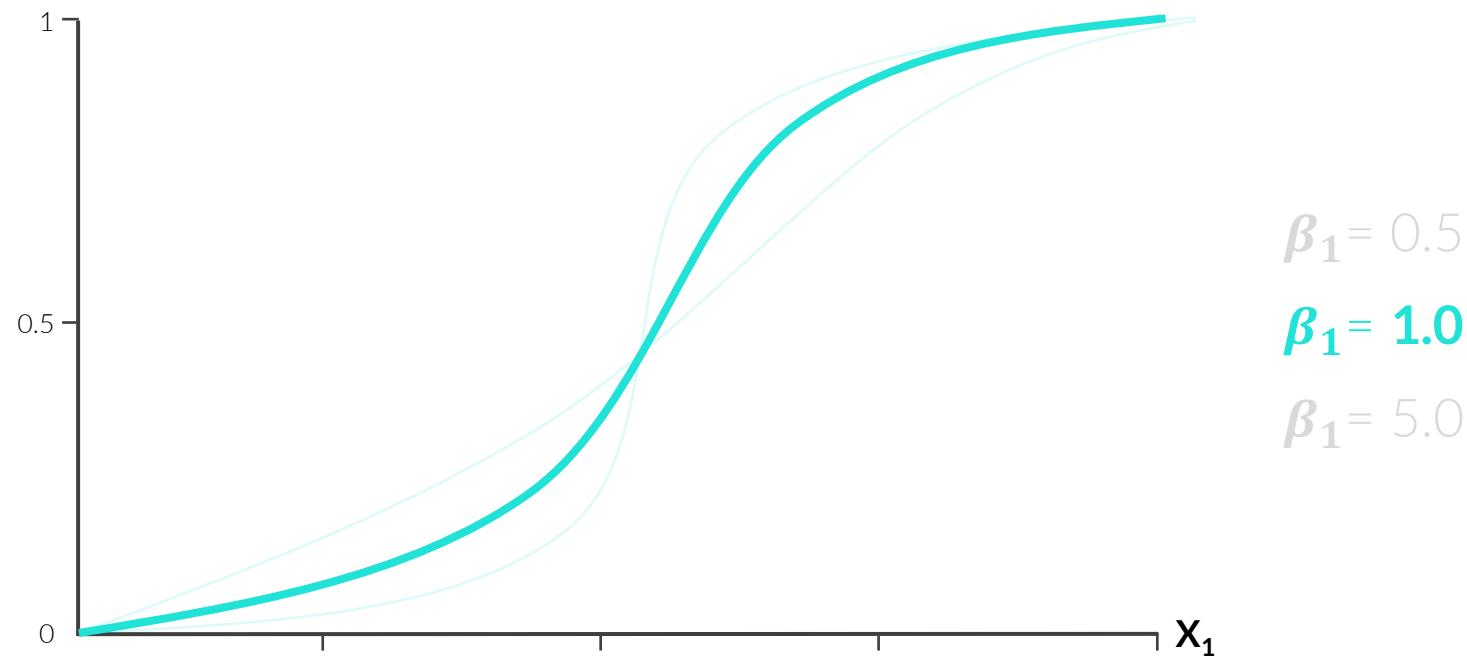
**Logistic Regression**

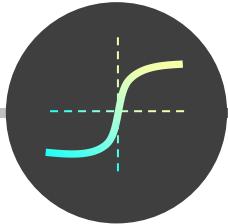
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

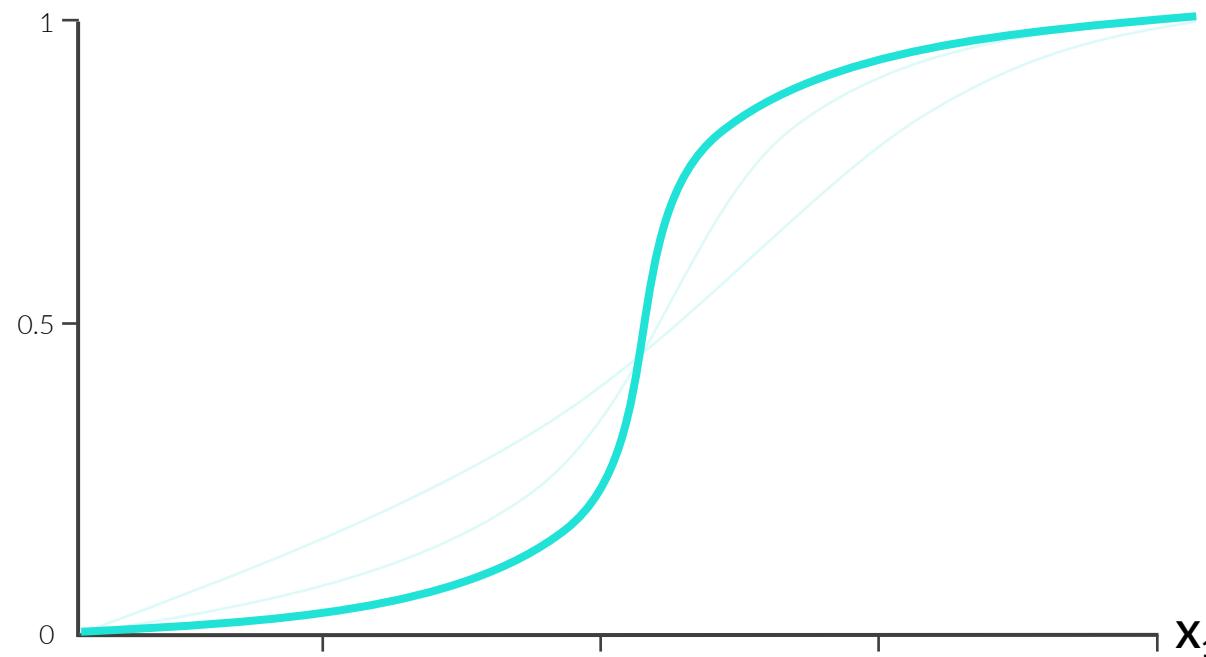
**Logistic Regression**

Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

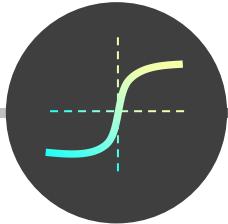
Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)



$$\beta_1 = 0.5$$

$$\beta_1 = 1.0$$

$$\beta_1 = 5.0$$



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

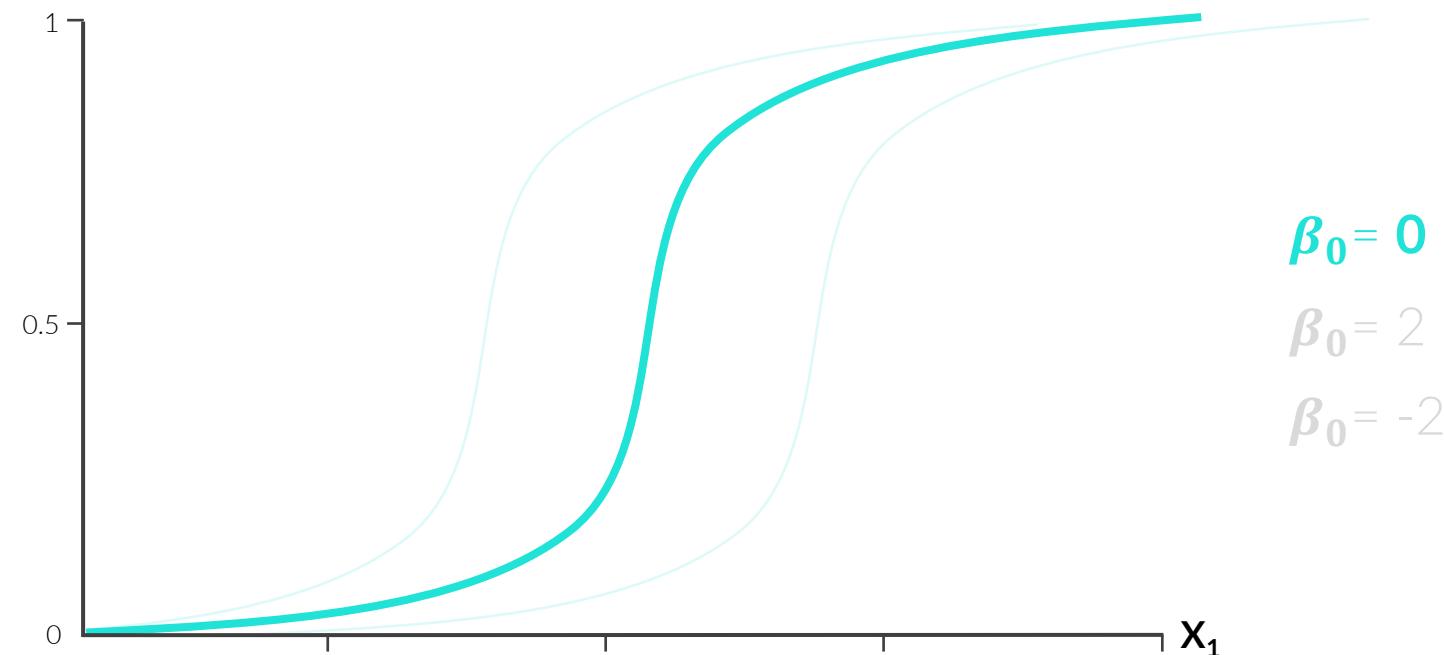
**Logistic Regression**

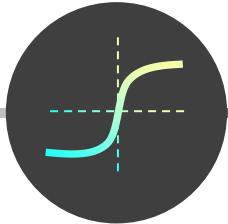
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

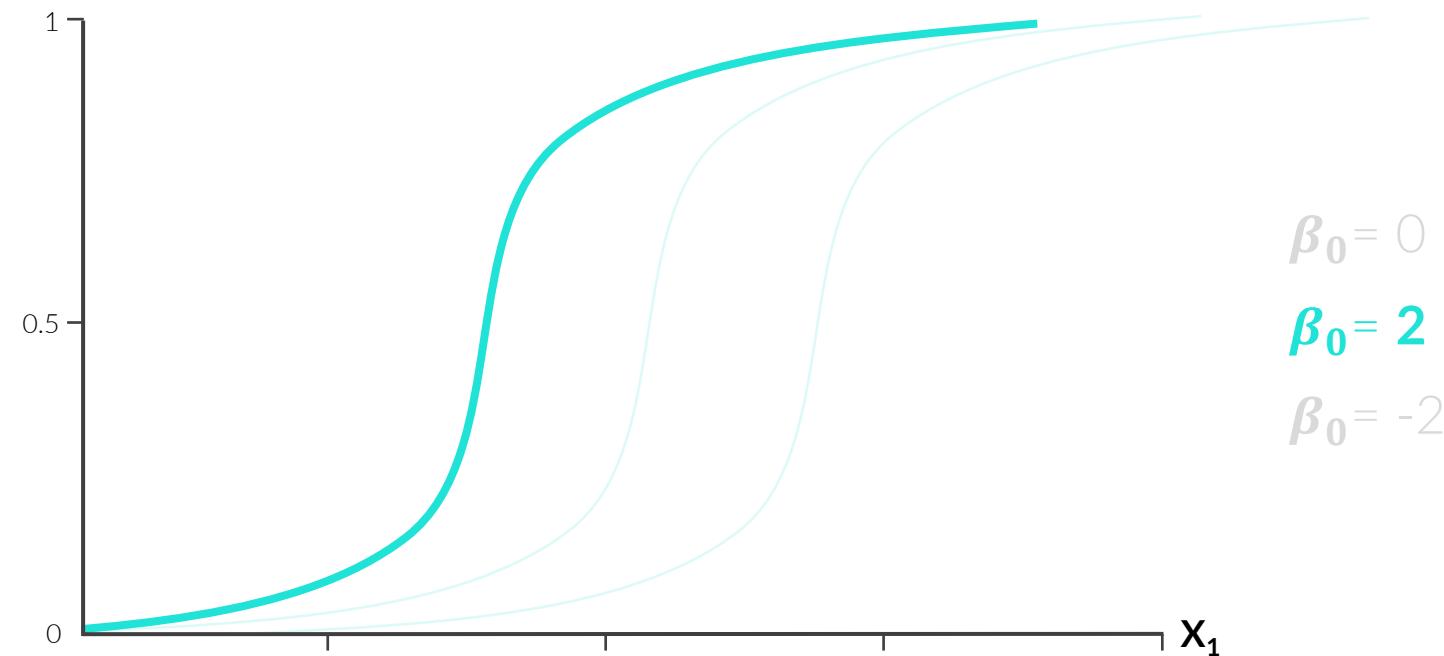
**Logistic Regression**

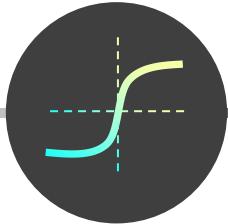
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

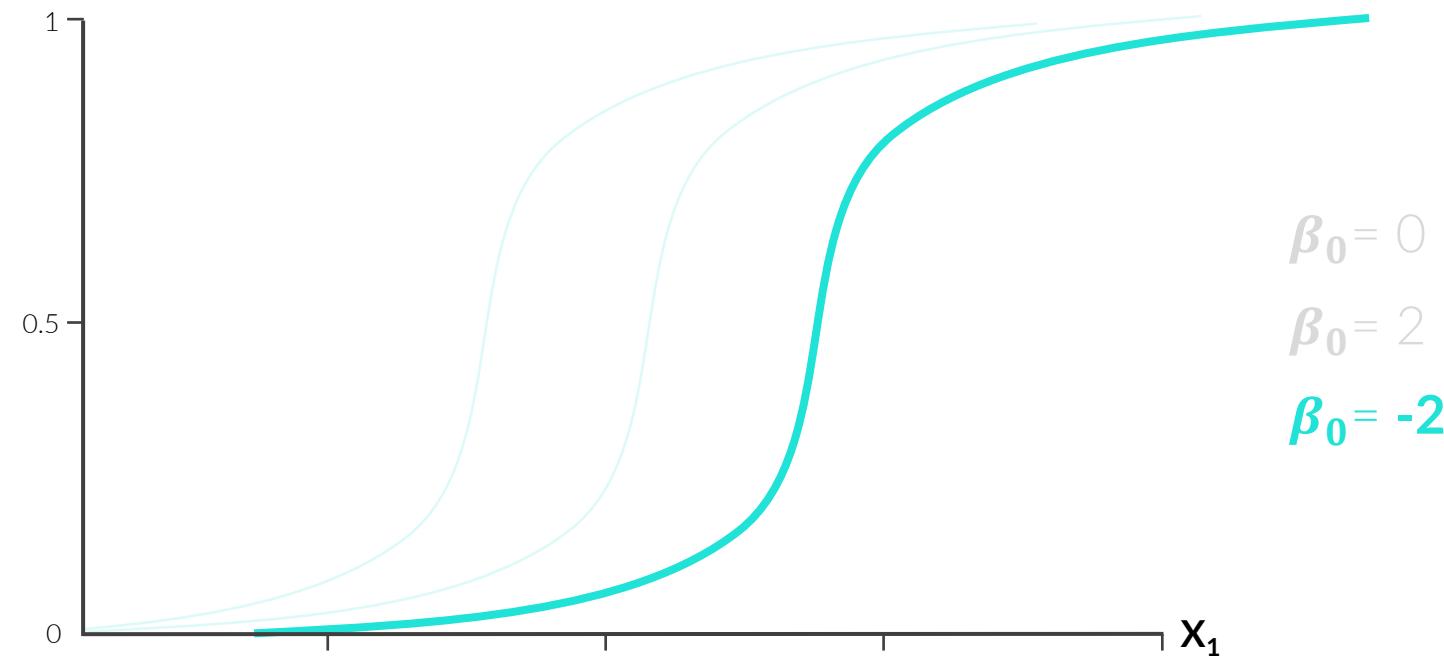
**Logistic Regression**

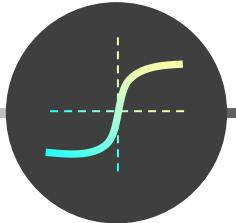
Sentiment Analysis

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

Linear equation, where  
 $\beta_0$  is the intercept,  $X$  is  
the IV value and  $\beta_1$  is  
the weight (or slope)





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

**Logistic Regression**

Sentiment Analysis

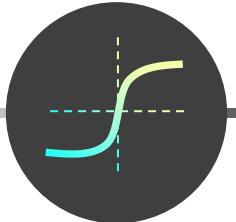


How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting  $Y$ , based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values

		Actual Observation	
		$Y = 1$	$Y = 0$
Model Output	$\sim 1$	HIGH	LOW
	$\sim 0$	LOW	HIGH

- When our model output is close to the actual  $Y$ , we want likelihood to be **HIGH** (near **1**)
- When our model output is far from the actual  $Y$ , we want likelihood to be **LOW** (near **0**)



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



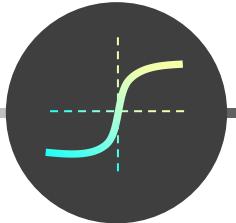
How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting Y, based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values

		Actual Observation	
		Y = 1	Y = 0
Model Output	~1	HIGH	LOW
	~0	LOW	HIGH

LIKELIHOOD FUNCTION:

$$\text{(output)}^y * (1 - \text{output})^{1-y}$$



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

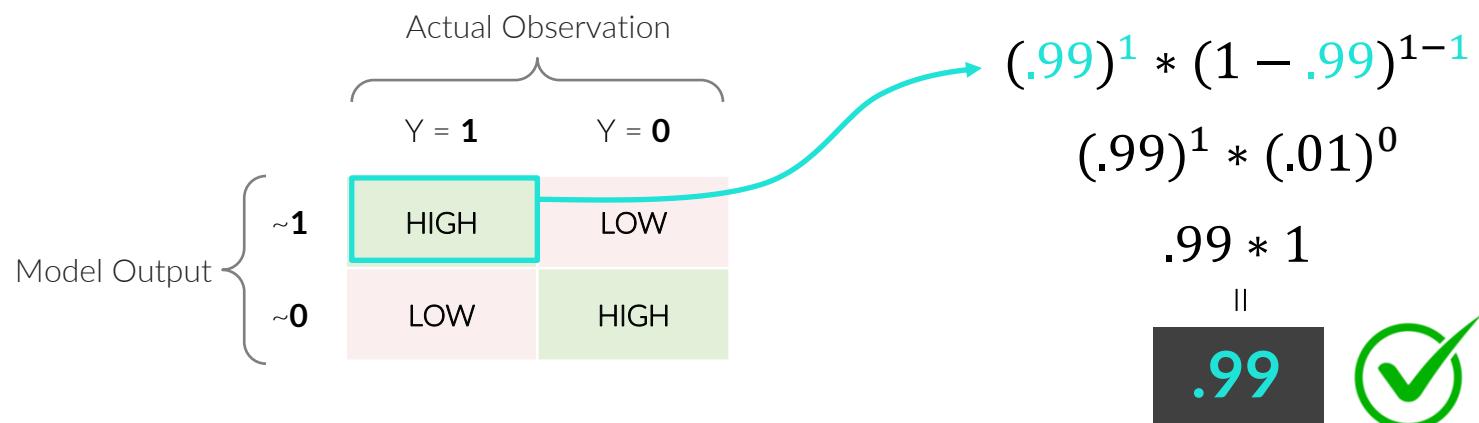
Logistic Regression

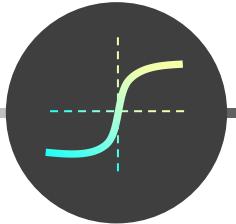
Sentiment Analysis



How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting Y, based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

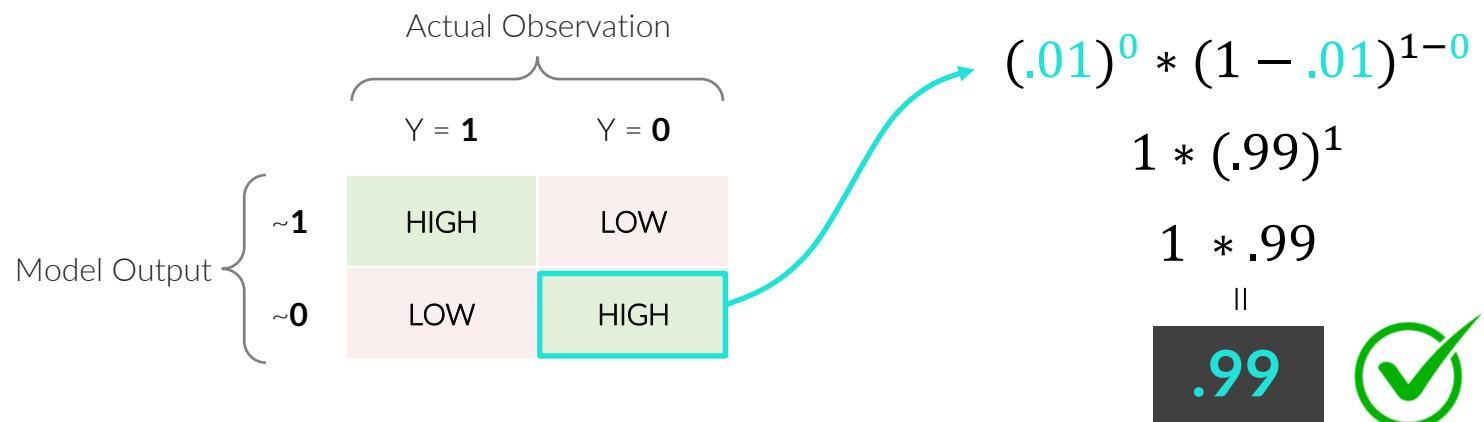
**Logistic Regression**

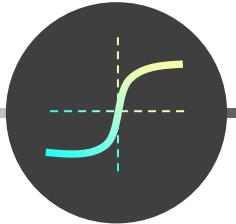
Sentiment Analysis



How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting Y, based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting Y, based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values

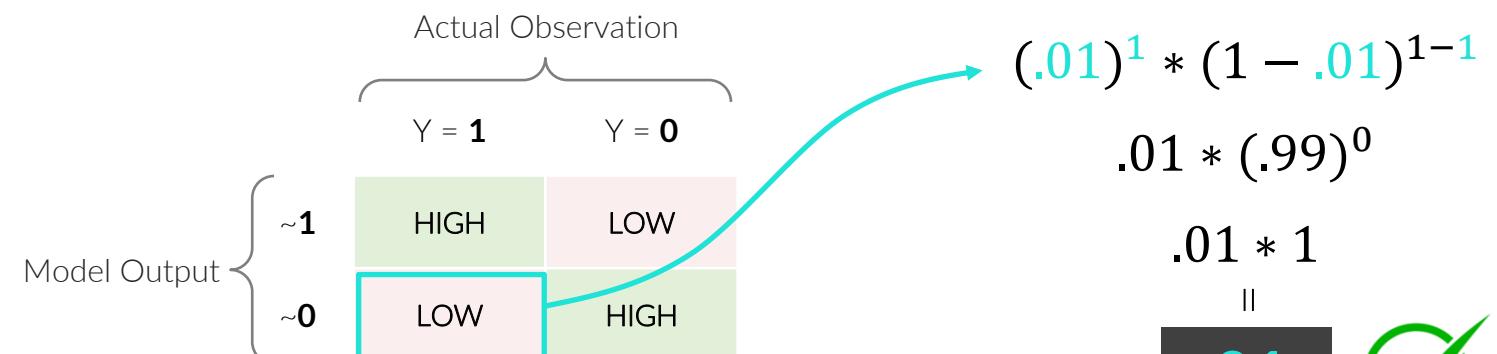
$$(output)^y * (1 - output)^{1-y}$$

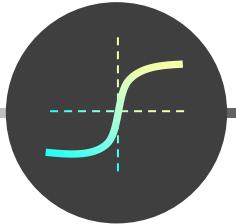
$$(.01)^1 * (1 - .01)^{1-1}$$

$$.01 * (.99)^0$$

$$.01 * 1$$

||  
.01





# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

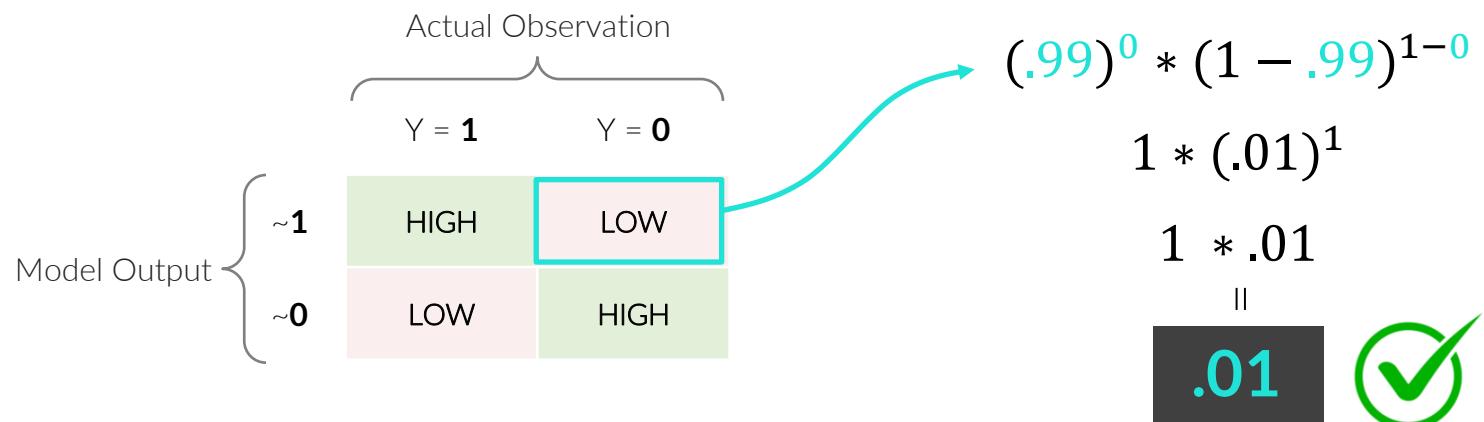
Logistic Regression

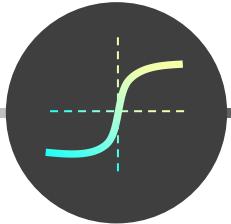
Sentiment Analysis



How do we determine the “right” values for  $\beta$ ? **LIKELIHOOD!**

- **Likelihood** is a metric that tells us how good our model is at correctly predicting Y, based on the shape of the curve
- This is where **machine learning** comes in; instead of human trial-and-error, an algorithm determines the best weights to maximize likelihood using observed values





# LOGISTIC REGRESSION

K-Nearest Neighbors

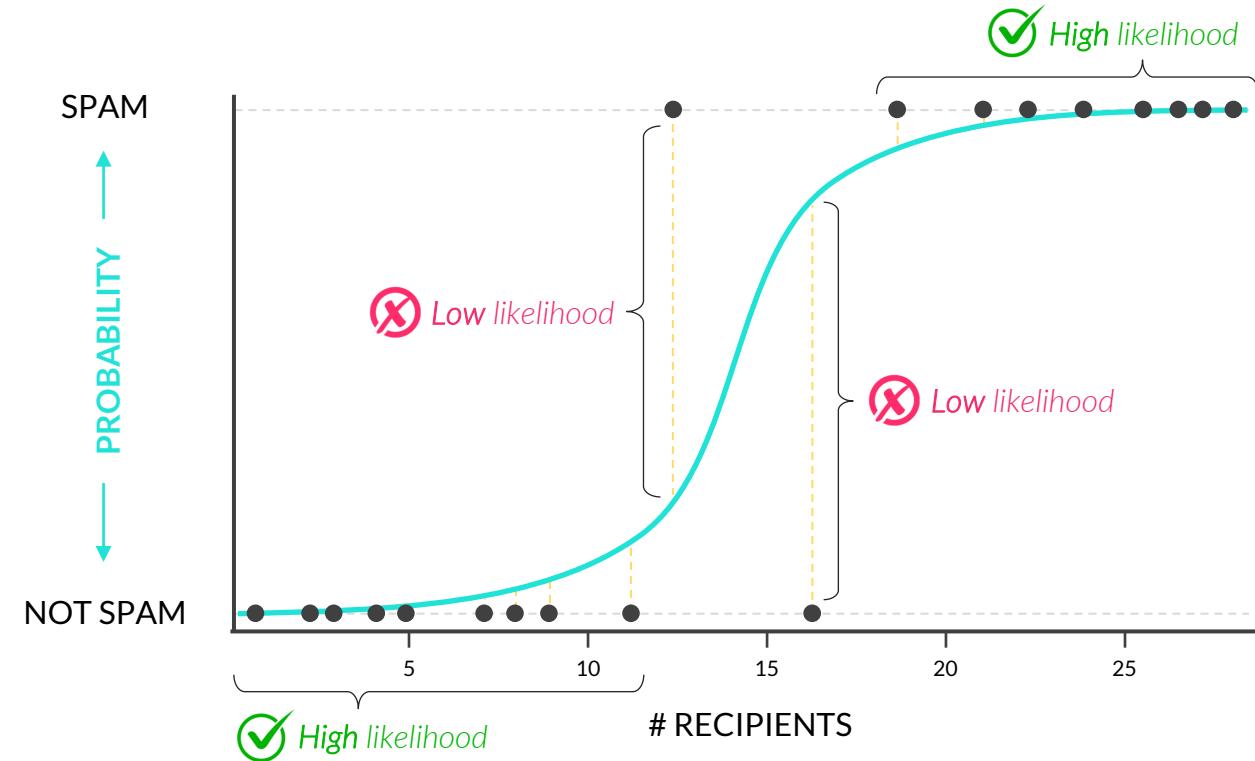
Naïve Bayes

Decision Trees

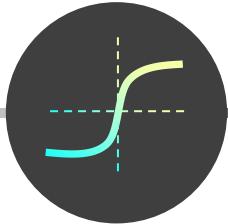
Random Forests

**Logistic Regression**

Sentiment Analysis



- Observations closest to the curve have the **highest likelihood values** (and vice versa), so maximizing total likelihood allows us to find the curve that fits our data best



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

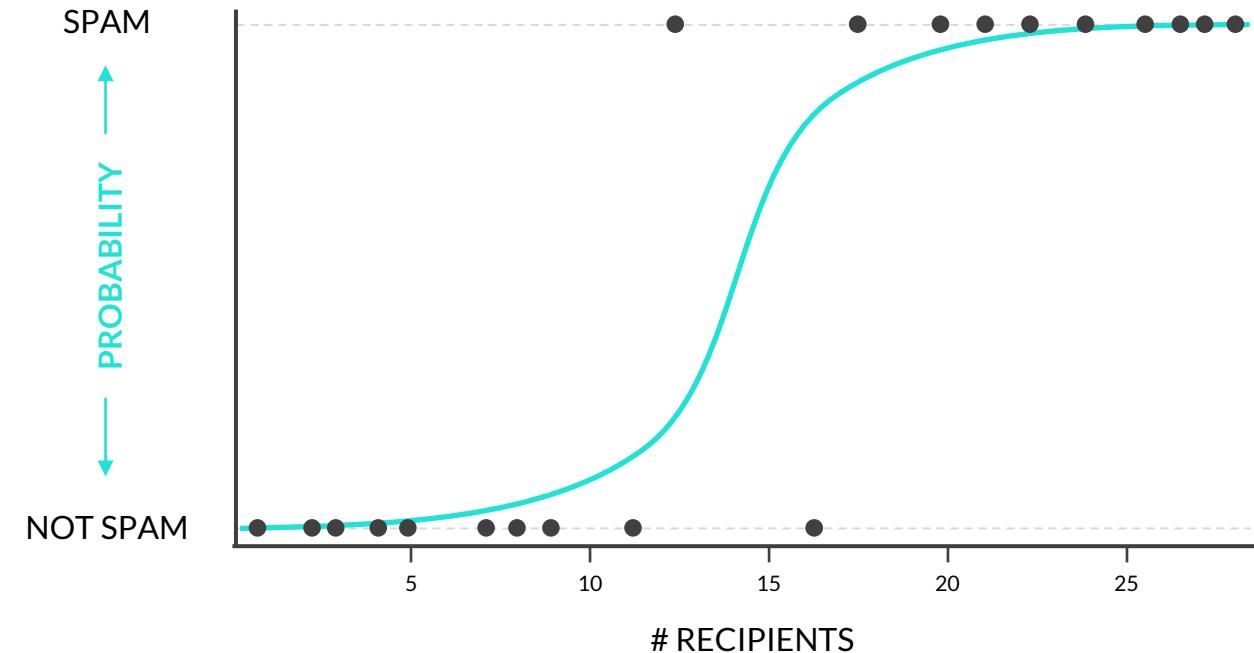
Random Forests

Logistic Regression

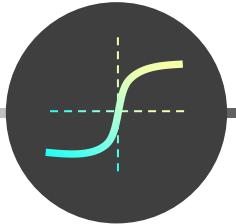
Sentiment Analysis



But what if *multiple X variables* could help predict Y?



- **# of Recipients** can help us detect spam, but so can other variables like the **number of typos, count of words like “free” or “bonus”, sender reputation score**, etc.



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

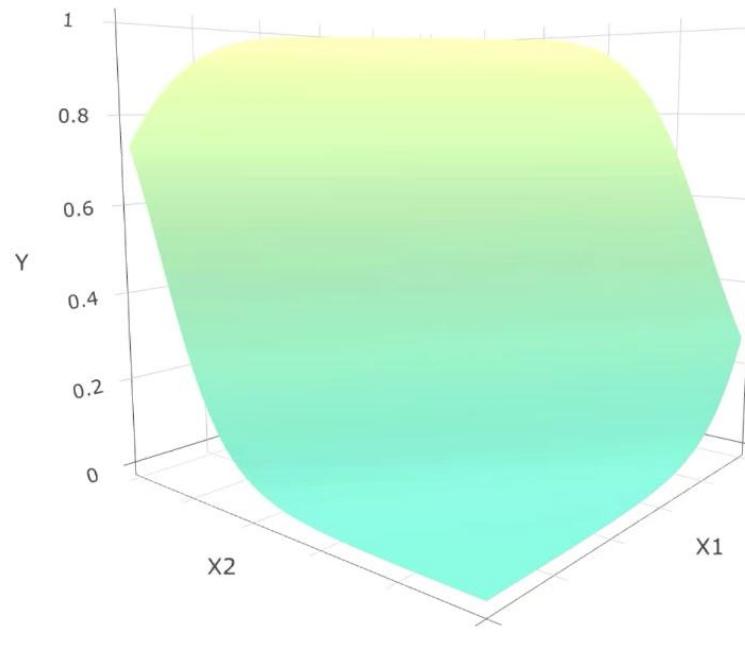
Random Forests

**Logistic Regression**

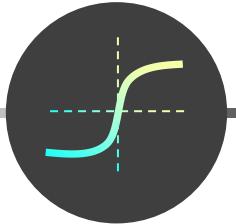
Sentiment Analysis



But what if **multiple X variables** could help predict Y?



- Logistic regression can handle multiple independent variables, but the visual interpretation breaks down at >2 IV's (*this is why we need machine learning!*)



# LOGISTIC REGRESSION

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



But what if **multiple X variables** could help predict Y?

Makes the output fall  
between **0** and **1**

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n)}}$$

A diagram illustrating the logistic regression formula. A horizontal arrow points from the term  $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n$  to the number 1. A curly brace under this same term is labeled "Weighted independent variables ( $x_1, x_2 \dots x_n$ )".

- Logistic regression is about finding the **best combination of weights** ( $\beta_1, \beta_2 \dots \beta_n$ ) for a given set of independent variables ( $x_1, x_2 \dots x_n$ ) to maximize the likelihood function

# CASE STUDY: LOGISTIC REGRESSION



## THE SITUATION

You've just been promoted to Marketing Manager for **Lux Dining**, a wildly popular international food blog.



## THE ASSIGNMENT

The CMO is concerned about unsubscribe rates and thinks it may be related to the frequency of emails your team has been sending.

Your job is to use **logistic regression** to plot this relationship and predict if a user will unsubscribe based on the number of weekly emails received.



## THE OBJECTIVES

1. Collect customer data containing email frequency and subscription activity
2. Plot the logistic regression curve which maximizes likelihood
3. Determine the appropriate email frequency (*based on a 50% decision point*)

# SENTIMENT ANALYSIS



# SENTIMENT ANALYSIS

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

**Sentiment analysis** is a technique used to determine the emotional tone or “sentiment” behind text-based data

- Sentiment analysis often falls under **Natural Language Processing** (NLP), but is typically applied as a classification technique
- Sentiment models typically use a “**bag of words**” approach, which involves calculating the frequency or presence (1/0) of key words to convert text (which *models struggle with*) into numerical inputs
- Unlike other classification models, you must “hand-score” the sentiment (DV) values for your Training data, which your model will learn from

**Example use cases:**

- Understanding the tone of product reviews posted by customers
- Analyzing open-ended survey responses



# SENTIMENT ANALYSIS

K-Nearest Neighbors

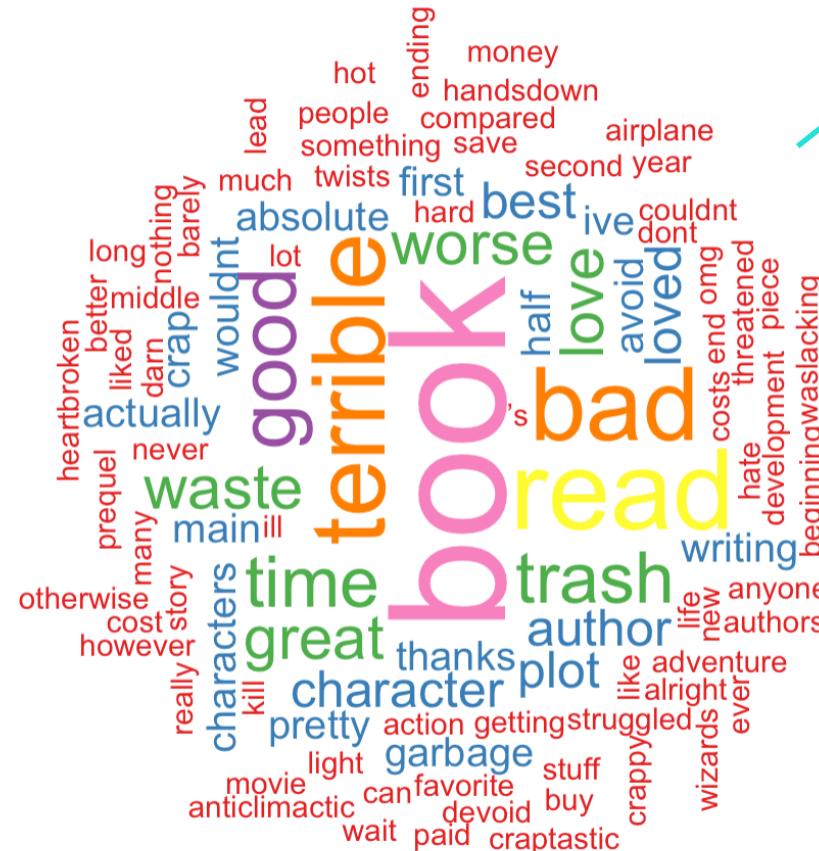
Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



This **word cloud** is \*technically\* a very simple version of sentiment analysis

## Limitations:

- Based entirely on word count
- Straight-forward but not flexible
- Requires manual interpretation
- Subject to inconsistency/human error



# SENTIMENT ANALYSIS

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

The first step in any sentiment analysis is to **clean** and **QA** the text to remove noise and isolate the most meaningful information:

- Remove **punctuation, capitalization** and **special characters**
- Correct **spelling** and **grammatical errors**
- Use proper **encoding** (i.e. UTF-8)
- **Lemmatize** or **stem** (remove grammar tense, convert to “root” term)
- Remove **stop words** (“a”, “the”, “or”, “of”, “are”, etc.)

*The computer is running hot because I'm mining bitcoin!*

*the computer is running hot because I'm mining bitcoin!*

**computer run hot mine bitcoin**

**NOTE:** This process can vary based on the context; for example, you may want to preserve capitalization or punctuation if you care about measuring intensity (i.e. “**GREAT!!**” vs. “**great**”), or choose to allow specific stop words or special characters



# SENTIMENT ANALYSIS

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis

Once the text has been cleaned, we can transform our text into numeric data using a “**bag of words**” approach:

- Split cleaned text into individual words (*this is known as **tokenization***)
- Create a new column with a **binary flag (1/0)** for each word
- Manually **assign sentiment** for observations in your Training data
- Apply any **classification technique** to predict sentiment for unobserved text

Each word is an *independent variable*

	hate	garbage	love	book	sentiment
I HATE this garbage!	1	1	0	0	negative
I love this book.	0	0	1	1	positive
This book is garbage	0	1	0	1	???

*Sentiment is our dependent variable*

*Observed values*

*Unobserved value*



# SENTIMENT ANALYSIS

K-Nearest Neighbors

Naïve Bayes

Decision Trees

Random Forests

Logistic Regression

Sentiment Analysis



How do you address **language nuances** like ambiguity, double negatives, slang or sarcasm?

- Generally speaking, the more observations you score in your Training data, the better your model will be at detecting these types of nuances
- No sentiment model is perfect, but feature engineering and advanced techniques can help improve accuracy for more complex cases

"If you like watching paint dry, you'll love this movie!"



If **you** like watching **ing** paint **dry**  
**you'll** love **this** movie



**like watch paint**  
**dry love movie**

"The new version is awfully good, not as bad as expected!"



The **new** version **is** awfully  
good **not as** bad **as** expected



**new version awful**  
**good bad expect**

# CASE STUDY: SENTIMENT ANALYSIS

---



## THE SITUATION

You're an accomplished author and creator of the hit series **Bark Twain the Data Dog**, featuring a feisty chihuahua who uses machine learning to solve crimes.



## THE ASSIGNMENT

Reviews for the latest book in the Bark Twain series are coming in, and they aren't looking great...

To apply a bit more rigor to your analysis and automate scoring for future reviews, you've decided to use this feedback to build a basic **sentiment model**.



## THE OBJECTIVES

1. Collect raw text reviews and tokenize key words
2. Hand-score the sentiment for observed values in the Training data
3. Apply any classification technique and validate using Test data

# MODEL SELECTION & TUNING

# MODEL SELECTION & TUNING



In this section we'll discuss techniques for **selecting & tuning** classification models, including hyperparameter optimization, class balancing, confusion matrices and more

## TOPICS WE'LL COVER:

Hyperparameters

Imbalanced Classes

Confusion Matrix

Selection & Drift

## COMMON USE CASES:

- *Adjusting model parameters to optimize performance*
- *Rebalancing classes to reduce bias*
- *Comparing performance across multiple classification models to select the best performing option*
- *Retraining models with fresh data to minimize drift*

# HYPERPARAMETERS

## Hyperparameters

## Imbalanced Classes

## Confusion Matrix

## Model Selection

## Model Drift

**Hyperparameters** are inputs/settings you can control while training a model

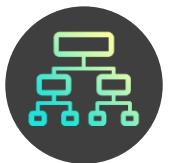
- Adjusting parameters to maximize accuracy is known as “*hyperparameter optimization*”



K-Nearest  
Neighbors



Naïve  
Bayes



Decision  
Trees



Random  
Forests



Logistic  
Regression

Examples:

- K
- Distance definition
- NN Algorithm

Examples:

- Smoothing parameters

Examples:

- Criterion (entropy, gini)
- Tree depth
- Leaf size
- Max Features
- Min gain

Examples:

- # Trees
- # Samples
- # Features
- Re-Sampling

Examples:

- Intercept
- Penalty Term
- Solver

# IMBALANCED CLASSES

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

**Imbalanced classes** occur when you are predicting outcomes with *drastically* different frequencies in the Training data

- The class or outcome which occurs more frequently is known as the **majority class**, while the class which occurs less frequently is the **minority class**
- Imbalanced classes can bias a model towards *always* predicting the majority class, since it often yields the best overall accuracy (*i.e.* 99%+)
- This is a significant issue when predicting very rare and very *important* events, like a nuclear meltdown

There are several ways to balance classes in your Training data, including **up-sampling**, **down-sampling** and **weighting**

# IMBALANCED CLASSES

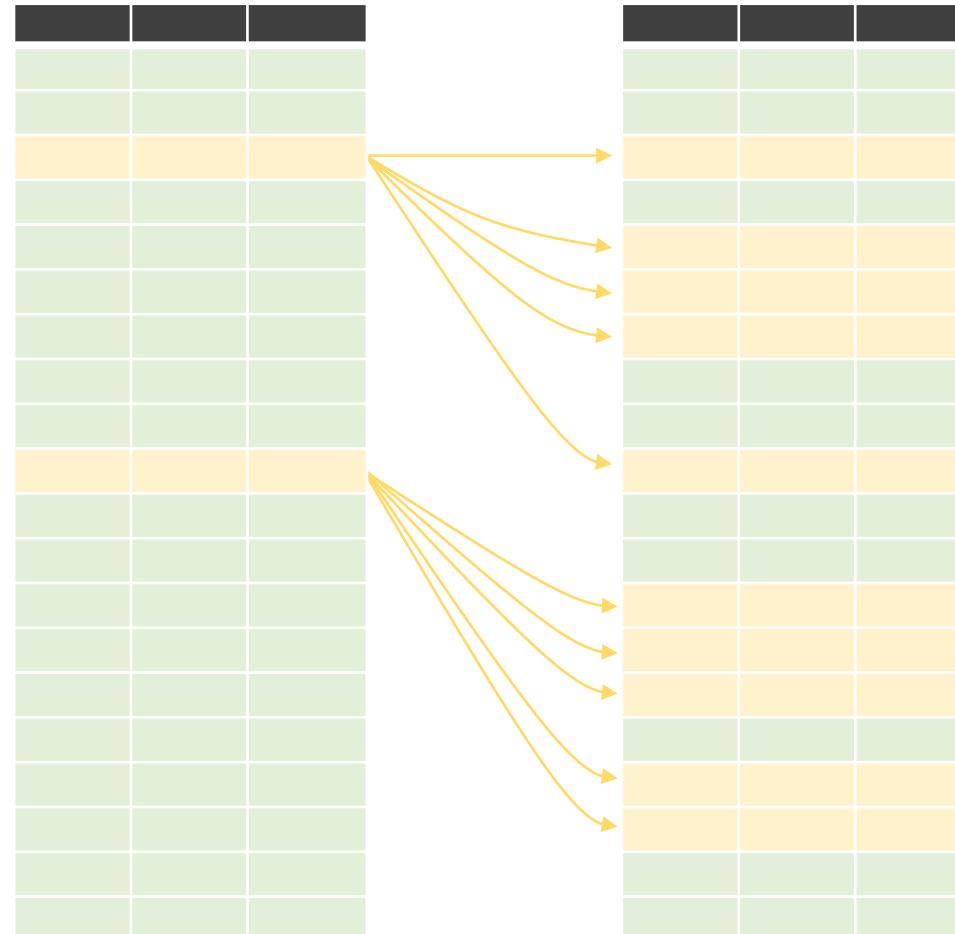
Hyperparameters

**Imbalanced Classes**

Confusion Matrix

Model Selection

Model Drift



**Up-sampling**

*Minority class observations are duplicated to balance the data*

**Down-sampling**

*Majority class observations are randomly removed to balance the data*

**Weighting**

*For models that randomly sample observations (random forests), increase the probability of selecting the minority class*

# IMBALANCED CLASSES

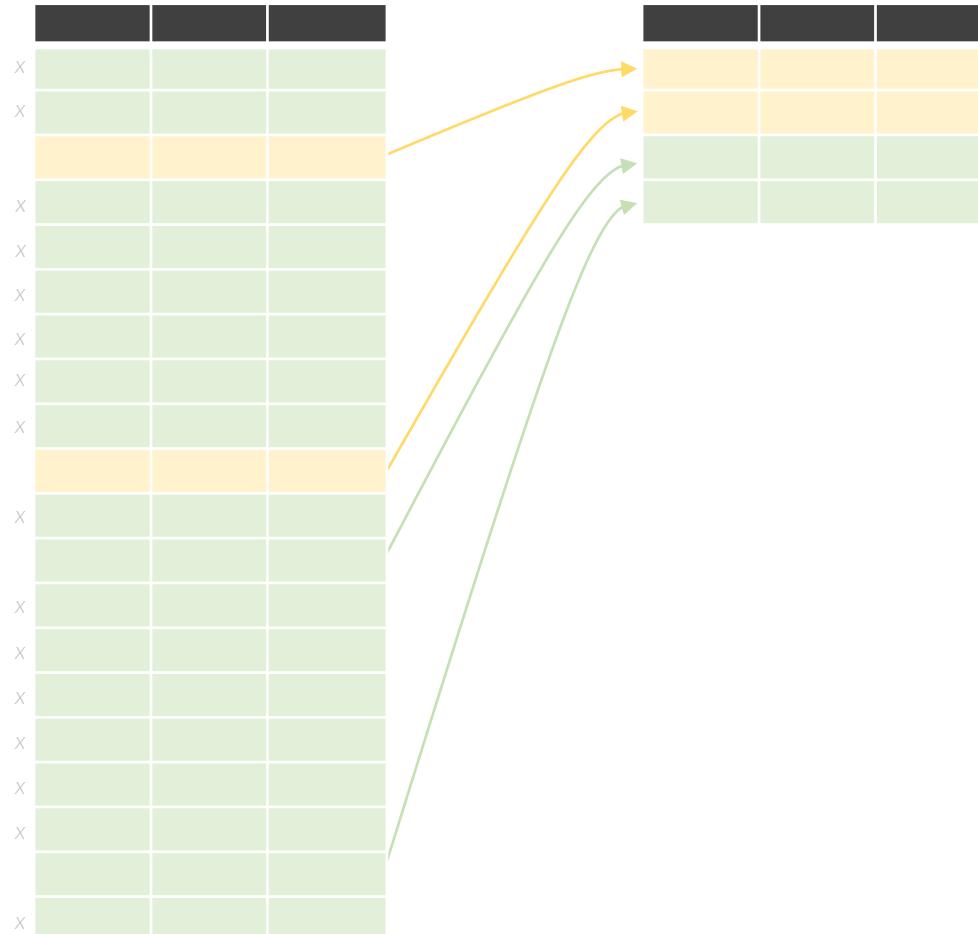
Hyperparameters

**Imbalanced Classes**

Confusion Matrix

Model Selection

Model Drift



**Up-sampling**

*Minority class observations are duplicated to balance the data*

**Down-sampling**

*Majority class observations are randomly removed to balance the data*

**Weighting**

*For models that randomly sample observations (random forests), increase the probability of selecting the minority class*

# IMBALANCED CLASSES

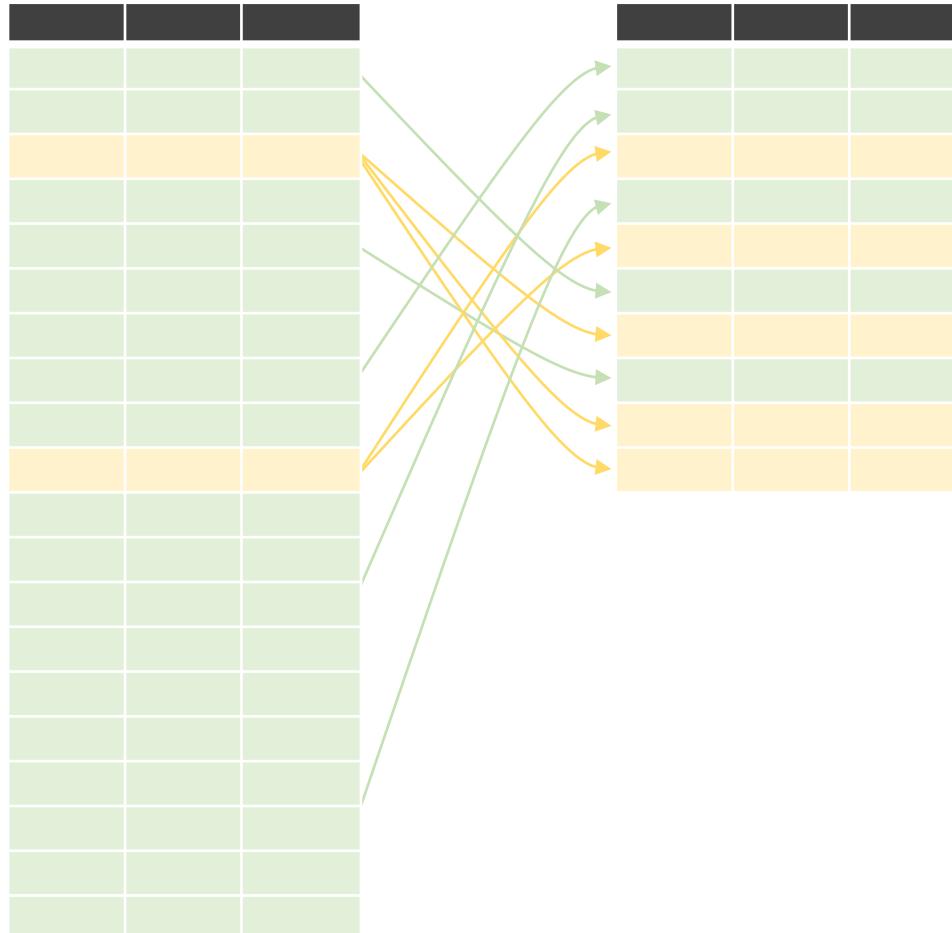
Hyperparameters

**Imbalanced Classes**

Confusion Matrix

Model Selection

Model Drift



**Up-sampling**

*Minority class observations are duplicated to balance the data*

**Down-sampling**

*Majority class observations are randomly removed to balance the data*

**Weighting**

*For models that randomly sample observations (i.e. random forests), increase the probability of selecting the minority class*

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

A **Confusion Matrix** is a table summarizing the frequency of *predicted* vs. *actual* classes in your Test data

- This is the most common and concise way to evaluate performance and compare classification models against one another
- Confusion matrices can be used to derive several types of model performance metrics, including **accuracy**, **precision** and **recall**

		ACTUAL CLASS			
		1	0		
PREDICTED CLASS	1	True Positive (TP)	False Positive (FP)	1	0
	0	False Negative (FN)	True Negative (TN)		0
		100	5		
		15	50		

# CONFUSION MATRIX

Hyperparameters

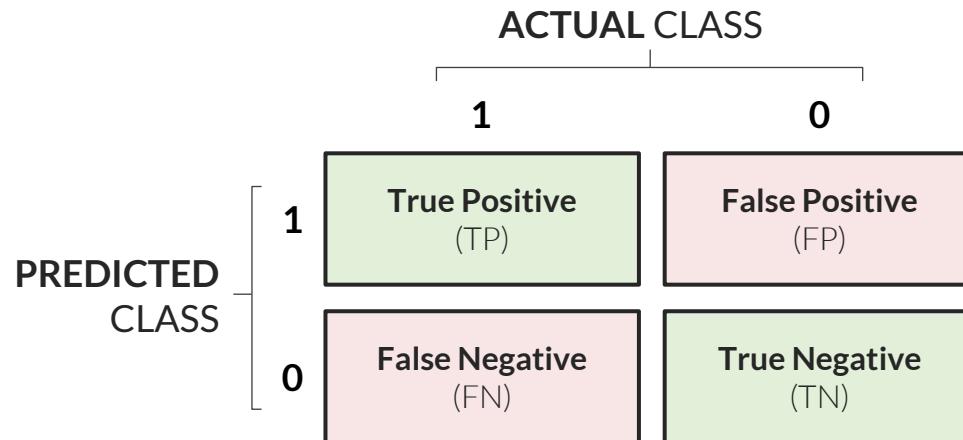
Imbalanced Classes

**Confusion Matrix**

Model Selection

Model Drift

Confusion matrices are used to derive model **performance metrics**



**Accuracy**

$$(TP+TN) / (TP+TN+FP+FN)$$

*Of all predictions, what % were correct?*

**Precision**

$$TP / (TP+FP)$$

*Of all predicted positives, what % were correct?*

**Recall**

$$TP / (TP+FN)$$

*Of all actual positives, what % were predicted correctly?*

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

**Confusion Matrix**

Model Selection

Model Drift

Confusion matrices are used to derive model **performance metrics**

		ACTUAL CLASS	
		1	0
PREDICTED CLASS	1	100	5
	0	15	50

**Accuracy**

$$(TP+TN) / (TP+TN+FP+FN)$$

||

$$(100+50)/(100+50+5+15)$$

||

**.88**

**Precision**

$$TP / (TP+FP)$$

||

$$100/(100+5)$$

||

**.95**

**Recall**

$$TP / (TP+FN)$$

||

$$100/(100+15)$$

||

**.87**

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

**Confusion Matrix**

Model Selection

Model Drift



Will a **Confusion Matrix** work for multi-class predictions too?

- **Yes!** Consider a model classifying multiple products (A, B, C, D); ideally, we want the highest frequencies along the top-left to bottom-right diagonal

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	214	1	8	2
	B	15	452	1	0
	C	0	0	1,123	19
	D	3	2	12	34



*Good confusion matrix*

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	14	1	27	67
	B	55	56	79	15
	C	11	201	90	100
	D	63	145	11	18



*Bad confusion matrix*

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

**Confusion Matrix**

Model Selection

Model Drift

A **confusion matrix** can also provide insight into relationships and interactions between classes, to help inform model improvements

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	37	2	1	0
	B	4	71	76	1
	C	3	69	66	0
	D	1	2	0	201

In this case our model has a hard time differentiating products **B** and **C**, often predicting the wrong one

- Are these products very similar? Do we need to predict them separately?
- If so, can we engineer features to help distinguish B from C, and improve model accuracy?

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	214	1	8	2
	B	15	452	1	0
	C	0	0	1,123	19
	D	3	2	12	34

In this case, (TN) includes all cases where Product A was not predicted OR observed

PRODUCT A:

Accuracy

$$( TP + TN ) / ( TP + TN + FP + FN )$$

$$( 214 + 452 + 1 + 1123 + 19 + 2 + 12 + 34 ) / ( 1886 )$$

.9846

Precision

$$TP / ( TP + FP )$$

$$( 214 ) / ( 214 + 1 + 8 + 2 )$$

.9511

Recall

$$TP / ( TP + FN )$$

$$( 214 ) / ( 214 + 15 + 3 )$$

.9224

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	214	1	8	2
	B	15	452	1	0
	C	0	0	1,123	19
	D	3	2	12	34

PRODUCT B:

Accuracy

$$( \text{TP} + \text{TN} ) / ( \text{TP} + \text{TN} + \text{FP} + \text{FN} )$$

$$(\ 452 + 214 + 8 + 2 + 1123 + 19 + 3 + 12 + 34 \ ) / ( \ 1886 \ )$$

.9899

Precision

$$\text{TP} / ( \text{TP} + \text{FP} )$$

$$( \ 452 \ ) / ( \ 452 + 15 + 1 \ )$$

.9658

Recall

$$\text{TP} / ( \text{TP} + \text{FN} )$$

$$( \ 452 \ ) / ( \ 452 + 1 + 2 \ )$$

.9934

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	214	1	8	2
	B	15	452	1	0
	C	0	0	1,123	19
	D	3	2	12	34

PRODUCT:

Accuracy

$$( \text{TP} + \text{TN} ) / ( \text{TP} + \text{TN} + \text{FP} + \text{FN} )$$

$$(\ 1123 + 214 + 1 + 2 + 15 + 452 + 3 + 2 + 34\ ) / ( \ 1886\ )$$

.9788

Precision

$$\text{TP} / ( \text{TP} + \text{FP} )$$

$$( \ 1123\ ) / ( \ 1123 + 19\ )$$

.9834

Recall

$$\text{TP} / ( \text{TP} + \text{FN} )$$

$$( \ 1123\ ) / ( \ 1123 + 8 + 1 + 12\ )$$

.9816

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

		ACTUAL PRODUCT			
		A	B	C	D
PREDICTED PRODUCT	A	214	1	8	2
	B	15	452	1	0
	C	0	0	1,123	19
	D	3	2	12	34

PRODUCT D:

Accuracy

$$( TP + TN ) / ( TP + TN + FP + FN )$$

$$( 34 + 214 + 1 + 8 + 15 + 452 + 1 + 1123 ) / ( 1886 )$$

.9799

Precision

$$TP / ( TP + FP )$$

$$( 34 ) / ( 34 + 3 + 2 + 12 )$$

.6667

Recall

$$TP / ( TP + FN )$$

$$( 34 ) / ( 34 + 2 + 19 )$$

.6182

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

214	1	8	2
15	452	1	0
0	0	1,123	19
3	2	12	34

	# Obs.	Accuracy	Precision	Recall
A	232	.9846	.9511	.9224
B	455	.9899	.9658	.9934
C	1,144	.9788	.9834	.9816
D	55	.9799	.6667	.6182
WEIGHTED AVG:		.9822	.9659	.9666

# CONFUSION MATRIX

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

To score a **multi-class confusion matrix**, calculate metrics for each predicted class, then take a weighted average to evaluate the model as a whole

14	1	27	67
55	56	79	15
11	201	90	100
63	145	11	18

	# Obs.	Accuracy	Precision	Recall
A	143	.7500	.1284	.0979
B	403	.4795	.2732	.1390
C	207	.5498	.2239	.4348
D	200	.5792	.0759	.0900
WEIGHTED AVG:		.5563	.1994	.1868

# MODEL SELECTION

---

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

Model Drift

**Model selection** describes the process of training multiple models, comparing performance on Test data, and choosing the best option

When comparing performance, it's important to prioritize the *most relevant* metrics based on the context of your model

- **RECALL** may be the best metric if it's critical to predict ALL positive outcomes correctly, and false negatives are a major risk (*i.e. nuclear reactor*)
- **PRECISION** may be the best metric if false negatives aren't a big deal, but false positives are a major risk (*i.e. spam filter or document search*)
- **ACCURACY** may be the best metric if you care about predicting positive and negative outcomes equally, or if the risk of each outcome is comparable

# MODEL DRIFT

Hyperparameters

Imbalanced Classes

Confusion Matrix

Model Selection

**Model Drift**

**Drift** is when a trained model gradually becomes less accurate over time, even when all variables and parameters remain the same

- As a best practice, all ML models used for ongoing prediction should be updated or retrained on a regular basis to combat drift

## Tips to correct drift:

- Benchmark your model performance (*accuracy, precision, recall*) on Day 1
- Monitor performance against your benchmark over time
- If you notice drift compared to your benchmark, retrain the model using updated Training data and consider discarding old records (*if you have enough volume*)
- Conduct additional feature engineering as necessary

PART 3:

# REGRESSION



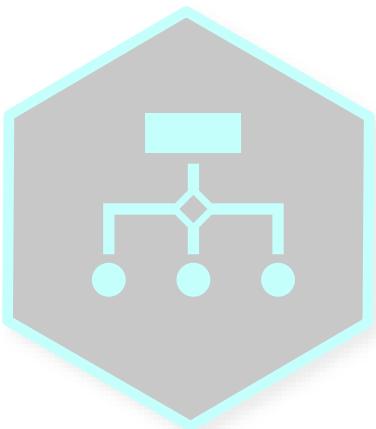
# ABOUT THIS SERIES

This is **Part 3** of a **4-Part series** designed to help you build a deep, foundational understanding of machine learning, including data QA & profiling, classification, forecasting and unsupervised learning



## PART 1

QA & Data Profiling



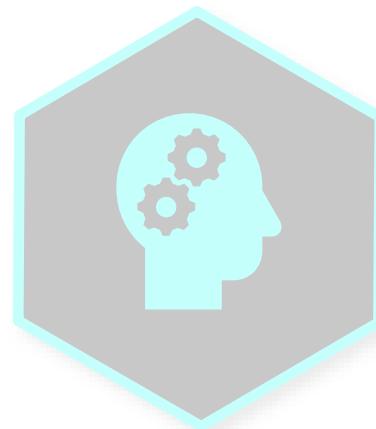
## PART 2

Classification



## PART 3

Regression & Forecasting



## PART 4

Unsupervised Learning

# COURSE OUTLINE

---

1

## Intro to Regression

Review the ML landscape and key supervised learning concepts, including regression vs. classification, feature engineering, overfitting, etc.

2

## Regression Modeling

Learn the key building blocks of regression analysis, including linear relationships, least squared error, multivariate and non-linear models

3

## Model Diagnostics

Understand and interpret model outputs and common diagnostic metrics like R-squared, mean error, F-significance, P-values, and more

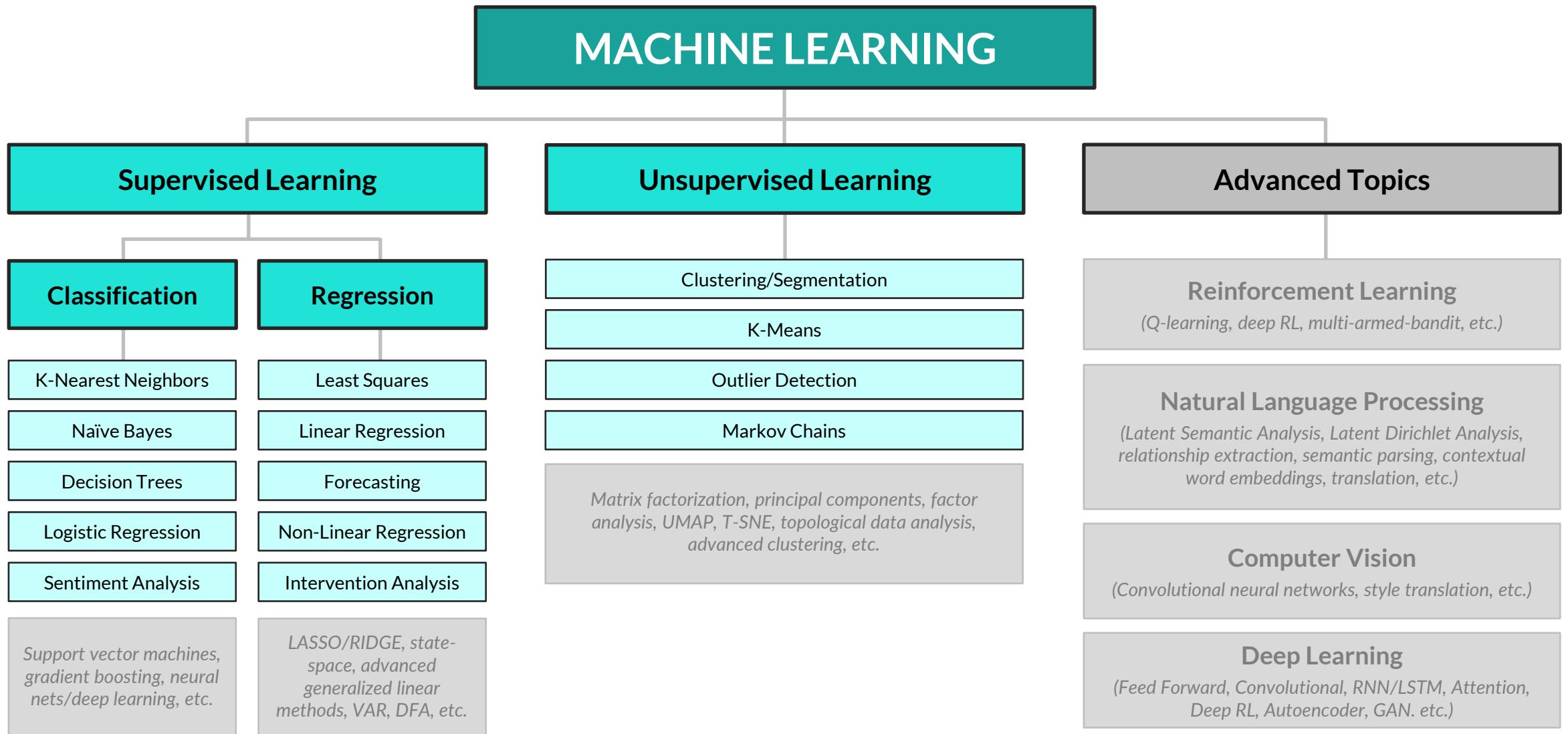
4

## Time-Series Forecasting

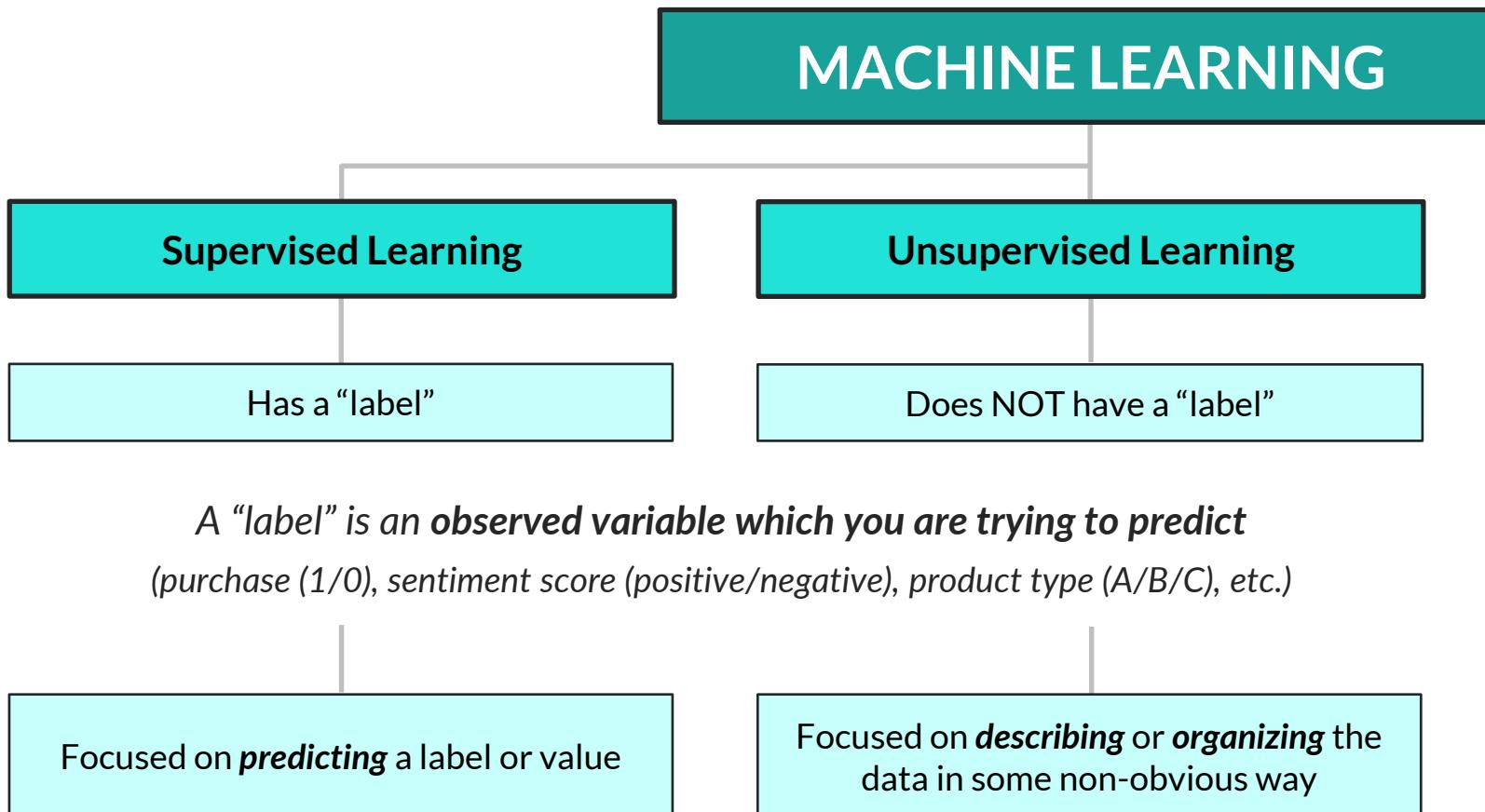
Explore powerful tools & techniques for time-series forecasting, including seasonality, linear and non-linear trends, intervention analysis, etc.

# INTRO TO REGRESSION

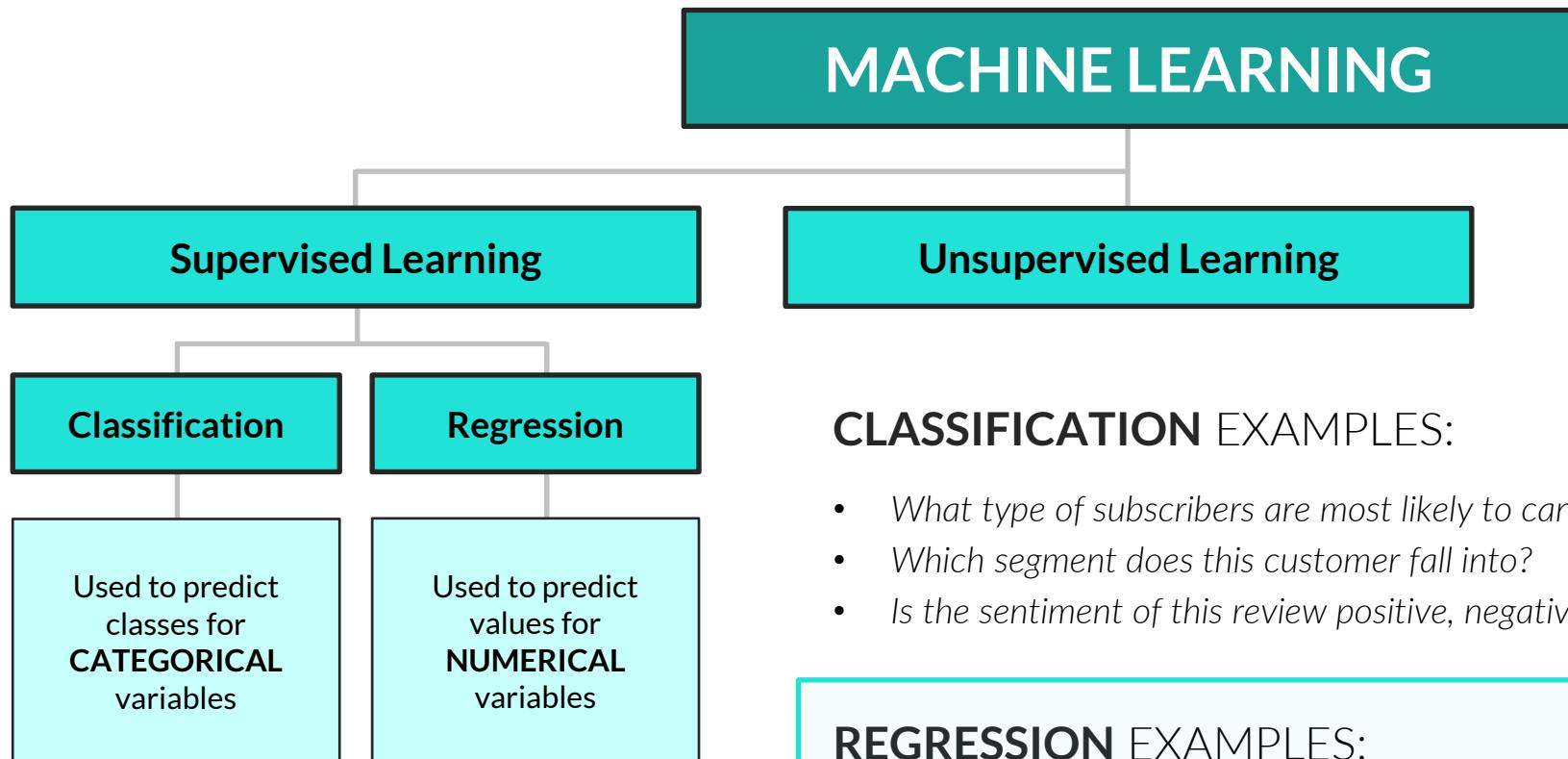
# MACHINE LEARNING LANDSCAPE



# MACHINE LEARNING LANDSCAPE



# MACHINE LEARNING LANDSCAPE



## CLASSIFICATION EXAMPLES:

- What type of subscribers are most likely to cancel or churn?
- Which segment does this customer fall into?
- Is the sentiment of this review positive, negative or neutral?

## REGRESSION EXAMPLES:

- How much revenue are we forecasting for the quarter?
- How many units do we expect to sell next season?
- How will seasonality impact the size of our customer base?

# RECAP: KEY CONCEPTS



## All tables contain rows and columns

- Each row represents an individual **record/observation**
- Each columns represent an individual **variable**



## Variables can be categorical or numerical

- Categorical variables contain **classes** or **categories** (used for filtering)
- Numerical variables contain **numbers** (used for aggregation)



## QA and data profiling comes first, every time

- Without quality data, you can't build quality models ("garbage in, garbage out")
- Profiling is the first step towards **conditioning** (or *filtering*) on key variables to understand their impact



## Machine Learning is needed when visual analysis falls short

- Humans aren't capable of thinking beyond 3+ dimensions, but machines are built for it
- ML is a natural extension of visual analysis and univariate/multivariate profiling

Session ID	Browser	Time (Sec)	Pageviews	Purchase
1	Chrome	354	11	1
2	Safari	94	4	1
3	Safari	36	2	0
4	IE	17	1	0
5	Chrome	229	9	1

These are **categorical** variables

These are **numerical** variables

Each **row** represents a record of a unique web session

# REGRESSION 101

---

The goal of regression is to predict a **numeric dependent variable** using **independent variables**

## **y** Dependent variable (DV) – for regression, this must be numerical (not categorical)!

- This is the variable **you're trying to predict**
- The dependent variable is commonly referred to as "Y", "predicted", "output", or "target" variable
- Regression is about understanding how the numerical DV is impacted by, or *dependent* on, other variables in the model

## **X** Independent variables (IVs)

- These are the variables **which help you predict the dependent variable**
- Independent variables are commonly referred to as "X's", "predictors", "features", "explanatory variables", or "covariates"
- Regression is about understanding how the IVs impact, or *predict*, the DV

# REGRESSION 101

**EXAMPLE:** Using marketing and sales data (*sample below*) to predict revenue for a given month

Month ID	Social Posts	Competitive Activity	Marketing Spend	Promotion Count	Revenue
1	30	High	\$130,000	12	\$1,300,050
2	15	Low	\$600,000	5	\$11,233,310
3	11	Medium	\$15,000	10	\$1,112,050
4	22	Medium	\$705,000	11	\$1,582,077
5	41	High	\$3,000	3	\$1,889,053
6	5	High	\$3,500	3	\$1,200,089
7	23	Low	\$280,000	2	\$1,300,080
8	12	Low	\$120,000	11	\$700,150
9	8	Low	\$1,000,000	51	\$41,011,150
10	19	Medium	\$43,000	10	\$1,000,210

**Revenue** is our **dependent variable**, since it's what we want to predict

Since Revenue is **numerical**, we'll use regression (vs. classification) to predict it

**Social Posts, Competitive Activity, Marketing Spend and Promotion Count** are all **independent variables**, since they can help us explain, or predict, monthly Revenue

# REGRESSION 101

**EXAMPLE:** Using marketing and sales data (*sample below*) to predict revenue for a given month

Month ID	Social Posts	Competitive Activity	Marketing Spend	Promotion Count	Revenue
1	30	High	\$130,000	12	\$1,300,050
2	15	Low	\$600,000	5	\$11,233,310
3	11	Medium	\$15,000	10	\$1,112,050
4	22	Medium	\$705,000	11	\$1,582,077
5	41	High	\$3,000	3	\$1,889,053
6	5	High	\$3,500	3	\$1,200,089
7	23	Low	\$280,000	2	\$1,300,080
8	12	Low	\$120,000	11	\$700,150
9	8	Low	\$1,000,000	51	\$41,011,150
10	19	Medium	\$43,000	10	\$1,000,210
11	7	High	\$320,112	8	???

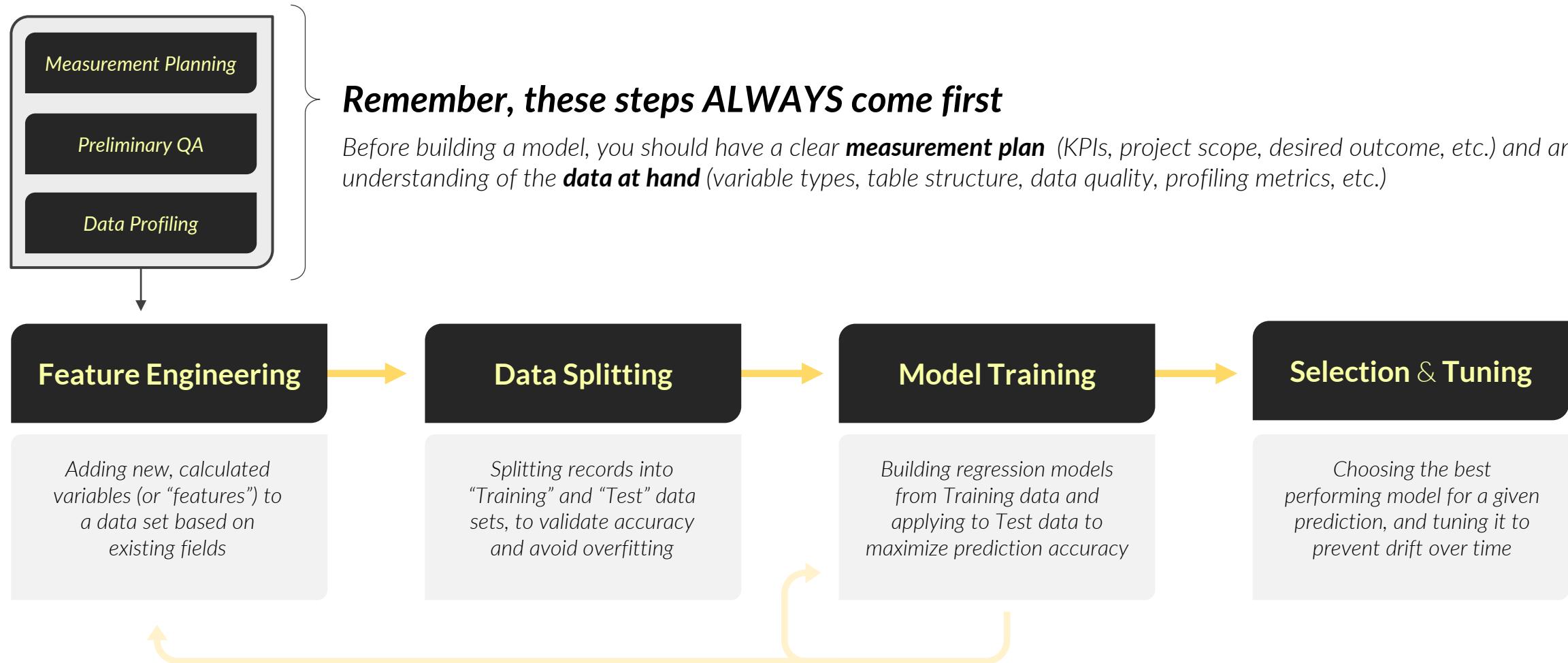


We'll use records with **observed values** for both independent and dependent variables to "train" our regression model...

...then apply that model to new, **unobserved values** containing IVs but no DV

***This is what our regression model will predict!***

# REGRESSION WORKFLOW



# FEATURE ENGINEERING



**Feature engineering** is the process of enriching a data set by creating additional independent variables based on existing fields

- New features can help improve the accuracy and predictive power of your ML models
- Feature engineering is often used to convert fields into “model-friendly” formats; for example, **one-hot encoding** transforms categorical variables into individual, binary (1/0) fields

Original features						Engineered features				
Month ID	Social Posts	Competitive Activity	Marketing Spend	Promotion Count	Revenue	Competitive High	Competitive Medium	Competitive Low	Promotion >10 & Social > 25	Log Spend
1	30	High	\$130,000	12	\$1,300,050	1	0	0	1	11.7
2	15	Low	\$600,000	5	\$11,233,310	0	0	1	0	13.3
3	8	Medium	\$15,000	10	\$1,112,050	0	1	0	0	9.6
4	22	Medium	\$705,000	11	\$1,582,077	0	1	0	0	13.5
5	41	High	\$3,000	3	\$1,889,053	1	0	0	0	8.0

# DATA SPLITTING



**Splitting** is the process of partitioning data into separate sets of records for the purpose of training and testing machine learning models

- As a rule of thumb, ~70-80% of your data will be used for **Training** (which is what your model learns from), and ~20-30% will be reserved for **Testing** (to validate the model's accuracy)

Month ID	Social Posts	Competitive Activity	Marketing Spend	Promotion Count	Revenue
1	30	High	\$130,000	12	\$1,300,050
2	15	Low	\$600,000	5	\$11,233,310
3	11	Medium	\$15,000	10	\$1,112,050
4	22	Medium	\$705,000	11	\$1,582,077
5	41	High	\$3,000	3	\$1,889,053
6	5	High	\$3,500	3	\$1,200,089
7	23	Low	\$280,000	2	\$1,300,080
8	12	Low	\$120,000	11	\$700,150
9	8	Low	\$1,000,000	51	\$41,011,150
10	19	Medium	\$43,000	10	\$1,000,210



**Test data is NOT used to optimize models**

**Training**  
data

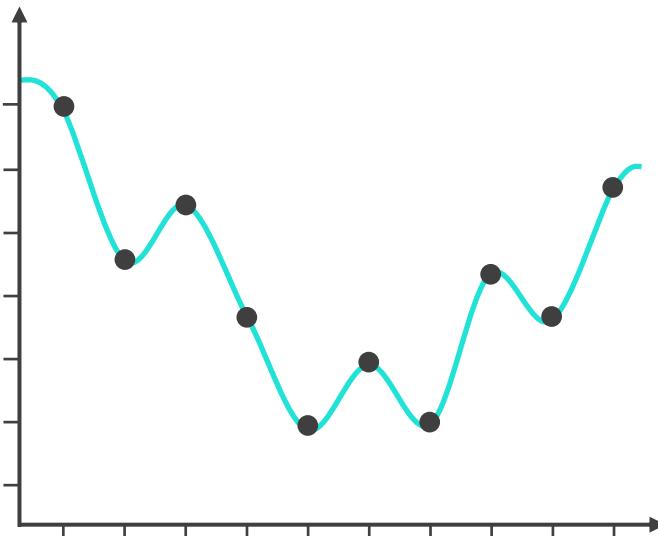
**Test**  
data

Using **Training** data for optimization and **Test** data for validation ensures that your model can accurately predict both known and *unknown* values, which helps to prevent **overfitting**

# OVERFITTING

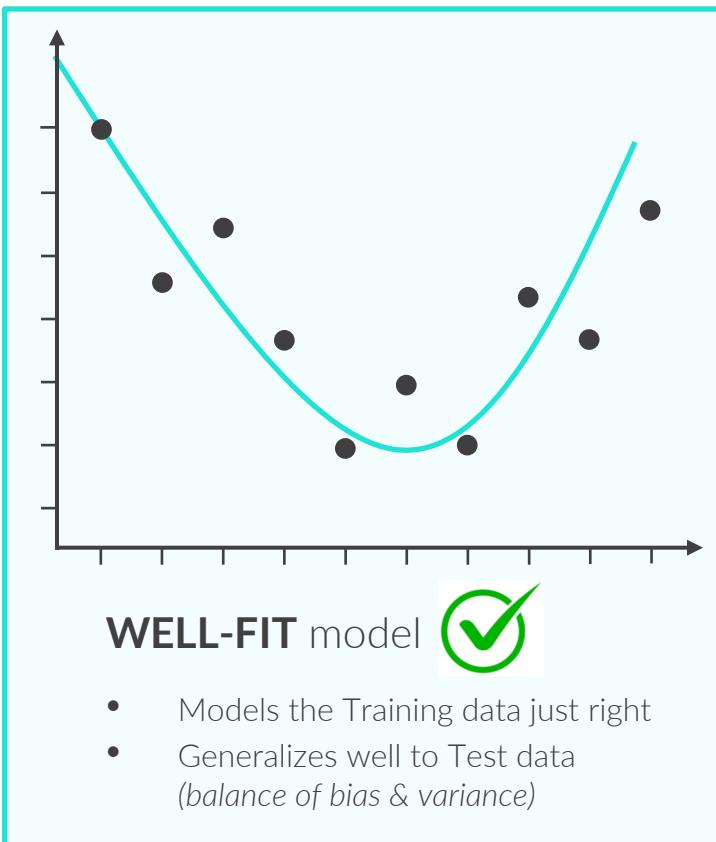
Splitting is primarily used to avoid **overfitting**, which is when a model predicts known (*Training*) data very well but unknown (*Test*) data poorly

- Think of overfitting like memorizing the answers to a test instead of actually *learning* the material; you'll ace the test, but lack the ability to **generalize** and apply your knowledge to unfamiliar questions



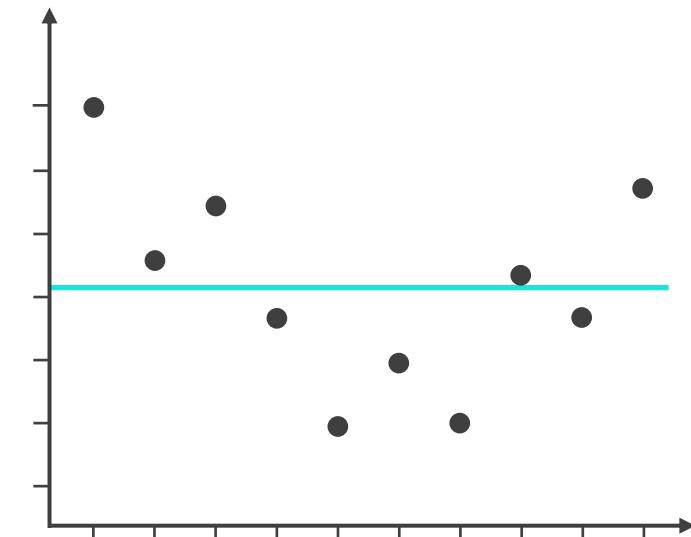
**OVERFIT** model

- Models the Training data too well
- Doesn't generalize well to Test data (*high variance, low bias*)



**WELL-FIT** model

- Models the Training data just right
- Generalizes well to Test data (*balance of bias & variance*)



**UNDERFIT** model

- Doesn't model Training data well enough
- Doesn't generalize well to Test data (*high bias, low variance*)

# REGRESSION

There are two common use cases for linear regression: **prediction** and **root-cause analysis**



## PREDICTION

- Used to **predict** or **forecast** a numerical dependent variable
- Goal is to make accurate predictions, even if causality cannot necessarily be proven

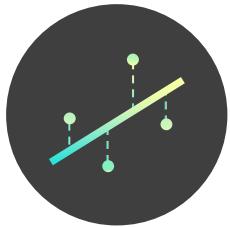


## ROOT-CAUSE ANALYSIS

- Used to determine the **causal impact** of individual model inputs
- Goal is to prove causality by comparing the sensitivity of each IV on the DV

# REGRESSION MODELING

# REGRESSION MODELING



In this section we'll introduce the basics of **regression modeling**, including linear relationships, least squared error, simple and multiple regression and non-linear models

## TOPICS WE'LL COVER:

Linear Relationships

Least Squared Error

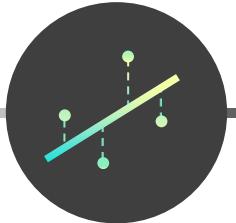
Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

## GOALS FOR THIS SECTION:

- Understand basic linear relationships and the concept of least squared errors
- Explore the difference between simple (univariate) and multiple linear regression
- Demonstrate how regression can be used to model linear or non-linear relationships



# LINEAR RELATIONSHIPS

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

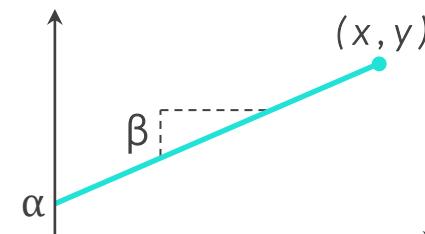
Non-Linear Regression

A **linear relationship** means that when you increase or decrease your X variable, your Y variable increases or decreases at a steady rate

- If you plot a linear relationship over many X/Y values, it will look like a straight line
- Mathematically, linear relationships can be described using the following equation:

$$y = \alpha + \beta x$$

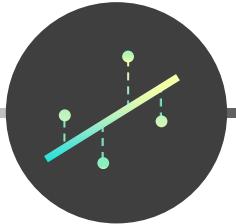
Y-intercept  
X value  
Y value  
Slope (rise/run)



- **NOTE:** Not all relationships are linear (*more on that later!*)

## Common examples:

- Taxi Mileage (X) and Total Fare (Y)
- Units Sold (X) and Total Revenue (Y)



# LINEAR RELATIONSHIPS

Linear Relationships

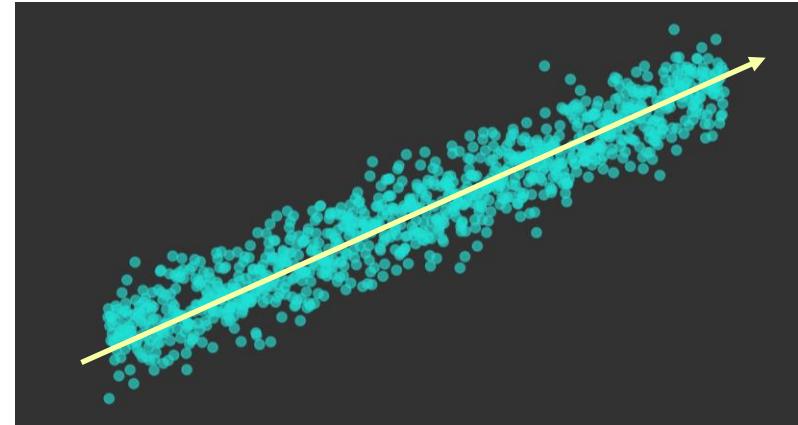
Least Squared Error

Univariate Linear Regression

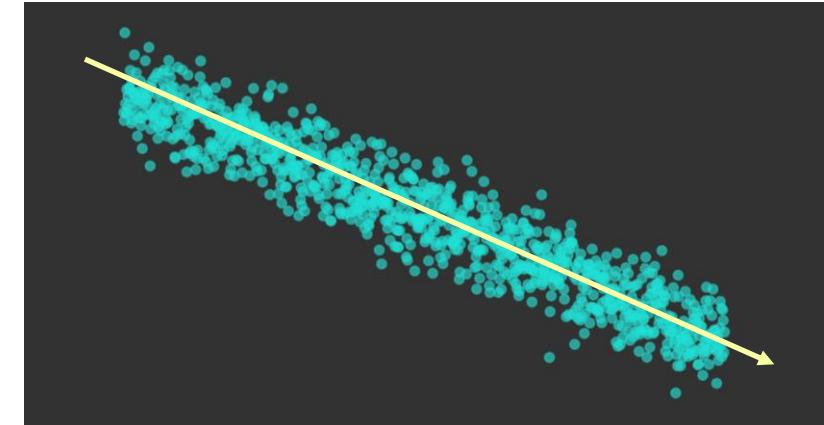
Multiple Linear Regression

Non-Linear Regression

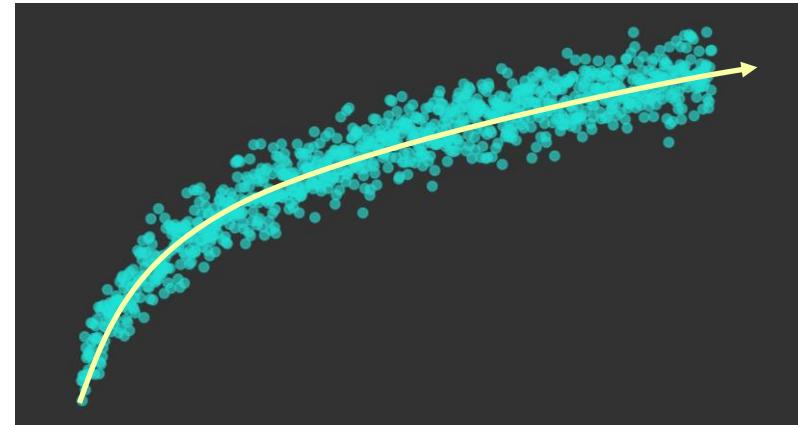
Positive Linear



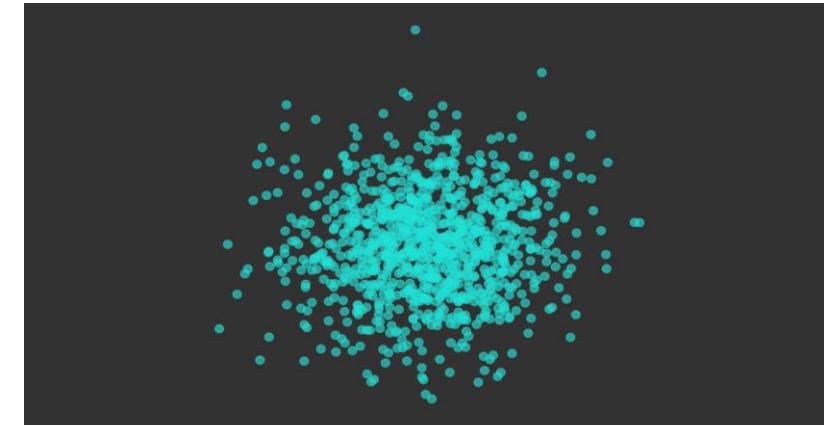
Negative Linear



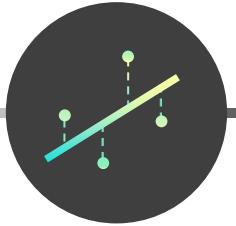
Non-Linear (logarithmic)



No Relationship



Sometimes there's **no relationship at all!**



# LINEAR RELATIONSHIPS

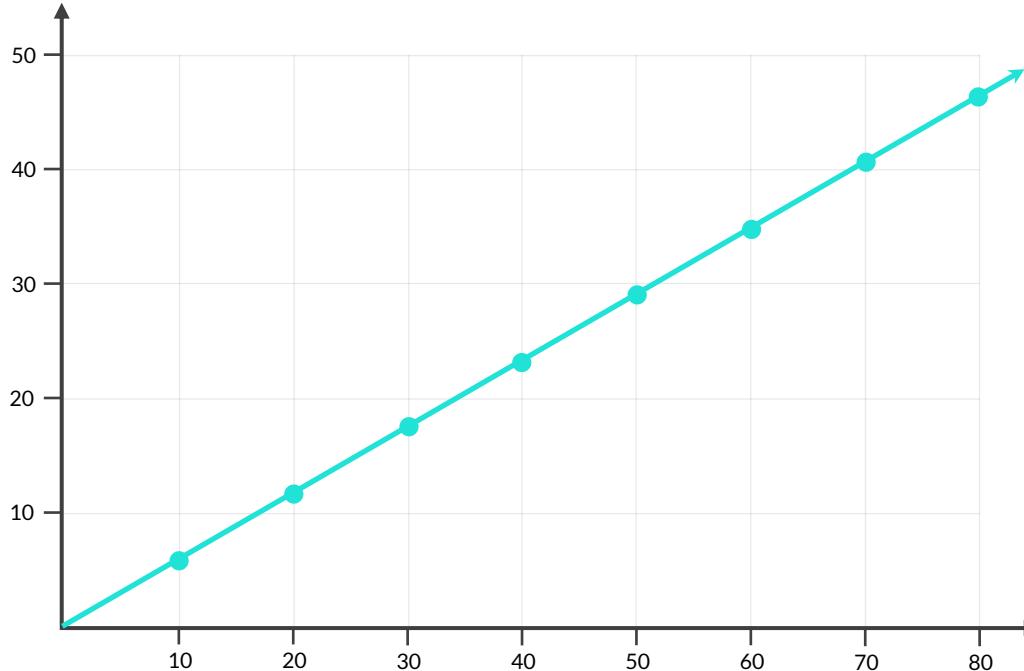
Linear Relationships

Least Squared Error

Univariate Linear Regression

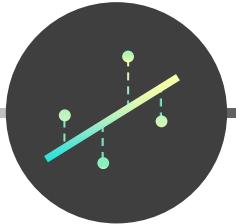
Multiple Linear Regression

Non-Linear Regression



Consider a line that fits every *single* point in the plot  
This is known as a **perfectly linear relationship**

In this case you can simply calculate the **exact value of Y** for any given value of X (no Machine Learning needed, just simple math!)



# LINEAR RELATIONSHIPS

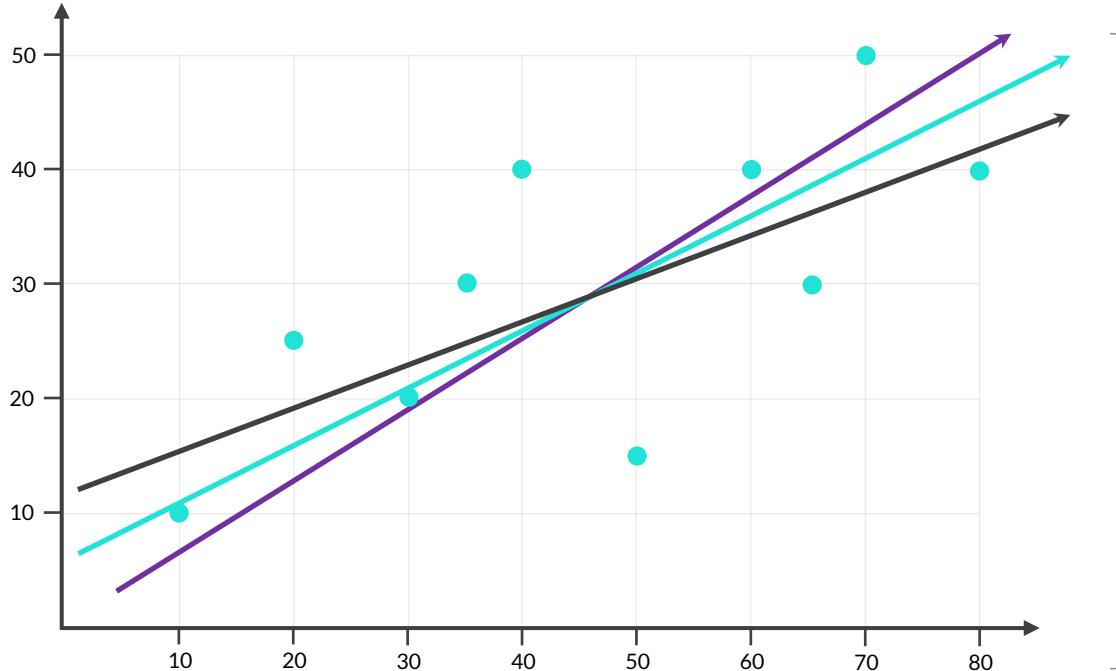
Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

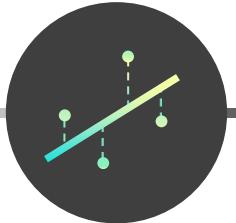
Non-Linear Regression



In the real world, things aren't quite so simple

When you add **variance**, it means that *many* different lines could potentially fit through the plot

To find the equation of the line with the best possible fit, we can use a technique known as **least squared error** or “**least squares**”



# LEAST SQUARED ERROR

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

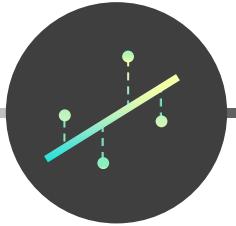
**Least squared error** is used to mathematically determine the line that best fits through a series of data points

- Imagine drawing a line through a scatterplot, and measuring the distance between your line and each point (*these distances are called **errors**, or **residuals***)
- Now square each of those residuals, add them all up, and adjust your line until you've minimized that sum; **this is how least squares works!**



## Why “squared” error?

- Squaring the residuals converts them into **positive values**, and prevents positive and negative distances from cancelling each other out (*this helps the model optimize more efficiently, too*)



# LEAST SQUARED ERROR

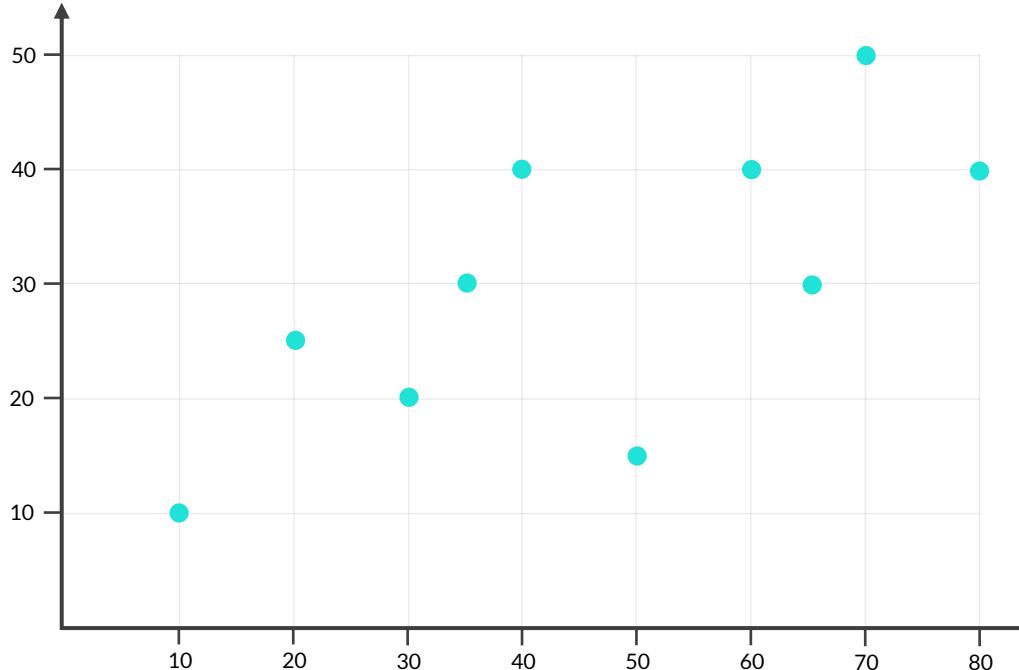
Linear Relationships

Least Squared Error

Univariate Linear Regression

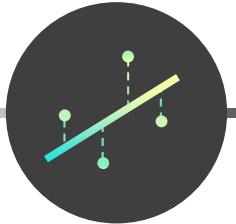
Multiple Linear Regression

Non-Linear Regression



X	Y (actual)
10	10
20	25
30	20
35	30
40	40
50	15
60	40
65	30
70	50
80	40

**STEP 1:** Plot each data point on a scatterplot, and record the X and Y values



# LEAST SQUARED ERROR

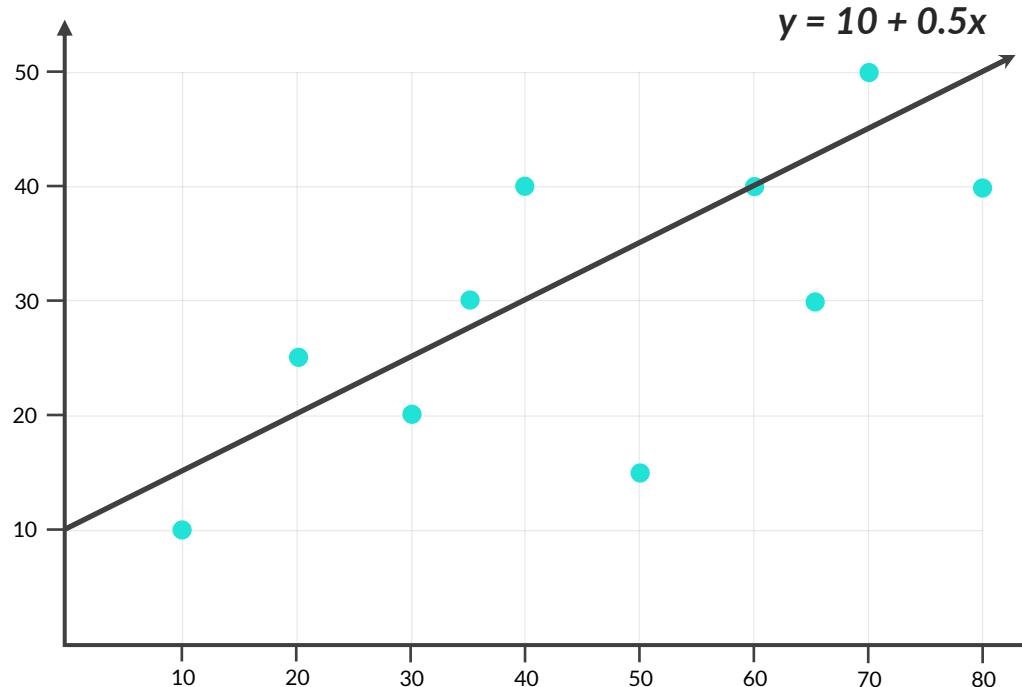
Linear Relationships

Least Squared Error

Univariate Linear Regression

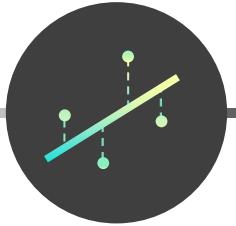
Multiple Linear Regression

Non-Linear Regression



X	Y (actual)
10	10
20	25
30	20
35	30
40	40
50	15
60	40
65	30
70	50
80	40

**STEP 2:** Draw a straight line through the points in the scatterplot, and calculate the Y values derived by your linear equation



# LEAST SQUARED ERROR

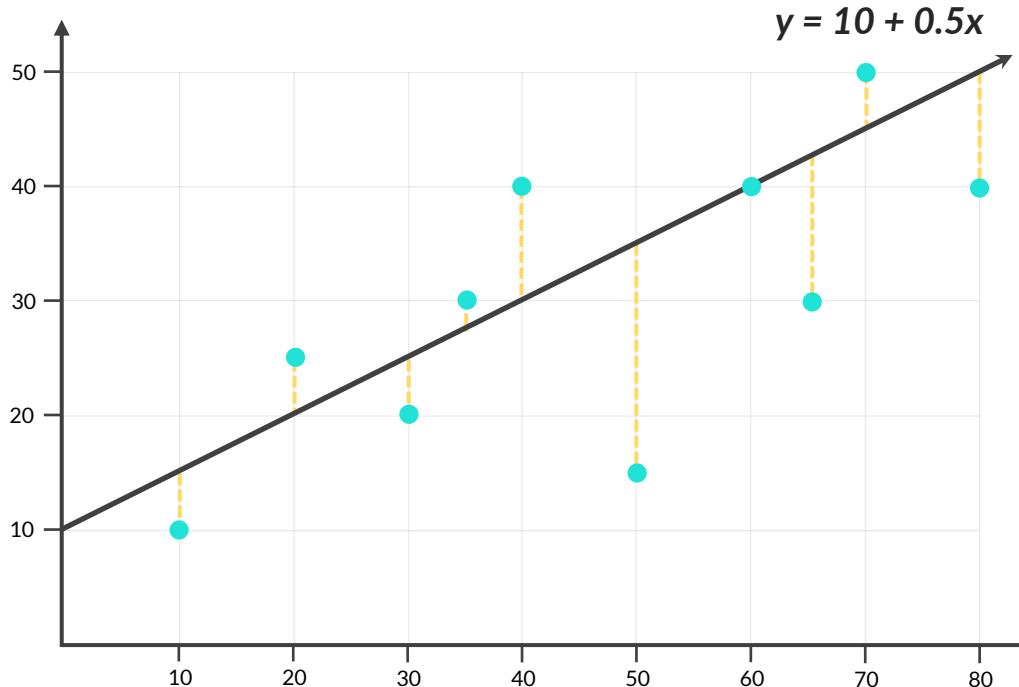
Linear Relationships

Least Squared Error

Univariate Linear Regression

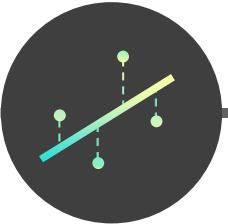
Multiple Linear Regression

Non-Linear Regression



X	Y (actual)	Y (line)
10	10	15
20	25	20
30	20	25
35	30	27.5
40	40	30
50	15	35
60	40	40
65	30	42.5
70	50	45
80	40	50

**STEP 3:** For each value of X, calculate the error (or residual) by comparing the actual Y value against the Y value produced by your linear equation



# LEAST SQUARED ERROR

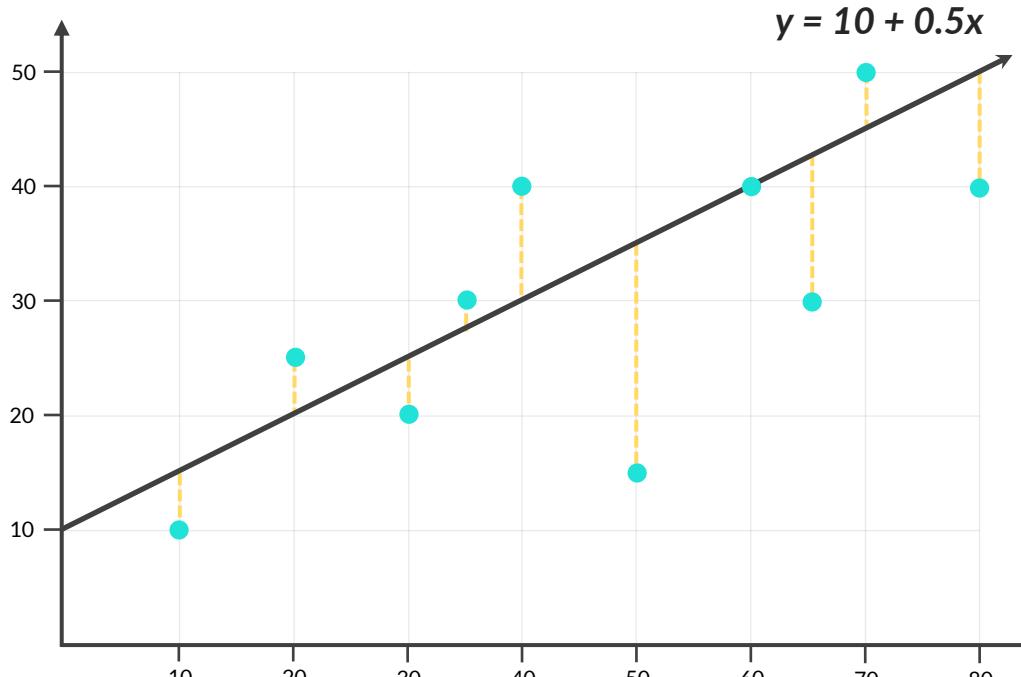
Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

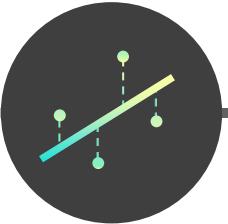


X	Y (actual)	Y (line)	Error
10	10	15	5
20	25	20	-5
30	20	25	5
35	30	27.5	-2.5
40	40	30	-10
50	15	35	20
60	40	40	0
65	30	42.5	12.5
70	50	45	-5
80	40	50	10

SUM OF SQUARED ERROR: **862.5**

**STEP 4:** Square each individual residual and add them up to determine the **sum of squared error (SSE)**

- This defines exactly how well your line “fits” the plot (or in other words, how well the linear equation describes the relationship between X and Y)



# LEAST SQUARED ERROR

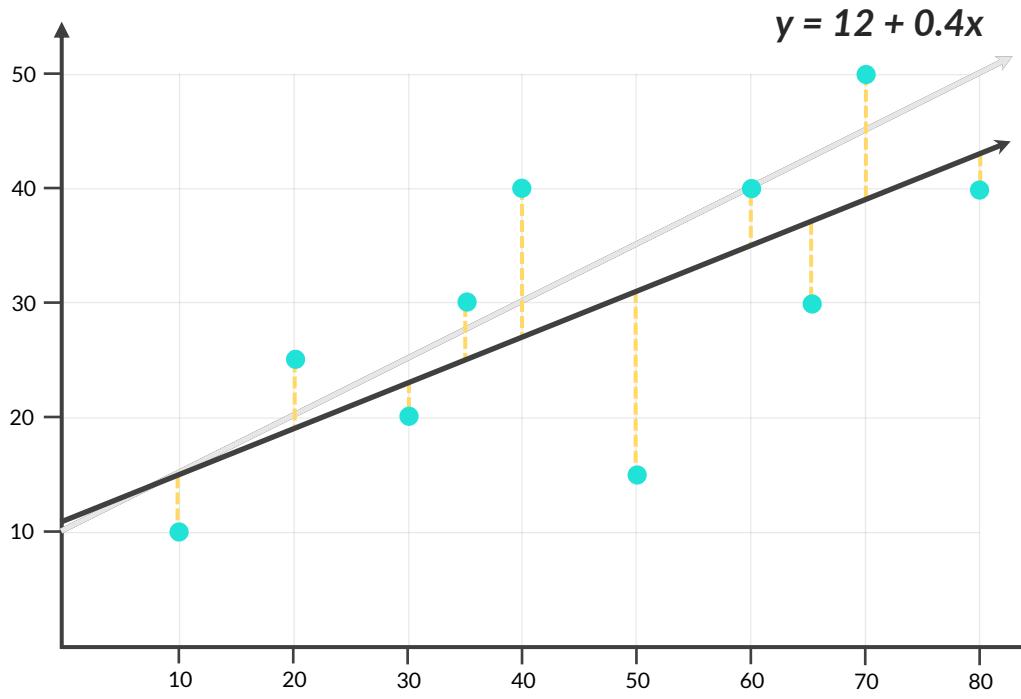
Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression



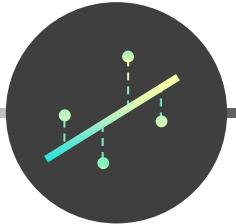
X	Y (actual)	Y (line)	Error	Sq. Error
10	10	16	6	36
20	25	20	-5	25
30	20	24	4	16
35	30	26	-4	16
40	40	28	-12	144
50	15	32	17	289
60	40	36	-4	16
65	30	38	8	64
70	50	40	-10	100
80	40	44	4	16

SUM OF SQUARED ERROR: **722**

II

**STEP 5:** Plot a new line, repeat Steps 1-4, and continue the process until you've found the line that **minimizes the sum of squared error**

- This is where **Machine Learning** comes in; human trial-and-error is completely impractical, but machines can find an optimal linear equation in seconds



# UNIVARIATE LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

**Univariate (“simple”) linear regression** is used for predicting a numerical output (DV) based on a *single* independent variable

- Univariate linear regression is simply an extension of least squares; you use the linear equation that minimizes SSE to predict an output (**Y**) for any given input (**X**)

$$y = \alpha + \beta x + \varepsilon$$

Annotations for the equation:

- Dependent variable (DV)
- Coefficient/parameter (sensitivity of Y to X)
- Error/residual
- Y-intercept
- Independent variable (IV)

This is just the equation of a line, plus an error term

Simple linear regression is rarely used on its own; think of it as a primer for understanding more complex topics like non-linear and multiple regression

# CASE STUDY: UNIVARIATE LINEAR REGRESSION

---



## THE SITUATION

You are the proud owner of **The Cone Zone**, a mobile ice cream cart operating on the Atlantic City boardwalk.



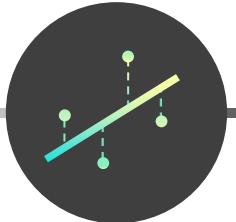
## THE ASSIGNMENT

You've noticed that you tend to sell more ice cream on hot days, and want to understand how temperature and sales relate. Your goal is to build a simple linear regression model that you can use to **predict sales based on the weather forecast**.



## THE OBJECTIVES

1. Use a scatterplot to visualize a sample of daily temperatures (X) and sales (Y)
2. Do you notice a clear pattern or trend? How might you interpret this?
3. Find the line that best fits the data by minimizing SSE, then confirm by plotting a linear trendline on the scatter plot



# MULTIPLE LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression



Can we use **more than one IV** to predict the DV?

**Multiple linear regression** is used for predicting a numerical output (DV) based on *multiple* independent variables

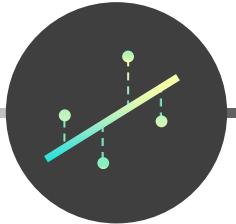
- In its simplest form, multiple linear regression is simply univariate linear regression with additional x variables:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_n x_n + \epsilon$$

Annotations for the equation:

- A blue arrow labeled "DV" points to the variable  $y$ .
- A blue bracket under the terms  $\beta_1 x_1, \beta_2 x_2, \dots, \beta_n x_n$  is labeled "Y-intercept".
- A blue arrow labeled "Error/residual" points to the term  $\epsilon$ .

Instead of just 1 IV, we have a **whole set of independent variables** (and associated coefficients/weights) to help explain our DV



# MULTIPLE LINEAR REGRESSION

Linear Relationships

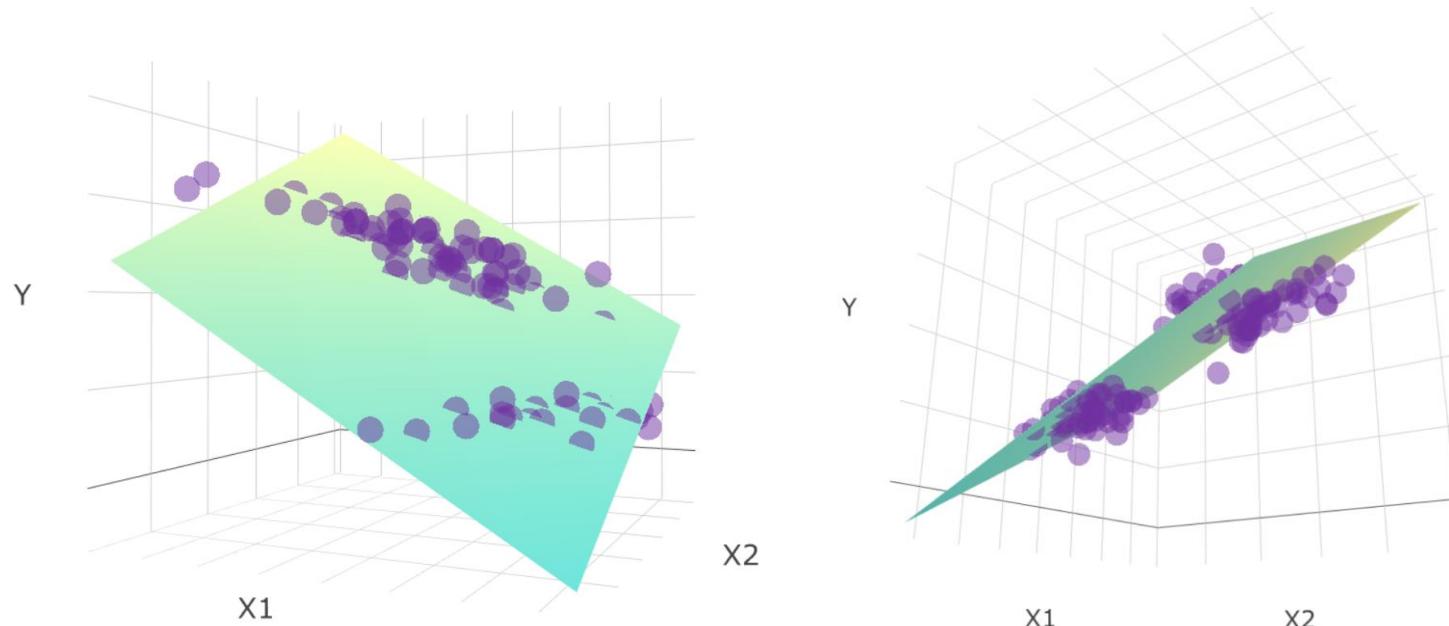
Least Squared Error

Univariate Linear Regression

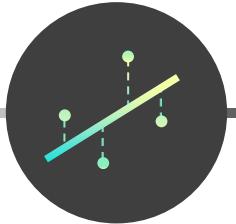
Multiple Linear Regression

Non-Linear Regression

To visualize how multiple regression works with 2 independent variables, imagine fitting a plane (vs. a line) through a 3D scatterplot:



Multiple regression can scale well beyond 2 variables, but this is where visual analysis breaks down (and why we need **machine learning!**)



# MULTIPLE LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

## EXAMPLE

You are preparing to list a new property on **AirBnB**, and want to estimate (or predict) an appropriate price using the listing data below

list_id	city	district	room_type	accommodates	bedrooms	entire.place	hotel.room	private.room	bronx	brooklyn	manhattan	queens	price
3831	New York	Brooklyn	Entire place	3	1	1	0	0	0	1	0	0	73
5121	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	60
5178	New York	Manhattan	Private room	2	1	0	0	1	0	0	1	0	79
5203	New York	Manhattan	Private room	1	1	0	0	1	0	0	1	0	75
5803	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	83
6872	New York	Manhattan	Private room	1	1	0	0	1	0	0	1	0	65
6990	New York	Manhattan	Private room	1	1	0	0	1	0	0	1	0	62
7097	New York	Brooklyn	Entire place	4	1	1	0	0	0	0	1	0	199
7750	New York	Manhattan	Private room	1	2	0	0	1	0	0	1	0	96
8490	New York	Brooklyn	Entire place	5	1	1	0	0	0	1	0	0	120
9657	New York	Manhattan	Entire place	3	1	1	0	0	0	0	1	0	150
9704	New York	Manhattan	Private room	2	1	0	0	1	0	0	1	0	55
10452	New York	Brooklyn	Private room	3	1	0	0	1	0	1	0	0	70
10962	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	83
11943	New York	Brooklyn	Private room	1	1	0	0	1	0	1	0	0	150
12192	New York	Manhattan	Private room	2	1	0	0	1	0	0	1	0	40
12343	New York	Manhattan	Entire place	3	1	1	0	0	0	0	1	0	150
12937	New York	Queens	Private room	4	1	0	0	1	0	0	0	1	130
12940	New York	Brooklyn	Entire place	2	1	1	0	0	0	1	0	0	99
13121	New York	Queens	Private room	4	1	0	0	1	0	0	0	1	75
13394	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	80
13808	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	81
14290	New York	Brooklyn	Entire place	2	2	1	0	0	0	1	0	0	157
14314	New York	Brooklyn	Entire place	2	1	1	0	0	0	1	0	0	99
14322	New York	Manhattan	Entire place	1	1	1	0	0	0	0	1	0	200
15385	New York	Brooklyn	Private room	1	1	0	0	1	0	1	0	0	80
15711	New York	Manhattan	Entire place	6	2	1	0	0	0	0	1	0	206
16326	New York	Brooklyn	Entire place	8	4	1	0	0	0	1	0	0	200
1633	New York	Brooklyn	Private room	2	1	0	0	1	0	1	0	0	45



# MULTIPLE LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

## EXAMPLE

You are preparing to list a new property on **AirBnB**, and want to estimate (or predict) an appropriate price using the listing data below

**MODEL 1:** Predict **price** (Y) based on **accommodation** ( $X_1$ ):

$$\hookrightarrow Y = 55.71 + (16.6 \cdot X_1)$$

**MODEL 2:** Predict **price** (Y) based on **accommodation** ( $X_1$ ) and number of **bedrooms** ( $X_2$ ):

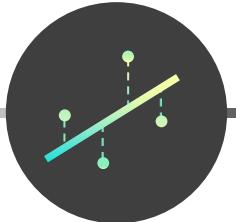
$$\hookrightarrow Y = 52.59 + (15.4 \cdot X_1) + (5.1 \cdot X_2)$$

**MODEL 3:** Predict **price** (Y) based on **accommodation** ( $X_1$ ), number of **bedrooms** ( $X_2$ ), and **room type** (entire.place ( $X_3$ ), hotel.room ( $X_4$ ), private.room ( $X_5$ )):

$$\hookrightarrow Y = 43.82 + (5.7 \cdot X_1) + (5.1 \cdot X_2) + (63.7 \cdot X_3) + (65.4 \cdot X_4) + (9.8 \cdot X_5)$$

**MODEL 4:** Predict **price** (Y) based on **accommodation** ( $X_1$ ), number of **bedrooms** ( $X_2$ ), **room type** (entire.place ( $X_3$ ), hotel.room ( $X_4$ ), private.room ( $X_5$ )), and **district** (manhattan ( $X_6$ ), Brooklyn ( $X_7$ )):

$$\hookrightarrow Y = 26.1 + (6.3 \cdot X_1) + (6.7 \cdot X_2) + (60.5 \cdot X_3) + (54.5 \cdot X_4) + (10.6 \cdot X_5) + (28.5 \cdot X_6) + 9.8 \cdot X_7$$



# MULTIPLE LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

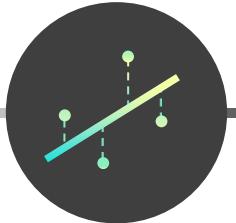
Non-Linear Regression

## EXAMPLE

You are preparing to list a new property on **AirBnB**, and want to estimate (or predict) an appropriate price using the listing data below

list_id	accommodates	bedrooms	entire.place	hotel.room	private.room	brooklyn	manhattan	price	Model 1	Model 2	Model 3	Model 4
3831	3	1	1	0	0	1	0	\$73	\$106	\$104	\$130	\$122
5121	2	1	0	0	1	1	0	\$60	\$89	\$88	\$70	\$66
5178	2	1	0	0	1	0	1	\$79	\$89	\$88	\$70	\$84
5203	1	1	0	0	1	0	1	\$75	\$72	\$73	\$64	\$78
5803	2	1	0	0	1	1	0	\$83	\$89	\$88	\$70	\$66
6872	1	1	0	0	1	0	1	\$65	\$72	\$73	\$64	\$78
6990	1	1	0	0	1	0	1	\$62	\$72	\$73	\$64	\$78
7097	4	1	1	0	0	0	1	\$199	\$122	\$119	\$135	\$128
7750	1	2	0	0	1	0	1	\$96	\$72	\$78	\$69	\$85
8490	5	1	1	0	0	1	0	\$120	\$139	\$135	\$141	\$134
9657	3	1	1	0	0	0	1	\$150	\$106	\$104	\$130	\$140
9704	2	1	0	0	1	0	1	\$55	\$89	\$88	\$70	\$84
10452	3	1	0	0	1	1	0	\$70	\$106	\$104	\$76	\$72
10962	2	1	0	0	1	1	0	\$83	\$89	\$88	\$70	\$66
11943	1	1	0	0	1	1	0	\$150	\$72	\$73	\$64	\$59
12192	2	1	0	0	1	0	1	\$40	\$89	\$88	\$70	\$84
12343	3	1	1	0	0	0	0	\$150	\$106	\$104	\$130	\$140
12937	4	1	0	0	1	0	0	\$130	\$122	\$119	\$82	\$68
12940	2	1	1	0	0	1	0	\$99	\$89	\$88	\$124	\$116
13121	4	1	0	0	1	0	0	\$75	\$122	\$119	\$82	\$68
13394	2	1	0	0	1	1	0	\$80	\$89	\$88	\$70	\$66
13808	2	1	0	0	1	1	0	\$81	\$89	\$88	\$70	\$66
14290	2	2	1	0	0	1	0	\$177	\$94	\$91	\$177	\$177

Model 1      Model 2      Model 3      Model 4  
Sum of Squared Error (SSE): **226,577**    **224,259**    **172,201**    **158,591**



# NON-LINEAR REGRESSION

Linear Relationships

Least Squared Error

Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression



What if the relationship between variables **isn't linear**?

**Non-linear regression** is used when variables have a non-linear relationship, but can be transformed to create a linear one

- This works exactly like linear regression, except you use *transformed* versions of your dependent or independent variables:

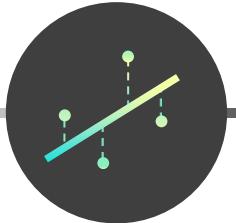
$$y = \alpha + \beta * \ln(x) + \varepsilon$$

Annotations for the equation:

- Dependent variable (DV)
- Coefficient/parameter
- Error/residual
- Y-intercept
- Log-transformed independent variable (IV)



All we're really doing is transforming the data to create linear relationships between each IV and the DV, then applying a **standard linear regression model** using those transformed values



# NON-LINEAR REGRESSION

Linear Relationships

Least Squared Error

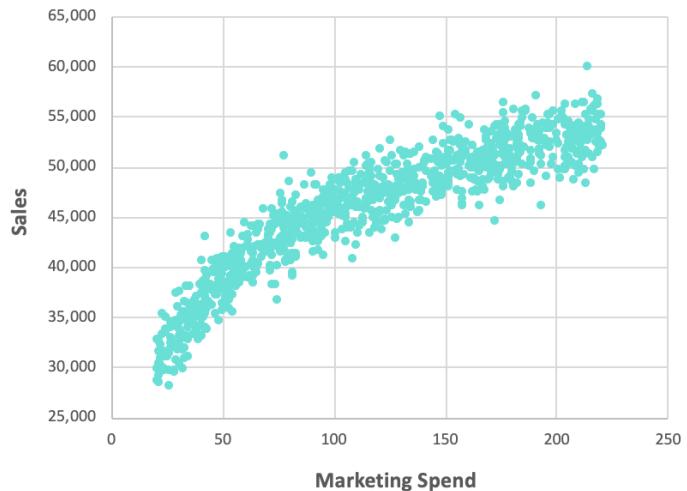
Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

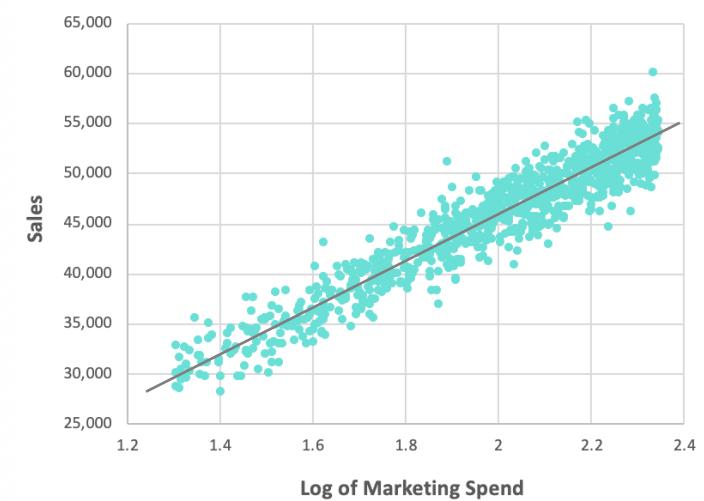
## EXAMPLE #1

You are predicting **Sales** (Y) using **Marketing Spend** (X). As you spend more on marketing, the impact on sales eventually begins to diminish.



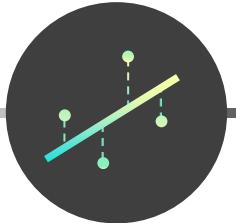
The relationship between Sales and Marketing Spend is **non-linear (logarithmic)**...

$$y = \alpha + \beta x + \varepsilon$$



...but the relationship between Sales and the **log of Marketing Spend** is **linear!**

$$y = \alpha + \beta^* \ln(x) + \varepsilon$$



# NON-LINEAR REGRESSION

Linear Relationships

Least Squared Error

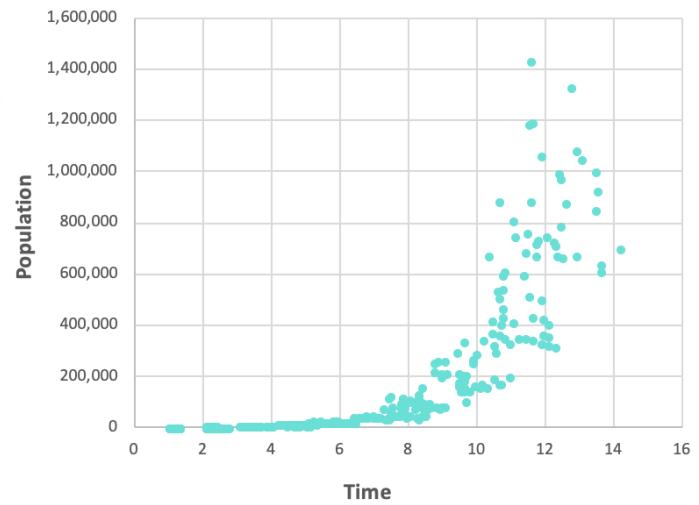
Univariate Linear Regression

Multiple Linear Regression

Non-Linear Regression

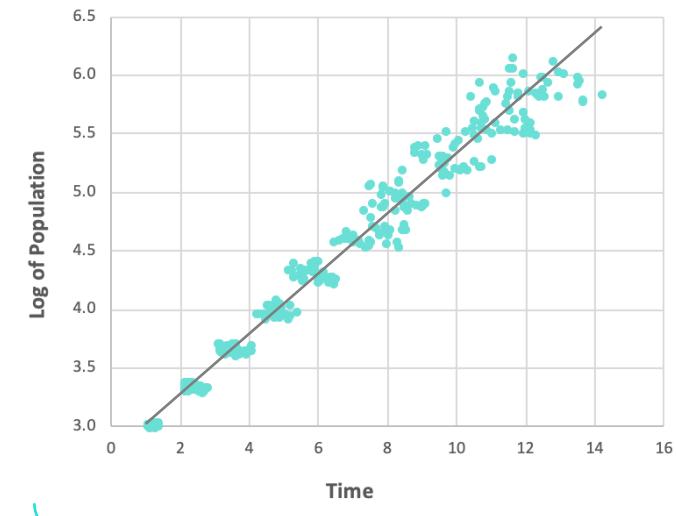
## EXAMPLE #2

You are predicting **population growth** (Y) over **time** (X) and notice an increasing rate of growth as the population size increases.



The relationship between Time and Population is **non-linear (exponential)**...

$$y = \alpha + \beta x + \varepsilon$$



...but the relationship between Time and the **log of Population** is **linear!**

$$\ln(y) = \alpha + \beta x + \varepsilon$$

**NOTE:** There are multiple ways to transform variables based on the type of relationship (log, exponential, cubic, etc.), and multiple techniques to model them (more on that later!)

# CASE STUDY: NON-LINEAR REGRESSION

---



## THE SITUATION

You work as a Marketing Analyst for **Maven Marketing**, an international advertising agency based in London.



## THE ASSIGNMENT

Your client has asked you to help set media budgets and estimate sales for an upcoming campaign. Using historical ad spend and revenue, your goal is to **build a regression model to help predict campaign performance.**



## THE OBJECTIVES

1. Collect historical data for daily ad spend and revenue
2. Create a linear regression, and gauge the fit. Does it look accurate?
3. Transform the ad spend values to fit a logarithmic relationship, and re-run the model. How does this compare to the linear version?

# MODEL DIAGNOSTICS

# MODEL DIAGNOSTICS



In this section we'll explore **common diagnostic metrics** used to evaluate regression models and ensure that predictions are stable and accurate

## TOPICS WE'LL COVER:

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation Factor

## GOALS FOR THIS SECTION:

- Understand how to interpret regression model outputs
- Define common diagnostic metrics like  $R^2$ , MSE/MAE/MAPE, P-Values, F-statistics, etc.
- Explore the difference between homoskedasticity and heteroskedasticity in a linear regression model
- Understand how to identify and measure multicollinearity using variance inflation factor (VIF)



# SAMPLE MODEL OUTPUT

## Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

```
Call:  
lm(formula = price ~ accommodates + bedrooms + Entire.place +  
    host_has_profile_pic + Hotel.room + Private.room, data = d.2)  
  
Residuals:  
    Min      1Q      Median      3Q      Max  
-135.057 -27.091   -8.091   20.909  194.388  
  
Coefficients:  
            Estimate Std. Error t value P-value  
(Intercept) 47.9501    4.4393 10.801 0.0000000  
accommodates  5.7166    0.2238 25.540 0.0000000  
bedrooms      5.0933    0.5478  9.298 0.0000000  
Entire.place   63.6845   1.6472 38.663 0.0000000  
host_has_profile_pic -4.1485   4.1430 -1.001 0.3135200  
Hotel.room     65.4399   3.5595 18.385 0.0000000  
Private.room    9.7627   1.6065  6.077 0.0000000  
---  
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  
  
Residual standard error: 10.21 on 29797 degrees of freedom  
Multiple R-squared:  0.416431  
F-statistic: 40.31352 on 6 and 29797 degrees of freedom
```

A	B	C	D	E	F	G	H	I
SUMMARY OUTPUT								
Regression Statistics								
1	2	3	4	5	6	7	8	9
Multiple R	0.645315							
R Square	0.416431							
Adjusted R Square	0.416314							
Standard Error	40.31352							
Observations	29797							
ANOVA								
10	11	df	SS	MS	F	Significance F		
	12	Regression	6	34548002	5758000	3542.993	0	

OLS Regression Results	
Dep. Variable:	y
Model:	OLS
Method:	Least Squares
Date:	Tue, 02 Feb 2021
Time:	06:52:01
No. Observations:	50
Df Residuals:	46
Df Model:	3
Covariance Type:	nonrobust

	coef	std err	t	P> t	[0.025	0.975]
x1	0.4687	0.026	17.751	0.000	0.416	0.522
x2	0.4836	0.104	4.659	0.000	0.275	0.693
x3	-0.0174	0.002	-7.507	0.000	-0.022	-0.013
const	5.2058	0.171	30.405	0.000	4.861	5.550



# R-SQUARED

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

**R-Squared** measures how well your model explains the variance in the dependent variable you are predicting

- The higher the R-Squared, the “better” your model predicts variance in the DV and the more confident you can be in the accuracy of your predictions
- **Adjusted R-Squared** is often used as it “penalizes” the R-squared value based on the number of variables included in the model

$$R^2 = 1 - \frac{SSE}{TSS} = \frac{\text{Sum of Squared Error}}{\text{Total Sum of Squares}} = \frac{\sum_i (y_i - prediction_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Total distance between predicted and actual values, squared (aka **squared error**)

Total distance between each  $y$  value and the mean, squared (basically **variance**, without dividing by  $n$ )



# R-SQUARED EXAMPLE

Sample Model Output

R-Squared

Mean Error Metrics

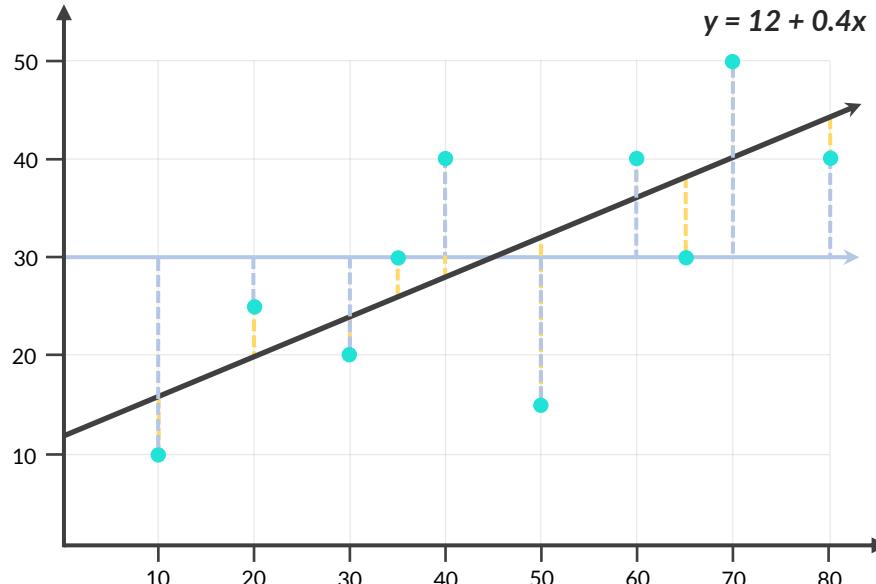
Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation



x	y	prediction	$(y - \text{prediction})^2$	$\bar{y}$	$(y - \bar{y})^2$
10	10	16	36	30	400
20	25	20	25	30	25
30	20	24	16	30	100
35	30	26	16	30	0
40	40	28	144	30	100
50	15	32	289	30	225
60	40	36	16	30	100
65	30	38	64	30	0
70	50	40	100	30	400
80	40	44	16	30	100

$$\left. \begin{aligned} \text{SSE} &= 722 \\ \text{TSS} &= 1,450 \end{aligned} \right\}$$

$$R^2 = 1 - \frac{\text{SSE}}{\text{TSS}} = \frac{\sum_i (y_i - \text{prediction}_i)^2}{\sum_i (y_i - \bar{y})^2}$$
$$1 - \frac{722}{1,450} = 0.502$$



# MEAN ERROR METRICS

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

**Mean error metrics** measure how well your regression model **predicts**, as opposed to how well it explains variance (like R-Squared)

- There are many variations, but the most common ones are **Mean Squared Error** (MSE), **Mean Absolute Error** (MAE) and **Mean Absolute Percentage Error** (MAPE)
- These metrics provide “standards” which can be used to compare predictive accuracy across multiple regression models

**MSE**

$$\frac{\sum_i (y_i - \text{prediction}_i)^2}{n}$$

Average of the **squared** distance between actual & predicted values

**MAE**

$$\frac{\sum_i |y_i - \text{prediction}_i|}{n}$$

Average of the **absolute** distance between actual & predicted values

**MAPE**

$$\frac{\sum_i \frac{|y_i - \text{prediction}_i|}{y_i}}{n}$$

Mean Absolute Error, converted to a **percentage**



Mean error metrics can be used to evaluate regression models just like performance metrics like **accuracy**, **precision** and **recall** can be used to evaluate classification models



# MSE EXAMPLE

Sample Model Output

R-Squared

Mean Error Metrics

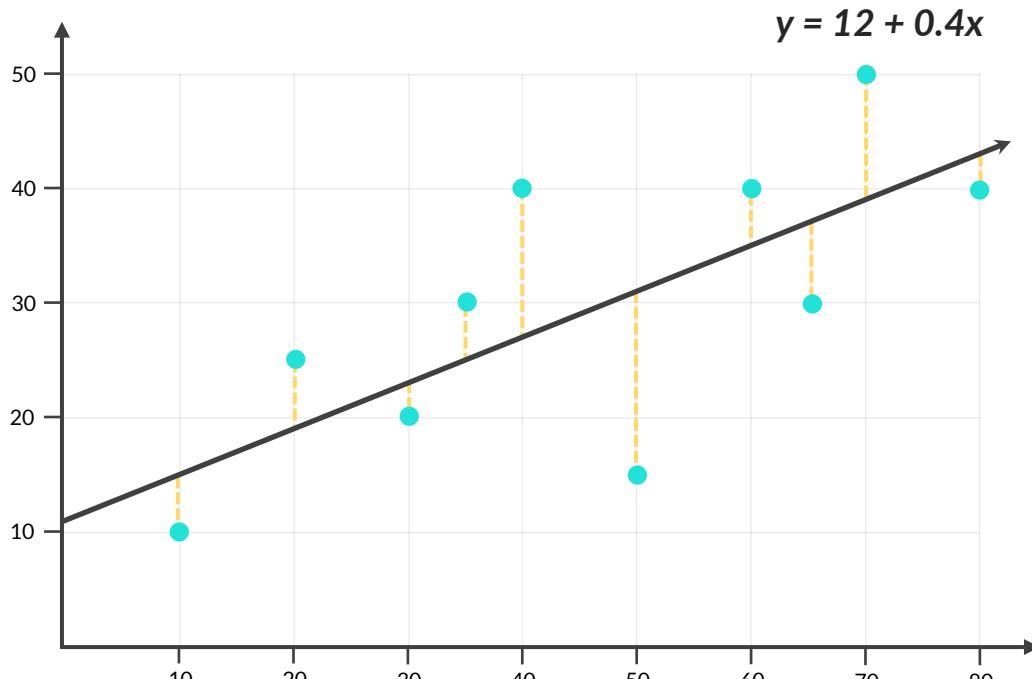
Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation



MSE

$$\text{MSE} = \frac{\sum_i (y_i - \text{prediction}_i)^2}{n} = \frac{722}{10} = 72.2$$

X	Y (actual)	Y (line)	Error	Sq. Error
10	10	16	6	36
20	25	20	-5	25
30	20	24	4	16
35	30	26	-4	16
40	40	28	-12	144
50	15	32	17	289
60	40	36	-4	16
65	30	38	8	64
70	50	40	-10	100
80	40	44	4	16

II  
SUM OF SQUARED ERROR: 722



# MAE EXAMPLE

Sample Model Output

R-Squared

Mean Error Metrics

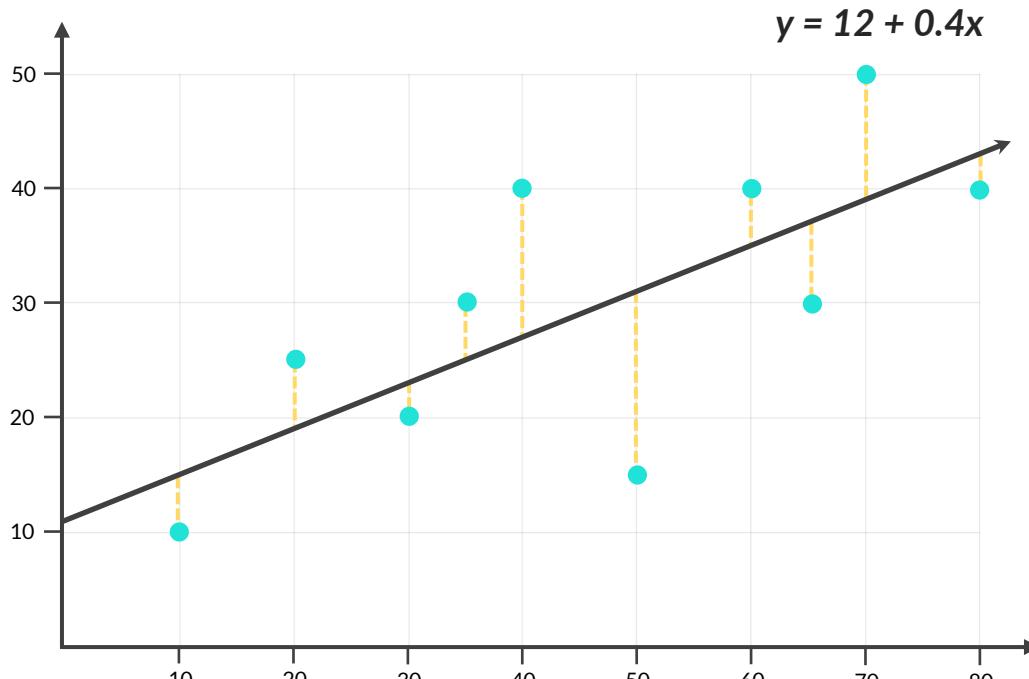
Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation



MAE

$$\text{MAE} = \frac{\sum_i |y_i - \text{prediction}_i|}{n} = \frac{74}{10} = 7.4$$

X	Y (actual)	Y (line)	Error	Abs Error
10	10	16	6	6
20	25	20	-5	5
30	20	24	4	4
35	30	26	-4	4
40	40	28	-12	12
50	15	32	17	17
60	40	36	-4	4
65	30	38	8	8
70	50	40	-10	10
80	40	44	4	4

SUM OF ABSOLUTE ERROR: **74**

II



# MAPE EXAMPLE

Sample Model Output

R-Squared

Mean Error Metrics

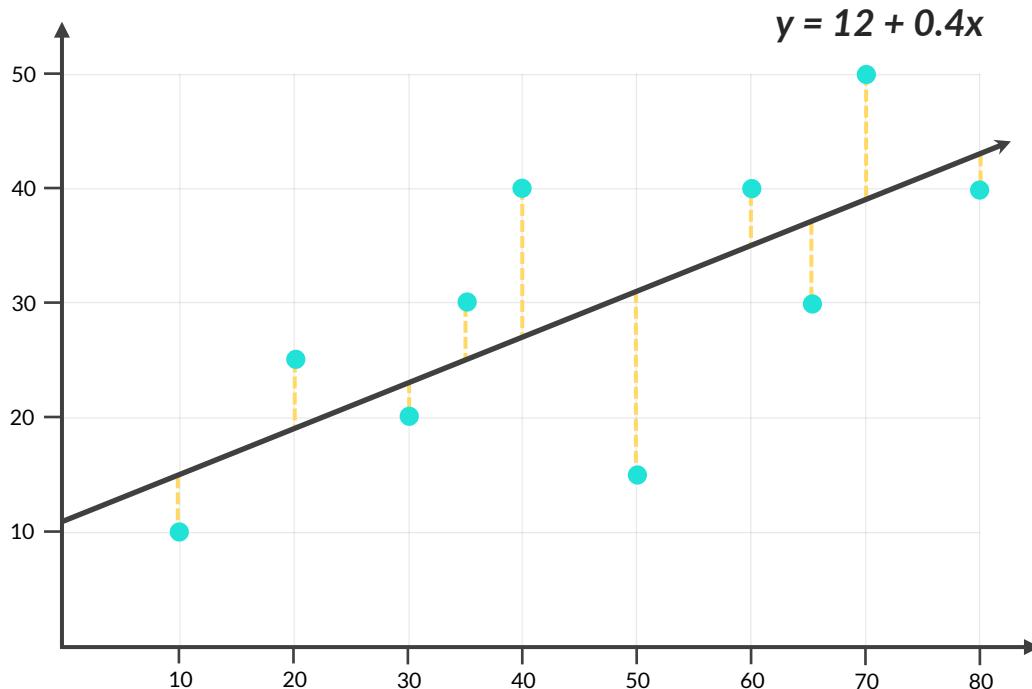
Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation



MAPE

$$\text{MAPE} = \frac{\sum_i \frac{|y_i - \text{prediction}_i|}{y_i}}{n} = \frac{3.233}{10} = 32.33\%$$

X	Y (actual)	Y (line)	Error	Abs Error	Abs % Error
10	10	16	6	6	0.6
20	25	20	-5	5	0.2
30	20	24	4	4	0.2
35	30	26	-4	4	0.133
40	40	28	-12	12	0.3
50	15	32	17	17	1.133
60	40	36	-4	4	0.1
65	30	38	8	8	0.267
70	50	40	-10	10	0.2
80	40	44	4	4	0.1

SUM OF ABSOLUTE % ERROR: 3.233

II



# MEAN ERROR METRICS

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation



When should I use each type of error metric?

- **Mean Squared Error** (MSE) is particularly useful when outliers or extreme values are important to predict
- **Mean Absolute Error** (MAE) is useful if you want to minimize the impact of outliers on model selection
- **Mean Absolute Percentage Error** (MAPE) is useful when your DV is on a very large scale, or if you want to compare models on a more intuitive scale



**PRO TIP:** In general we recommend considering **all of them**, since they can be calculated instantly and each provide helpful context into model performance



# HOMOSKEDASTICITY

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

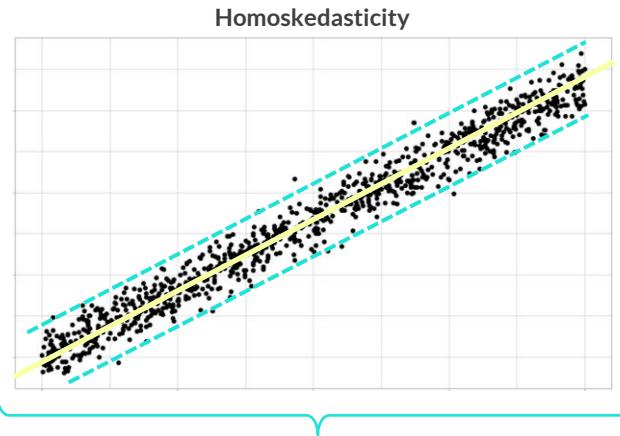
P-Values & T-Statistics

Multicollinearity

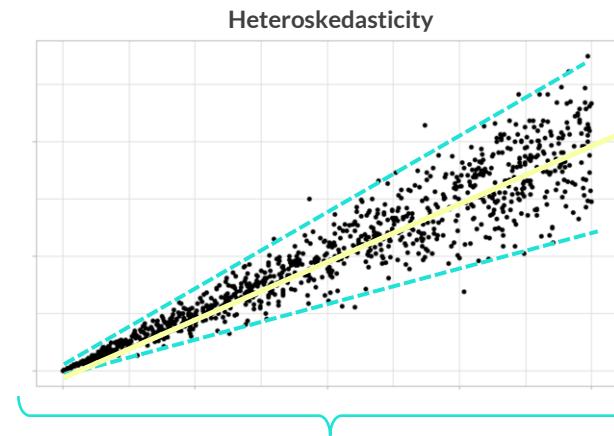
Variance Inflation

**Homoskedasticity** is a term used to describe a model with consistent “scatter”, or variance in residuals, across all IV values

- In order to make accurate predictions across the full range of IV values, models should have normally distributed residuals (errors) with a zero mean
- **Heteroskedasticity** describes a model with *inconsistent* residual variance, meaning that it predicts poorly for certain IV values and will fail to generalize



Residuals are consistent over the entire IV range



Residuals increase at higher IV values, **indicating that there is some variance that the IVs are unable to explain**



Breusch-Pagan tests can report a formal calculation of Heteroskedasticity, but usually a simple visual check is enough

# NULL HYPOTHESIS

---



## NULL HYPOTHESIS ( $H_0$ ) [null hy·poth·e·sis]

---

**noun**

1. In a statistical test, the hypothesis that there is no significant difference between specified populations, any observed difference being due to sampling or experimental error \*



The hypothesis that your model is garbage

Our goal is to **reject the null hypothesis** and prove (with a high level of confidence) that our model can produce accurate, statistically significant predictions and not just random outputs



# F-STATISTIC & F-SIGNIFICANCE

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

The **F-Statistic** and associated **P-Value** (aka **F-Significance**) help us understand the predictive power of the model *as a whole*

- **F-Significance** is technically defined as “*the probability that the null hypothesis cannot be rejected*”, which can be interpreted as **the probability that your model predicts poorly**
- The smaller the F-Significance, the more useful your regression is for prediction
- **NOTE:** It’s common practice to use a P-Value of **.05** (aka **95%**) as a threshold to determine if a model is “statistically significant”, or valid for prediction



**PRO TIP:** F-Significance should be the **first thing** you check when you evaluate a regression model; if it’s above your threshold, you may need more training, if it’s below your threshold, move on to coefficient-level significance (up next!)



# T-STATISTICS & P-VALUES

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

**T-Statistics** and their associated **P-Values** help us understand the predictive power of the *individual model coefficients*

```
call:  
lm(formula = price ~ accommodates + bedrooms + Entire.place +  
    host_has_profile_pic + Hotel.room + Private.room, data = d.2)  
  
Residuals:  
    Min      1Q      Median      3Q      Max  
-135.057 -27.091   -8.091   20.909  194.388  
  
Coefficients:  
            Estimate Std. Error t value Pr(>|t|)  
(Intercept) 47.9501   4.4393 10.801 < 2e-16 ***  
accommodates  5.7166   0.2238 25.540 < 2e-16 ***  
bedrooms     5.0933   0.5478  9.298 < 2e-16 ***  
Entire.place  63.6845   1.6472 38.663 < 2e-16 ***  
host_has_profile_pic -4.1485   4.1430 -1.001 0.317  
Hotel.room    65.4399   3.5595 18.385 < 2e-16 ***  
Private.room   9.7627   1.6065  6.077 1.24e-09 ***  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 40.31 on 29790 degrees of freedom  
Multiple R-squared:  0.4164, Adjusted R-squared:  0.4163  
F-statistic: 3543 on 6 and 29790 DF, p-value: < 2.2e-16
```

**T-Statistics** tell us the degree to which we can “trust” the coefficient estimates

- Calculated by dividing the coefficient by its standard error
- Primarily used as a stepping-stone to calculate P-Values, which are easier to interpret and more commonly used for diagnostics

**P-Values** tell us the probability that the coefficient is meaningless, statistically speaking

- The smaller the P-value, the more confident you can be that the coefficient is valid (*not 0*)
- Statistical significance** is calculated as **(1 - P)**

P-Value	Statistical Significance
0.001	99.9%***
0.01	99%**
0.05	95%*



# MULTICOLLINEARITY

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

**Multicollinearity** occurs when two or more independent variables are highly correlated, leading to untrustworthy model coefficients

- Correlation means that one IV can be used to predict another (*i.e. height* and **weight**), leading to many combinations of coefficients that predict equally well
- This leads to unreliable coefficient estimates, and means that your model will fail to generalize when applied to non-training data



How do I measure multicollinearity, and what can I do about it?

- **Variance Inflation Factor** (VIF) can help you quantify the degree of multicollinearity, and determine which IVs to exclude from the model



# VARIANCE INFLATION FACTOR

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

To calculate **Variance Inflation Factor** (VIF) you treat each individual IV as the *dependent* variable, and use the  $R^2$  value to measure how well you can predict them using the other IVs in the model

$$Y = \alpha + \beta_1x_1 + \beta_2x_2 + \beta_3x_3 + \dots + \beta_nx_n + \varepsilon$$



$$x_1 = \alpha + \beta_2x_2 + \beta_3x_3 + \dots + \beta_nx_n + \varepsilon \longrightarrow \frac{1}{1-R^2} \rightarrow \text{VIF for } X_1$$

$$x_2 = \alpha + \beta_1x_1 + \beta_3x_3 + \dots + \beta_nx_n + \varepsilon \longrightarrow \frac{1}{1-R^2} \rightarrow \text{VIF for } X_2$$



**PRO TIP:** As a rule of thumb, a **VIF >10** indicates that multicollinearity is a problem, and that one or more IVs is redundant and should be removed from the model



# VARIANCE INFLATION FACTOR

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

## EXAMPLE

You are predicting the **price of an AirBnB listing**, and notice the following results after calculating the VIF for each independent variable:

accommodates	bedrooms	Entire.place	Hotel.room	Private.room	Manhattan	Brooklyn
2.144259	1.720523	12.304416	1.246123	11.791492	1.988467	1.949623

**Entire.place** and **Private.room** produce high VIF values, since they essentially measure the same thing;  
if a listing isn't a private room, there's a high probability that it's an entire place (and vice versa)

**SOLUTION:** Pick one high-VIF variable to remove (arbitrarily), re-run the model,  
and recalculate the VIF values to see if multicollinearity is gone

accommodates	bedrooms	Entire.place	Hotel.room	Manhattan	Brooklyn
2.143976	1.720513	1.473047	1.013757	1.988459	1.949182



Adios multicollinearity!



**PRO TIP:** Use a frequency  
table to confirm correlation!

		Private.room	
		NO	YES
Entire.place	NO	816	15,815
	YES	13,166	0



# RECAP: SAMPLE MODEL OUTPUT

Sample Model Output

R-Squared

Mean Error Metrics

Homoskedasticity

F-Significance

P-Values & T-Statistics

Multicollinearity

Variance Inflation

Formula for the regression (variables & data set)

call:  
lm(formula = price ~ accommodates + bedrooms + Entire.place +  
host\_has\_profile\_pic + Hotel.room + Private.room, data = d.2)

Profile of  
Residuals/Errors

Residuals:

Min	1Q	Median	3Q	Max
-135.057	-27.091	-8.091	20.909	194.388

Y-Intercept & IV  
coefficients

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	47.9501	4.4393	10.801	< 2e-16	***
accommodates	5.7166	0.2238	25.540	< 2e-16	***
bedrooms	5.0933	0.5478	9.298	< 2e-16	***
Entire.place	63.6845	1.6472	38.663	< 2e-16	***
host_has_profile_pic	-4.1485	4.1430	-1.001	0.317	
Hotel.room	65.4399	3.5595	18.385	< 2e-16	***
Private.room	9.7627	1.6065	6.077	1.24e-09	***

Coefficient  
P-Values

R<sup>2</sup> and  
Adjusted R<sup>2</sup>

Residual standard error: 40.31 on 29790 degrees of freedom  
Multiple R-squared: 0.4164, Adjusted R-squared: 0.4163  
F-statistic: 3543 on 6 and 29790 DF, p-value: < 2.2e-16

Coefficient  
Standard Errors  
& T-Values

F-Statistic and P-Value (aka F-Significance)

# TIME-SERIES FORECASTING

# TIME-SERIES FORECASTING



In this section we'll explore common **time-series forecasting** techniques, which use regression models to predict future values based on seasonality and trends

## TOPICS WE'LL COVER:

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis

## GOALS FOR THIS SECTION:

- Understand when, why and how to use common time-series forecasting techniques
- Learn how to identify and quantify seasonality using auto correlation and one-hot encoding
- Explore common models for non-linear forecasting (like ADBUDG and Gompertz)
- Apply forecasting techniques to analyze the impact of key business decisions (aka “interventions”)



# FORECASTING 101

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

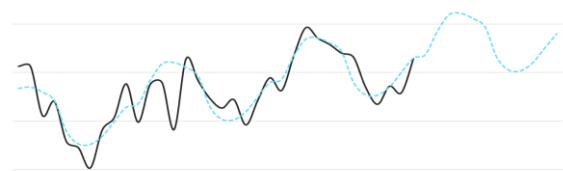
Intervention Analysis

**Time-series forecasting** is all about predicting future values of a single, numeric dependent variable

- Forecasting works just like any other regression model, except your data must contain multiple observations of your DV over time (aka **time-series** data)
- Time-series forecast models look for patterns in the observed data – like **seasonality** or **linear/non-linear trends** – to accurately predict future values

## Common Examples:

- Forecasting revenue for the next fiscal year
- Predicting website traffic growth over time
- Estimating sales for a new product launch





# SEASONALITY

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

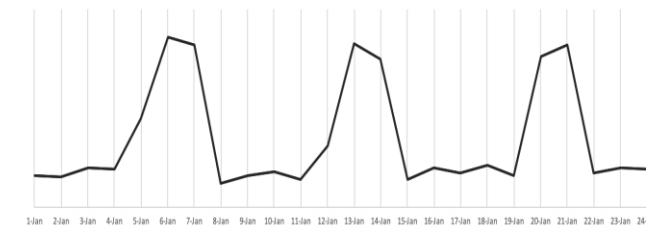
Intervention Analysis

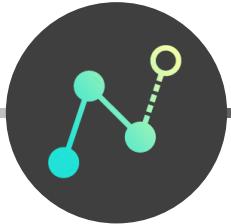
**Seasonality** is a repeatable, predictable pattern that a dependent variable may follow over time

- Seasonality often aligns with specific time periods (*calendar months, fiscal periods, days of the week, hours of the day, etc.*) but that isn't always the case
- We can identify seasonal patterns using an **Auto Correlation Function (ACF)**, then apply that seasonality to forecasts using techniques like **one-hot encoding** or **moving averages** (*more on that soon!*)

## Common examples:

- Website traffic by hour of the day
- Seasonal product sales
- Airline ticket prices around major holidays





# AUTO CORRELATION FUNCTION

Forecasting 101

Seasonality

Linear Trending

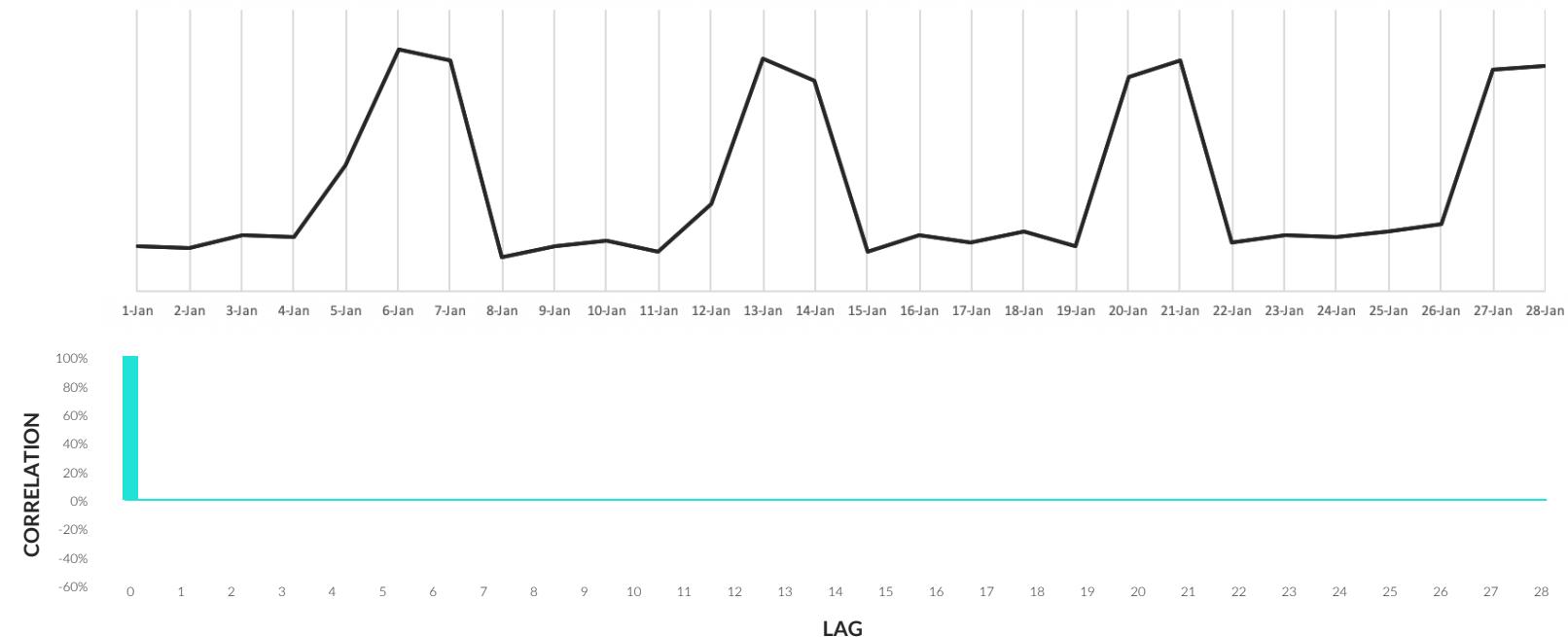
Smoothing

Non-Linear Trends

Intervention Analysis

**ACF** essentially involves calculating the correlation between time-series data and lagged versions of itself, then plotting those correlations

- This allows you to visualize which lags are highly correlated with the original data, and reveals the length (or period) of the seasonal trend





# AUTO CORRELATION FUNCTION

Forecasting 101

Seasonality

Linear Trending

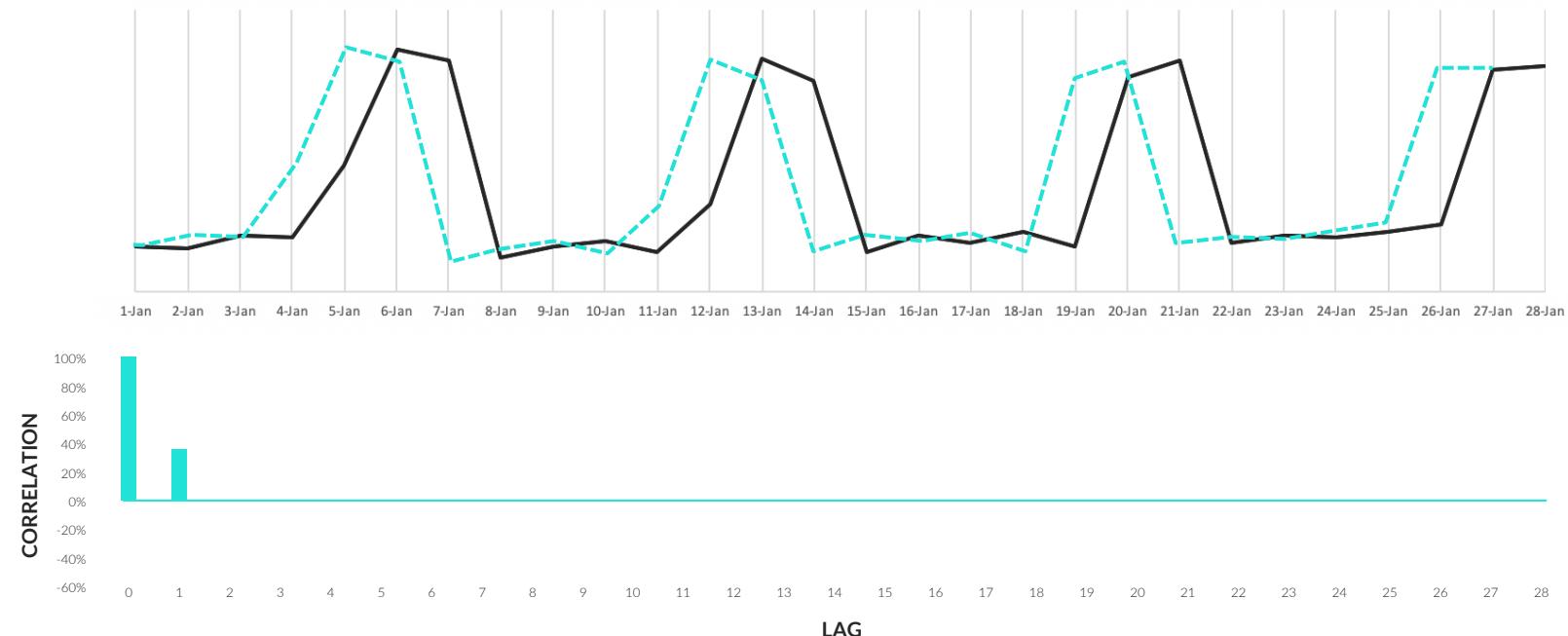
Smoothing

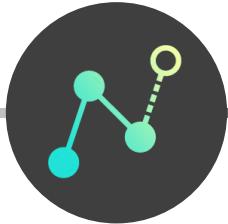
Non-Linear Trends

Intervention Analysis

**ACF** essentially involves calculating the correlation between time-series data and lagged versions of itself, then plotting those correlations

- This allows you to visualize which lags are highly correlated with the original data, and reveals the length (or period) of the seasonal trend





# AUTO CORRELATION FUNCTION

Forecasting 101

Seasonality

Linear Trending

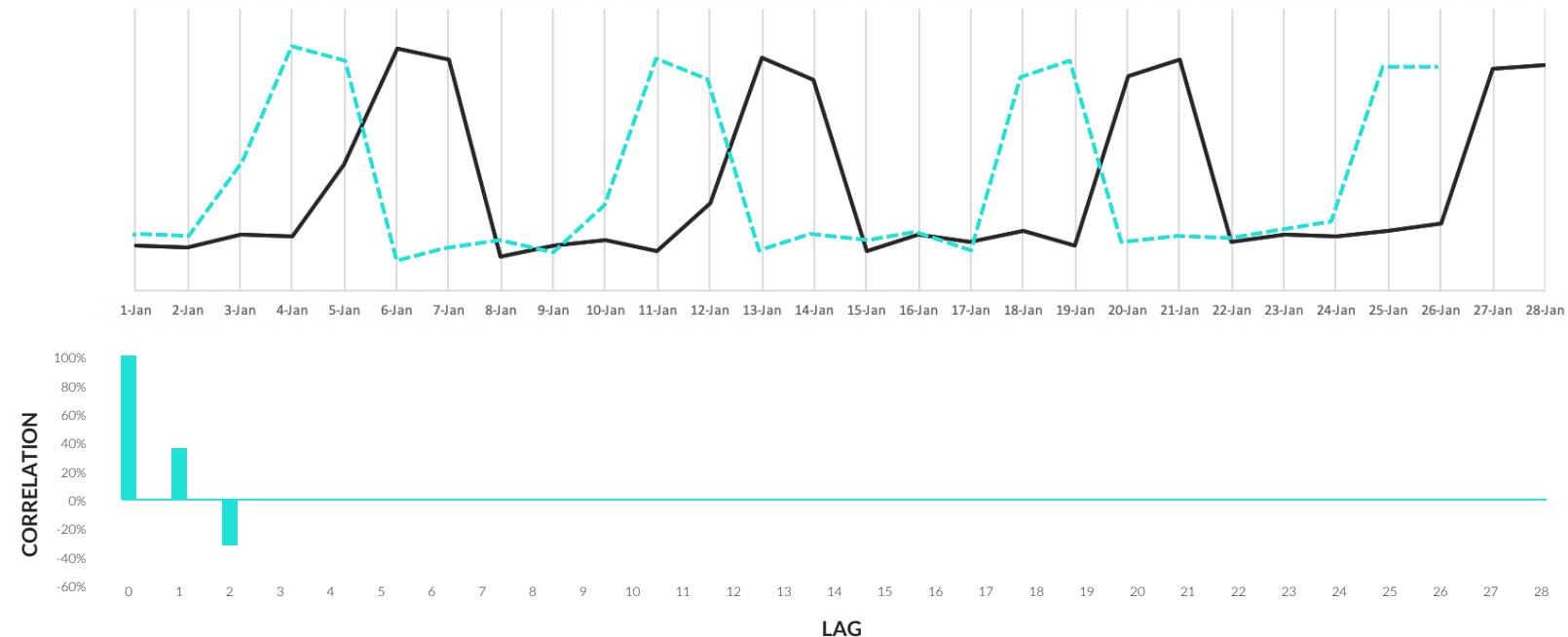
Smoothing

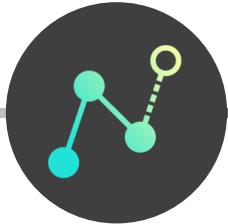
Non-Linear Trends

Intervention Analysis

**ACF** essentially involves calculating the correlation between time-series data and lagged versions of itself, then plotting those correlations

- This allows you to visualize which lags are highly correlated with the original data, and reveals the length (or period) of the seasonal trend





# AUTO CORRELATION FUNCTION

Forecasting 101

Seasonality

Linear Trending

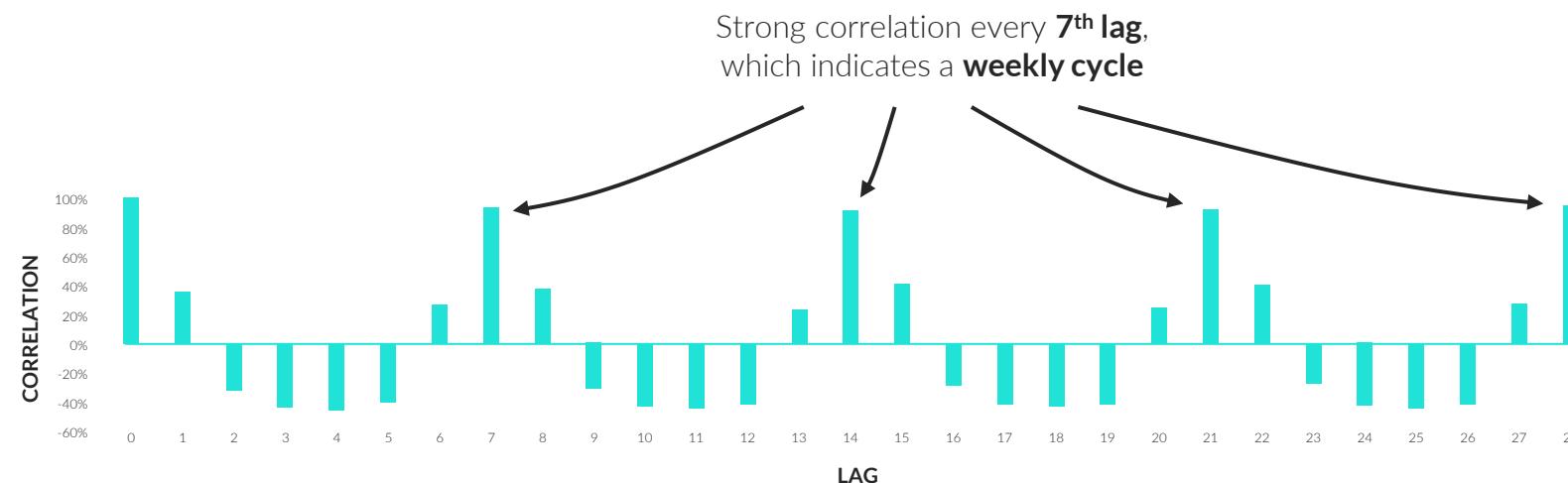
Smoothing

Non-Linear Trends

Intervention Analysis

**ACF** essentially involves calculating the correlation between time-series data and lagged versions of itself, then plotting those correlations

- This allows you to visualize which lags are highly correlated with the original data, and reveals the length (or period) of the seasonal trend



# CASE STUDY: AUTO CORRELATION



## THE SITUATION

You are a Business Intelligence Analyst for **Accucorp Accounting**, a national firm specializing in tax preparation services.



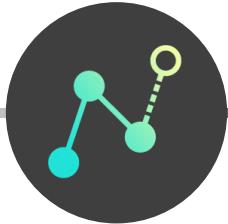
## THE ASSIGNMENT

The marketing team is hoping to better understand customer behavior, and would like to know if online searches for “tax”-related keywords follow a seasonal pattern. Your goal is to **analyze monthly search data and use auto correlation to test for seasonality**.



## THE OBJECTIVES

1. Collect monthly search volume for relevant tax keywords
2. Create 36 monthly lags, and calculate the correlation between each lag and the original values
3. Plot the correlations and determine the period of seasonality, if it exists (*monthly, quarterly, annual, etc.*)



# SEASONALITY: ONE-HOT ENCODING

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis

Once you've identified a seasonal pattern, you can use **one-hot encoding** to create independent variables which capture the effect of those time periods (days, weeks, months, quarters, etc.)

- **NOTE:** When you use one-hot encoding, you **must exclude one of the options** rather than encoding all of them (*it doesn't matter which one you exclude*)
- Consider the equation **A+B=5**; there are an *infinite* combination of A & B values that can solve it. One-hot encoding all options creates a similar problem for regression

Quarter ID	Revenue	Q1	Q2	Q3	Q4
1	\$1,300,050	1	0	0	0
2	\$11,233,310	0	1	0	0
3	\$1,112,050	0	0	1	0
4	\$1,582,077	0	0	0	1



**PRO TIP:** If your data contains multiple seasonal patterns (i.e. hour of day + day of week), include both dimensions in the model as one-hot encoded independent variables

# CASE STUDY: ONE-HOT ENCODING

---



## THE SITUATION

You're a Senior Analyst for **Weather Trends**, a Brazilian weather station boasting the longest and most accurate forecasts in the biz.



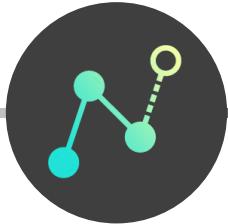
## THE ASSIGNMENT

You've been asked to help prepare temperature forecasts for the upcoming year. To do this, you'll need to analyze ~5 years of historical data from Rio de Janeiro, and use regression to **predict monthly average temperatures**.



## THE OBJECTIVES

1. Collect historical average monthly temperatures from Rio de Janeiro
2. Create independent variables for each month using one-hot encoding
3. Build a regression model to produce a forecast for the following year



# LINEAR TRENDING

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis



What if the data includes both seasonality and a linear trend?

**Trend** describes an overarching direction or movement in a time series, not counting seasonality

- Trends are often linear (up/down), but can be non-linear as well (*more on that later!*)
- To account for linear trending in a regression, you can include a **time step** IV; this is simply an index value that starts at 1 and increments with each time period
- If the time step coefficient isn't statistically significant, it means you don't have a meaningful linear trend



**PRO TIP:** It's common for time-series models to include trending AND seasonality; in this case, use a combination of **one-hot encoding** and **time step variables** to account for both!



# LINEAR TRENDING

# Forecasting 101

## Seasonality

## Linear Trending

## Smoothing

## Non-Linear Trends

## Intervention Analysis



What if the data includes both seasonality and a linear trend?

**Trend** describes an overarching direction or movement in a time series, not counting seasonality

# CASE STUDY: SEASONALITY + TREND

---



## THE SITUATION

You are a Business Intelligence Analyst for **Maven Muscles**, a large national chain of fitness centers.



## THE ASSIGNMENT

The Analytics Director needs your help **building a monthly revenue forecast for the upcoming year**. Memberships follow a clear seasonal pattern, and revenue has been steadily rising as the gym continues to open new locations.



## THE OBJECTIVES

1. Collect historical monthly revenue data
2. Use one-hot encoding and a time-step variable to account for both seasonality and linear trending
3. Fit a regression model to forecast revenue for the upcoming year



# SMOOTHING

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis



What if my data is so noisy that I can't tell if a trend exists?

If your data is highly volatile, **smoothing techniques** can be used to reveal underlying patterns or trends

- Common techniques like **moving averages** or **weighted smoothing** remove random variation and “noise” from the data to help expose seasonality or trends
- If the volatility is truly just noise (*and something we want our model to ignore*), averaging or weighting the values around each time step can help us “smooth” the data and produce more accurate forecasts



**PRO TIP:** Smoothing is a great way to expose patterns and trends that otherwise might be tough to see; make this part of your data profiling process!



# SMOOTHING

Forecasting 101

Seasonality

Linear Trending

Smoothing

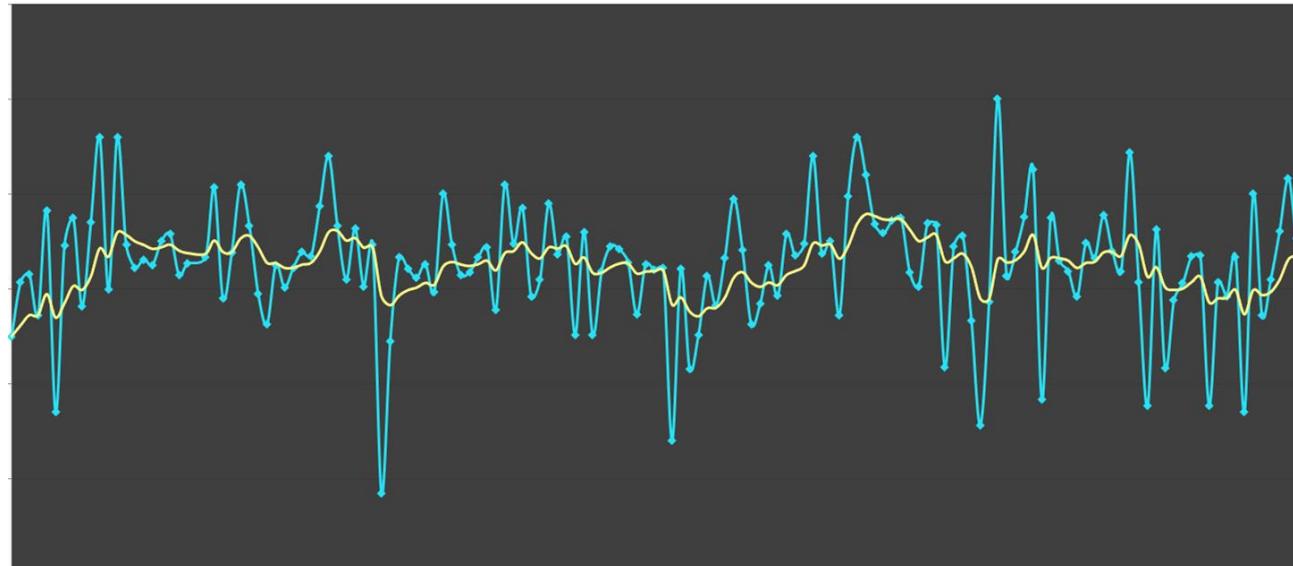
Non-Linear Trends

Intervention Analysis



What if my data is so noisy that I can't tell if a trend exists?

If your data is highly volatile, **smoothing techniques** can be used to reveal underlying patterns or trends



# CASE STUDY: SMOOTHING

---



## THE **SITUATION**

You've just been hired as an analytics consultant for **Maven Motel**, a struggling national motel chain.



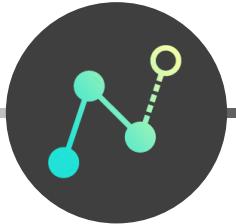
## THE **ASSIGNMENT**

Management has been taking steps to improve guest satisfaction, and has asked you to analyze daily data to determine how ratings are trending. Your task is to **use a moving average calculation to discern if an underlying trend is present.**



## THE **OBJECTIVES**

1. Collect daily average guest ratings for the motel. Do you see any clear patterns or trends?
2. Calculate a moving average, and compare various windows from 1-12 weeks
3. Determine if an underlying trend is present. How would you describe it?



# NON-LINEAR TRENDS

Forecasting 101

Seasonality

Linear Trending

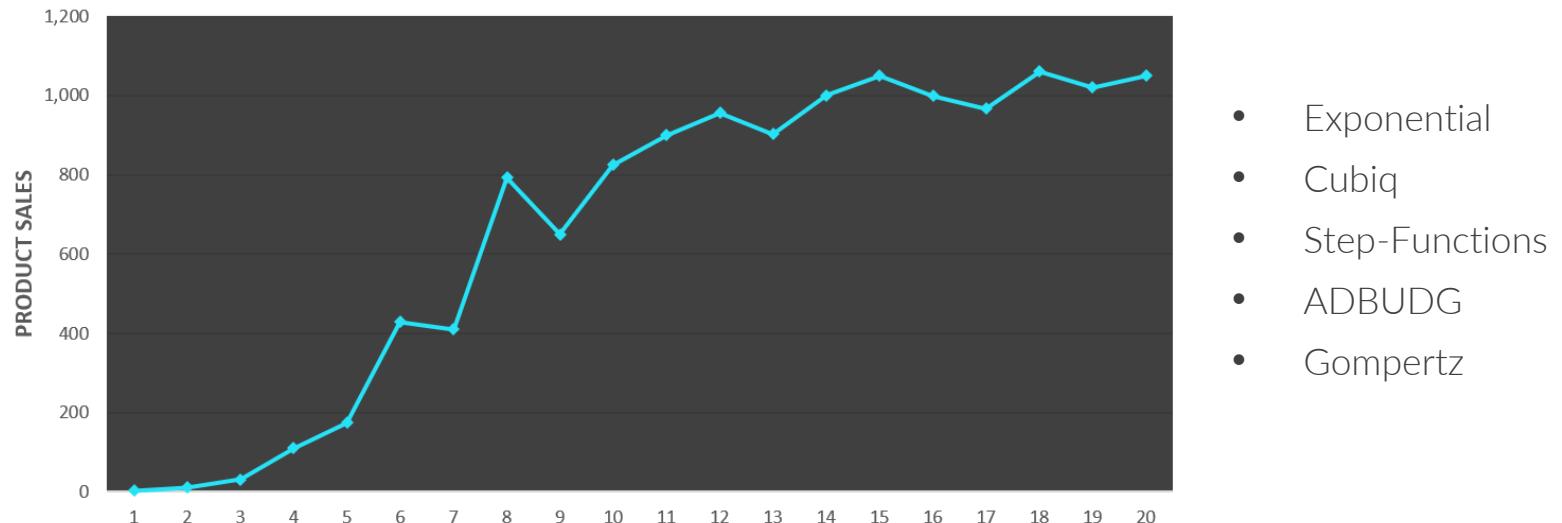
Smoothing

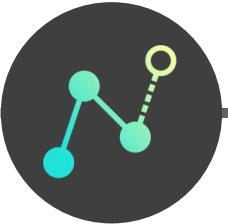
Non-Linear Trends

Intervention Analysis

Time-series data won't always follow a seasonal pattern or linear trend; it may follow a **non-linear trend**, or have no predictable trend at all

- There are many formulas designed to forecast common non-linear trends:





# NON-LINEAR TRENDS

Forecasting 101

Seasonality

Linear Trending

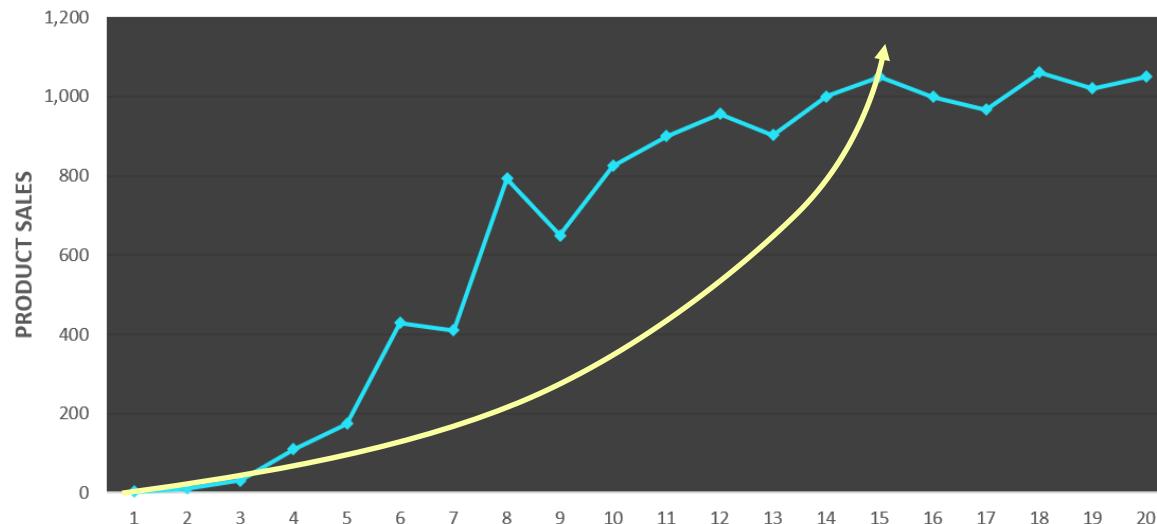
Smoothing

Non-Linear Trends

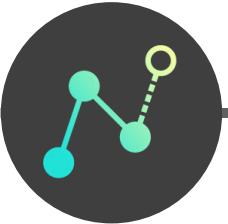
Intervention Analysis

Time-series data won't always follow a seasonal pattern or linear trend; it may follow a **non-linear trend**, or have no predictable trend at all

- There are many formulas designed to forecast common non-linear trends:



- **Exponential**
- Cubic
- Step-Functions
- ADBUDG
- Gompertz



# NON-LINEAR TRENDS

Forecasting 101

Seasonality

Linear Trending

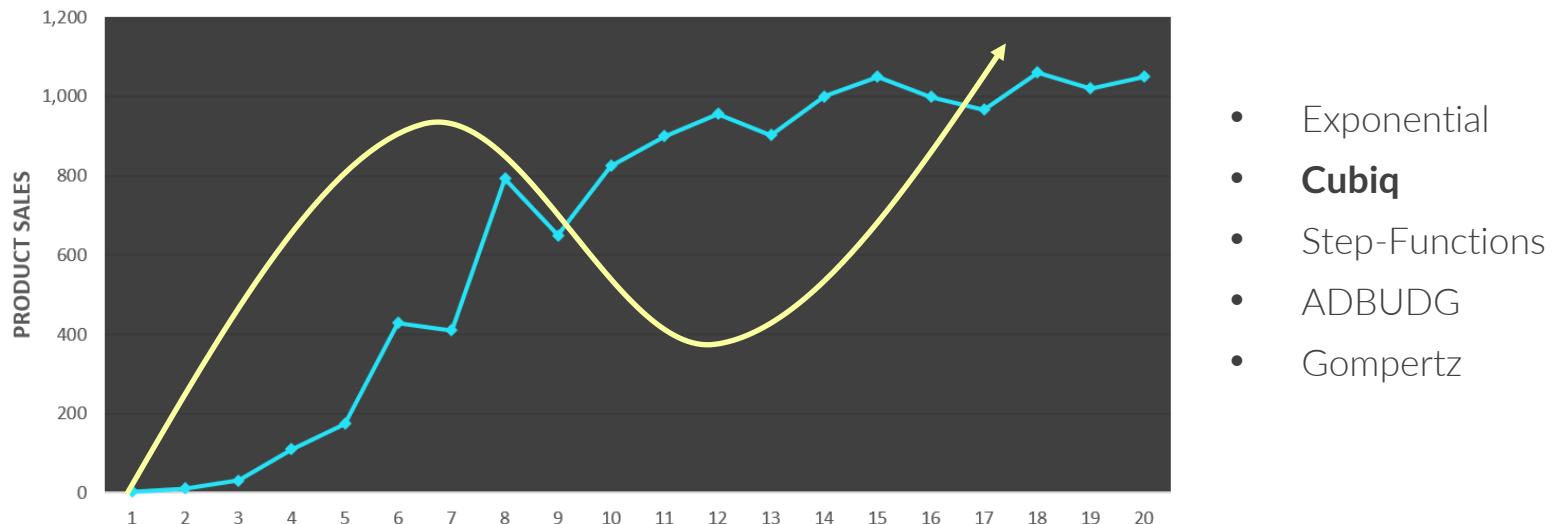
Smoothing

Non-Linear Trends

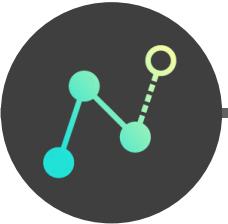
Intervention Analysis

Time-series data won't always follow a seasonal pattern or linear trend; it may follow a **non-linear trend**, or have no predictable trend at all

- There are many formulas designed to forecast common non-linear trends:



- Exponential
- **Cubiq**
- Step-Functions
- ADBUDG
- Gompertz



# NON-LINEAR TRENDS

Forecasting 101

Seasonality

Linear Trending

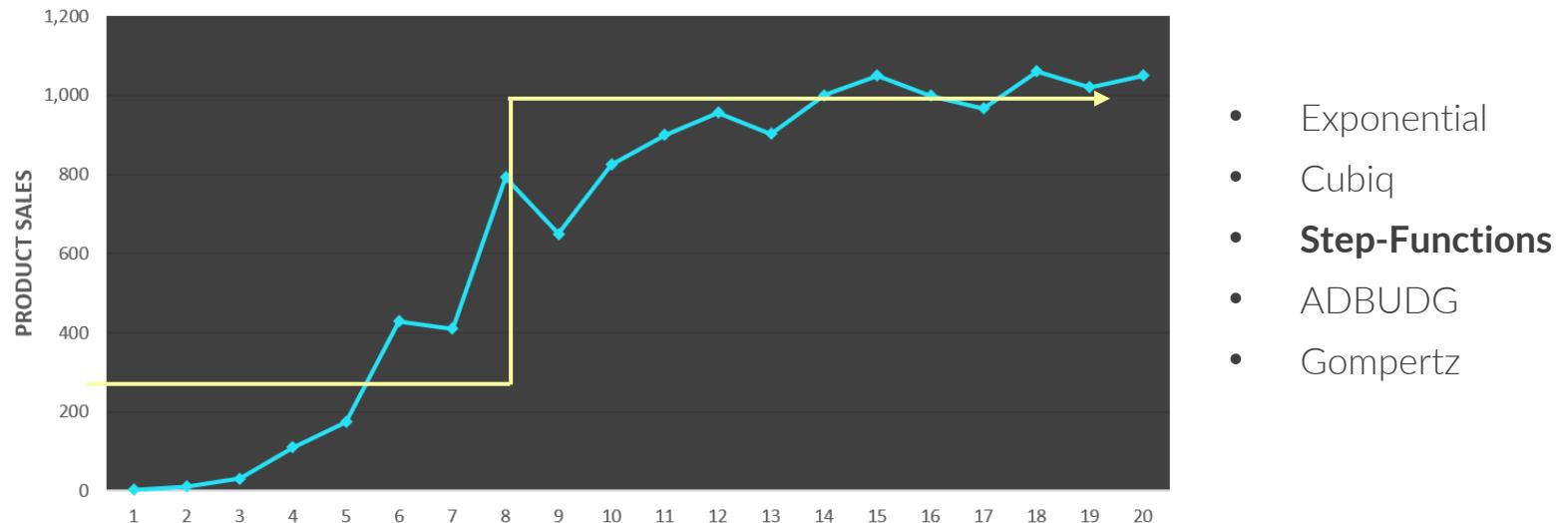
Smoothing

Non-Linear Trends

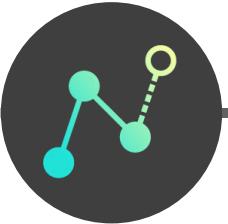
Intervention Analysis

Time-series data won't always follow a seasonal pattern or linear trend; it may follow a **non-linear trend**, or have no predictable trend at all

- There are many formulas designed to forecast common non-linear trends:



- Exponential
- Cubiq
- **Step-Functions**
- ADBUDG
- Gompertz



# NON-LINEAR TRENDS

Forecasting 101

Seasonality

Linear Trending

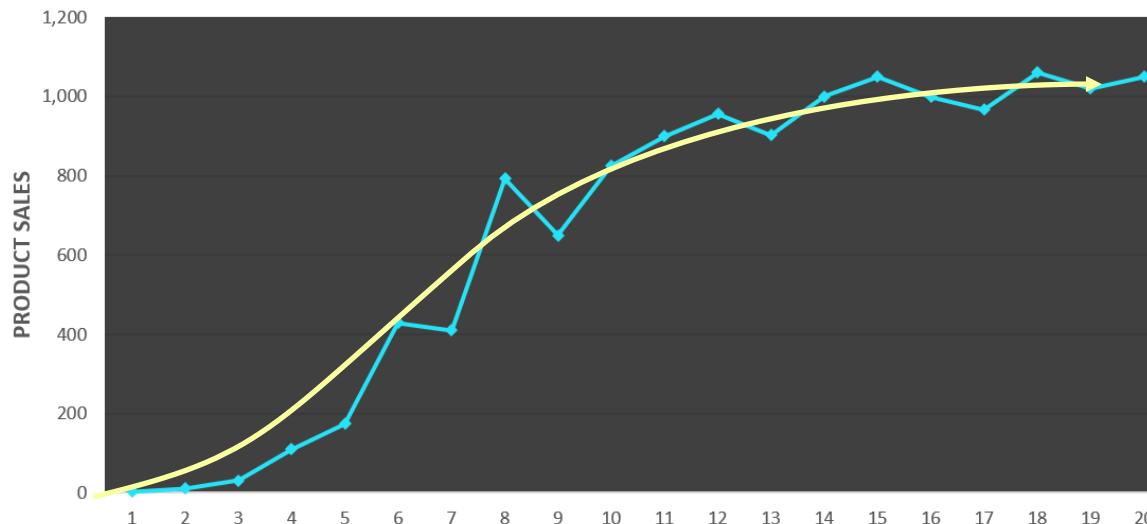
Smoothing

Non-Linear Trends

Intervention Analysis

Time-series data won't always follow a seasonal pattern or linear trend; it may follow a **non-linear trend**, or have no predictable trend at all

- There are many formulas designed to forecast common non-linear trends:



- Exponential
- Cubic
- Step-Functions
- **ADBUDG**
- **Gompertz**



**PRO TIP:** ADBUDG and Gompertz are more flexible versions of a logistic curve, and are commonly seen in BI use cases (*product launches, diminishing returns, etc.*)

# CASE STUDY: NON-LINEAR TREND



## THE SITUATION

The team at **Cat Slacks** just launched a new product poised to revolutionize the world of feline fashion: a lightweight, breathable jogging short designed for active cats who refuse to compromise on quality.



## THE ASSIGNMENT

You've been asked to **provide a weekly sales forecast to help the manufacturing and warehouse teams with capacity planning**. You only have 8 weeks of data to work with, but expect the launch to follow a logistic curve.



## THE OBJECTIVES

1. Collect sales data for the first 8 weeks since the launch
2. Apply a Gompertz curve to fit a logistic trend
3. Adjust parameters to compare various capacity limits and growth rates



# INTERVENTION ANALYSIS

Forecasting 101

Seasonality

Linear Trending

Smoothing

Non-Linear Trends

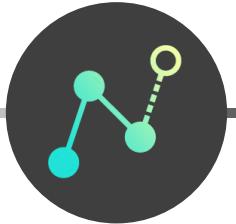
Intervention Analysis

**Intervention analysis** is a technique used to estimate the impact of a specific change (or “intervention”) on the dependent variable

- Simply put, intervention analysis is about predicting what **would have happened** if the change or intervention never took place
- By fitting a model to the “pre-intervention” data (up to the date of the change), you can compare predicted vs. actual values after that date to estimate the impact of the intervention

## Common examples:

- Measuring the impact of a new website or check-out page on conversion rates
- Quantifying the impact of a new HR program to reduce employee churn



# INTERVENTION ANALYSIS

Forecasting 101

Seasonality

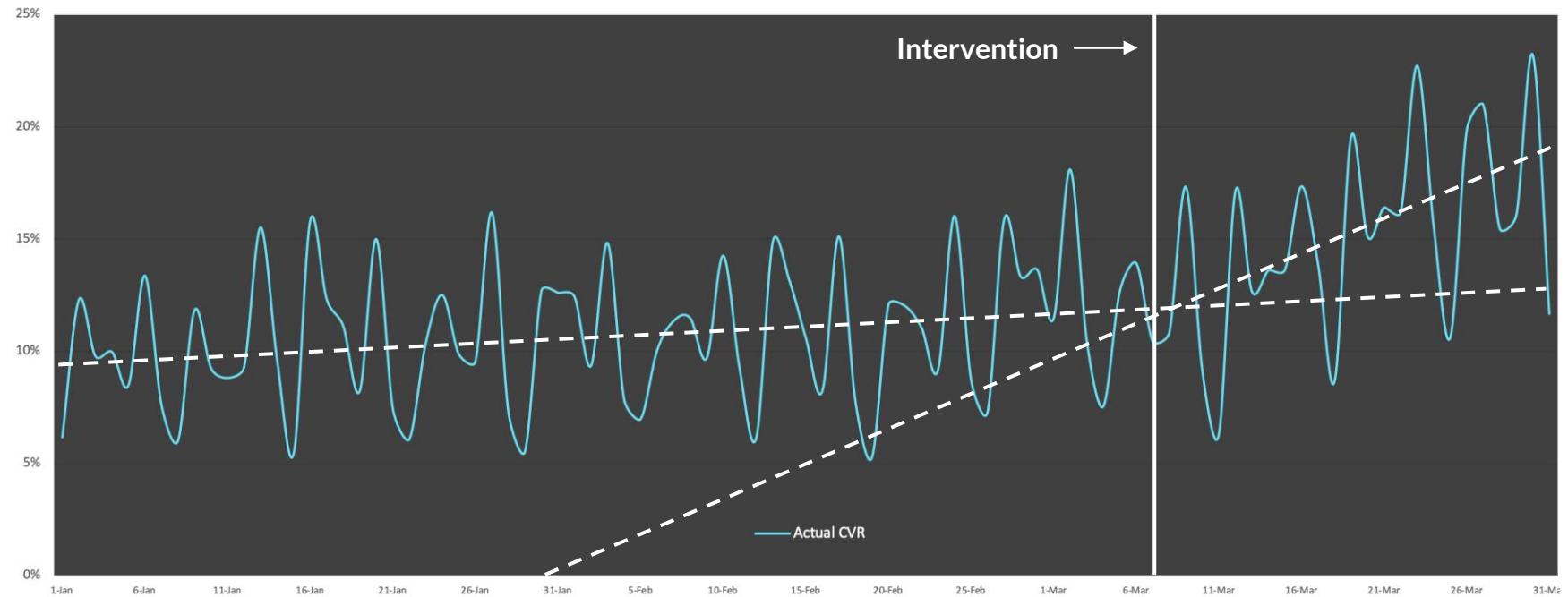
Linear Trending

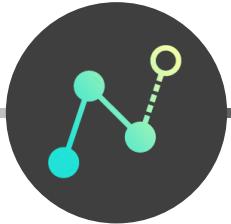
Smoothing

Non-Linear Trends

Intervention Analysis

**Intervention analysis** is a technique used to estimate the impact of a specific change (or “intervention”) on the dependent variable





# INTERVENTION ANALYSIS

Forecasting 101

Seasonality

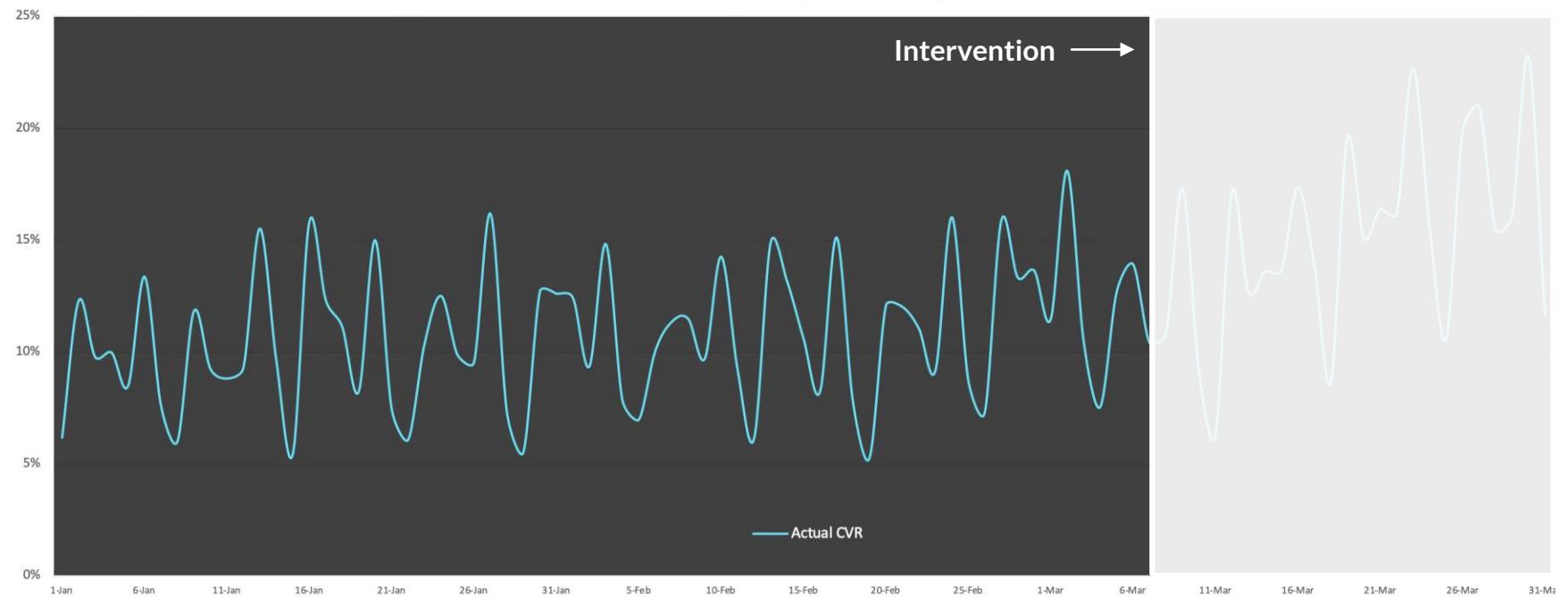
Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis

**STEP 1:** Fit a regression model to the data, using only observations from the **pre-intervention** period:





# INTERVENTION ANALYSIS

Forecasting 101

Seasonality

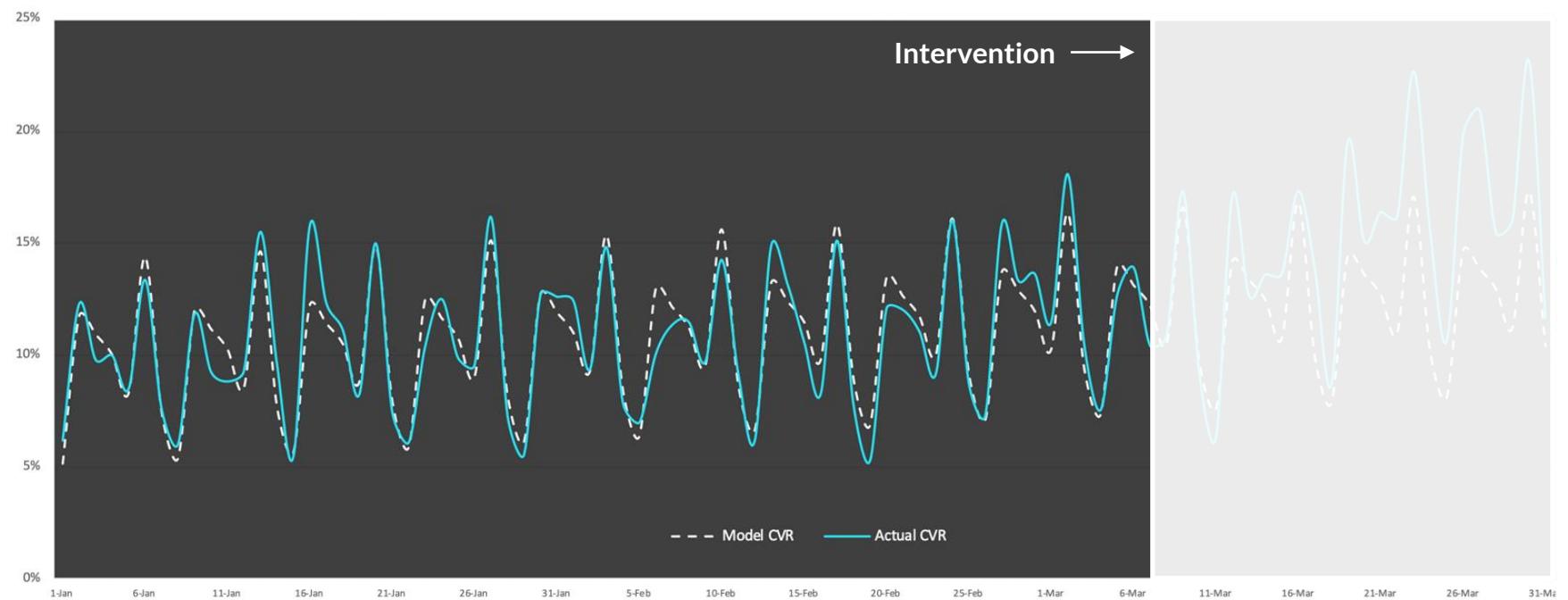
Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis

**STEP 1:** Fit a regression model to the data, using only observations from the **pre-intervention** period:





# INTERVENTION ANALYSIS

Forecasting 101

Seasonality

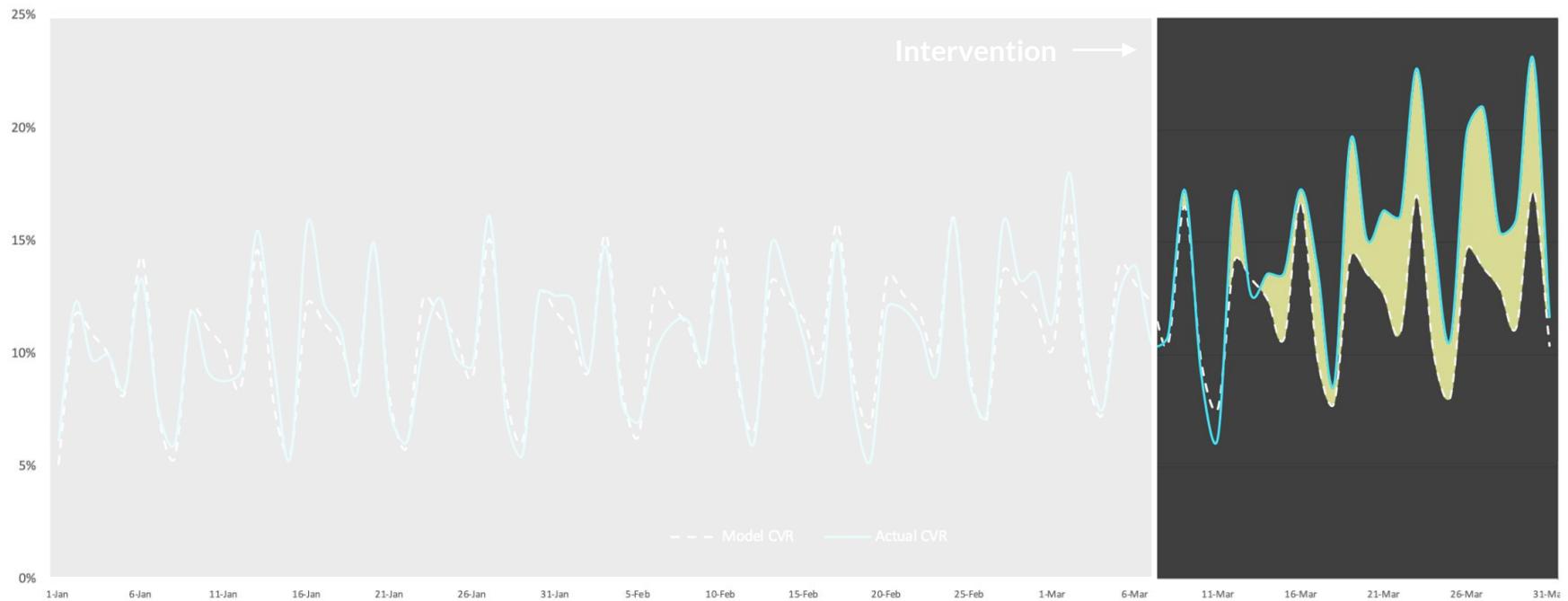
Linear Trending

Smoothing

Non-Linear Trends

Intervention Analysis

**STEP 2:** Compare the predicted and observed values in the **post-intervention** period, and sum the daily residuals to estimate the impact of the change:



# CASE STUDY: INTERVENTION ANALYSIS



## THE SITUATION

You are a Web Analyst for **Alpine Supplies**, an online retailer specializing in high-end camping and hiking gear.



## THE ASSIGNMENT

The company recently rolled out a new product landing page, and the CMO has asked you to help **quantify the impact on conversion rate (CVR) and sales**. You'll need to conduct an intervention analysis, and make sure to capture day of week seasonality and trending in your model.



## THE OBJECTIVES

1. Collect data to track sessions and CVR before and after the change
2. Fit a regression model to predict CVR using data from before the redesign
3. Use the model to forecast "baseline" CVR after the change, and calculate both incremental daily sales and the total cumulative impact

PART 4:

# UNSUPERVISED



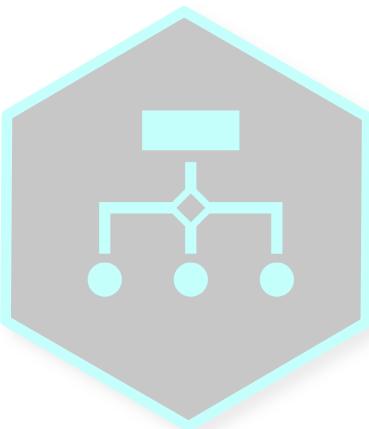
# ABOUT THIS SERIES

This is **Part 4** of a **4-Part series** designed to help you build a deep, foundational understanding of machine learning, including data QA & profiling, classification, forecasting and unsupervised learning



## PART 1

QA & Data Profiling



## PART 2

Classification



## PART 3

Regression & Forecasting



## PART 4

Unsupervised Learning

# COURSE OUTLINE

---

1

## Intro to Unsupervised ML

Review the ML landscape and key unsupervised learning concepts, including unsupervised vs. supervised, feature engineering, workflow, etc.

2

## Segmentation & Clustering

Learn the key building blocks of segmentation and clustering, and review two of the most-used algorithms: K-Means and Hierarchical Clustering

3

## Association Mining

Explore common techniques for association mining and when to use each of them, including A Priori and Markov models

4

## Outlier Detection

Learn how to detect cross-sectional and time-series outliers using techniques like Nearest Neighbor Distance and Time-Series models

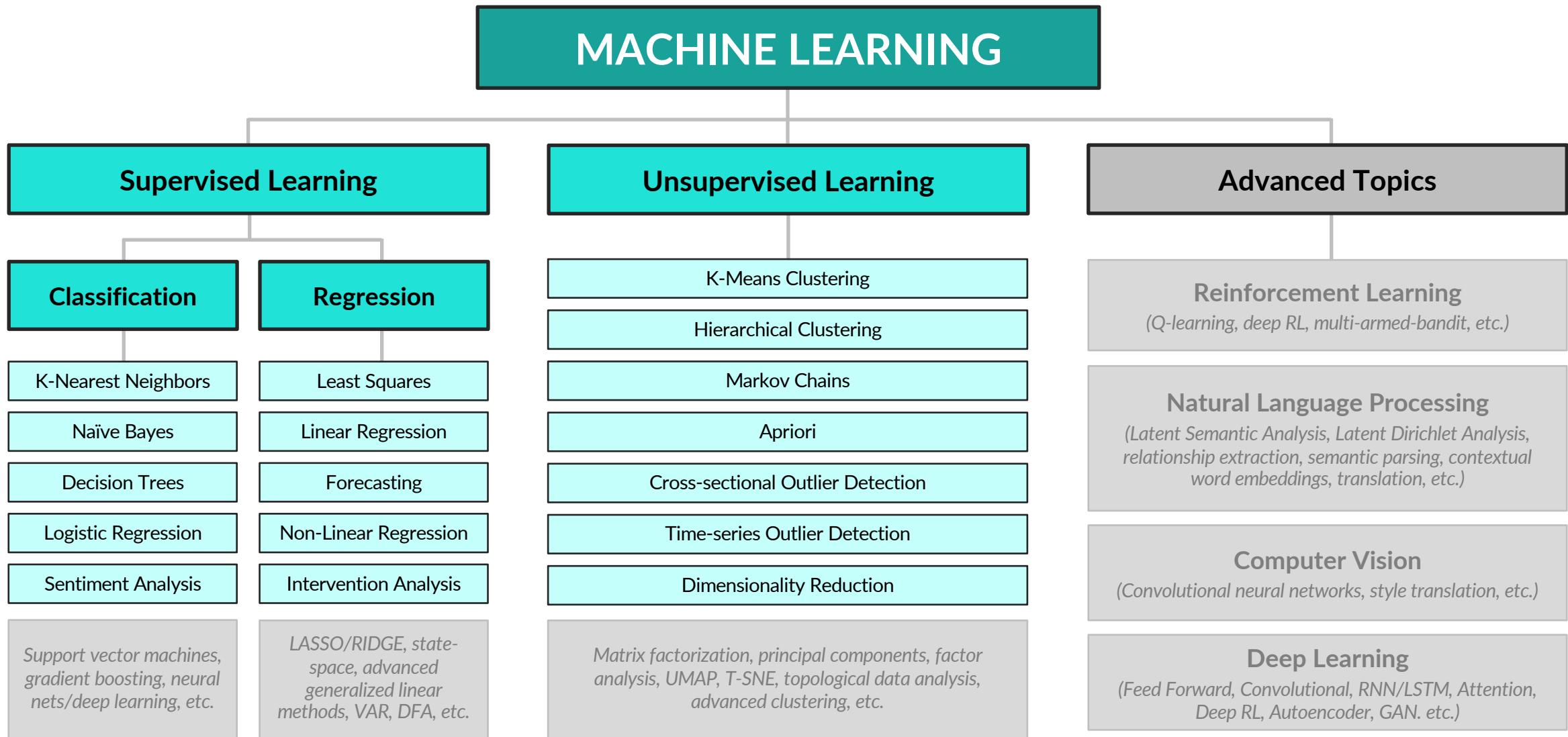
5

## Dimensionality Reduction

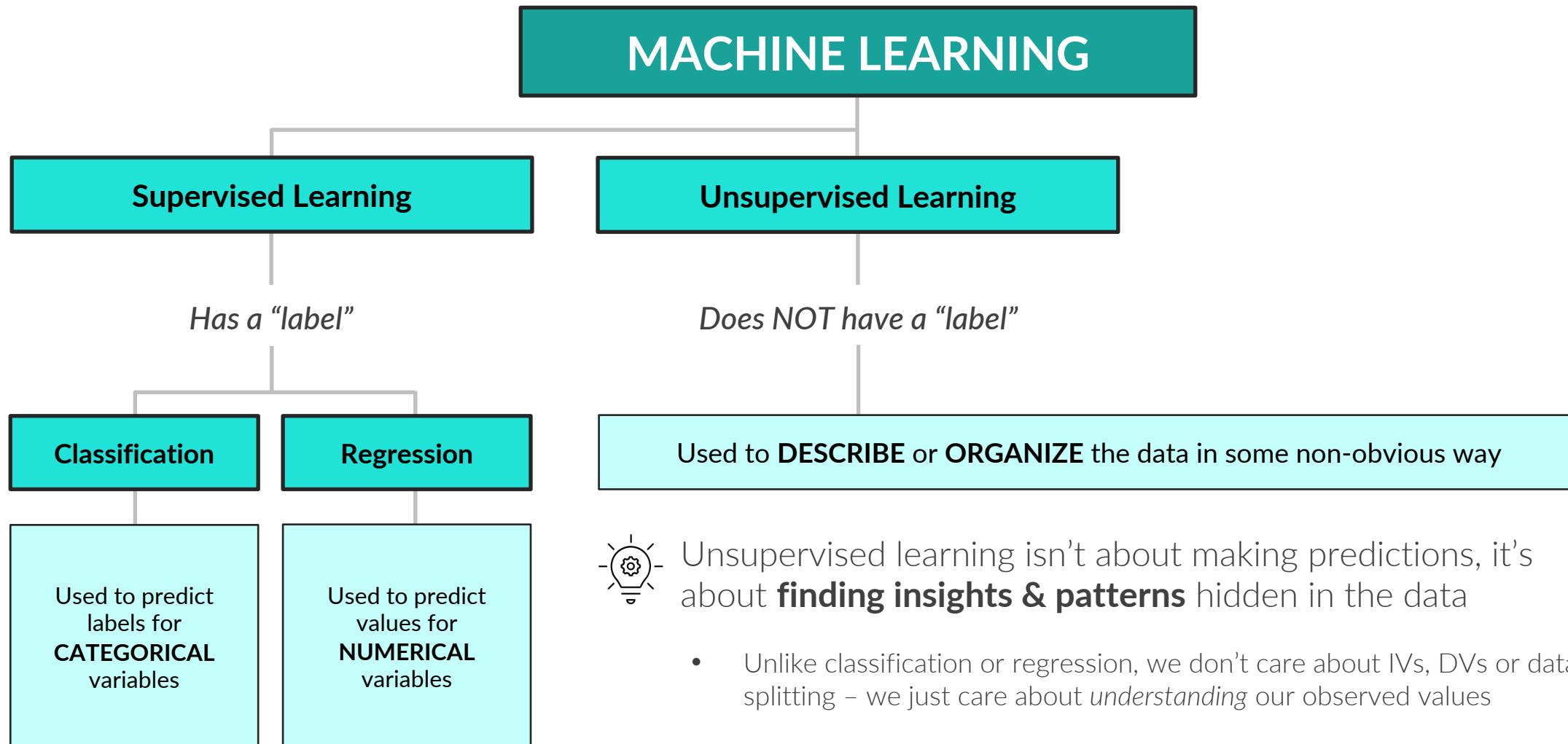
Understand the core foundations of dimensionality reduction and how it can be used in the context of BI and unsupervised learning

# INTRO TO UNSUPERVISED ML

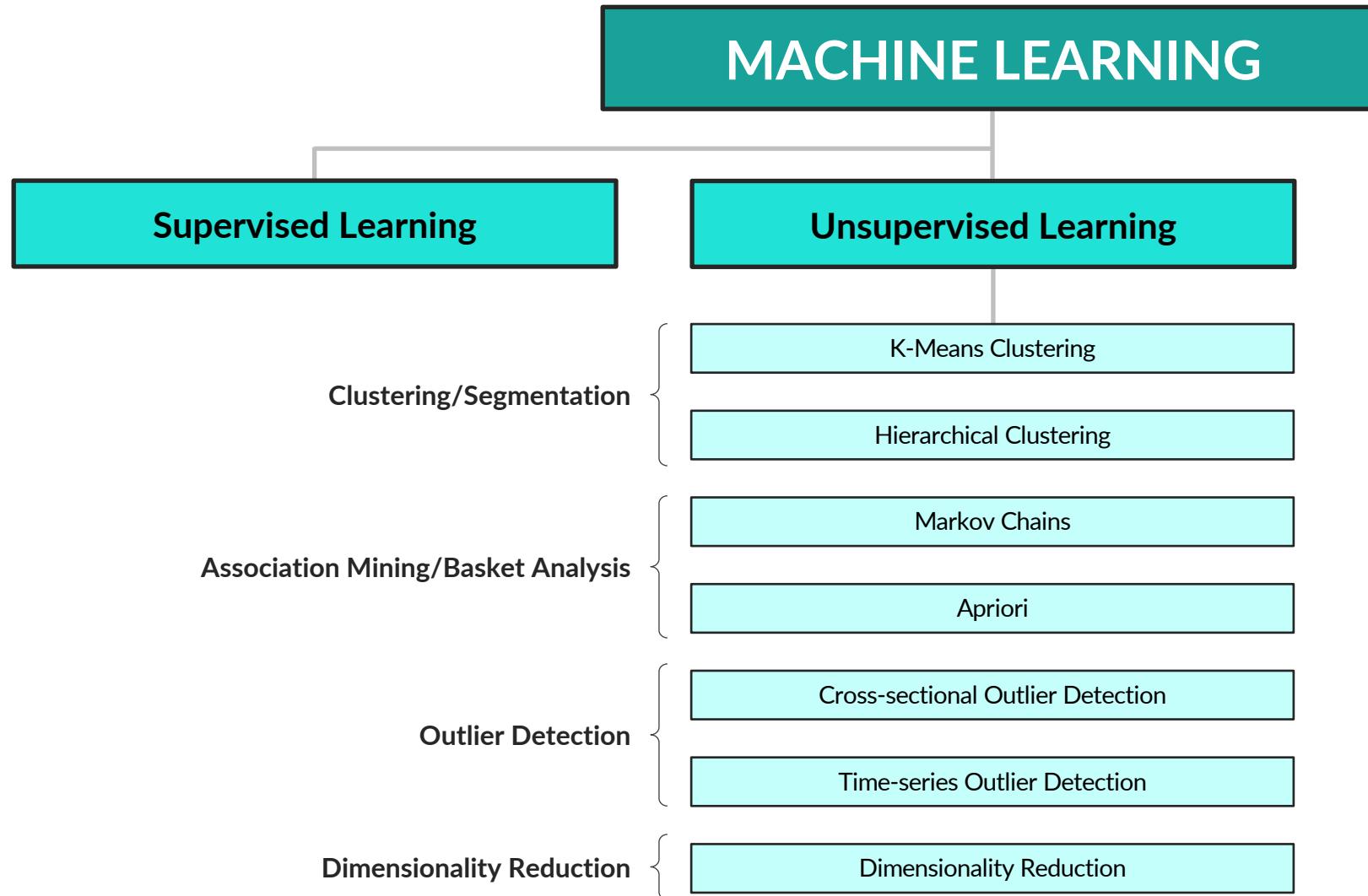
# MACHINE LEARNING LANDSCAPE



# MACHINE LEARNING LANDSCAPE



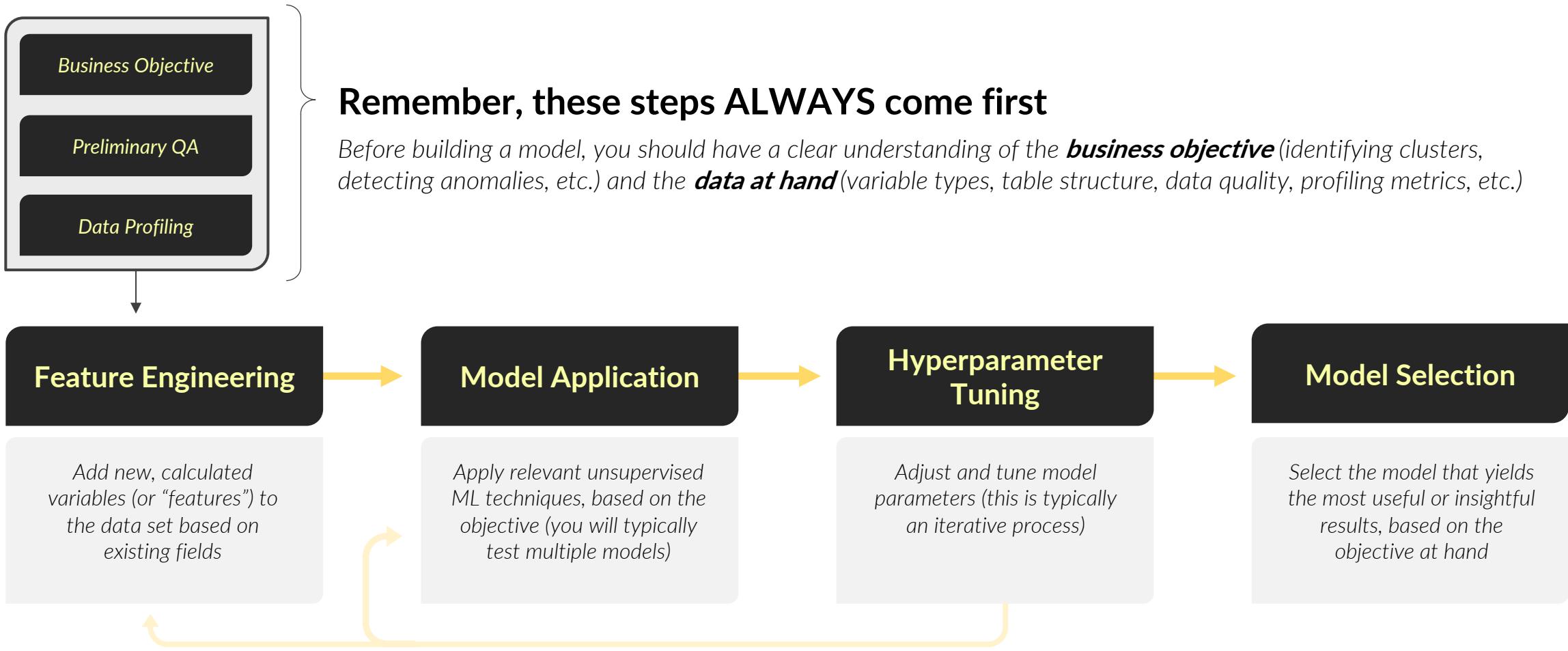
# MACHINE LEARNING LANDSCAPE



## UNSUPERVISED EXAMPLES:

- Are there specific groups or “clusters” of customers who behave in similar ways?
- When customers purchase a particular item, which other items tend to be purchased as well?
- Are there any anomalies in the data that we wouldn’t expect to see under normal conditions?
- Can we combine or reduce the number of dimensions in our data to better understand it?

# UNSUPERVISED LEARNING WORKFLOW



 There often are **no strict rules** to determine which model is best; it's about which one helps you **best answer the question at hand**

# RECAP: FEATURE ENGINEERING



**Feature engineering** is the process of enriching a data set by creating additional variables (or *dimensions*) based on existing fields

- When preparing data for segmentation/clustering, features should be **intuitive** and **actionable**; they should help you clearly describe each cluster and make practical business recommendations

Original features						Engineered features				
Month ID	Social Posts	Competitive Activity	Marketing Spend	Promotion Count	Revenue	Competitive High	Competitive Medium	Competitive Low	Promotion >10 & Social > 25	Log Spend
1	30	High	\$130,000	12	\$1,300,050	1	0	0	1	11.7
2	15	Low	\$600,000	5	\$11,233,310	0	0	1	0	13.3
3	8	Medium	\$15,000	10	\$1,112,050	0	1	0	0	9.6
4	22	Medium	\$705,000	11	\$1,582,077	0	1	0	1	13.5
5	41	High	\$3,000	3	\$1,889,053	1	0	0	0	8.0

# KEY TAKEAWAYS

---



Unsupervised learning is about **finding patterns**, not making predictions

- *Unlike classification or regression, it's not about labels, dependent/independent variables or splitting test and training data; it's about better understanding the observed values in a data set*



For many unsupervised techniques, there's no **“right” or “wrong”** solution

- *Unsupervised learning is often less about optimizing specific metrics or parameters, and more about using Machine Learning to clearly answer the business question at hand*



Always focus on the desired **business outcome**

- *Make sure that you can clearly interpret your model output and translate it into intuitive, practical insights and recommendations (otherwise what's the point?)*

# CLUSTERING & SEGMENTATION

# CLUSTERING & SEGMENTATION



In this section we'll introduce the fundamentals of **clustering** & **segmentation** and compare two common unsupervised models: K-means and hierarchical clustering

## TOPICS WE'LL COVER:

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways

## GOALS FOR THIS SECTION:

- Understand how segmentation and clustering models fundamentally work
- Explore common models including K-Means and hierarchical clustering
- Compare and contrast common clustering models



Remember, there's no "right" answer or single optimization metric when it comes to clustering and segmentation; the best outputs are the ones which help you answer the question at hand and make practical, data-driven business decisions



# CLUSTERING BASICS

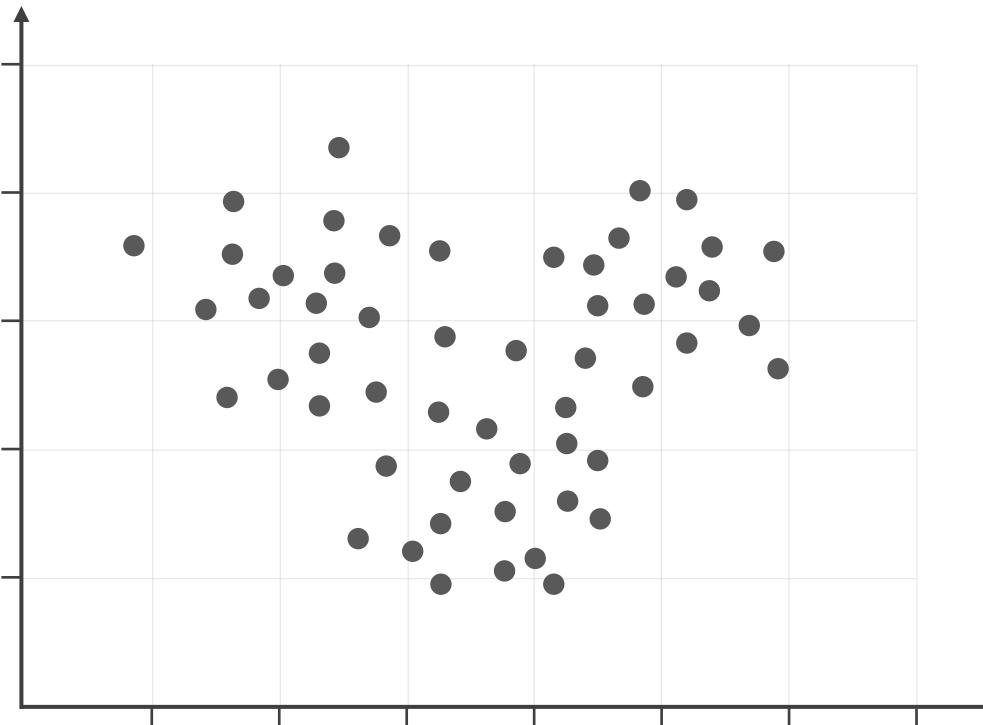
Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways

**Clustering** allows you to find concentrations or groups of observations which are similar to one another but distinct from other groups





# CLUSTERING BASICS

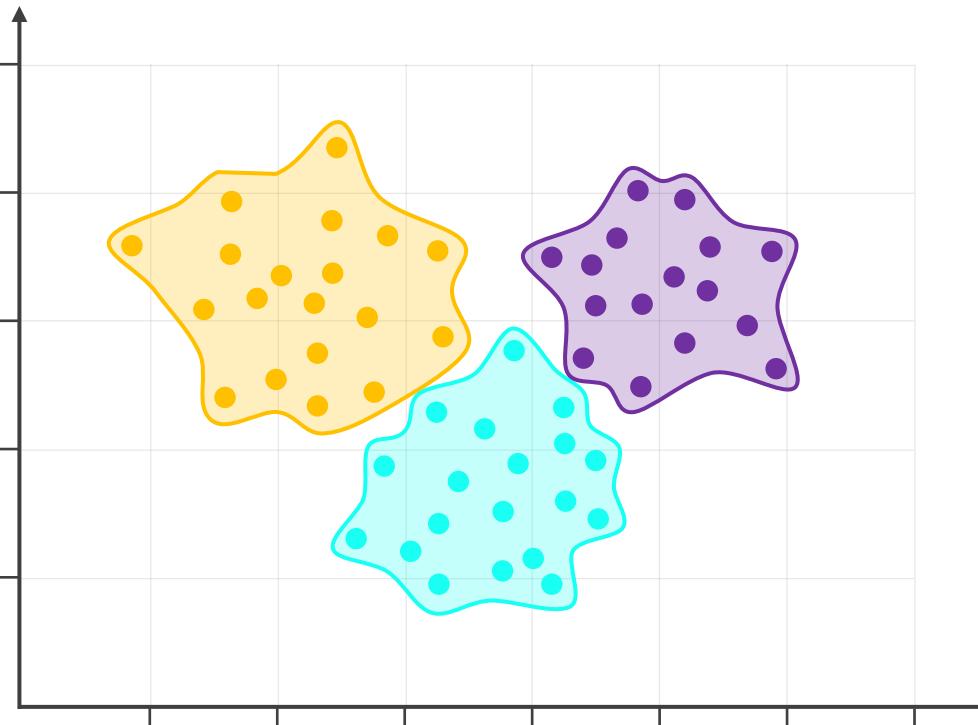
Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways

**Clustering** allows you to find concentrations or groups of observations which are similar to one another but distinct from other groups





# CLUSTERING BASICS

Clustering Basics

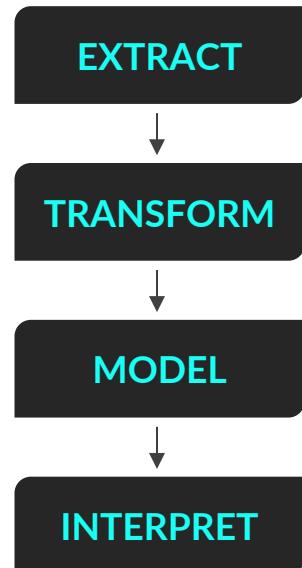
K-Means

Hierarchical Clustering

Key Takeaways

**Clustering** allows you to find concentrations or groups of observations which are similar to one another but distinct from other groups

Multiple models can be used for clustering, but the general workflow consists of the following steps:



**Collect the raw data** you need for analysis (i.e. transactions, customer records, survey results, website pathing, etc.)

Use **feature engineering** to create clear, intuitive dimensions which can help you interpret each cluster and make actionable recommendations

Apply one or more **clustering models** and determine an appropriate number of clusters based on the model output and business context

Use **multivariate analysis** to understand the characteristics of each cluster (profiling metrics, distributions, etc.). **The model won't do this for you!**



# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways

**K-Means** is a common algorithm which assigns each observation in a data set to a specific cluster, where **K** represents the number of clusters

In its simplest form (2 dimensions), here's how it works:

1. Select K arbitrary locations in a scatterplot as cluster centers (or **centroids**), and assign each observation to a cluster based on the closest centroid
2. Recalculate and relocate each centroid to the mean of the observations assigned to it, then reassign each observation to its new closest centroid
3. Repeat the process until observations no longer change clusters

## Example use cases:

- Identifying customer segments for targeted marketing campaigns
- Clustering store locations based on factors like sales, ratings, size, etc.



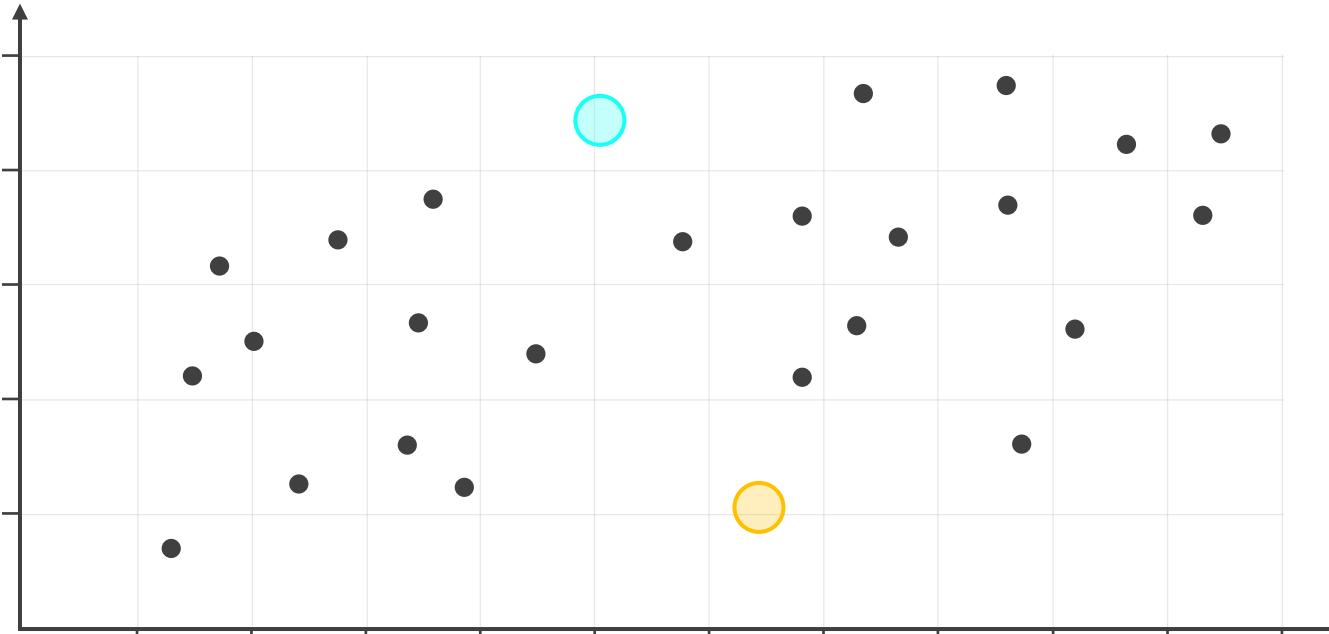
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 1:** Determine what you think might be an appropriate number of clusters (*in this case 2*), and select arbitrary locations as initial centroids



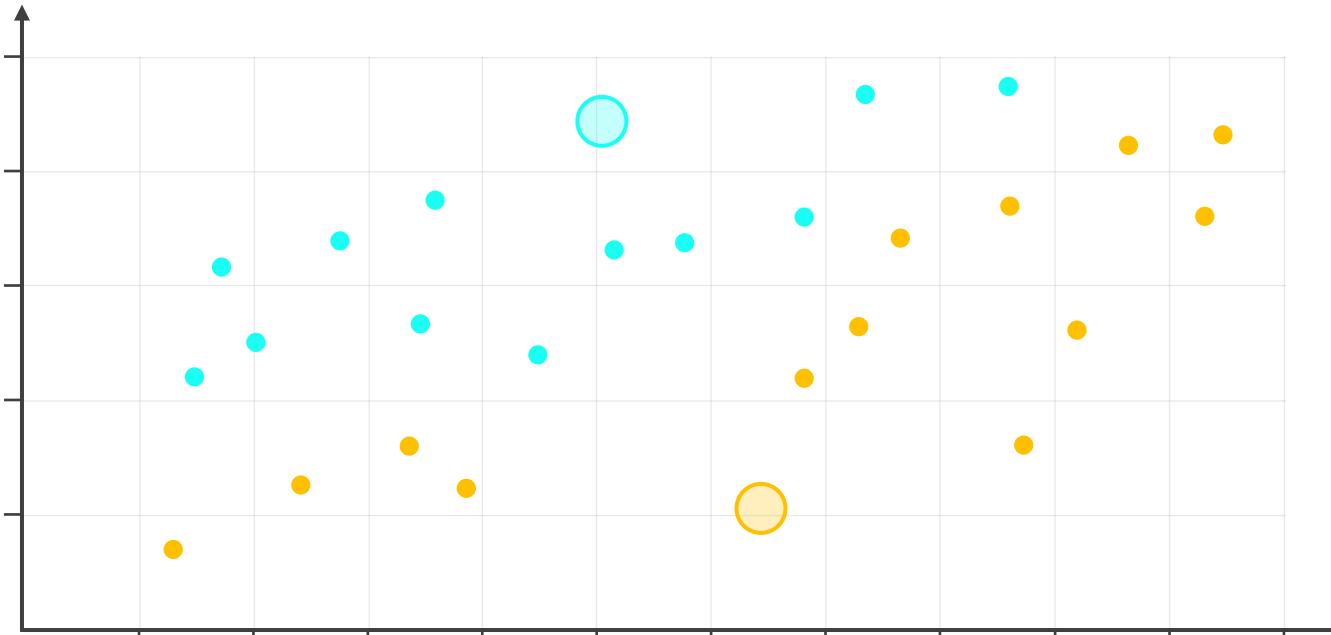
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 2:** Assign each observation to a cluster, based on the closest centroid



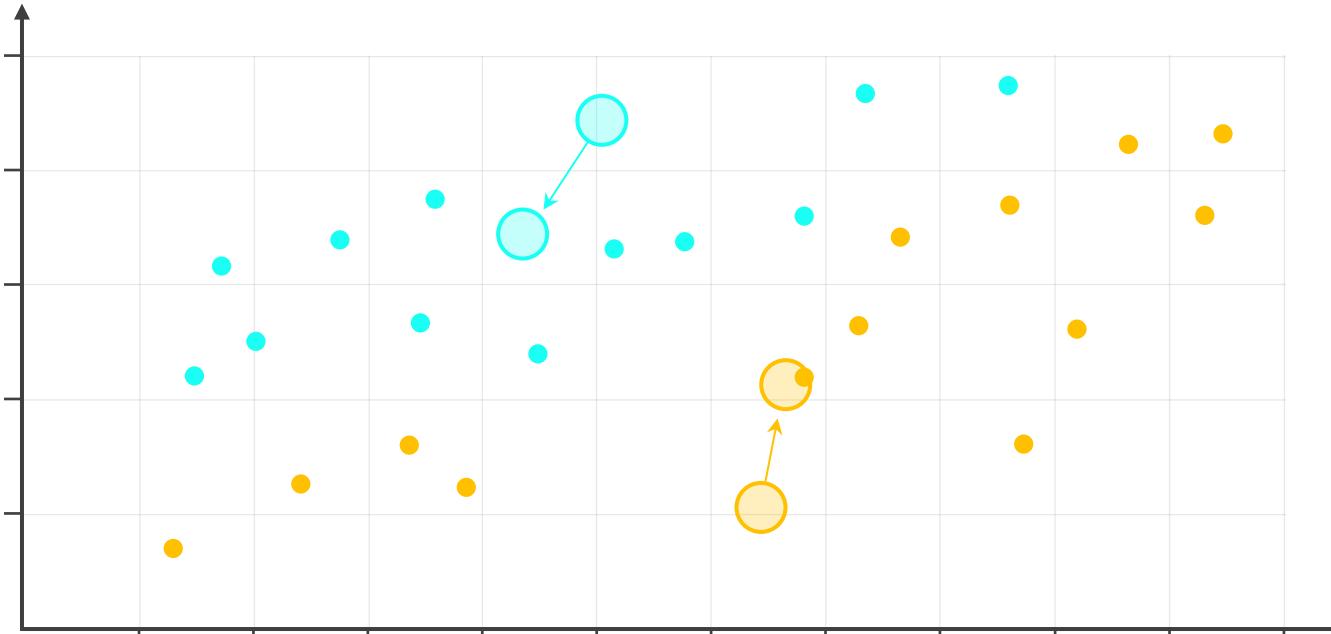
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 3:** Relocate each centroid to the mean of its assigned observations, and reassign each observation to the new closest centroid



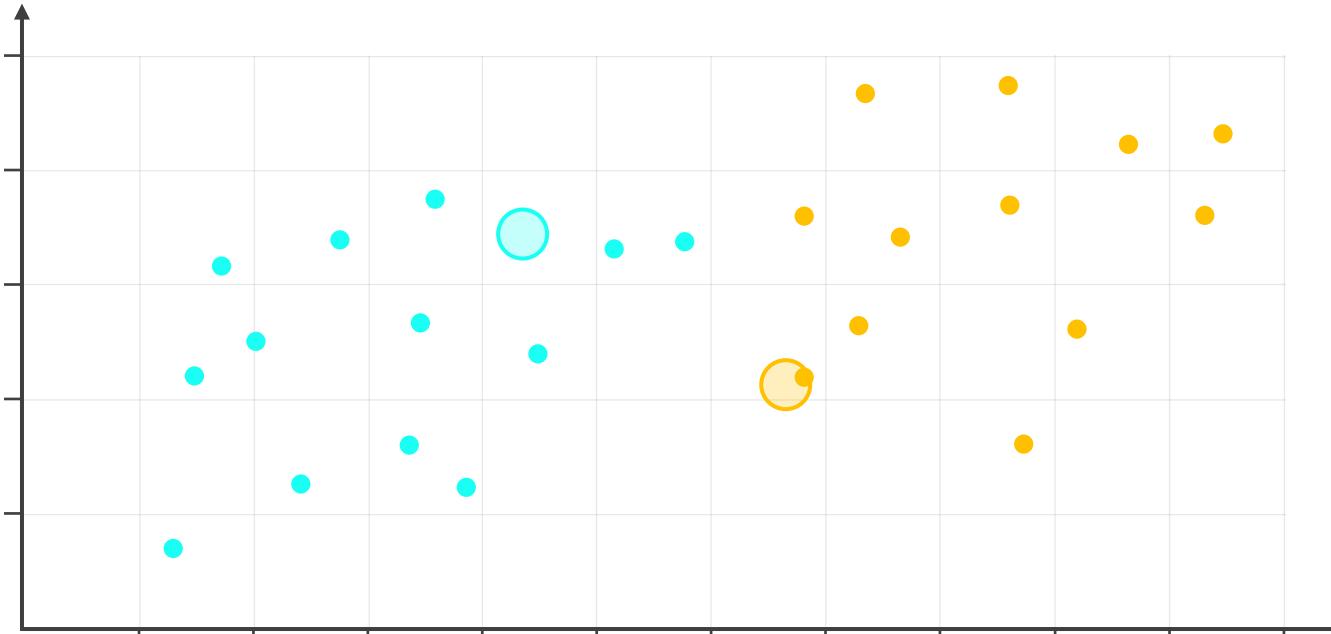
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 3:** Relocate each centroid to the mean of its assigned observations, and reassign each observation to the new closest centroid



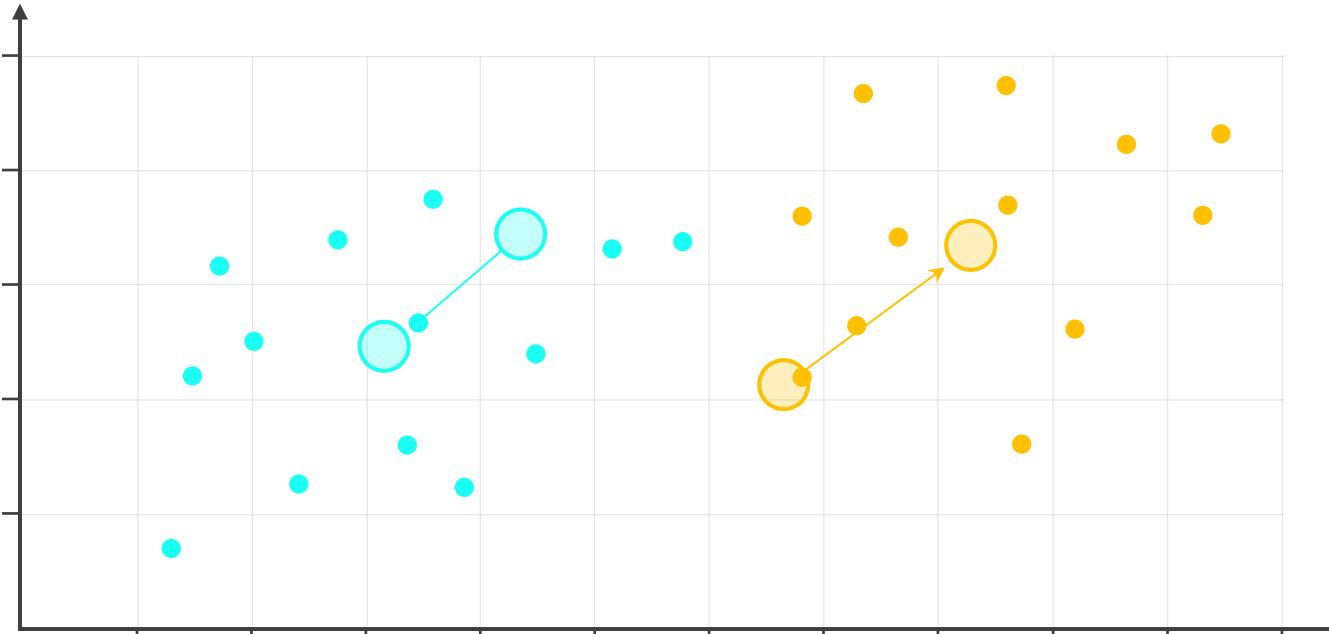
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 4:** Continue to relocate each centroid to the mean of its assigned observations, until the clusters no longer change



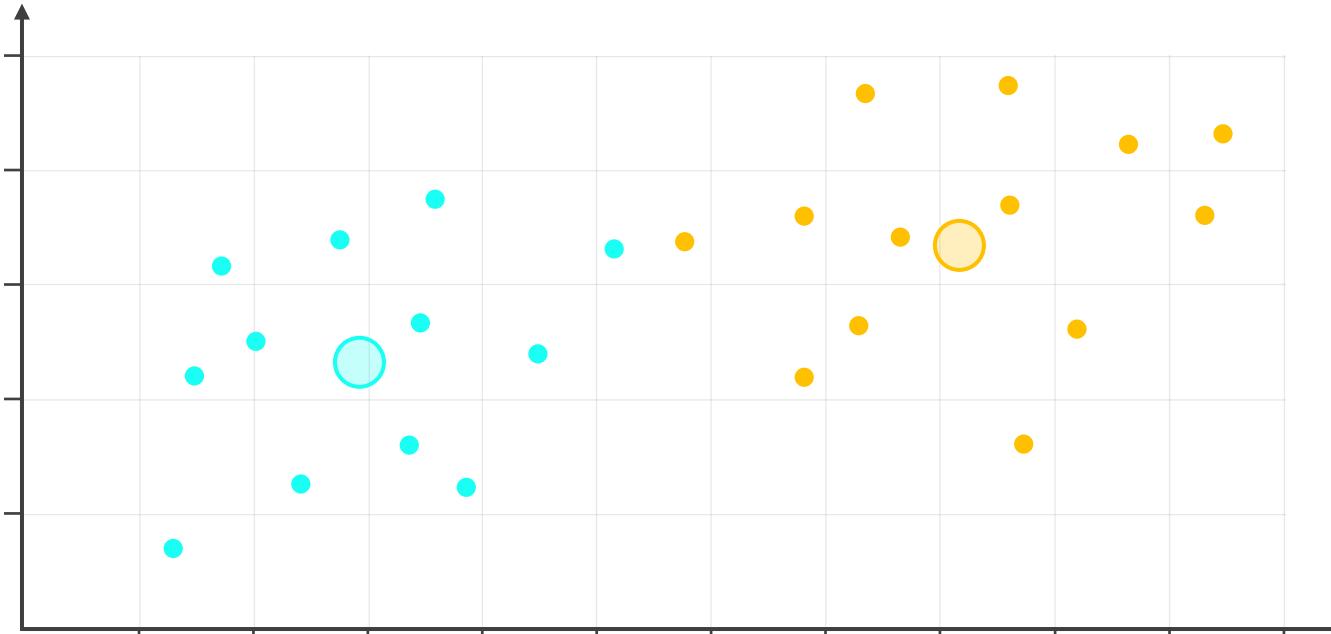
# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 4:** Continue to relocate each centroid to the mean of its assigned observations, until the clusters no longer change



# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

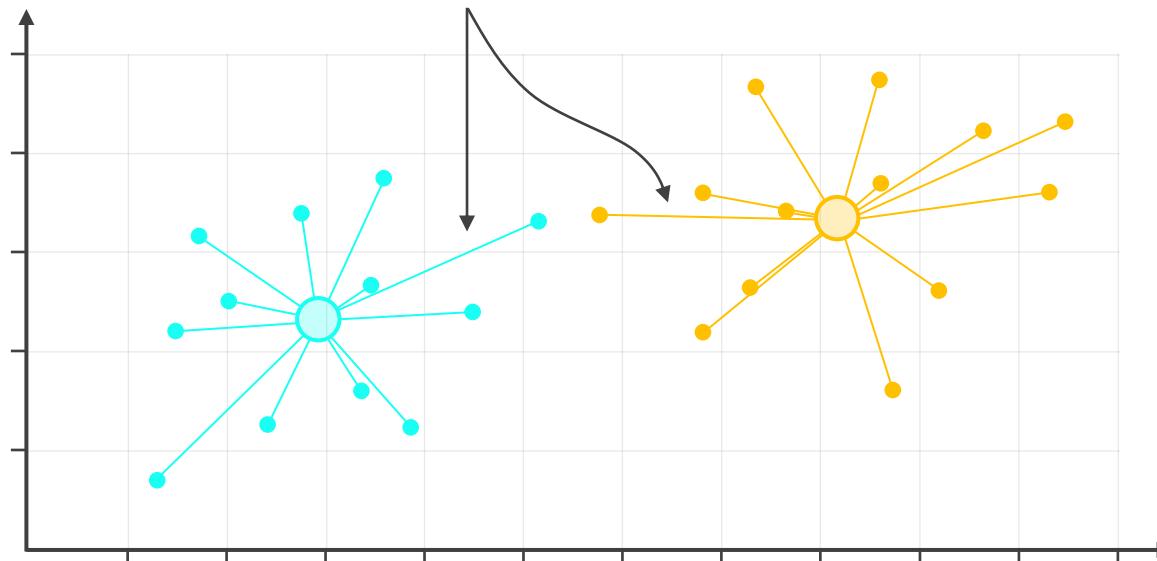
Key Takeaways



How do we know what's the “right” number of clusters (K)?

- While there is no “right” or “wrong” number of clusters, you can use the **within-cluster sum of squares (WSS)** to help inform your decision

*Square these distances and sum them to calculate WSS for two clusters (K=2)*





# K-MEANS

Clustering Basics

K-Means

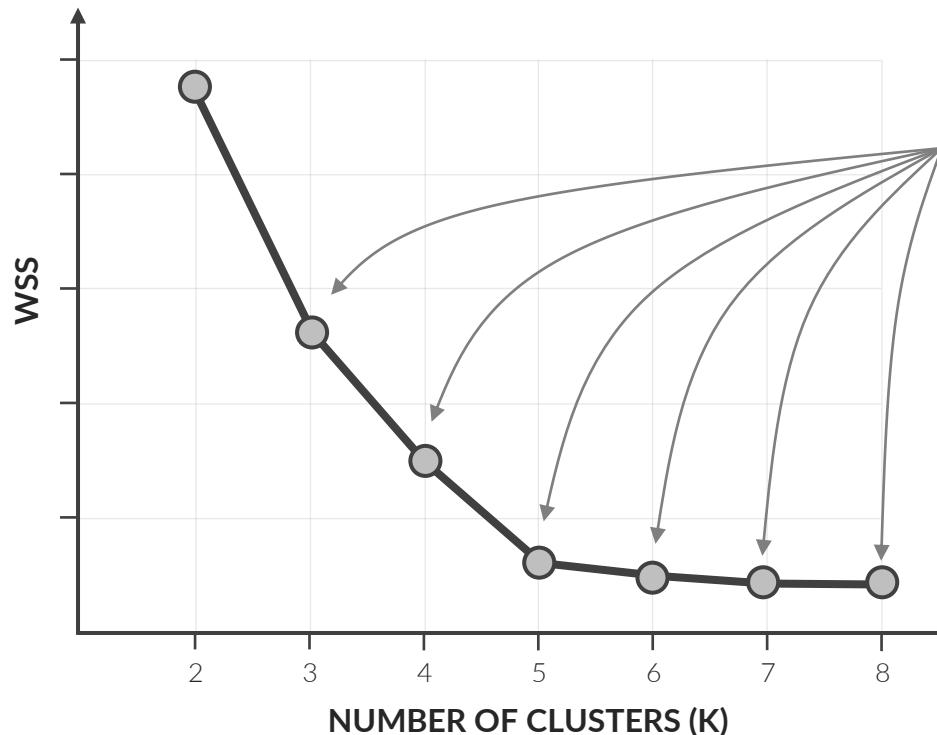
Hierarchical Clustering

Key Takeaways



How do we know what's the “right” number of clusters (K)?

- While there is no “right” or “wrong” number of clusters, you can use the **within-cluster sum of squares (WSS)** to help inform your decision



Rerun the model with additional clusters, and plot the WSS for each value of K



# K-MEANS

Clustering Basics

K-Means

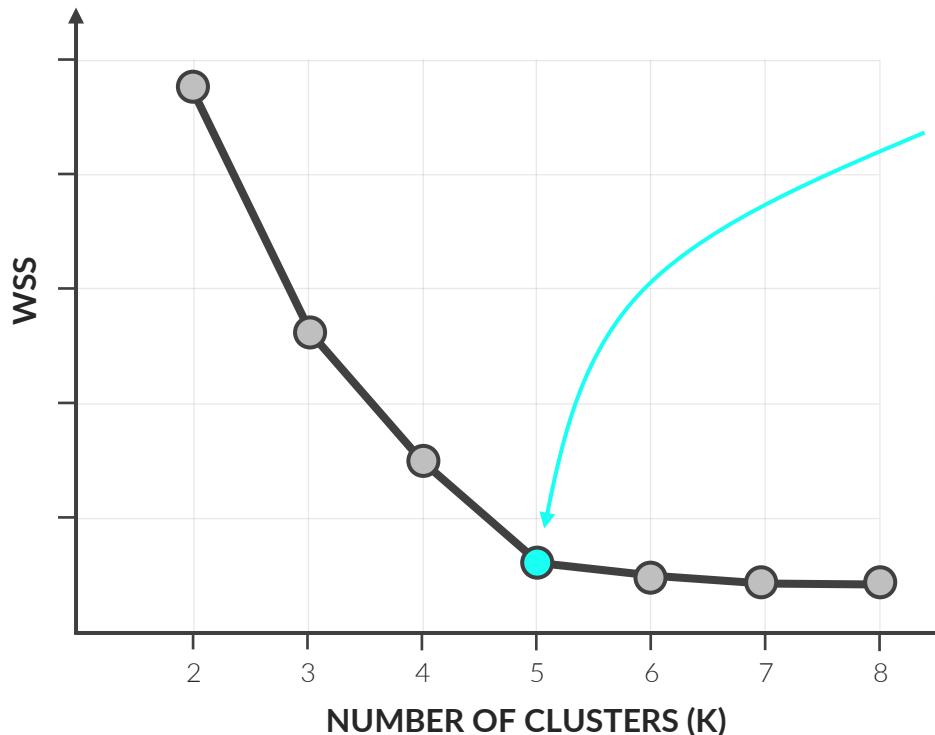
Hierarchical Clustering

Key Takeaways



How do we know what's the “right” number of clusters (K)?

- While there is no “right” or “wrong” number of clusters, you can use the **within-cluster sum of squares (WSS)** to help inform your decision



Look for an “elbow” or inflection point, where adding another cluster has a relatively small impact on WSS (in this case where **K=5**)



**PRO TIP:** Think of this as a guideline, not a strict rule



# K-MEANS

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



Can we create clusters based on **more than 2 dimensions?**

- Yes! K-Means and WSS plots can scale to *any* number of dimensions
- At higher dimensions, profiling metrics are critical for understanding what each cluster represents, since you can't compare them visually



Does the **shape** of the clusters matter?

- Yes, K-Means works best when the clusters are mostly circular in shape; but other tools like Hierarchical Clustering (*up next!*) can address this



Does the **initial centroid location** make a difference?

- It can, so a common best practice is to re-run the model K times to ensure that you are minimizing WSS for the selected number of clusters

# CASE STUDY: K-MEANS

---



## THE SITUATION

You've just been hired as a Data Science Intern for **Pet Palz**, an ecommerce start-up selling custom 3-D printed models of household pets.



## THE ASSIGNMENT

You've been asked to analyze customer-level data to help **identify meaningful audience segments and refine the company's marketing strategy**.

As a first step, you'll be exploring the relationship between how often customers engage with promotional emails and how much they actually spend, using a K-means model for clustering.



## THE OBJECTIVES

1. Collect sample data containing email engagement and spend by customer
2. Use K-means to identify clusters
3. Interpret each cluster and suggest next steps based on your analysis



# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways

**Hierarchical clustering** is an unsupervised learning technique that creates clusters by grouping similar data points together\*

In its simplest form (2 dimensions), here's how it works:

1. Create a scatterplot, find the 2 closest points, and group them into a cluster
2. Then find the next two closest points or clusters, and group them to a cluster
3. Repeat the process of combining the closest pairs of points or clusters until you eventually end up with one single cluster

This process is visualized using a tree diagram called a **dendrogram**, which shows the hierarchical relationship between clusters

\*This is known as agglomerative or “bottom-up” clustering (vs. divisive or “top-down” clustering, which is much less common)



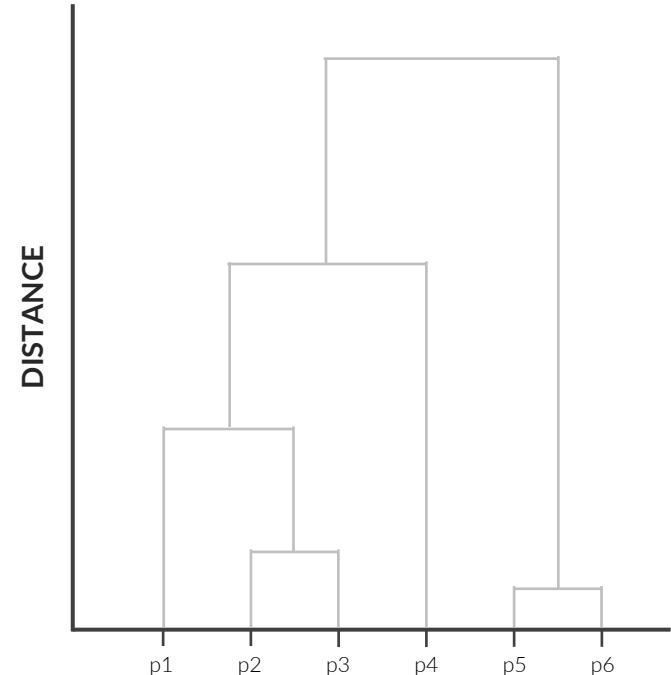
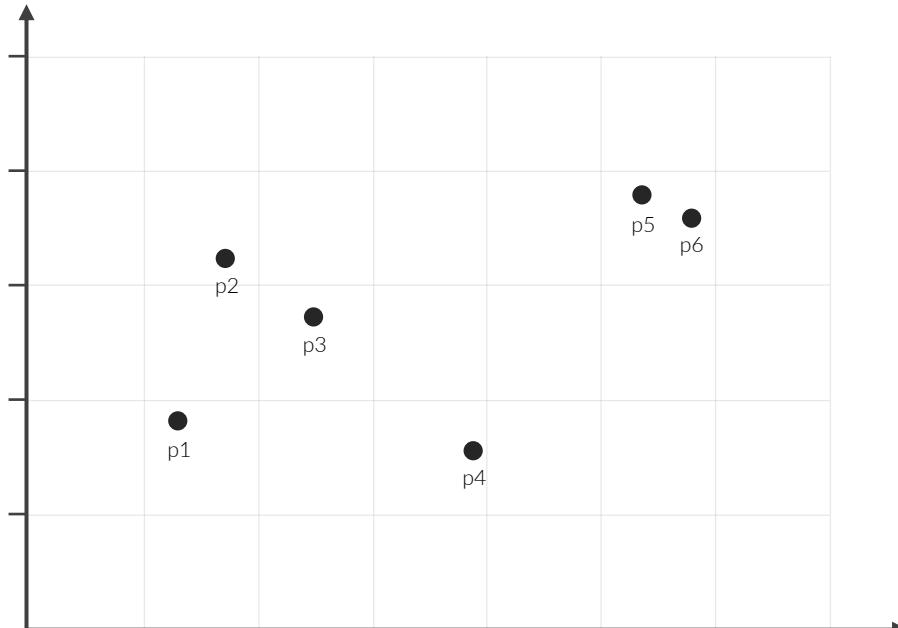
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 1:** Find the two closest points, and group them into a cluster



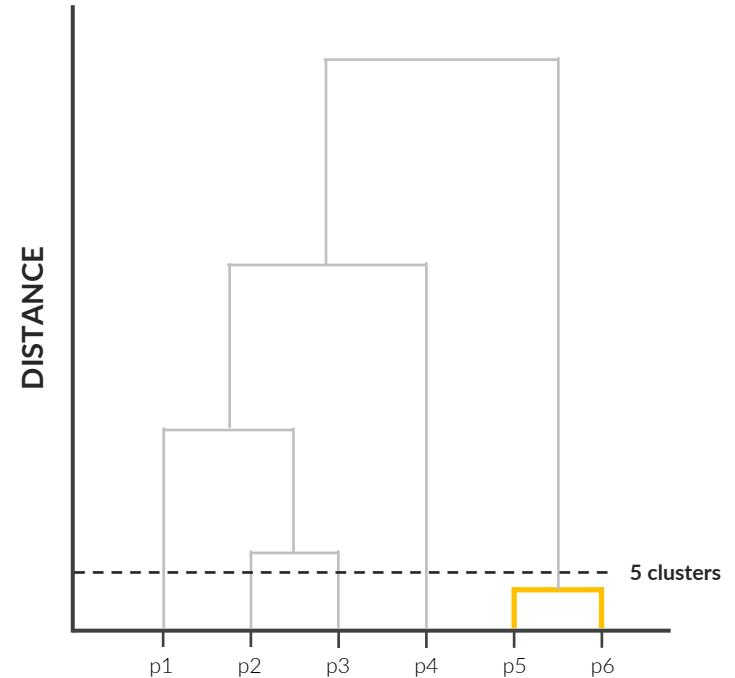
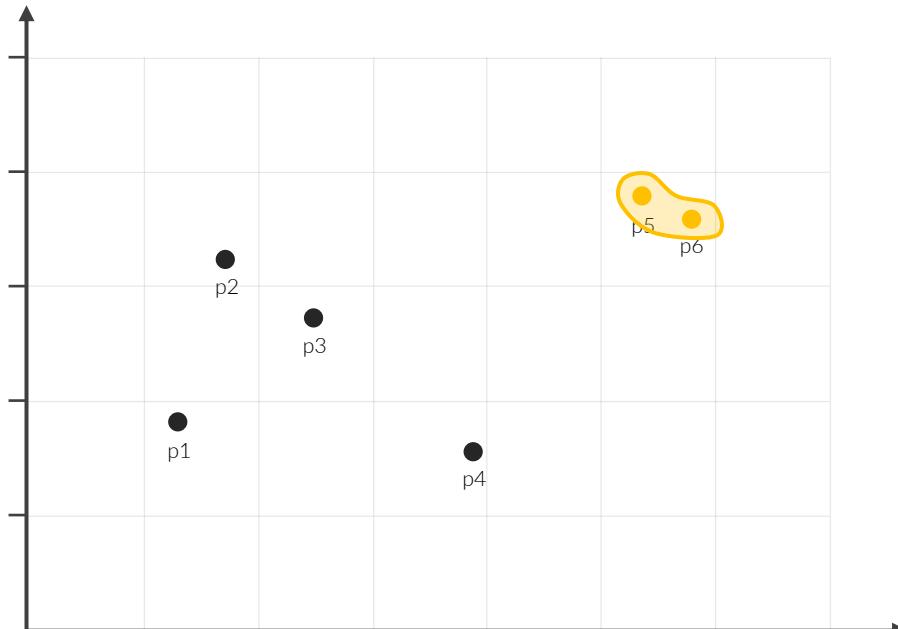
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 1:** Find the two closest points, and group them into a cluster



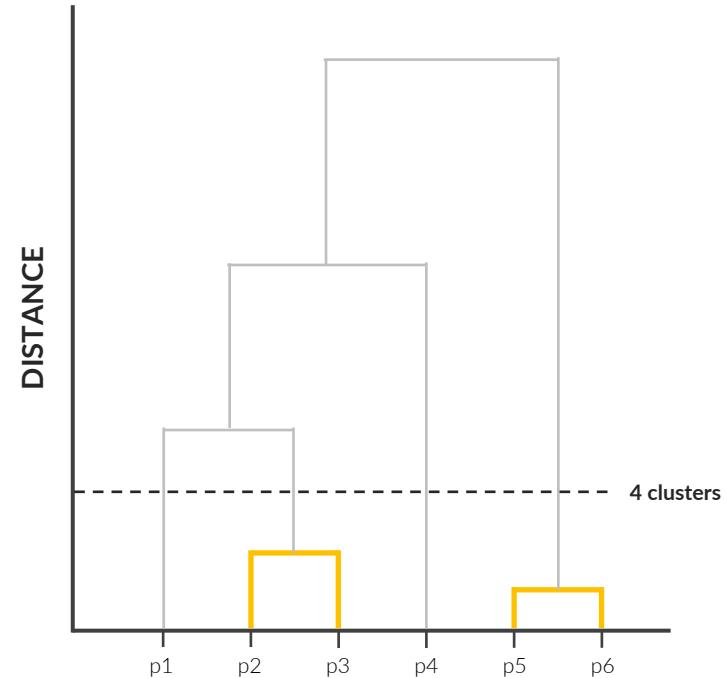
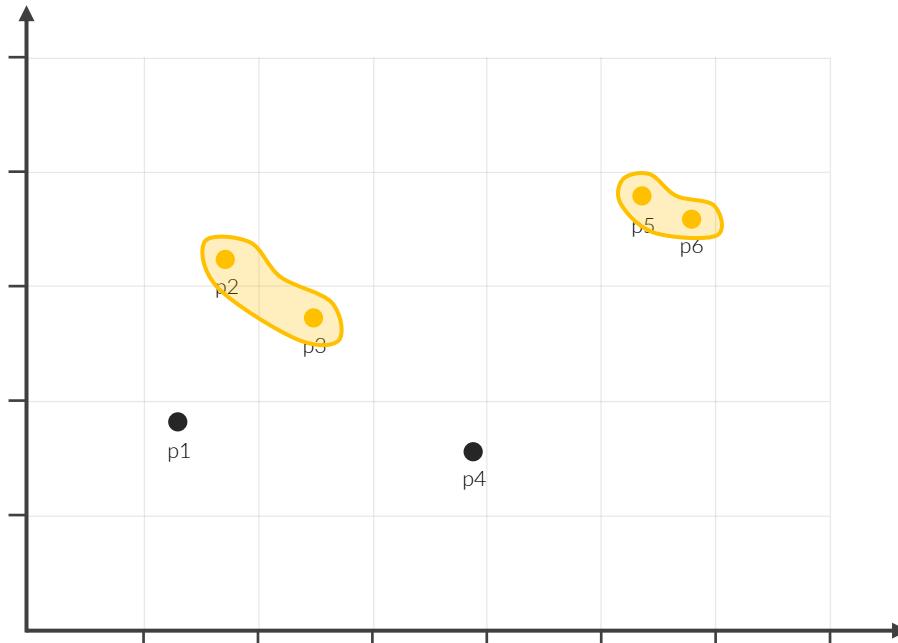
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 2:** Find the *next* two closest points/clusters, and group them together



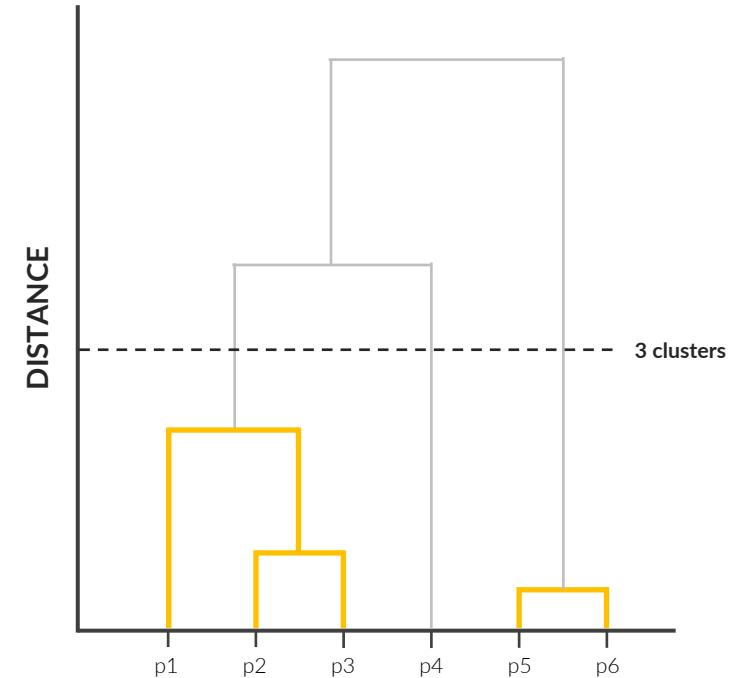
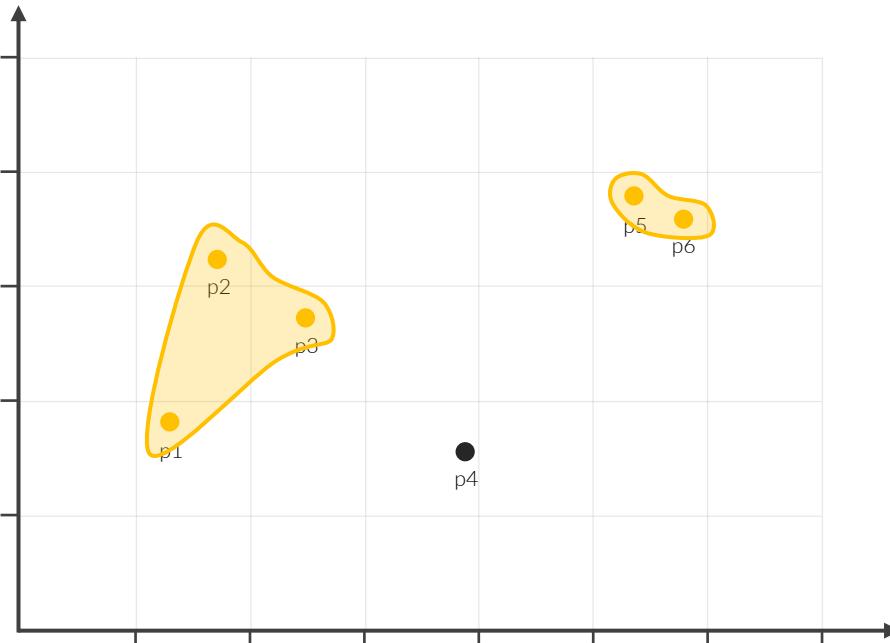
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 3:** Repeat the process until all points are part of the same cluster



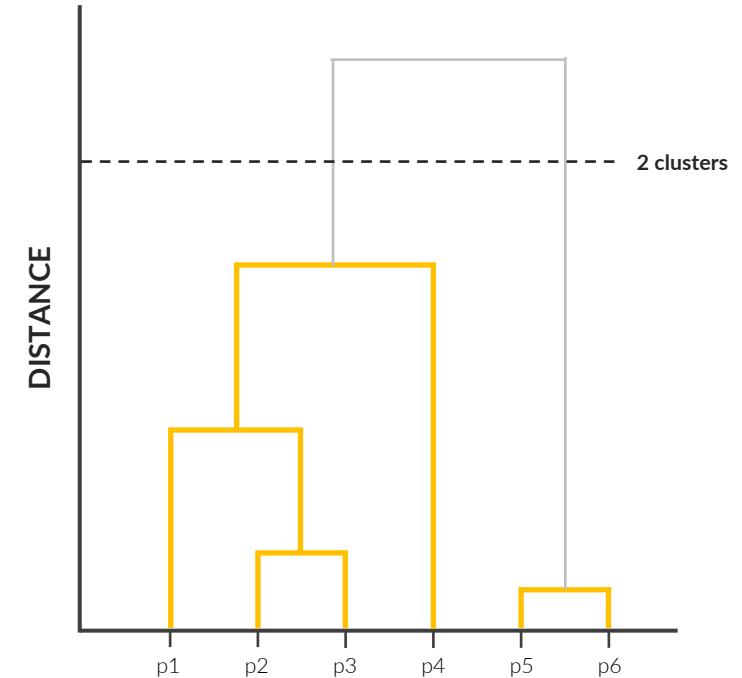
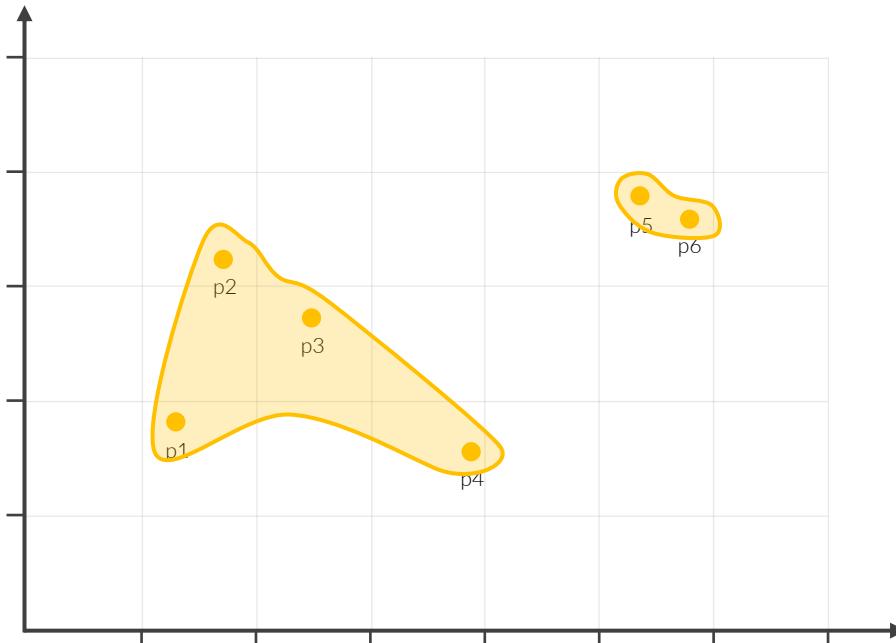
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 3:** Repeat the process until all points are part of the same cluster



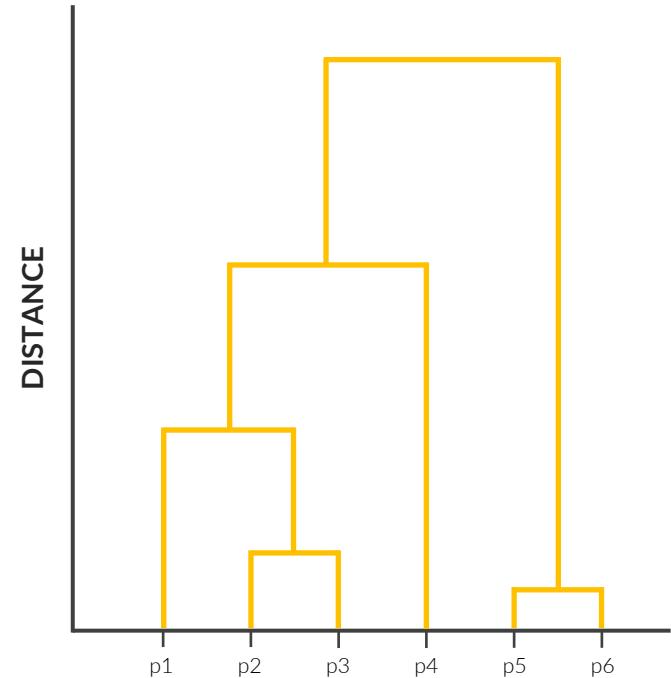
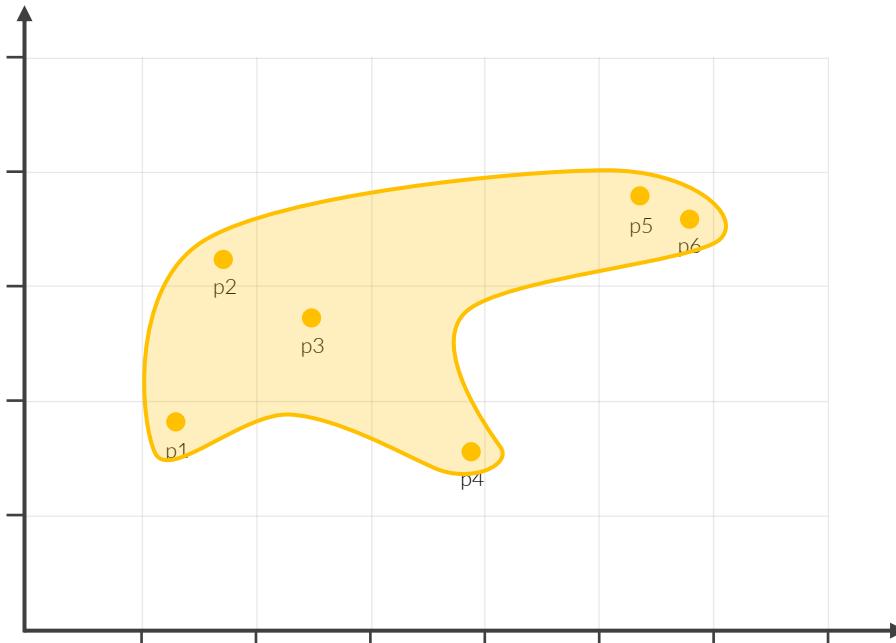
# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



**STEP 3:** Repeat the process until all points are part of the same cluster



# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

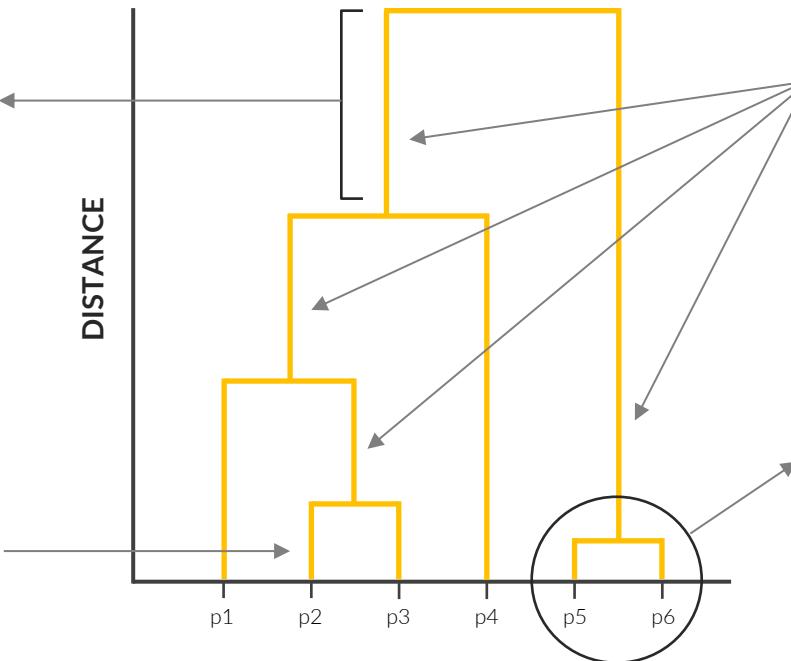
Key Takeaways



How do you know when to stop creating new clusters?

The **height** of each branch tells us how close the clusters/data points are to each other (**taller = longer distance**)

At the end of each clade is a **leaf**, representing a single data point



Vertical branches that lead to splits are called **clades**

Clades help us understand **similarity**; two leaves in the same clade are more similar to each other than they are to the leaves in another clade



# HIERARCHICAL CLUSTERING

Clustering Basics

K-Means

Hierarchical Clustering

Key Takeaways



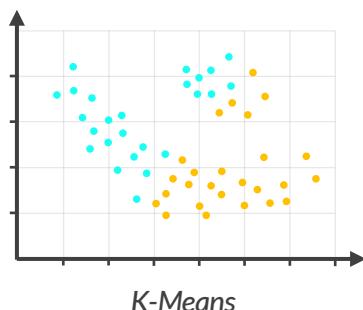
How exactly do you define the **distance** between clusters?

- There are a number of valid ways to measure the distance between clusters, which are often referred to as "**linkage methods**"
- Common methods include measuring the closest *min/max* distance between clusters, the lowest *average* distance, or the distance between cluster centroids

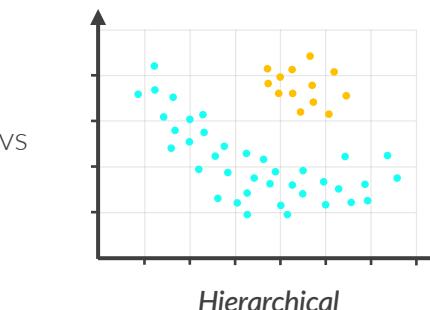


When might you use **hierarchical clustering** over **K-Means**?

- Run both models and compare outputs; Hierarchical clustering may produce more meaningful results if clusters are not circular or uniform in shape



K-Means



Hierarchical

vs

# KEY TAKEAWAYS

---



Cluster analysis is about finding concentrations of observations which are **similar to one another** and **distinct from other groups**

- Common use cases include identifying key customer segments, clustering stores based on performance, etc.



Two common techniques are **K-means** and **Hierarchical clustering**

- In practice, we recommend testing multiple models since each technique has strengths and weaknesses



Models **won't tell you how to interpret** what each cluster represents

- Use multivariate analysis and profiling techniques to understand the characteristics of each cluster

# ASSOCIATION MINING

# ASSOCIATION MINING



In this section we'll introduce the fundamentals of **association mining** and **basket analysis**, and compare common techniques including Apriori and Markov Chains

## TOPICS WE'LL COVER:

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

## GOALS FOR THIS SECTION:

- Understand the basics of association mining and introduce several real-world use cases
- Explore common association mining techniques like Apriori and Markov Chains
- Compare and contrast common unsupervised models for basket analysis



# ASSOCIATION MINING BASICS

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

**Association Mining** is used to reveal underlying patterns, relationships and correlations in the data, and translate them into IF/THEN association rules (i.e. “**If** a customer buys product A, **then** they will likely buy product B”)

- Association Mining is often applied to transactional data, to analyze which products tend to be purchased together (known as “**basket analysis**”)

## Common use cases:

- Building product recommendation systems (i.e. Amazon, Netflix)
- Sending targeted product offers based on purchase history
- Optimizing physical store layouts to maximize cross-selling



Association mining is NOT about trying to prove or establish causation; it's just about identifying frequently occurring patterns and correlations in large datasets



# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

The **Apriori** algorithm is commonly used to analyze how frequently items are purchased together in a single transaction (i.e. beer and diapers, peanut butter and jelly, etc.)

- In its simplest form (2-item sets), Apriori models compare the frequency of transactions containing **item A**, **item B**, and both **items A and B**
- This allows you to understand how often these items tend to be purchased together, and calculate the strength of the association between them

Apriori models typically include **three key metrics**:

- **Support** (frequency of transactions containing a given item or set of items) / total transactions
- **Confidence** (conditional probability of item B occurring, given item A)
- **Lift** (measures the importance or “strength” of the association between items)

$$\frac{\text{Support } (\mathbf{A},\mathbf{B})}{\text{Support } (\mathbf{A}) \times \text{Support } (\mathbf{B})}$$



# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

TRANS.	ITEM 1	ITEM 2
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

## EXAMPLE #1

Calculating the association between **bacon** & **eggs**

$$1) \text{ Support}(\text{bacon}) = 10/20 = 0.5$$

$$\text{Support}(\text{eggs}) = 7/20 = 0.35$$

$$\text{Support}(\text{bacon}, \text{eggs}) = 6/20 = 0.3$$

$$2) \text{ Confidence}(\text{bacon} \rightarrow \text{eggs}) = \frac{\text{Support}(\text{bacon}, \text{eggs})}{\text{Support}(\text{bacon})} = \frac{0.3}{0.5} = 60\%$$

$$3) \text{ Lift}(\text{bacon} \rightarrow \text{eggs}) = \frac{\text{Support}(\text{bacon}, \text{eggs})}{\text{Support}(\text{bacon}) \times \text{Support}(\text{eggs})} = \frac{0.3}{0.5 \times 0.35} = 1.7$$



Since **Lift > 1**, we can interpret the association between bacon and eggs as **real and informative** (*eggs are likely to be purchased with bacon*)



# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

TRANS.	ITEM 1	ITEM 2
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

## EXAMPLE #2

Calculating the association between **bacon** & **basil**

1) Support () =  $10/20 = 0.5$

Support () =  $5/20 = 0.25$

Support (, ) =  $1/20 = 0.05$

2) Confidence (  $\rightarrow$  ) =  $\frac{\text{Support}(\text{Bacon}, \text{Basil})}{\text{Support}(\text{Bacon})} = \frac{0.05}{0.5} = 10\%$

3) Lift (  $\rightarrow$  ) =  $\frac{\text{Support}(\text{Bacon}, \text{Basil})}{\text{Support}(\text{Bacon}) \times \text{Support}(\text{Basil})} = \frac{0.05}{0.5 \times 0.25} = 0.4$



Since **Lift < 1**, we can conclude that there is no positive association between bacon and basil (*basil is unlikely to be purchased with bacon*)



# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways

TRANS.	ITEM 1	ITEM 2
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

## EXAMPLE #3

Calculating the association between **bacon** & **water**

1) Support () =  $10/20 = 0.5$

Support () =  $1/20 = 0.05$

Support (, ) =  $1/20 = 0.05$

2) Confidence (  $\rightarrow$  ) =  $\frac{\text{Support}(\text{Bacon}, \text{Water})}{\text{Support}(\text{Bacon})} = \frac{0.05}{0.5} = 10\%$

3) Lift (  $\rightarrow$  ) =  $\frac{\text{Support}(\text{Bacon}, \text{Water})}{\text{Support}(\text{Bacon}) \times \text{Support}(\text{Water})} = \frac{0.05}{0.5 \times 0.05} = 2$



Since **Lift = 2** we might assume a strong association between bacon and water, but this is skewed since water only appears in one transaction



# APRIORI

Association Mining Basics

Apriori

Markov Chains

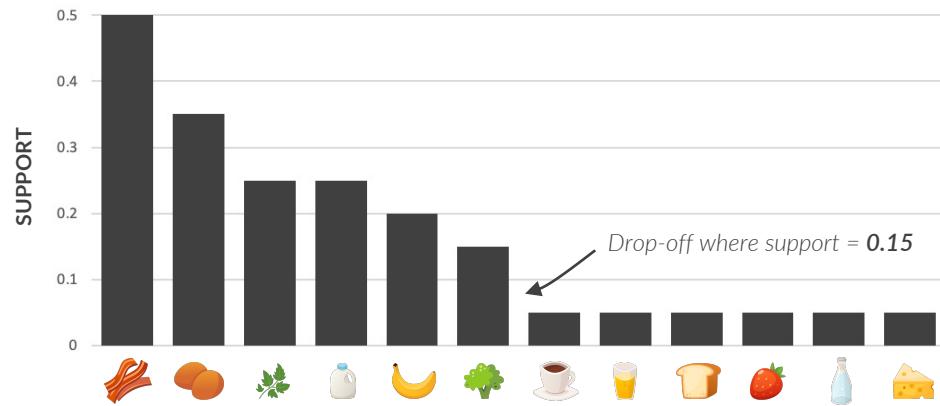
Key Takeaways

TRANS.	ITEM 1	ITEM 2
1	🥓	🥔
2	🍌	🌿
3	🥓	🥔
4	🥛	----
5	🥓	🌿
6	----	菔
7	🥓	🥔
8	----	----
9	🥓	🥔
10	🥛	----
11	🥓	🥔
12	🍌	🥛
13	🥓	🥔
14	🌿	----
15	菔	----
16	🌿	----
17	菔	----
18	----	🌿
19	🥓	🥛
20	banana	🌿



How do we account for **infrequently purchased items?**

To filter low-volume purchases, you can plot support for each item and determine a threshold or cutoff value:



In this case we might filter out any transactions containing items with support **<0.15** (transactions 4, 6, 8, 10, 15, 17)

- This helps us avoid misleading confidence and lift calculations, and reduces the number of transactions we need to analyze



# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

Consider a shop that sells **4 items**  
(chocolate, cheese, bananas & bread)

There are **15 possible combinations**  
or itemsets (*ignoring duplicates*)





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

## STEP 1:

Calculate support for each **1-item** set, and filter out *all* transactions containing items below the threshold (*in this case cheese*)





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

## STEP 2:

Based on the remaining itemsets, calculate support for each **2-item** set, and filter out transactions containing any pairs below the threshold (*in this case chocolate & bread*)





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

## STEP 3:

Repeat until all infrequent itemsets have been eliminated, and filter transactions accordingly





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

## STEP 4:

Based on the filtered transactions, you can use an apriori model to calculate confidence and lift and identify the strongest associations

1-item sets:



2-item sets:



3-item sets:



4-item sets:





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



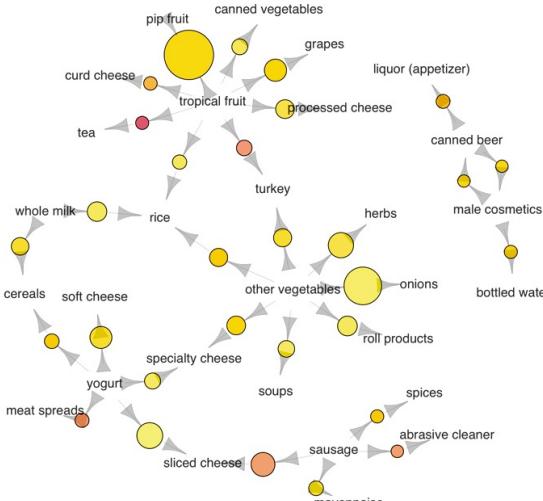
Wouldn't you want to analyze *all* possible item combinations, instead of calculating individual associations?

- Yes! In reality that's how Apriori works. But since the number of configurations increases exponentially with each item, this is impractical for humans to do
- To reduce or "prune" the number of itemsets to analyze, we can use the **apriori principle**, which basically states that if an item is infrequent, any combinations containing that item must also be infrequent

## STEP 4:

Based on the filtered transactions, you can use an apriori model to calculate confidence and lift and identify the strongest associations

Associations can be visualized using **network graphs** like this one





# APRIORI

Association Mining Basics

Apriori

Markov Chains

Key Takeaways



Can you calculate associations between **multiple items**, like coffee being purchased with bacon and eggs?

- Yes, you can calculate support, confidence and lift using the same exact logic as you would with individual items

ID	ITEM 1	ITEM 2	ITEM 3
1	🥓	🍊	☕
2	🍌	🌿	
3	🥓		
4	🍞	🥛	🍓
5	🥓	☕	
6	🍓	🌿	
7	🥓	🍊	☕
8	☕	🥛	
9	🥓	🍊	🌿
10	🍺	🥛	

$$\text{Support} (\text{🥓, 🥩}) = 3/10 = \mathbf{0.3}$$

$$\text{Support} (\text{☕}) = 4/10 = \mathbf{0.4}$$

$$\text{Support} (\text{🥓, 🥩, ☕}) = 2/10 = \mathbf{0.2}$$

$$\text{Confidence} (\text{🥓 & 🥩} \rightarrow \text{☕}) = \frac{\text{Support} (\text{🥓, 🥩, ☕})}{\text{Support} (\text{🥓, 🥩})} = \frac{0.2}{0.3} = \mathbf{67\%}$$

$$\text{Lift} (\text{🥓 & 🥩} \rightarrow \text{☕}) = \frac{\text{Support} (\text{🥓, 🥩, ☕})}{\text{Support} (\text{🥓, 🥩}) \times \text{Support} (\text{☕})} = \frac{0.2}{0.12} = \mathbf{1.7}$$



Calculating all possible associations would be impossible for a human; this is where Machine Learning comes in!

# CASE STUDY: APRIORI

---



## THE SITUATION

You are the proud owner of **Coastal Roasters**, a local coffee shop and bakery based in the Pacific Northwest.



## THE ASSIGNMENT

You'd like to better understand **which items customers tend to purchase together**, to help inform product placement and promotional strategies.

As a first step, you've decided to analyze a sample of ~10,000 transactions over a 6-month period, and conduct a simple basket analysis using an apriori model.



## THE OBJECTIVES

1. Collect transaction-level data, including date, time, and items purchased
2. Determine an appropriate support threshold, based on product frequency
3. Calculate confidence and lift to measure association between popular items



# MARKOV CHAINS

Association Mining Basics

Apriori

Markov Chains

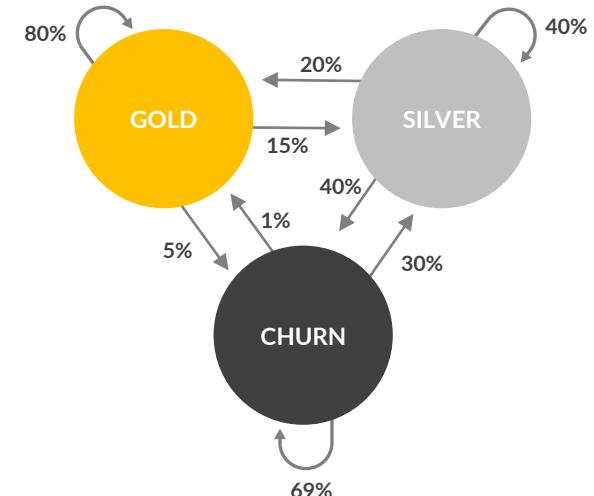
Key Takeaways

**Markov Chains** are used to describe the flow or transition between “states” of a categorical variable

- Markov Chains **calculate the probability of moving between states** (subscribers renewing or churning, customers purchasing one product after another, etc.)
- Observations must be tracked over time and tied to a unique ID, so that all state transitions can be recorded as probabilities in a matrix (where all rows = 100%)

**EXAMPLE:** Monthly subscribers transitioning between **Gold**, **Silver** and **Churn** states each month:

		TO STATE		
		Gold	Silver	Churn
FROM STATE	Gold	0.8	0.15	0.05
	Silver	0.2	0.4	0.4
	Churn	0.01	0.3	0.69





# MARKOV CHAINS

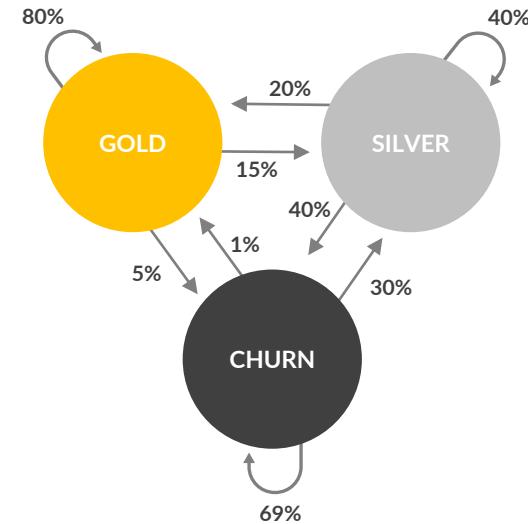
Association Mining Basics

Apriori

Markov Chains

Key Takeaways

		TO STATE		
		Gold	Silver	Churn
FROM STATE	Gold	<b>0.8</b>	<b>0.15</b>	<b>0.05</b>
	Silver	<b>0.2</b>	<b>0.4</b>	<b>0.4</b>
	Churn	<b>0.01</b>	<b>0.3</b>	<b>0.69</b>



## Example insights & recommendations:

- Most customers who churn stayed churned, but **31%** do come back; of those who return, nearly all of them re-subscribe to a Silver plan (vs. Gold)
  - ✓ **RECOMMENDATION:** Launch targeted marketing to recently churned customers, offering a discount to resubscribe to a Silver membership plan
- Once customers upgrade to a Gold membership, the majority **80%** renew each month
  - ✓ **RECOMMENDATION:** Offer a one-time discount for Silver customers to upgrade to Gold; while you may sacrifice some short-term revenue, it will likely be profitable in the long term



To account for prior transitions (*vs. just the previous*) you can use more complex “higher-order” Markov Chains

# CASE STUDY: MARKOV CHAINS

---



## THE SITUATION

You've just been promoted to Senior Web Analyst at **Alpine Supplies**, an online retailer specializing in equipment and supplies for outdoor enthusiasts.



## THE ASSIGNMENT

The VP of Sales just shared a sample of ~15,000 customer purchase paths, and would like you to analyze the data to **help inform a new cross-sell sales strategy**.

Your goal is to explore the data and build a simple Markov model to predict which product an existing customer is most likely to purchase next.



## THE OBJECTIVES

1. Collect sample data containing customer-level purchase paths
2. Calculate a frequency and probability matrix for popular products
3. For any given product, predict the most likely future purchase

# KEY TAKEAWAYS

---



Association mining reveals **patterns, relationships & correlations** in the data, and translates them into IF/THEN association rules

- Common use cases include building recommendation systems (i.e. Amazon, Netflix), targeting product offers based on purchase history, optimizing physical store layouts to maximize cross-selling, etc.



**Apriori models** measure how frequently specific items tend to be purchased together (known as “basket analysis”)

- Rather than analyzing every possible combination of itemsets, the apriori principle can be used to speed up calculations by filtering out infrequent transactions



**Markov Chains** can be used to measure the probability of transitioning between “states” of a categorical variable

- Markov models analyze sequences of events over time to determine what is most likely to happen next

# OUTLIER DETECTION

# OUTLIER DETECTION



In this section we'll introduce the concept of **statistical outliers**, and review common methods for detecting both cross-sectional and time-series outliers and anomalies

## TOPICS WE'LL COVER:

Outlier Detection Basics

Cross-sectional Outliers

Time-Series Outliers

Key Takeaways

## GOALS FOR THIS SECTION:

- Understand different types of outliers, and how they can impact an analysis
- Learn how to identify outliers in a cross-sectional dataset using a “nearest neighbors” approach
- Learn how to use regression analysis to detect outliers in a time-series dataset



The terms **outlier detection**, **anomaly detection**, and **rare-event detection** are often used interchangeably; they all focus on finding observations which are materially different than the others (*outliers are also sometimes called “pathological” data*)



# OUTLIER DETECTION BASICS

Outlier Detection Basics

Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

**Outlier detection** is used to identify observations in a dataset which are either unexpected or statistically different from the others

There are **two general types** of outliers you may encounter:

- **Data/recording issues:** Values which were captured or categorized incorrectly, and which must be removed or excluded to avoid skewing the analysis
- **Business outliers:** Legitimate values which provide real and meaningful information, such as a fraudulent transaction or an unexpected spike in sales

**Common use cases:**

- Identifying store locations which are behaving abnormally compared to others
- Detecting web traffic anomalies which would be impossible to detect otherwise
- Flagging potentially fraudulent credit card transactions



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

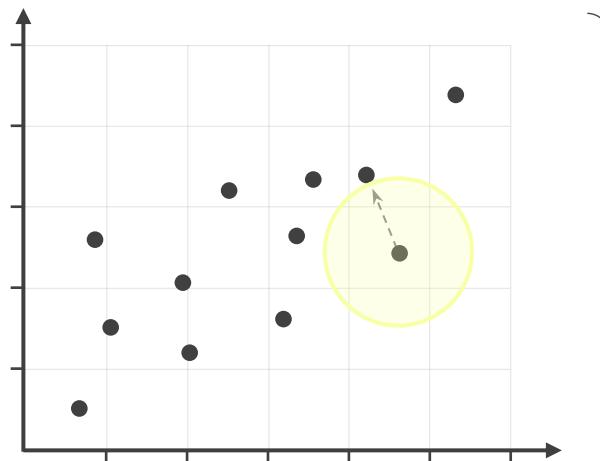
Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

**Cross-sectional outlier detection** is used to measure the similarity between observations in a dataset, and identify observations which are unusually different or dissimilar

- Detecting outliers in one or two dimensions is often trivial, using basic profiling metrics (*i.e. interquartile range*) or visual analysis (scatterplots, box plots, etc.)
- Detecting outliers 3+ dimensions is trickier, and often requires more sophisticated techniques; **this is where machine learning comes in!**



A common technique is to calculate the distance between each observation and its **nearest neighbor**, and plot those distances to reveal outliers



**NOTE:** Raw data should first be **scaled** to produce consistent and meaningful distance calculations



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615

In this example we're looking at normalized sales for 3 product categories (**x\_1**, **x\_2** and **x\_3**) across **101** different store locations

- In cases like these, detecting outliers by visualizing distributions or scatterplots can be difficult (and often impossible)



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

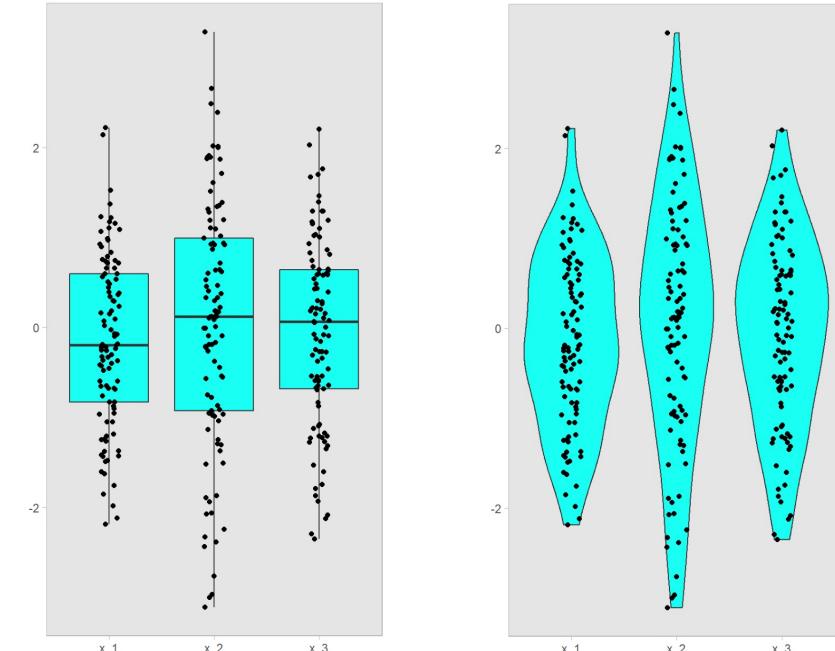
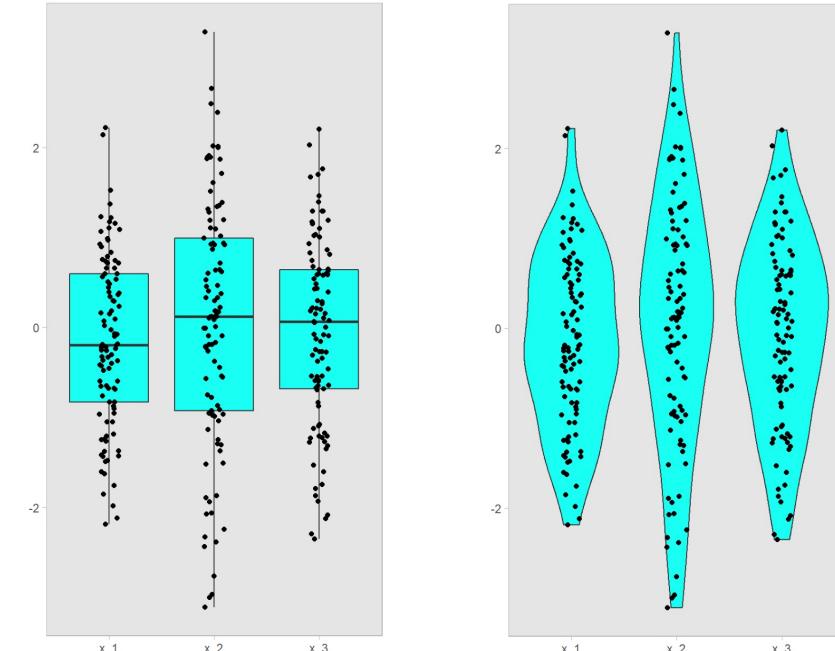
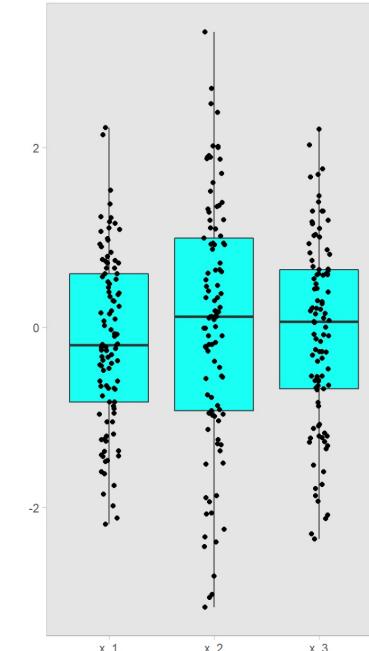
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



In this case, **box and violin plots** don't reveal any obvious outliers...



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

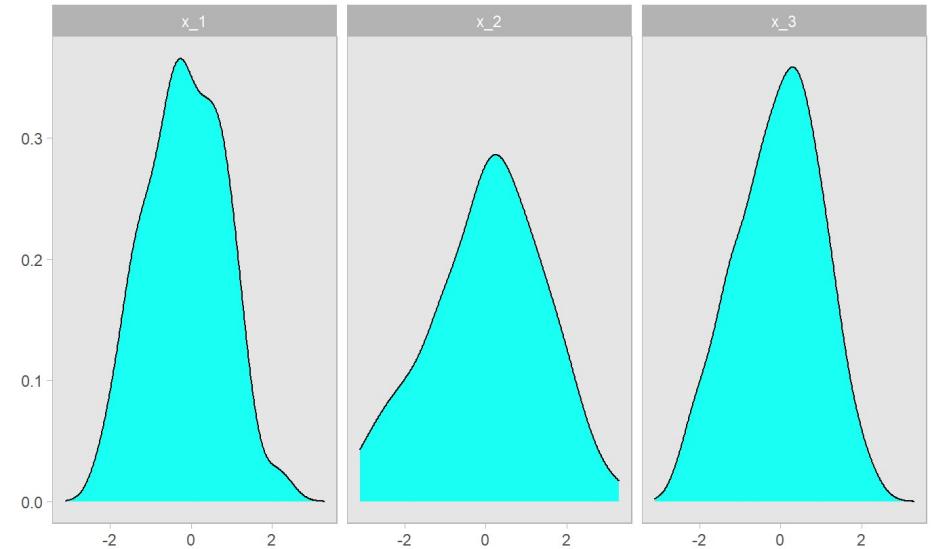
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



**Kernel density plots** show relatively normal distributions for each variable...



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

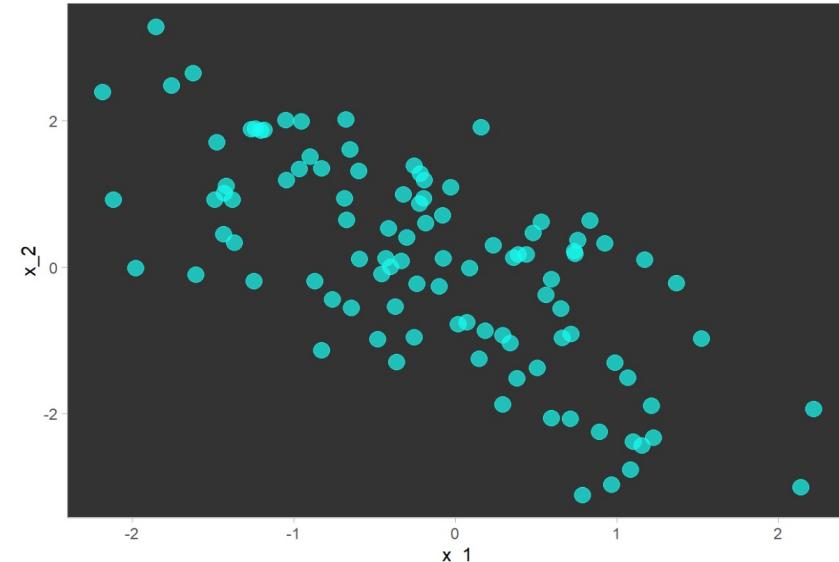
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



And **scatter plots** fail to expose any anomalies in the data



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

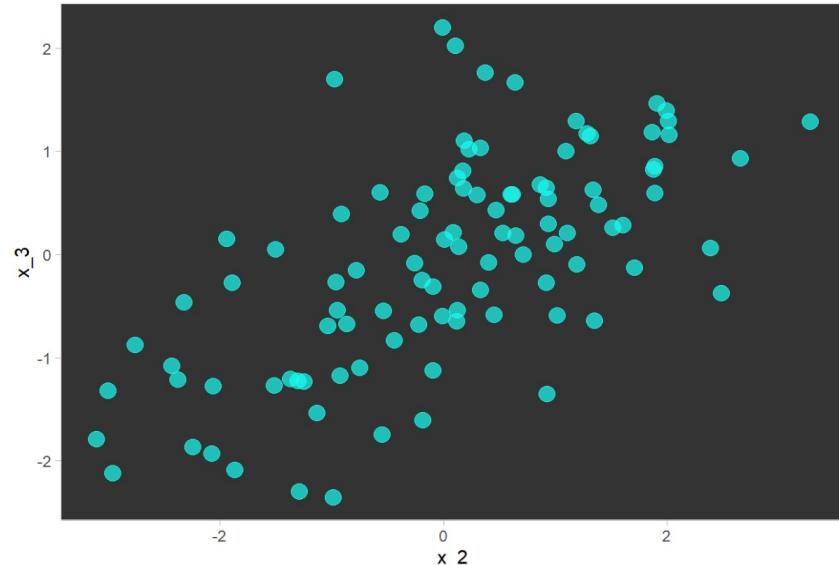
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



And **scatter plots** fail to expose any anomalies in the data



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

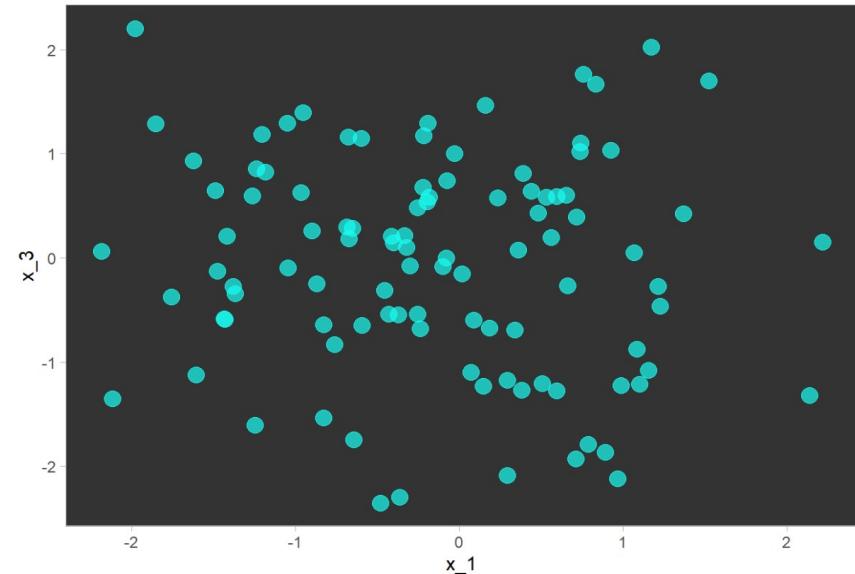
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



And **scatter plots** fail to expose any anomalies in the data



# CROSS-SECTIONAL OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

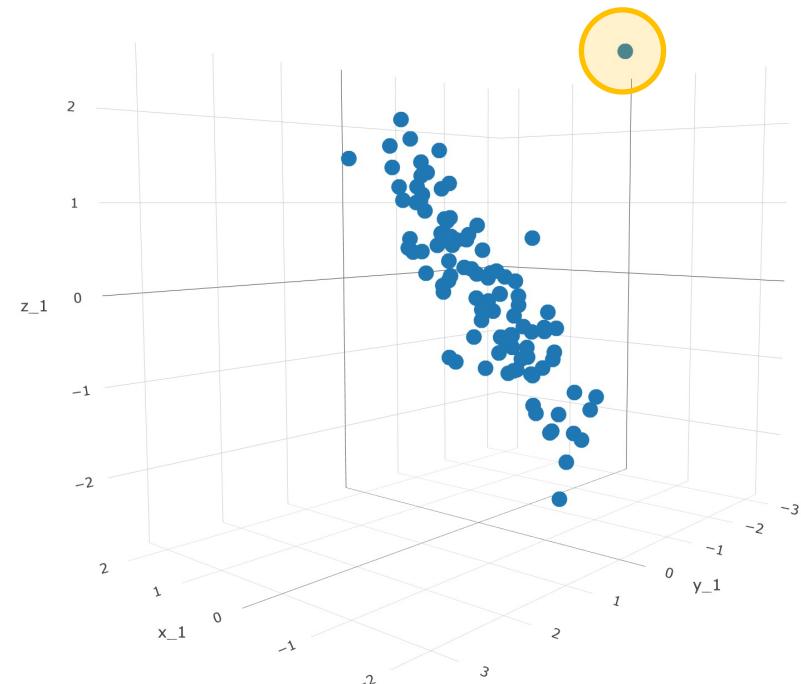
Time-Series Outliers

Key Takeaways

## EXAMPLE

Normalized store sales by product category ( $n=101$ )

	A	B	C	D
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>
2	1	0.0187	-0.7806	-0.1540
3	2	-0.1843	0.6036	0.5848
4	3	-1.3713	0.3314	-0.3448
5	4	-0.5992	1.3107	1.1476
6	5	0.2945	-0.9278	-1.1736
7	6	0.3898	0.1734	0.8111
8	7	-1.2081	1.8691	1.1873
9	8	-0.3637	-1.2944	-2.2954
10	9	-1.6267	2.6548	0.9313
11	10	-0.2565	1.3844	0.4804
12	11	1.1018	-2.3819	-1.2092
13	12	0.7558	0.3731	1.7597
14	13	-0.2382	-0.2259	-0.6799
15	14	0.9874	-1.3032	-1.2271
16	15	0.7414	0.1829	1.1006
17	16	0.0893	-0.0122	-0.5971
18	17	-0.9549	1.9949	1.3938
19	18	-0.1952	0.9370	0.5364
20	19	0.9255	0.3300	1.0325
21	20	0.4830	0.4679	0.4303
22	21	-0.5963	0.1149	-0.6460
23	22	-2.1853	2.3882	0.0615



But there *is* an outlier in the sample, which we can clearly see when we visualize a **3rd dimension**



# CROSS-SECTIONAL OUTLIERS

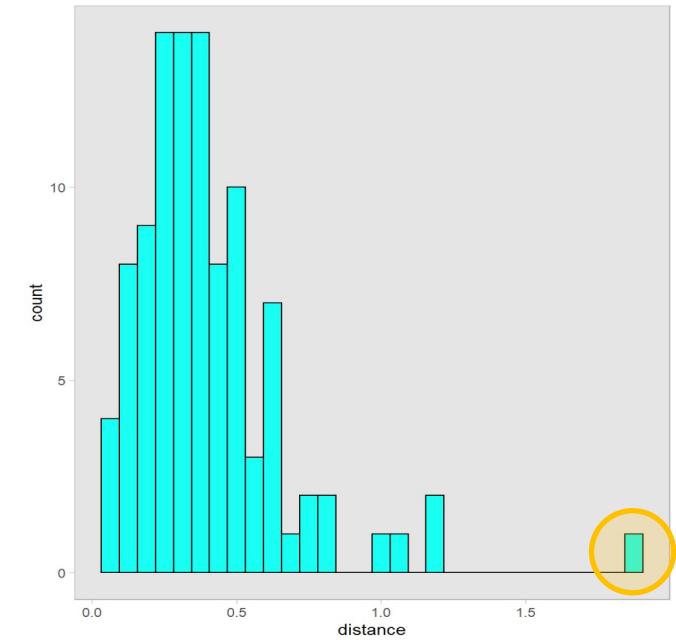
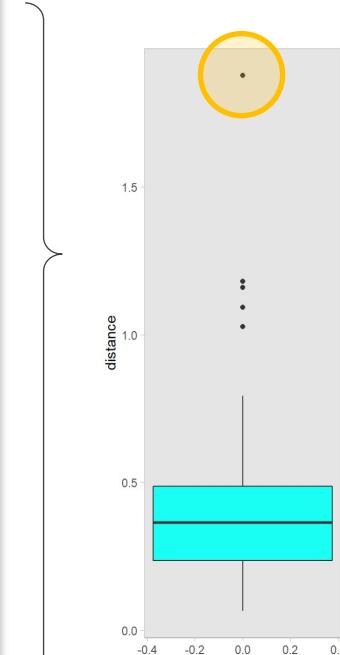
Outlier Detection Basics

Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

	A	B	C	D	E
1	<b>id</b>	<b>x_1</b>	<b>x_2</b>	<b>x_3</b>	<b>NN</b>
2	1	0.0187	-0.7806	-0.1540	0.505
3	2	-0.1843	0.6036	0.5848	0.280
4	3	-1.3713	0.3314	-0.3448	0.277
5	4	-0.5992	1.3107	1.1476	0.383
6	5	0.2945	-0.9278	-1.1736	0.297
7	6	0.3898	0.1734	0.8111	0.178
8	7	-1.2081	1.8691	1.1873	0.236
9	8	-0.3637	-1.2944	-2.2954	0.511
10	9	-1.6267	2.6548	0.9313	0.758
11	10	-0.2565	1.3844	0.4804	0.455
12	11	1.1018	-2.3819	-1.2092	0.148
13	12	0.7558	0.3731	1.7597	0.290
14	13	-0.2382	-0.2259	-0.6799	0.368
15	14	0.9874	-1.3032	-1.2271	0.487
16	15	0.7414	0.1829	1.1006	0.091
17	16	0.0893	-0.0122	-0.5971	0.400
18	17	-0.9549	1.9949	1.3938	0.142
19	18	-0.1952	0.9370	0.5364	0.158
20	19	0.9255	0.3300	1.0325	0.218
21	20	0.4830	0.4679	0.4303	0.219
22	21	-0.5963	0.1149	-0.6460	0.196
23	22	-2.1853	2.3882	0.0615	0.619
24	23	-0.6749	0.6431	0.1845	0.284
25	24	-1.19	0.9135	-1.39	1.02



By calculating the distance from each point to its nearest neighbor and plotting the distribution, we can detect a **clear outlier** in the data

# CASE STUDY: OUTLIER DETECTION

---



## THE SITUATION

You are a Senior Data Analyst for **Brain Games**, a large chain of retail shops specializing in educational toys, puzzles and board games.



## THE ASSIGNMENT

You'd like to analyze store-level sales by product category, to identify patterns and **see if any locations show an unusual composition of revenue**.

To do this, you'll be exploring revenue data across 100 individual store locations, and detecting outliers using "nearest neighbor" calculations.



## THE OBJECTIVES

1. Collect category-level sales data by store location
2. Use scatterplots to visualize relationships and scan for outliers
3. Calculate the nearest neighbor for each observation and visualize the data using a box plot and histogram. Do you see any outliers now?



# TIME-SERIES OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

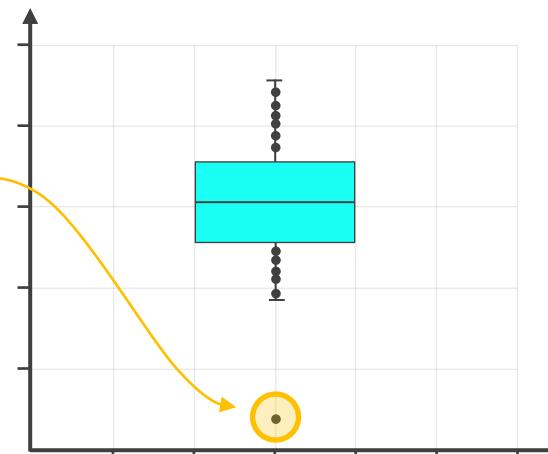
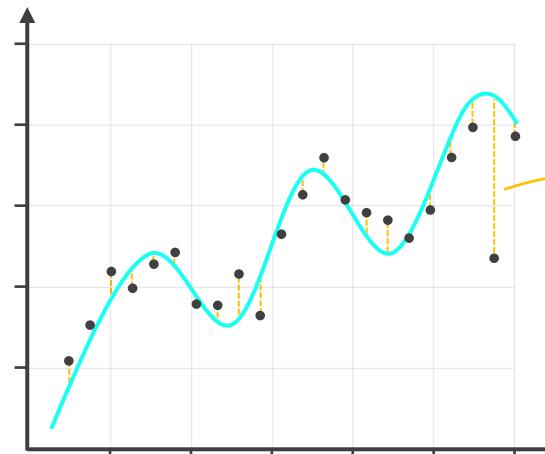
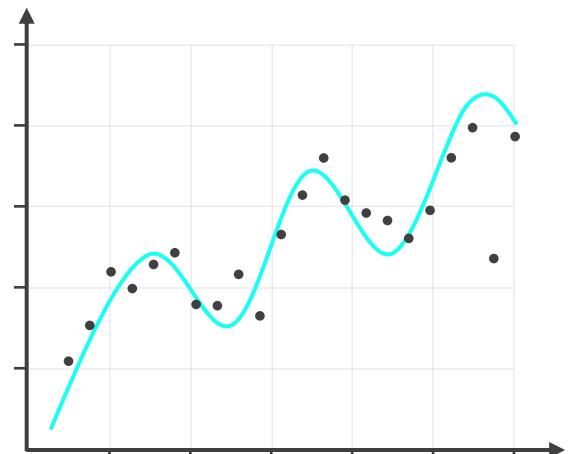
**Time-series outlier detection** is used to identify observations which fall well outside expectations for a particular point in time, after accounting for seasonality and trending

- In practice, this involves building a **time-series regression**, comparing each observation to its predicted value, and plotting the residuals to detect anomalies

Fit regression model

Calculate residuals

Plot residuals





# TIME-SERIES OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

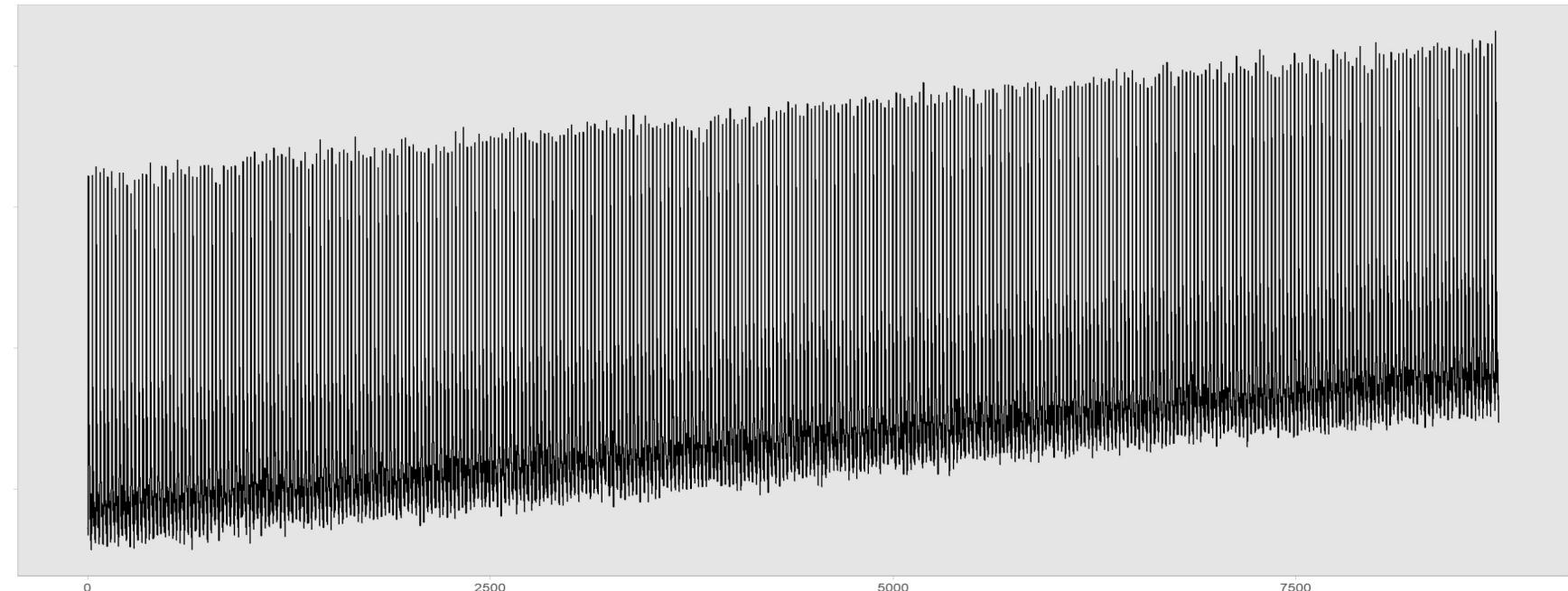
Time-Series Outliers

Key Takeaways

Real-world time-series data is often **extremely granular**, with recurring patterns and trends which make visual outlier detection virtually impossible

## EXAMPLE

*Hourly website sessions ( $n=8,760$ )*





# TIME-SERIES OUTLIERS

Outlier Detection Basics

Cross-Sectional Outliers

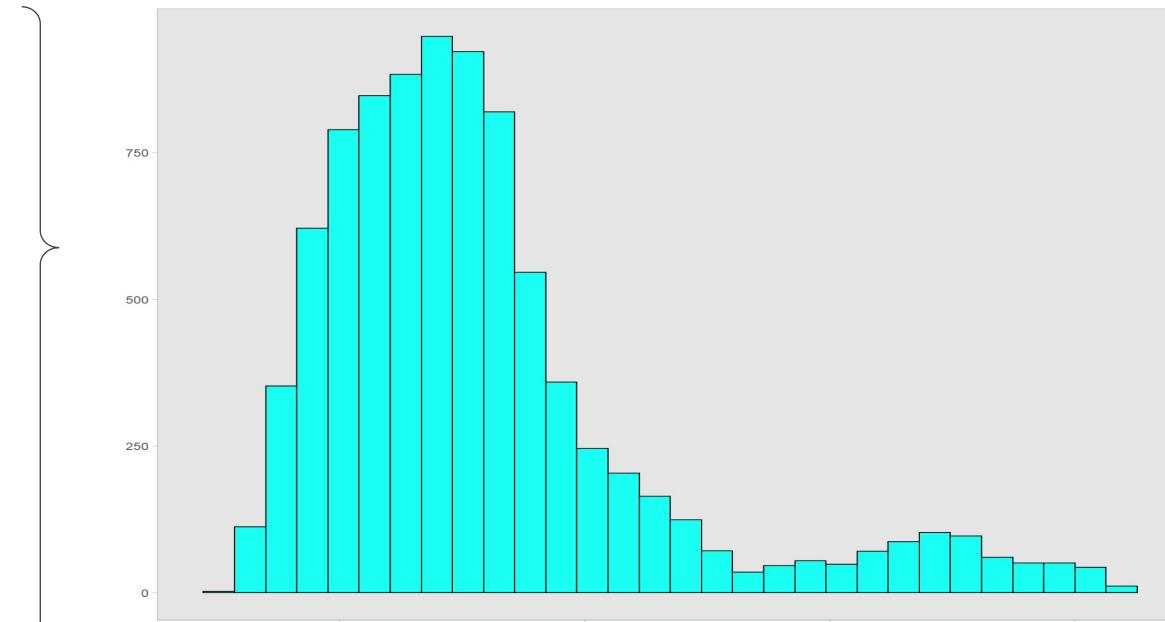
Time-Series Outliers

Key Takeaways

## EXAMPLE

Hourly website sessions ( $n=8,760$ )

	A	B
1	index	value
2	1	14.3710
3	2	13.4375
4	3	16.3675
5	4	20.6394
6	5	64.4129
7	6	53.9047
8	7	35.5245
9	8	31.9205
10	9	29.0358
11	10	28.9568
12	11	23.3266
13	12	23.3105
14	13	15.6372
15	14	17.7494
16	15	16.8970
17	16	18.6685
18	17	14.7505
19	18	13.3804
20	19	12.5986
21	20	19.3613
22	21	13.7367
23	22	11.2642
24	23	12.8758



Just like our cross-sectional example, univariate distributions don't reveal any outliers or anomalies



# TIME-SERIES OUTLIERS

# Outlier Detection Basics

## Cross-Sectional Outliers

# Time-Series Outliers

## Key Takeaways

Instead of simple visual analysis, we can fit a **linear regression model** to account for seasonality (hour of day) and trending

- From there, we can **plot the distribution of residuals** in order to quickly identify any observations which significantly deviated from the model's prediction



# TIME-SERIES OUTLIERS

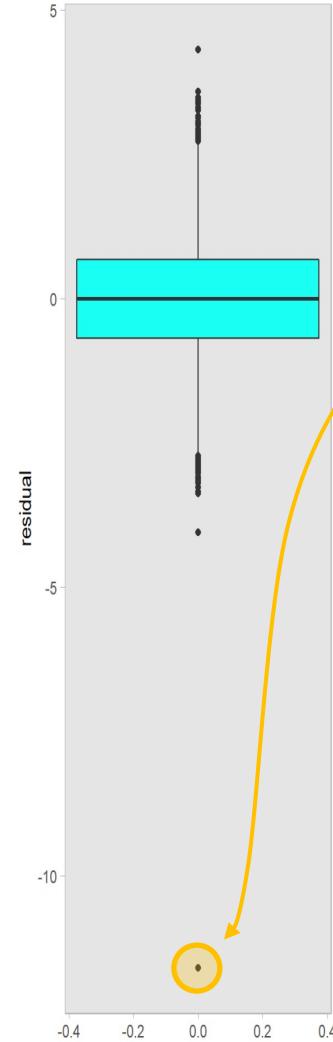
Outlier Detection Basics

Cross-Sectional Outliers

Time-Series Outliers

Key Takeaways

index	web_visits	prediction	residual
1	14.3710	12.9582	1.4127
2	13.4375	13.9417	-0.5042
3	16.3675	15.9503	0.4172
4	20.6394	20.0701	0.5693
5	64.4129	64.0274	0.3855
6	53.9047	53.9894	-0.0847
7	35.5245	34.0143	1.5102
8	31.9205	31.9487	-0.0281
9	29.0358	26.9367	2.0991
10	28.9568	28.9841	-0.0273
11	23.3266	21.9954	1.3312
12	23.3105	21.0162	2.2943
13	15.6372	17.0982	-1.4611
14	17.7494	18.0255	-0.2761
15	16.8970	17.0690	-0.1719
16	18.6685	18.0356	0.6329
17	14.7505	15.0293	-0.2788
18	13.3804	16.0404	-2.6599
19	12.5986	14.9461	-2.3475
20	19.3613	18.0447	1.3166
21	13.7367	14.0055	-0.2687
22	11.2642	13.0380	-1.7737
23	12.8758	12.9748	-0.0990
24	14.2646	12.9861	1.2785
25	14.9473	13.0103	1.9369



When we plot the distribution of model residuals, we detect a clear outlier in the data



**PRO TIP:** Not all outliers are bad! If you find an anomaly, understand what happened and how you can learn from it

# KEY TAKEAWAYS

---



**Outlier detection** is used to identify observations in a dataset which are either unexpected or statistically different from the others

- Common use cases include detecting web traffic anomalies, flagging fraudulent credit card transactions, identifying unusually high or low-performing store locations, etc.



For cross-sectional analysis, plotting “**nearest neighbor**” **distances** can help detect anomalies when visual profiling falls short

- This approach is ideal for exposing outliers in large, complex or highly-dimensional datasets



For time-series analysis, you can fit a **regression model** to the data and plot the residuals to clearly identify outliers

- This allows you to quickly find outliers, while controlling for seasonality and trending

# DIMENSIONALITY REDUCTION

# DIMENSIONALITY REDUCTION



In this section we'll review the fundamentals of **dimensionality reduction**, discuss common Machine Learning and business use cases, and introduce core techniques like Principal Component Analysis (PCA)

## TOPICS WE'LL COVER:

Dimensionality Reduction Basics

Principal Component Analysis

Advanced Techniques

Key Takeaways

## GOALS FOR THIS SECTION:

- Understand how dimensionality reduction can be applied in business and Machine Learning contexts
- Explore the intuition behind core linear dimensionality reduction techniques, including Principal Component Analysis (PCA)
- Introduce more advanced models, including techniques for non-linear dimensionality reduction



# DIMENSIONALITY REDUCTION

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

**Dimensionality reduction** involves reducing the number of columns (i.e. dimensions) in a dataset while losing as little information as possible

- This can be used strictly as a ML optimization technique, or to help develop a better understanding of the data for business intelligence, market research, etc.

## COMMON BUSINESS USE CASES:

- ✓ Summarizing item-level sales data to better understand group or category performance
- ✓ Analyzing survey data to understand overarching trends and findings without parsing through individual question responses
- ✓ Understanding the main traits or characteristics of your customers (website activity, transaction patterns, survey response data, etc.)

## COMMON ML USE CASES:

- ✓ Improving a model's predictive power by reducing redundant or unnecessary dimensions in the data
- ✓ Visualizing clusters in limited dimensions to spot-check cluster/segmentation outputs
- ✓ Reducing trivial correlations and removing multicollinearity to build more accurate and meaningful models



# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

One of the most common dimensionality reduction techniques is **Principal Component Analysis (PCA)**

- In the simplest form, PCA finds lines that best fit through the observations in a data set, and uses those lines to create new dimensions to analyze

STEP 1

STEP 2

STEP 3

STEP 4

STEP 5

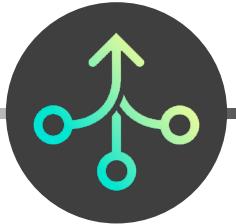
Use high-dimensional data that could be more useful if it could be accurately represented with fewer dimensions

Find “components” or lines that fit through those dimensions, to potentially use as new dimensions to represent the data

Interpret the weighting of each component on the original dimensions to build an intuition for what they represent

Select the “principal” components which explain a material proportion of the variance in the data

Use those principal components in place of existing dimensions for further analysis



# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

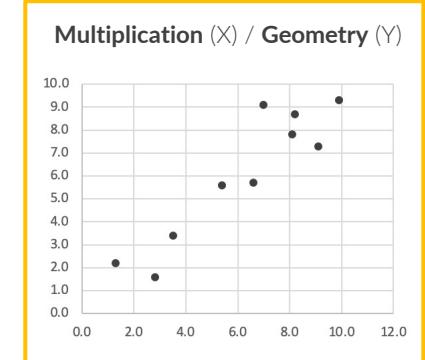
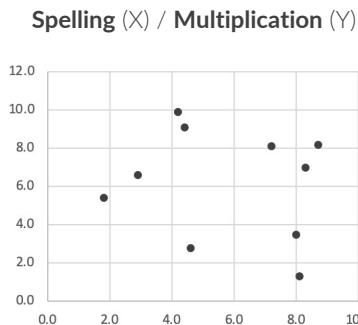
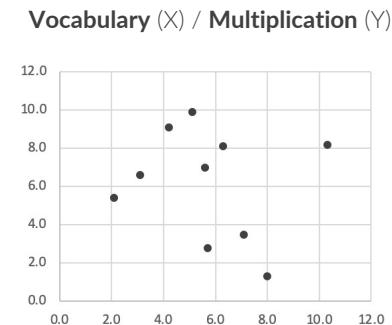
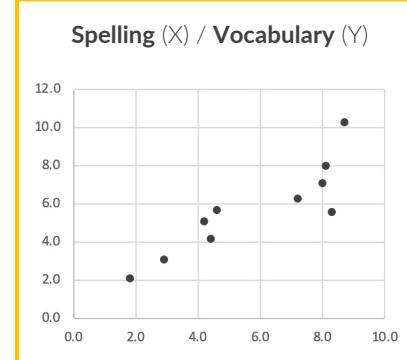
## EXAMPLE

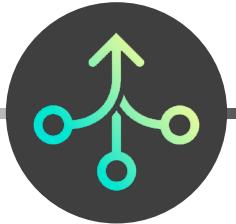
Student test scores by subject ( $n=10$ )

studentID	spelling	vocabulary	multiplication	geometry
1	1.8	2.1	5.4	5.6
2	2.9	3.1	6.6	5.7
3	4.6	5.7	2.8	1.6
4	8.0	7.1	3.5	3.4
5	8.1	8.0	1.3	2.2
6	8.3	5.6	7.0	9.1
7	4.4	4.2	9.1	7.3
8	7.2	6.3	8.1	7.8
9	4.2	5.1	9.9	9.3
10	8.7	10.3	8.2	8.7

Plotting pairs of dimensions reveals some correlation between **Spelling/Vocabulary** and **Multiplication/Geometry** test scores

- Linear dimensionality reduction techniques will find lines that fit through highly correlated dimensions first





# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

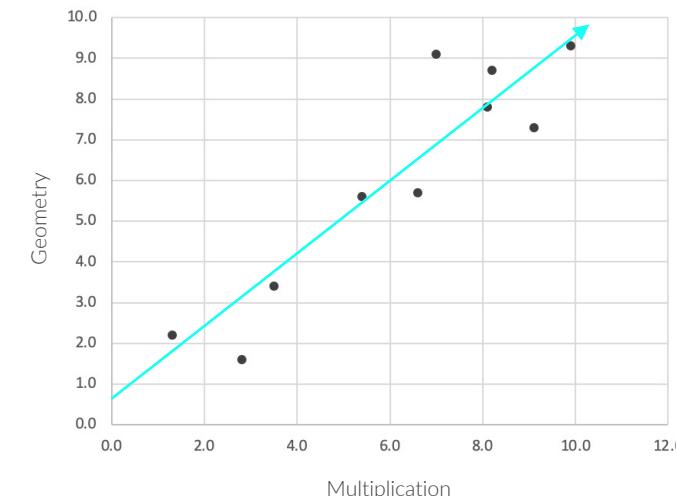
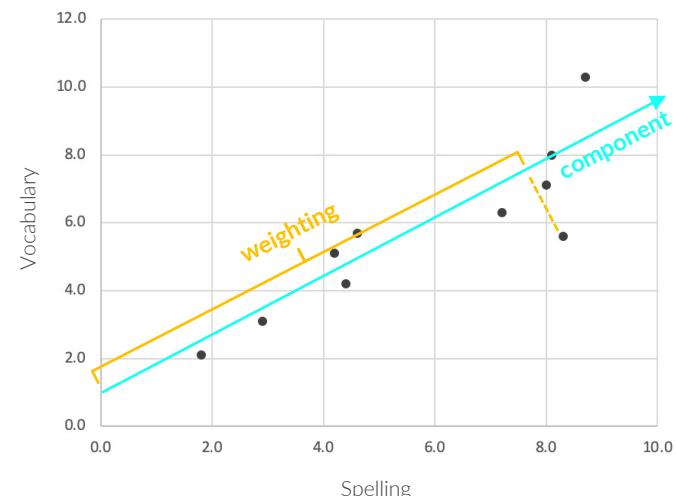
## EXAMPLE

Student test scores by subject ( $n=10$ )

studentID	spelling	vocabulary	multiplication	geometry
1	1.8	2.1	5.4	5.6
2	2.9	3.1	6.6	5.7
3	4.6	5.7	2.8	1.6
4	8.0	7.1	3.5	3.4
5	8.1	8.0	1.3	2.2
6	8.3	5.6	7.0	9.1
7	4.4	4.2	9.1	7.3
8	7.2	6.3	8.1	7.8
9	4.2	5.1	9.9	9.3
10	8.7	10.3	8.2	8.7

In the context of dimensionality reduction, these lines are called **components** or **factors**

Observations are given a **weighting** for each component, which essentially measures its position relative to the length of the line





# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

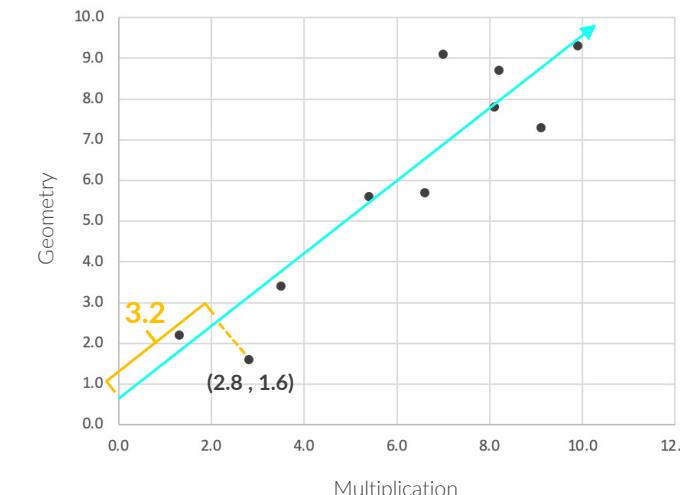
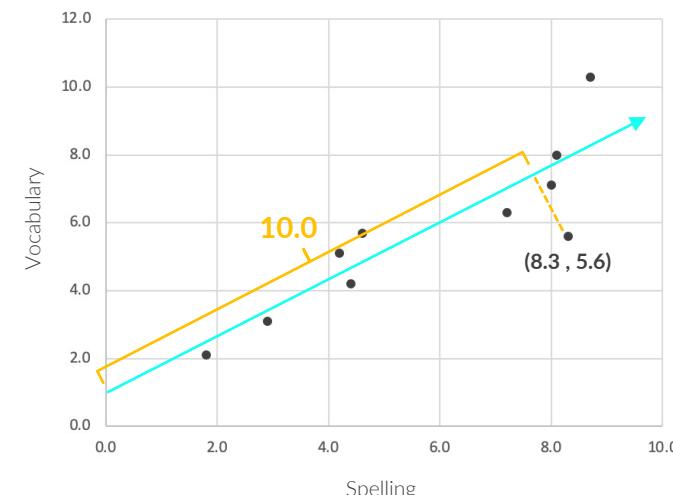
## EXAMPLE

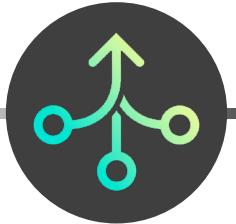
Student test scores by subject ( $n=10$ )

studentID	spelling	vocabulary	multiplication	geometry	component 1	component 2
1	1.8	2.1	5.4	5.6	2.8	7.8
2	2.9	3.1	6.6	5.7	4.3	8.7
3	4.6	5.7	2.8	1.6	7.3	3.2
4	8.0	7.1	3.5	3.4	10.7	4.9
5	8.1	8.0	1.3	2.2	11.4	2.6
6	8.3	5.6	7.0	9.1	10.0	11.5
7	4.4	4.2	9.1	7.3	6.1	11.6
8	7.2	6.3	8.1	7.8	9.6	11.3
9	4.2	5.1	9.9	9.3	6.6	13.6
10	8.7	10.3	8.2	8.7	13.5	12.0

Plot the component weightings for each point, and record them as **new dimensions** in the table

**NOTE:** We're looking at only 2 dimensions here, but PCA scales to **any number of dimensions**





# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways



How do you determine the meaning of each component?

- PCA will generate many potential components, and it won't be immediately clear what they actually represent
- In practice, PCA models calculate weights (or "**loadings**") to help explain the relationship between each component and the original dimensions (*think of loadings like coefficients in a linear regression*)

studentID	$X_1$ spelling	$X_2$ vocabulary	$X_3$ multiplication	$X_4$ geometry	$Y$ language	component 2
1	1.8	2.1	5.4	5.6	2.8	7.8
2	2.9	3.1	6.6	5.7	4.3	8.7
3	4.6	5.7	2.8	1.6	7.3	3.2
4	8.0	7.1	3.5	3.4	10.7	4.9
5	8.1	8.0	1.3	2.2	11.4	2.6
6	8.3	5.6	7.0	9.1	10.0	11.5
7	4.4	4.2	9.1	7.3	6.1	11.6
8	7.2	6.3	8.1	7.8	9.6	11.3
9	4.2	5.1	9.9	9.3	6.6	13.6
10	8.7	10.3	8.2	8.7	13.5	12.0

$$Y = (0.72)*X_1 + (0.69)*X_2 + (-0.02)*X_3 + (0.03)*X_4$$

High coefficients on  $X_1$  (spelling) and  $X_2$  (vocabulary), so we might interpret component 1 as "language"



# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways



How do you determine the meaning of each component?

- PCA will generate many potential components, and it won't be immediately clear what they actually represent
- In practice, PCA models calculate weights (or "**loadings**") to help explain the relationship between each component and the original dimensions (*think of loadings like coefficients in a linear regression*)

studentID	X <sub>1</sub> spelling	X <sub>2</sub> vocabulary	X <sub>3</sub> multiplication	X <sub>4</sub> geometry	Y language	Y math
1	1.8	2.1	5.4	5.6	2.8	7.8
2	2.9	3.1	6.6	5.7	4.3	8.7
3	4.6	5.7	2.8	1.6	7.3	3.2
4	8.0	7.1	3.5	3.4	10.7	4.9
5	8.1	8.0	1.3	2.2	11.4	2.6
6	8.3	5.6	7.0	9.1	10.0	11.5
7	4.4	4.2	9.1	7.3	6.1	11.6
8	7.2	6.3	8.1	7.8	9.6	11.3
9	4.2	5.1	9.9	9.3	6.6	13.6
10	8.7	10.3	8.2	8.7	13.5	12.0

$$Y = (0.004)*X_1 + (0.001)*X_2 + (0.68)*X_3 + (0.72)*X_4$$

High coefficients on X<sub>3</sub> (**multiplication**) and X<sub>4</sub> (**geometry**), so we might interpret component 2 as "**math**"



This requires **human intuition**; your model won't tell you what the components mean!



# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

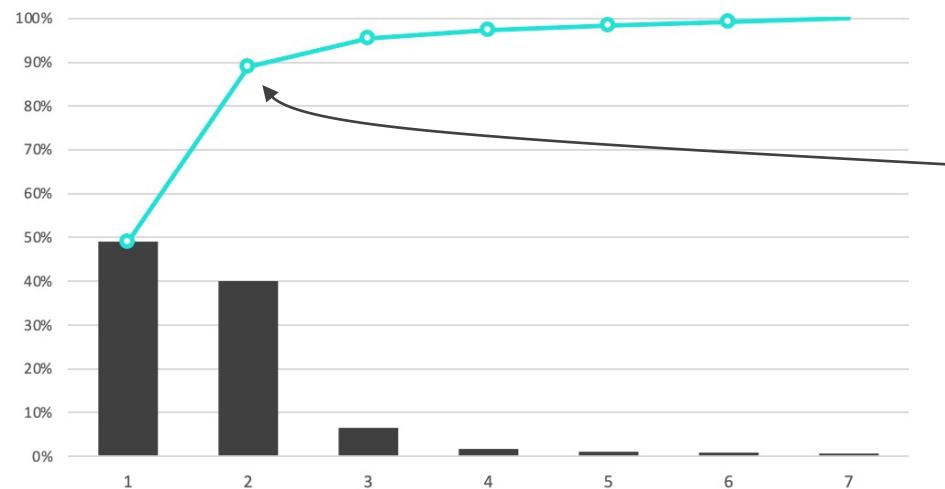
Advanced Techniques

Key Takeaways

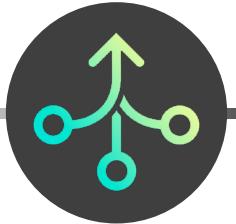


How do you know how many components to keep?

- You can use a **Scree Plot** to visualize the cumulative variance explained by each component, and look for the “elbow” where additional components add relatively little value (*similar to a WSS chart for clustering*)
- The components up to and including the “elbow” become your **principal** components, and others can be disregarded



In this plot, the first **2 components** explain most of the variance in the data, and additional components have minimal impact



# PRINCIPAL COMPONENT ANALYSIS

Dimensionality Reduction  
Basics

Principal Component  
Analysis

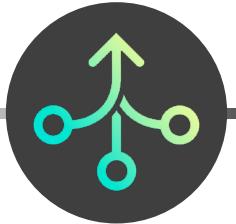
Advanced Techniques

Key Takeaways

studentID	spelling	vocabulary	multiplication	geometry	language	math
1	1.8	2.1	5.4	5.6	2.8	7.8
2	2.9	3.1	6.6	5.7	4.3	8.7
3	4.6	5.7	2.8	1.6	7.3	3.2
4	8.0	7.1	3.5	3.4	10.7	4.9
5	8.1	8.0	1.3	2.2	11.4	2.6
6	8.3	5.6	7.0	9.1	10.0	11.5
7	4.4	4.2	9.1	7.3	6.1	11.6
8	7.2	6.3	8.1	7.8	9.6	11.3
9	4.2	5.1	9.9	9.3	6.6	13.6
10	8.7	10.3	8.2	8.7	13.5	12.0

In this example, defining components for **language** and **math** helped us simplify and better understand the data set

- Using the new components we derived, we can conduct further analysis like predictive modeling, classification, clustering, etc.



# PRINCIPAL COMPONENT ANALYSIS

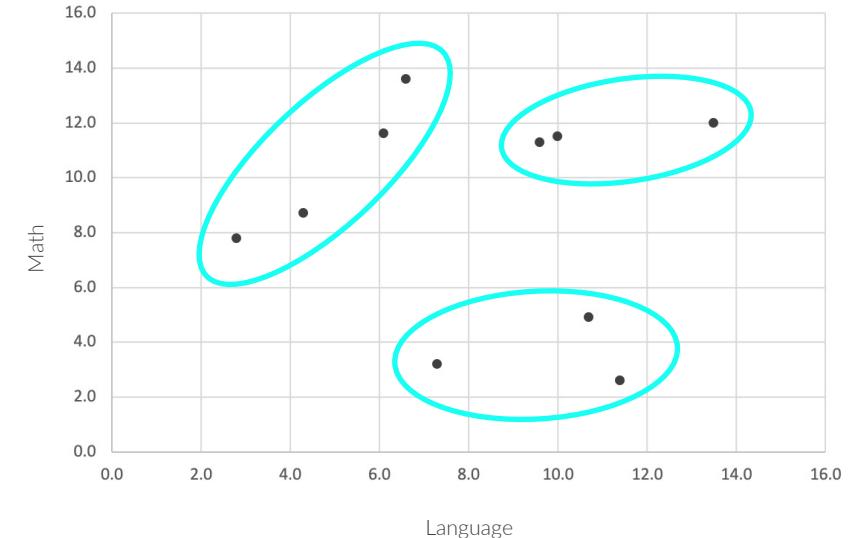
Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways

studentID	language	math
1	2.8	7.8
2	4.3	8.7
3	7.3	3.2
4	10.7	4.9
5	11.4	2.6
6	10.0	11.5
7	6.1	11.6
8	9.6	11.3
9	6.6	13.6
10	13.5	12.0



In this example, defining components for **language** and **math** helped us simplify and better understand the data set

- Using the new components we derived, we can conduct further analysis like predictive modeling, classification, clustering, etc.
- For example, clustering might help us understand student testing patterns (*most skew towards either language or math, while a few excel in both subjects*)



# ADVANCED TECHNIQUES

Dimensionality Reduction  
Basics

Principal Component  
Analysis

Advanced Techniques

Key Takeaways



What about *non-linear* dimensionality reduction?

- Linear dimensionality reduction is the most well-researched and widely used method, and can be implemented across a broad range of BI and ML use cases
- Other, non-linear dimensionality reduction techniques do exist, but they are significantly more specialized and complex in terms of the underlying algorithms and math:
  - Kernel PCA
  - T-Distributed Stochastic Neighbors (T-SNE)
  - Uniform Manifold Approximation & Projection (UMAP)
  - Self-Organizing Maps

# KEY TAKEAWAYS

---



**Dimensionality reduction** is used to reduce the number of columns in a data set while losing as little information as possible

- *This can be used for model optimization, or to identify overarching trends or patterns in a granular dataset*



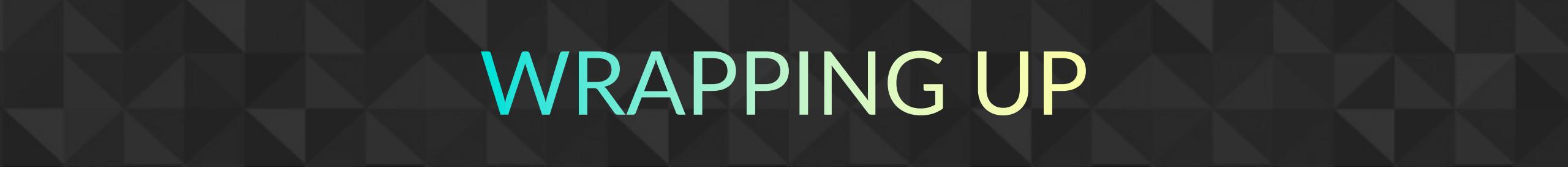
One of the most common dimensionality reduction techniques is known as **Principal Component Analysis** (PCA)

- *PCA essentially fits lines through the observations in a data set to create new dimensions to analyze*



Dimensionality reduction requires **human logic and intuition** to interpret what each component actually means

- *This will help you simplify your data, understand it on a deeper level, and conduct further analysis*



WRAPPING UP

# WRAPPING UP

---



## CONGRATULATIONS!

You now have a solid foundational understanding of unsupervised learning, including techniques like **clustering**, **association mining**, **outlier detection**, and **dimensionality reduction**.

We hope you've enjoyed the entire **Machine Learning for BI** series, and that you find an opportunity to put your skills to good use!



### PART 1

QA & Data Profiling



### PART 2

Classification



### PART 3

Regression & Forecasting



### PART 4

Unsupervised Learning