



Data Science & Deep Learning For Business™



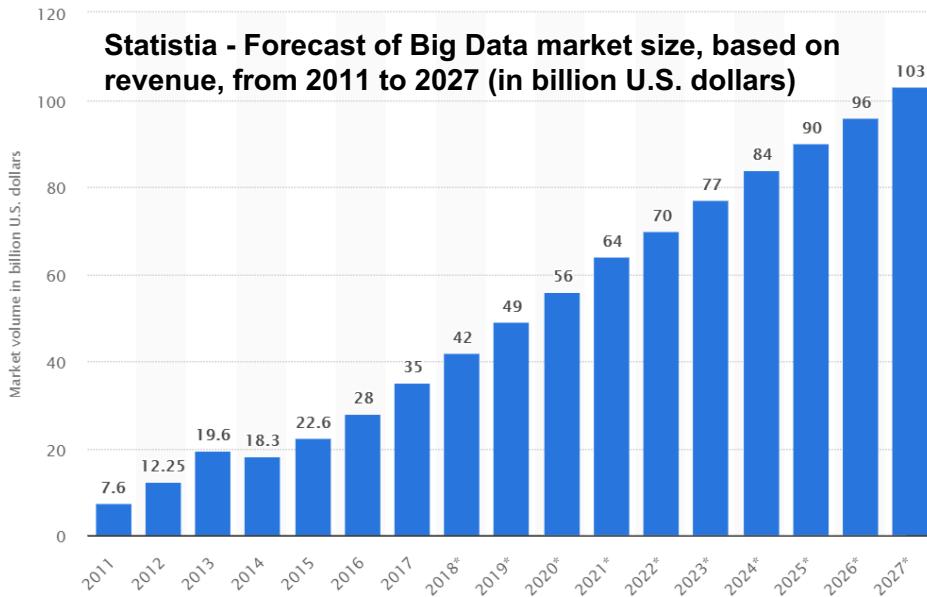
Data Science has been one of the biggest tech buzz word for the last 5-10 years!

- Data Science, Artificial Intelligence, Machine Learning, Big Data.
- Those terms have been bouncing around every tech site and been heavily glamourized (even vilified) by the media!





The Big Data Industry is Growing Rapidly!



IDC Forecasts Revenues for Big Data and Business Analytics Solutions Will Reach \$189.1 Billion This Year with Double-Digit Annual Growth Through 2022



The Demand for Data Scientists is only going up!

Demand for data scientists is booming and will only increase

Fueled by big data and AI, demand for data science skills is growing exponentially, according to job sites. The supply of skilled applicants, however, is growing at a slower pace.

18,000 views | Oct 14, 2019, 07:15am

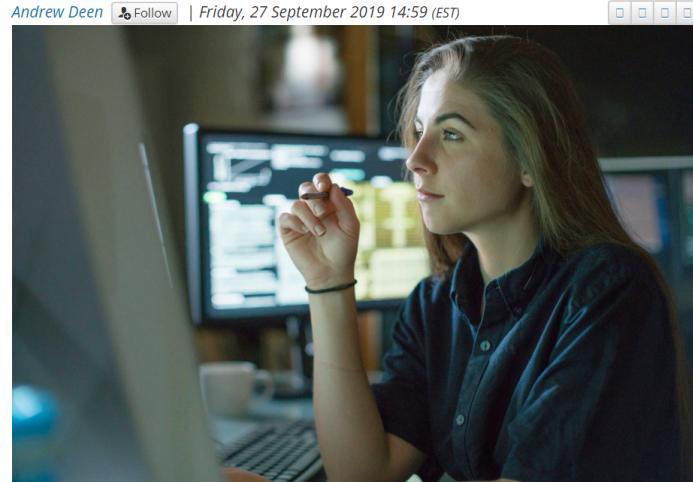
The Birth Of The Data Science Generation



Ike Kavas Forbes Councils Member
Forbes Technology Council COUNCIL POST | Paid Program
Innovation

Why Data Science Will be Among the Most Promising Careers in 2020

Andrew Deen Friday, 27 September 2019 14:59 (EST)





The Problems in the Industry and with Universities

- Too much Data Science hype?
- Confusion over what Data Scientists actually do
- Gatekeeping, Is it only for smart people, or people good in math?
- Is it just a fancy word for Analytics?
- Where do beginners even start?
- Every company can benefit from hiring a data scientist, but how?
- I did my degree and/or masters in Data Science and I still don't understand what to do in the work place

Data Science for the first time!





This course seeks to answer and fill in these Gaps!

You'll learn:

- How Data Science is used and applied across various businesses
- Go through a detailed Data Science learning path (more on that later!)
- How do approach business problems and solve them using Data Science Techniques
- You'll gain perspective on how Data Scientists fit into a tech world filled with Data Engineers, Data Analysts, Business Analysts
- How to apply the latest techniques in Deep Learning and solve some of our Business problems with 20 Case Studies



My In-Depth Data Science Learning Path



My Data Science Learn Path

1. Python Programming Introduction
2. Pandas, Numpy – Understand data frames and how to manipulate and wrangle data
3. Visualizations with Matplotlib, Seaborn, Plotly and Mapbox
4. Statistics, Probability and Hypothesis Testing
5. Machine Learning with Scikit-learn – Linear Regressions, Logistic Regressions, SVMs, Naïve Bayes, Random Forests etc
6. Deep Learning Neural Networks
7. Unsupervised Learning – Clustering and Recommendation Systems
8. Big Data using PySpark
9. Deployment into production using Cloud Services (create an ML API)



Carefully Selected Real World Case Studies

The case studies used in this course were carefully selected and chosen specifically because:

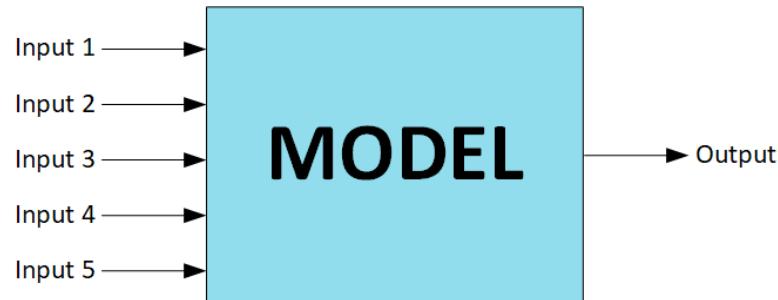
- The encompass some of the most common business problems and can be easily applied to your own business
- Taken from a wide variety of industries and separated into 7 sections:
 1. Predictive Modeling & Classifiers
 2. Data Science in Marketing
 3. Data Science in Retail – Clustering and Recommendation Systems
 4. Time Series Forecasting
 5. Natural Language Processing
 6. Big Data Projects with PySpark
 7. Deployment into Production



Case Studies Section 1 – Predictive Modeling & Classifiers

Predictive modeling encompasses using data inputs to produce an output. In this section we use a range of Machine Learning and Deep Learning Techniques.

1. Figuring Out Which Employees May Quit (Retention Analysis)
2. Figuring Out Which Customers May Leave (Churn Analysis)
3. Who do we target for Donations?
4. Predicting Insurance Premiums
5. Predicting Airbnb Prices
6. Detecting Credit Card Fraud



Case Studies Section 2 – Data Science in Marketing



In this section we solve a number of Marketing problems using Data Science and statistical techniques. We learn about the marketing process and Key Performance Indicators (KPIs) that drive Marketing teams.

1. Analyzing Conversion Rates of Marketing Campaigns
2. Predicting Engagement - What drives ad performance?
3. A/B Testing (Optimizing Ads)
4. Who are you best customers? & Customer Lifetime Values (CLV)





Case Studies Section 3 –Data Science in Retail

In this section we solve a number of Retail problems using Data Science and statistical techniques. We learn about how to track product and customer metrics, customer clustering and product recommendation,

1. Product Analytics (Exploratory Data Analysis Techniques)
2. Clustering Customer Data from Travel Agency
3. Product Recommendation Systems - Ecommerce Store Items
4. Movie Recommendation System using LiteFM

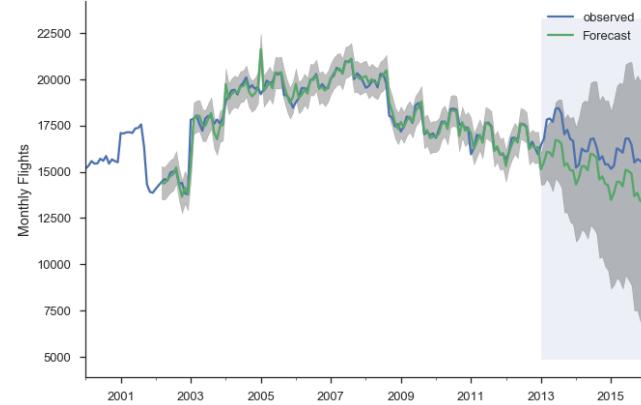




Case Studies Section 4 –Time Series Forecasting

In this section we solve a number of business problems where Time Series Data is important. This type of data can be used to solves issues in:

1. Sales/Demand Forecasting for a Store
2. Stock Trading using Re-Enforcement Learning

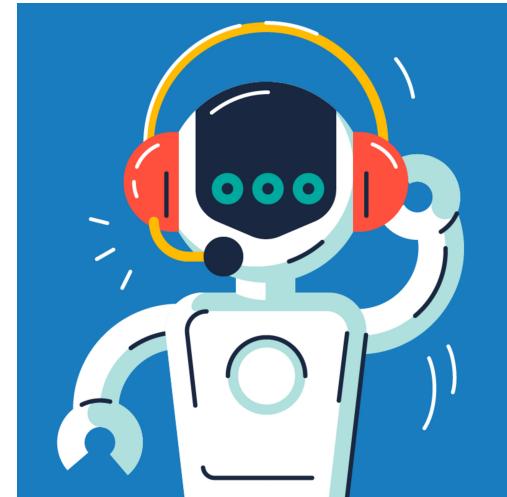


Case Studies Section 5 – Natural Language Processing



In this section we solve a number of business problems where text data becomes overwhelming for people to analyze. Our case studies involve.

1. Summarizing Reviews
2. Detecting Sentiment in text
3. Spam Filters



Case Studies Section 6 – Big Data with PySpark



In this section we look at a few real world projects using one of the best **Big Data** frameworks, Spark (using **PySpark**).

1. News Headline Classification





Case Studies Section 7 – Deployment Into Production

Here we do a simple project where we use **AWS** to create and deploy a real Machine Learning **API** that can be assessed anywhere in the world.





Hi

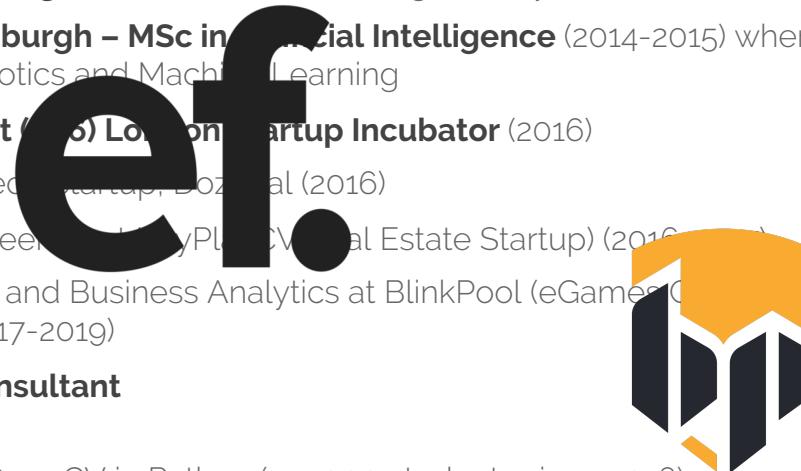
I'm Rajeev Ratan





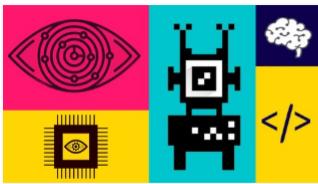
About me

- Radio Frequency Engineer in Trinidad & Tobago for 8 years
- University of Edinburgh – MSc in **Artificial Intelligence** (2014-2015) where I specialized in Robotics and Machine Learning
- Entrepreneur First (ef) London Startup Incubator (2016)
 - CTO at Edtech Startup, Bozzatool (2016)
 - VP of Engineering at KeyPlus CV (Real Estate Startup) (2016-2017)
 - Head of CV and Business Analytics at BlinkPool (eGaming Company Startup) (2017-2019)
- Data Scientist Consultant
- Udemy Courses
 1. Mastering OpenCV in Python (~15,000 students since 2016)
 2. Deep Learning Computer Vision™ CNN, OpenCV, YOLO, SSD & GANs (~5,000 students since 2018)





My Udemy Courses



Master Computer Vision™ OpenCV4 in Python with Deep Learning

116 lectures • 11 hours • All Levels

Learn OpenCV4, Dlib, Keras, TensorFlow & Caffe while completing over 21 **projects** such as classifiers, detectors & more! | By Rajeev Ratan

£10.99

£19.99

★★★★★ 4.1
(2,738 ratings)



I loved this course. Very detailed explanations and in depth with the latest technology and ideas. Very thoughtful help with ideas of implementation of course materials for real life projects. Thanks for a great insightful course. I'm looking forward to using the learnt material. Thank you.



Deep Learning Computer Vision™ CNN, OpenCV, YOLO, SSD & GANs

176 lectures • 14.5 hours • Intermediate

Go from beginner to Expert in using Deep Learning for **Computer Vision** (Keras & Python) completing 28 Real World Projects | By Rajeev Ratan

£12.64

£199.99

★★★★★ 4.3
(723 ratings)



It is really straightforward. Nice basics explaining with links for extension of topic knowledge. The exercising is effective and cool. I am really appreciate there are NOT boring parts.



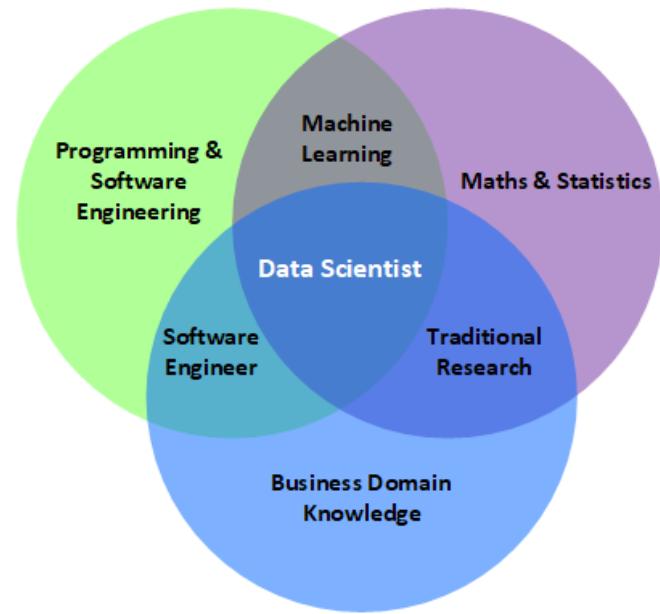
I'm amazed at the possibilities. Very educational, learning more than what I ever thought was possible. Now, being able to actually use it in a practical purpose is intriguing... much more to learn & apply.



What you'll be able to do after

Understand all aspects that make one a complete Data Scientist

- The software programming chops to mess with Data
- The Analytical and Statistical skills to make sense of Data
- All the Machine Learning Theory needed for Data Science
- Real World Understanding of Data and how to understand the business domain to better solve problems





Requirements

- **Some familiarity with programming**
 - Familiarity with any language helps, especially Python but it is **NOT** a prerequisite.
- **High School Level Math**
- **Little to no Data Science & Statistical or Machine Learning knowledge**
- **Interest and Passion in solving problems with Data**



What you're getting

- ~800+ Slides
- ~15 hours of video including
- Comprehensive Data Science & Deep Learning theoretical and practical learning path
- ~50+ ipython notebooks
- 20 Amazing Real World Case Studies



Course Outline

- 1. Course Introduction**
- 2. Python, Pandas and Visualizations**
- 3. Statistics, Machine Learning**
- 4. Deep Learning in Detail**
- 5. Predictive Modeling & Classifiers – 6 Case Studies**
- 6. Data Science in Marketing – 4 Case Studies**
- 7. Data Science in Retail – Clustering & Recommendation Systems – 3 Case Studies**
- 8. Time Series Forecasting – 2 Case Studies**
- 9. Natural Language Processing – 3 Case Studies**
- 10. Big Data Projects with PySpark – 2 Case Studies**
- 11. Deployment into Production**

Course Approach Options



Beginners – Do Everything!

- 1. Course Introduction**
- 2. Python, Pandas and Visualizations**
- 3. Statistics, Machine Learning**
- 4. Deep Learning in Detail**
- 5. Predictive Modeling & Classifiers – 6 Case Studies**
- 6. Data Science in Marketing – 4 Case Studies**
- 7. Data Science in Retail – Clustering & Recommendation Systems – 3 Case Studies**
- 8. Time Series Forecasting – 2 Case Studies**
- 9. Natural Language Processing – 3 Case Studies**
- 10. Big Data Projects with PySpark – 2 Case Studies**
- 11. Deployment into Production**



Did a few courses online or have a degree related to Data Science

- 1. Course Introduction**
- 2. Python, Pandas and Visualizations**
- 3. Statistics, Machine Learning**
- 4. Deep Learning in Detail**
- 5. Predictive Modeling & Classifiers – 6 Case Studies**
- 6. Data Science in Marketing – 4 Case Studies**
- 7. Data Science in Retail – Clustering & Recommendation Systems – 3 Case Studies**
- 8. Time Series Forecasting – 2 Case Studies**
- 9. Natural Language Processing – 3 Case Studies**
- 10. Big Data Projects with PySpark – 2 Case Studies**
- 11. Deployment into Production**



Junior Data Scientists – Just Look at the Case Studies in Any Order!

- 1. Course Introduction**
- 2. Python, Pandas and Visualizations**
- 3. Statistics, Machine Learning**
- 4. Deep Learning in Detail**
- 5. Predictive Modeling & Classifiers – 6 Case Studies**
- 6. Data Science in Marketing – 4 Case Studies**
- 7. Data Science in Retail – Clustering & Recommendation Systems – 3 Case Studies**
- 8. Time Series Forecasting – 2 Case Studies**
- 9. Natural Language Processing – 3 Case Studies**
- 10. Big Data Projects with PySpark – 2 Case Studies**
- 11. Deployment into Production**

Why Data is the New Oil and What Most Businesses are Doing wrong



Has this ever happened to you?

- You're thinking about something, maybe it's the new printer you wanted, or a skiing trip you were planning. Or perhaps thinking about starting a gym.
- Then BAM! You see an online Ad for the exact thing you wanted.





How do those online giants like Google and Facebook know you so well?

- Unfortunately, it's not magic
- It's DATA
- And it's everywhere (including yours)
- Let's take a look at 5 super interesting examples of how Data Driven companies are changing the business landscape





Online Advertising

- Many people still ask, how does Google and Facebook make money when everything is free?
- The answer is Advertising. These tech giants have become so good at targeting ads!
- Using their users data, both companies can target ads

Detailed Targeting INCLUDE people who match at least ONE of the following ⓘ

Add demographics, interests or behaviors | Suggestions | Browse

Connection

How Advanced Options

Placement

Income

Demographics

Financial

Income

Household income: top 10% of ZIP codes (US)

Household income: top 10%-25% of ZIP codes (US)

Household income: top 25%-50% of ZIP codes (US)

Household income: top 5% of ZIP codes (US)

Household income bracket	Checkboxes
top 10% of ZIP codes (US)	<input type="checkbox"/>
top 10%-25% of ZIP codes (US)	<input type="checkbox"/>
top 25%-50% of ZIP codes (US)	<input type="checkbox"/>
top 5% of ZIP codes (US)	<input type="checkbox"/>



How Targeted Are Online Ads?

- Let's say if you wanted to target grand parents, who live in one city who enjoy outdoor activities and barbeques and recently purchased a Mercedes Benz.....well you could!
- *"If you want to tailor a Facebook ad to a single user out of its universe of 2.2 billion, you could"*



Let's look at Uber

- Estimating that “Your driver will be in here in 6 minutes.” to estimating fares, showing up surge prices and heat maps to the drivers on where to position themselves within the city.
- Uber relies heavily on data to provide and optimize their services





Netflix's Recommendations

NETFLIX Home TV Shows Movies Recently Added My List

Because you watched Love, Death & Robots ›

Top Picks for Rajeev

Because you watched Better Call Saul

34



Amazon's Recommendations

Inspired by your shopping trends



Frequently bought together



Total price: £477.42

[Add all three to Basket](#)

i These items are dispatched from and sold by different sellers. [Show details](#)

This item: Canon EF 70-300 mm f/4-5.6 IS II USM Lens - Black £449.00

JJC Reversible Lens Hood Shade for Canon EF 70-300mm f/4-5.6 IS II USM Replaces Canon Lens Hood ET... £14.99

Hoya 67mm UV(C) Digital HMC Screw-in Filter,Y5UVC067 £13.43



YouTube GB

Search

7

Home

Trending

Subscriptions

Library

Recommended



JVNA Live Ep:008 | Alchemy | Melodic Dubstep, Future...
JVNA
53K views • 4 weeks ago



Relaxing Music Mix | BEAUTIFUL PIANO
Epic Music World
38M views • 3 years ago



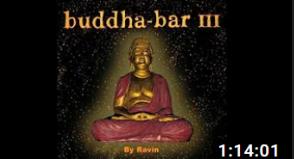
Soca 2019 / Soca 2020- Duty Free 2019 (The Best...)
REDYMIX DMTV
15K views • 1 month ago



MOTHERBOARD - Nuclear Fusion Energy: The Race to Create a Star on...
Motherboard
3.9M views • 2 years ago



Paramore: Playing God [OFFICIAL VIDEO]
Fueled By Ramen
68M views • 8 years ago



Buddha-Bar III - CD1
Buddha-Bar Official Channel
186K views • 5 months ago



YEAH yeah yeah yeah yeah
3:46



ピアノ曲勉強用
1:05:09



MACHINE LEARNING
Living in the Age of AI 41:17



Banking

- For better or worse, banks are harnessing their data to determine which customers to give loans to, which they should extend their credit line, and even who might file for bankruptcy





The Ubiquity of Data Opportunities

- Automated Recruitment of Employees
- Crypto, Forex & Stock Price Predictions
- Health Analytics – Disease Prediction, finding cures etc.
- Computer Vision – Understanding what is being seen to build things like self driving cars and facial recognition
- Agriculture
- Manufacturing
- Creating Art and Music
- Self Driving cars and Robots
- Chat Bots
- **And thousands more!** There is no shortage on areas we can apply Data Science

Here's how data science helped Zoomcar capture 75% of Indian market

*"Data lake and models built on Kafka helped us achieve enhanced customer experience at the best possible price,"
Arpit Agarwal, Director, Decision Science, Zoomcar, says.*

Nikhar Aggarwal | ETCIO | October 31, 2019, 09:23 IST

What Are Businesses Are Doing Wrong?

- Unfortunately, the data revolution hasn't been sparked into all businesses and industries.
- Why? Lack of competition so they sit comfy until a young fast moving startup gets their Series A funding and can compete.
- What are some common mistakes companies make?





Mistakes Businesses Make with Data

1. Not recording their data
2. Recording their data, but doing nothing substantial with it and then discarding it because of storage costs
3. Mistaking Business Analytic Reports as a substitute for data science.
4. Not Trusting the data and relying on their intuition
5. Relying too much on statistically flawed analysis

NOT DATA DRIVEN

Financial Metrics (2005-2009)





Data = Value = Better Decisions = More Profits!

Data is the new oil!



Defining Business Problems for Analytic Thinking & Data Driven Decision Making



Analytical Thinking Defined

Analytical Thinking is the ability to:

- identify and define problems
- extract key information from data and
- develop workable solutions for the problems identified
- in order to test and verify the cause of the problem and develop solutions to resolve the problems identified.”



Data-Analytic Thinking

- For Businesses who utilize data, Data Analysis and Data Driven Decision making is so critical that it **can almost blind them**.
- Having a deep understanding of the business problem, the domain and how the data is generated is **critical**. For example:
 - Image you built a model that was 97% accurate in diseases detection. Pretty good? Perhaps, but suppose it missed 3% of patients who actually had it. A model with 90% accuracy that never failed to detect a disease is better. False Positives here are better than missing when a person had a disease.
- Many times, companies can hire the best data scientists who are great technically, but they miss the big picture and lead to bad decisions.



Data-Analytic Thinking

- Data should be improving the decision making, not misleading or adding to confusion.

- A great Data Scientists needs to Understand
 - The business domain they're working in
 - The goal of his model (i.e. the problem they're solving)
 - The statistical weaknesses of their work
 - How to communicate this effectively to his superiors

10 Data Science Projects every Business should do!



Data for your business is growing all around you

- From customers data, to industry data.
- We live in age where there is almost no such thing as '**no data**'
- Data is produced by our own business systems, to external data from social media, Google Trends etc.
- Data can be gathered with every click!
- So what do we do with this data to **Add Value to businesses?**





10 Data Science Projects that can be applied to most businesses!

- Analytics Projects
 1. Determine your best (most profitable) customers
 2. Most Profitable items and item categories
 3. Customer Life Time Value
 4. Season Trends and Forecasting
- Machine Learning Prediction Projects
 5. Determine Customers likely to leave your business (retention)
 6. Customer Segments
 7. Recommendation Systems
 8. AB Testing Analysis of Ads or many other changes (UI, Logo etc.)
 9. Fraud Detection
 10. Natural Language Processing of Social Media Sentiment



1. Determine your best (most profitable) customers

Many people mistakenly think their best customers are the ones who spend the most. However, there are many exceptions and other metrics we can use to determine best or most valuable customers.

- In gambling , customers depositing the most are often the most skilled gamblers and can actually be profiting thus hurting your bottom line
- Retailers like Amazon may have customers who spend hundreds monthly, however, due to relaxed return policies, these people an be returning products regularly basically renting expensive items for 'free' and forcing you to sell your new stock as B-Stock or Used.
- Outside of profit from a customer, one can look at customers who garners the most referrals or perhaps has continuously increased their spending over time.



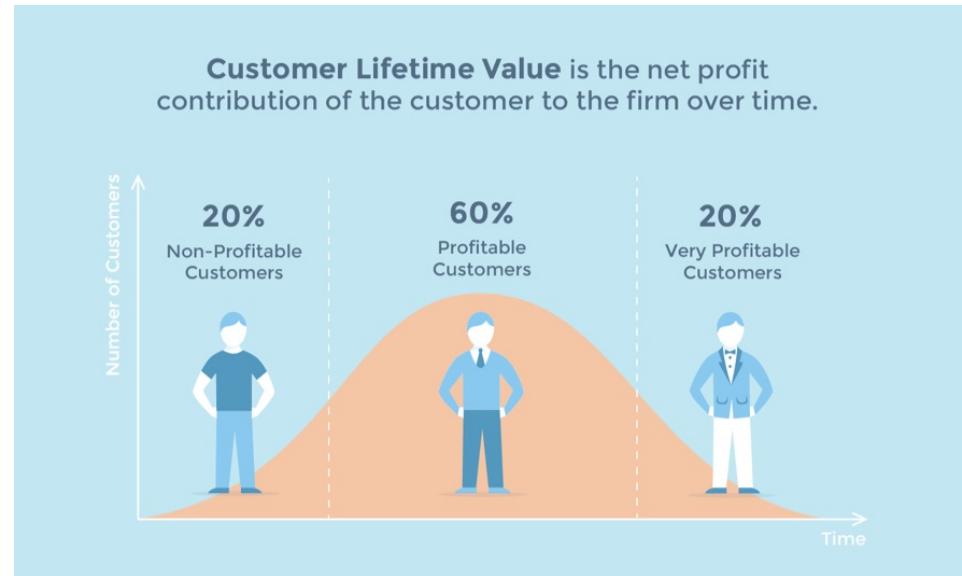
2. Most Profitable items and item categories

- It's very useful to understand what items are your number seller. However, many times analysts make simple mistakes that can be mislead executives.
- For instance, one supermarket thought one particular item was their top seller for years.
- However, when I dug in, I noticed a brand of beer that was sold in 3 variations - cans, medium and large glass bottles, was actually their top seller.
- Additionally, categorizing items (a tedious task if done manually) can shed more light on what customers want.



3. Customer Life Time Value

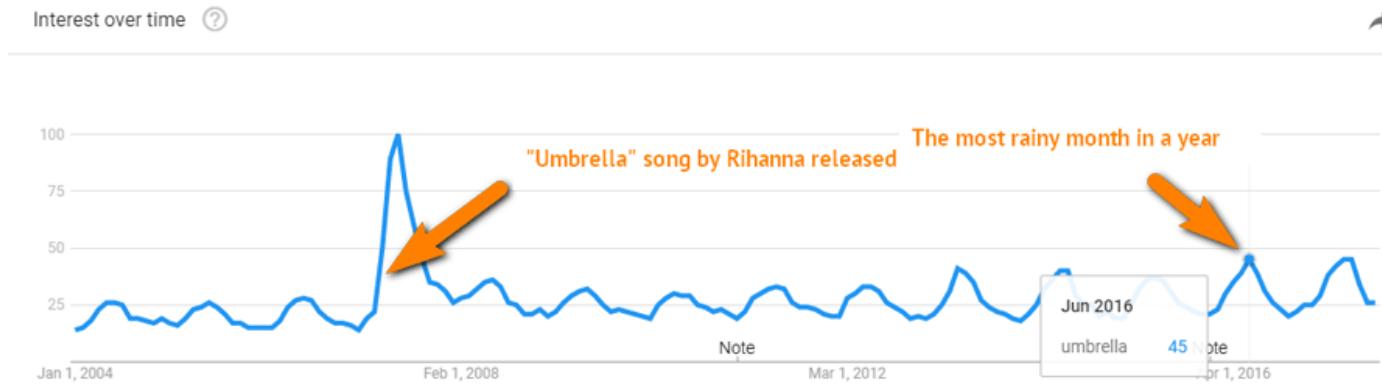
- A business that doesn't know their customer life time value will be unable to gauge how much to spend on marketing or customer retention.





4. Season Trends and Forecasting

- Understanding the seasonality of your business and the purchasing of different items can allow you to quickly position your business to profit and prepare for trends





5. Determine Customers likely to leave your business (retention)

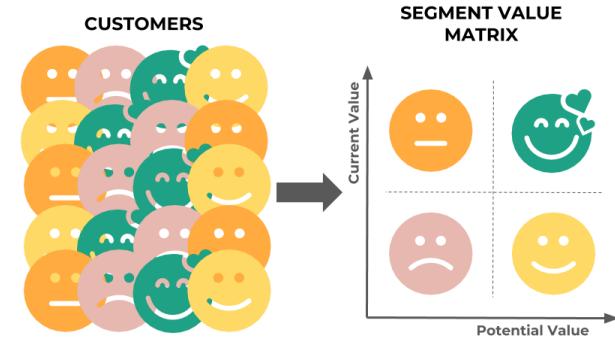
- Retention is an exercise often proposed and performed by businesses when they try prevent customers from dropping their service and/or going to a competitor.
- Imagine a retention strategy that involved gifts and a personal call to your customers – and you had 10,000 customers!
- That's going to be a waste of time and money
- It'd make far more sense to create a model to predict which customers were mostly likely going to leave and target those customers.





6. Customer Segments

- Who are your customers really?
- Performing advanced cluster analysis can reveal insightful revelations about your customer base. For instance a supermarket can have:
 - Middle Aged Women
 - Aged retired persons
 - School children





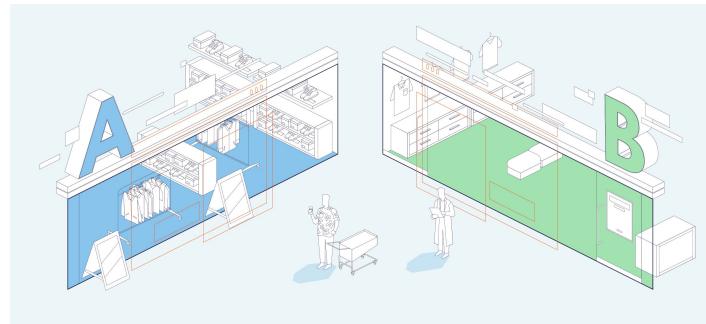
7. Recommendation Systems

- This is especially useful if you have an ecommerce site with a large variety of items.
- Creating a good Recommendation System can assist customers in discovering items they'd have a hard time finding on their own



8. A/B Testing

- A/B Testing is incredibly useful when testing our new features, designs, layouts etc.
- However, understanding the results of your test and even designing the test it self isn't as trivial as you'd think and Statistical knowledge is needed to make sense from the 'noise' you've observed





9. Fraud Detection

According to Wikipedia,

"Fraud is a billion-dollar business and it is increasing every year. The PwC global economic crime survey of 2016 suggests that more than one in three (36%) of organizations experienced economic crime."

- **Modern businesses need to be smart in detecting fraud that could potential hurt their business.**
- **Chargebacks with stolen credit cards, identity theft, affiliate fraud and many others are things we can use Machine Learning Models to detect.**

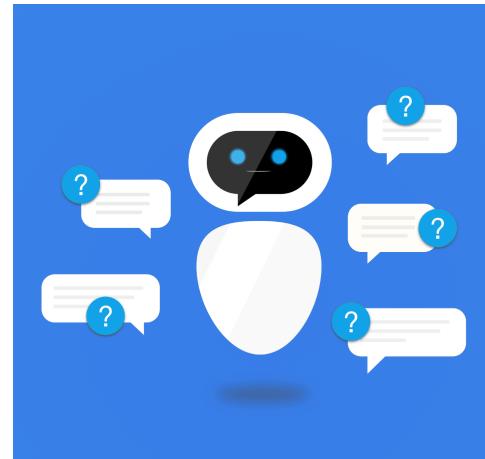




10. Natural Language Processing (NLP)

Natural Language Processing has numerous applications in modern businesses. Some examples of NLP projects that can be applied across many businesses are:

- Summarizing survey data
- Summarizing reviews
- Chat bots
- Social Media sentiment analysis



Making Sense of Buzz Words, Data Science, Big Data, Machine & Deep Learning



There are so many confusing buzz words in this industry.
Where does one begin?

- This industry is notorious for hyping and using words incorrectly, **re:marketing** speak
- However, you can decipher their meaning quite easily and knowing how to do so can help you understand roles in and tasks far better



Buzz Words

1. Deep Learning
2. Big Data
3. Machine Learning
4. Artificial Intelligence
5. Heuristic(s)
6. Neural Network
7. Algorithm
8. Modeling
9. Data Mining
10. Predictive Analysis
11. Cloud Computing/Distributed Computing
12. Data Science



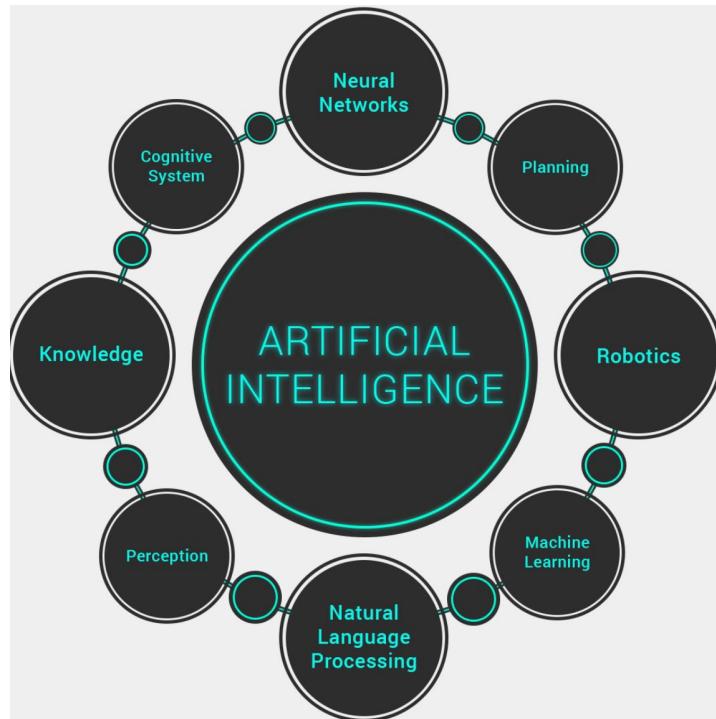
Buzz Words Explained

1. **Deep Learning** – A deep (re:complicated) Neural Network that learns to predict accurately after being trained on large sets of example data.
2. **Big Data** – Any dataset that can't really be stored and manipulated on one machine
3. **Machine Learning** – Wiki definition: *“Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence”*



Buzz Words Explained

4. **Artificial Intelligence** – is the simulation of human intelligence processes by machines. It is the broad field that encompasses and overlaps a lot with Data Science, but also includes robotics and other areas
5. **Heuristic(s)** - a heuristic is an educated approximation used when classic methods are too slow or fail to come to a definitive answer.
6. **Neural Network** – A machine learning algorithm, Deep Learning is essentially the same Neural Networks but Deep Learning tends to signify more complex models with more layers (the 'deep' in deeper)





Buzz Words Explained

7. **Algorithm** - An algorithm is a set of instructions that explain (in data science contexts, explain to computers) how to do something.
8. **Modeling** – A statistical representation of our data and thus can be used to make predictions
9. **Data Mining** - The implication with the term data mining is that all the discovery is driven by a person, which is one slight contrast between machine learning and data mining, as many of the algorithms or methods are similar between the two.



Buzz Words Explained

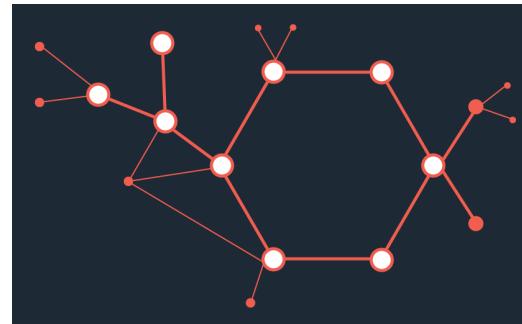
10. **Predictive Analysis** – a type of analysis that's meant to predict something
11. **Cloud Computing/Distributed Computing** – Using remote servers provided typically by Amazon, Google, Microsoft (many others too) to host, run processes on their machines. It's extremely useful as you can gain access to power
12. **Data Science** – Wiki definition: *Data science is a multi-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data. Data science is the same concept as data mining and big data: "use the most powerful hardware, the most powerful programming systems, and the most efficient algorithms to solve problems"*

How Deep Learning is Changing Everything!



The Power of Deep Learning

- Machine Learning has been around for decades with many of the established algorithms around since the 1960s.
- What brought on the Data Science revolution was:
 - Increasing Computer Power
 - Cheap Data Storage
 - Developed of software and tools that made it far more accessible
- Around ~2010 it was seen that typical Machine Learning models could only learn and do so much and some problems were just too difficult



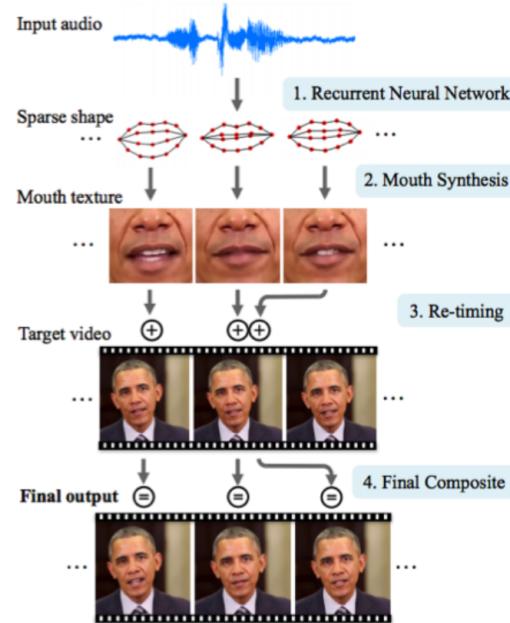
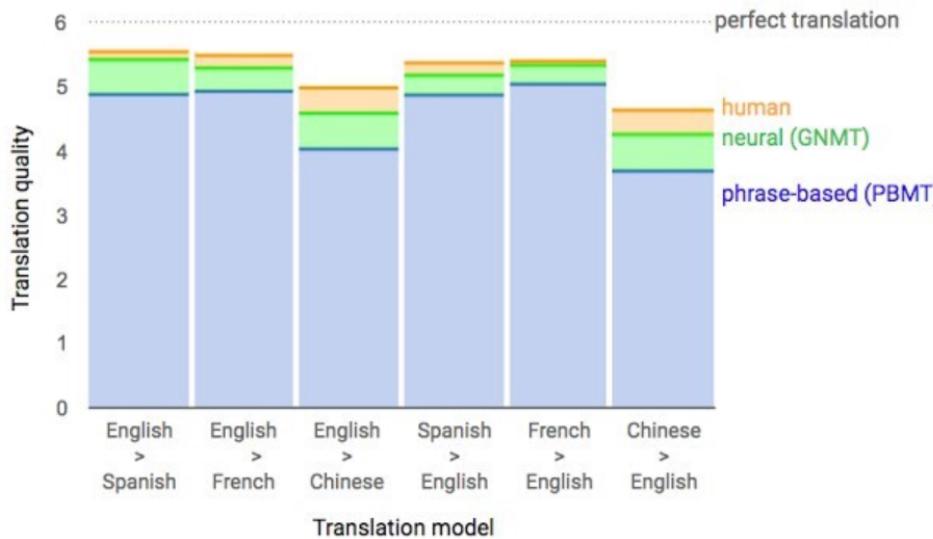


The Power of Deep Learning

- **Machine Learning algorithms lacked the complexity** to learn non-linear complex relationships
- **Deep Learning** solved this and has ushered in a new revolution in Data Science and Artificial Intelligence.
- It took our models from “that's pretty good” to “**that's scary good, better than what most humans can do**”. Deep Learning achieved far higher accuracy in a number of Computer Vision and NLP tasks and allowed Machine Learning experts to tackle even more difficult problems, problems once thought to be too challenging



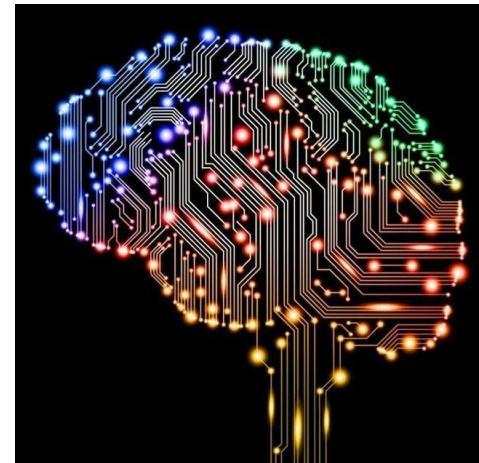
The Power of Deep Learning





How Does Deep Learning Work?

- We will go into extensive detail into Deep Learning, however for now, consider it Machine Learning on steroids.
- Deep Learning **achieved better than human performance** in isolated tasks. **But it came at a price.**
- **Slow training** time, demanding performance requirements (GPUs and TPUs instead of CPUs) and **it's need for vast amounts of data** are its **weaknesses**.
- Deep Learning achieves amazing results by **learning complex patterns, relationships** and is extremely versatile allowing it to be customizable and adaptable for a wide variety of applications





More Data Needed!

“The analogy to deep learning is that the rocket engine is the deep learning models and the fuel is the huge amounts of data we can feed to these algorithms.”

– Andrew Ng (source: [Wired](#))

The Roles of Data Analyst, Data Engineer & Data Scientists



Data Analysts

- Data analysts translate **raw numbers** into **comprehensible reports and/or insights**
- As we know, all businesses potentially collects data, from sales figures, market research, logistics, or transportation costs.
- Data analysts take that data and use it to help companies make better **business decisions**.
- They often can create simple **visual illustrations and charts** to convey the meaning in their data
- This could mean figuring out trends in cash flow, how to schedule employees, optimal pricing and more.



Data Engineers

Data engineers connect all parts of the data ecosystem within a company or institution and make it accessible.

- **Accessing, collecting, auditing, and cleaning data** from applications and systems into a usable state (ETL Pipelines)
- Creating, choosing and maintaining efficient **databases**
- Building **data pipelines** taking data from one system to next
- Monitoring and managing all the data systems (**scalability, security, DevOps**)
- **Deploying** Data Scientists' models
- They work with production and understand **Big Data** concepts

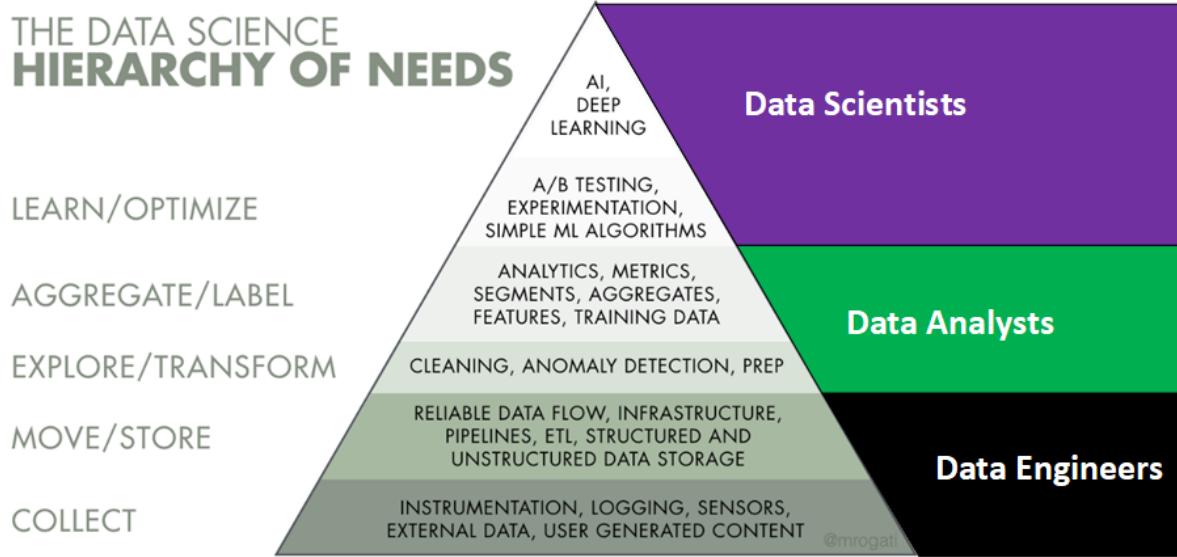


Data Scientists

- Clean up and pre-process (data wrangling) data from various systems in usable data for use in their algorithms
- Using Machine Learning and Deep Learning to build better prediction models
- Evaluating statistical models and results to determine the validity of analyses.
- Testing and continuously improving the accuracy of machine learning models.
- Much like Data Analysts, Data Scientists building data visualizations to summarize the conclusion of an advanced analysis.



The Data Science Hierarchy of Needs





Data Analysts vs. Data Scientists



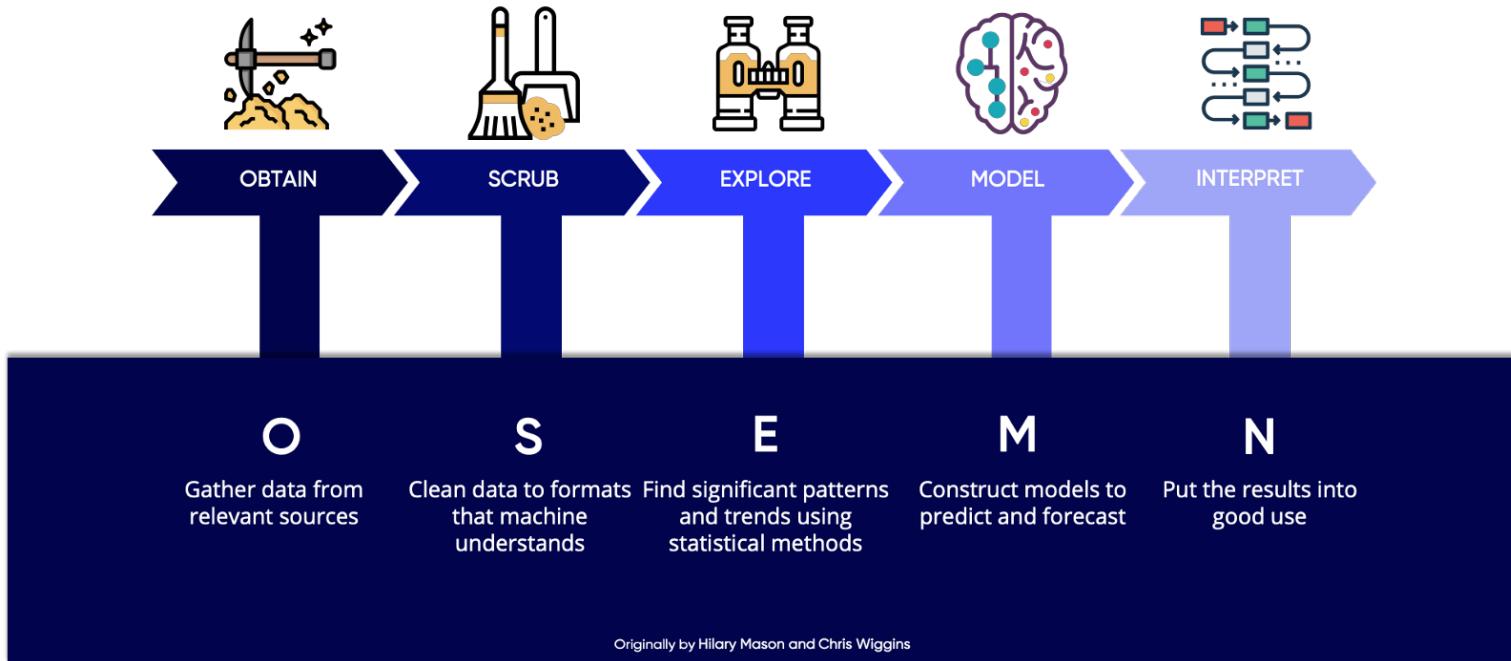
**No Machine Learning doesn't
need to be an ace programmer**

**Uses Machine Learning and needs
to be good at programming**

How Data Scientists Approach Problems



Data Science Process



Originally by Hillary Mason and Chris Wiggins



Obtain – Finding Data

- In every online course (including mine) you'll be given datasets to work with.
- However, in the real world **data doesn't come by so easily** (good data that is)
- In fact, obtaining data can be an arduous task, why? It can involve:
 - Complex SQL queries with deep understanding of how the data is being generated
 - Web Scrapping
 - Or even manually building a dataset



Data Wrangling

- A Data Scientists will spend almost **70%** of his time on Data Wrangling/Munching/Pre-processing/Feature Engineering.
- It's the **ugly** side of Data Science, but it is the **MOST important** step.
- Getting your data ready to be **inputted** into a Machine Learning algorithm, in a way that makes sense, is the hard part.
- This is why all those cloud based ML algorithm tools will never actually replace real data science work.



Exploratory Analysis

- Also called Exploratory Data Analysis, is an approach to analyzing data sets to summarize their main characteristics, often by using visual methods.
- These initial investigations are extremely important in discovering patterns, detect anomalies and is a good sanity check before running your data though an ML model





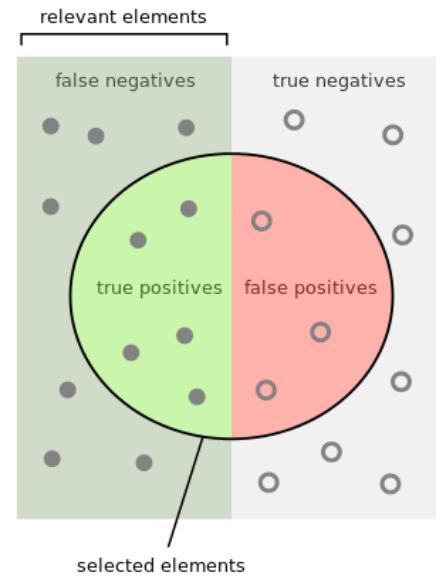
Model

- Modeling data is now the process where we begin to use our Machine Learning algorithms to create a model.
- The data is typically split into two (2) parts:
 - **Training Data Set** – This is the data we use to create our model.
 - **Test Data Set** – This is the data (unseen by our ML algorithm) used to test and validate our model's performance.



Interpret

- Having trained our model we need to understand its performance **strengths** and **weaknesses**.
- This step isn't too difficult, however it does require a **proper understanding of the domain** you're working in.
- Detecting Fraud or Diseases requires a model that rarely misses, and we will be ok with False Positives. Conversely, in a situation where False Positives are costly, you make want to adjust the model accordingly



How many selected items are relevant?

$$\text{Precision} = \frac{\text{green}}{\text{green} + \text{red}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{green}}{\text{green} + \text{black}}$$



Deploy into Production

- Deployment or Integration is something done more often by Data Engineers and Software Engineers. However, in recent times a Data Scientists portfolio of tasks often includes this process.
- In production, it is important to understand how his model scales, and its computation time.
- Additionally, real world data can often be different to the data used when training and it is important to monitor behavior and adjust the model accordingly.





Communication is Vital

“Which skill is more important for a data scientist: the ability to use the most sophisticated deep learning models, or the ability to make good PowerPoint slides?”

What is Python and Why do Data Scientists Love it?

Python



<https://www.python.org/>

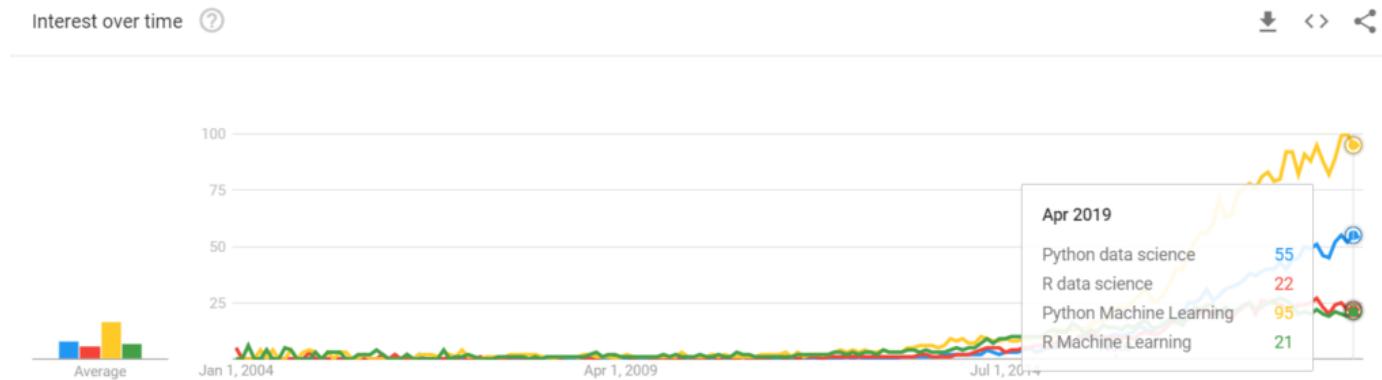


- Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes **code readability** with its notable use of significant whitespace
- Python is an interpreted, high-level, general-purpose programming language.
 - **Interpreted** – Instructions are execute directly without being compiled into machine-language instructions. Compiled languages unlike interpreted languages, are faster and give the developer more control over memory management and hardware resources.
 - **High-level** – allowing us to perform complex tasks easily and efficiently



Why Python for Data Science

- Python competes with many languages in the Data Science world, most notably R and to a much lesser degree MATLAB, JAVA and C++.





Why does Python Lead the Pack?

- It is the only general-purpose programming language that comes with a solid ecosystem of scientific computing libraries.
- Supports a number of popular Machine Learning, Statistical and Numerical Packages (Pandas, Numpy, Scikit-learn, TensorFlow, Cython)
- Supports easy to use iPython Notebooks, especially handy for view Data Science work.
- Quite easy to get started
- As a general purpose language it allows more flexibility such as building webservers, APIs and a plethora of other useful programming libraries.

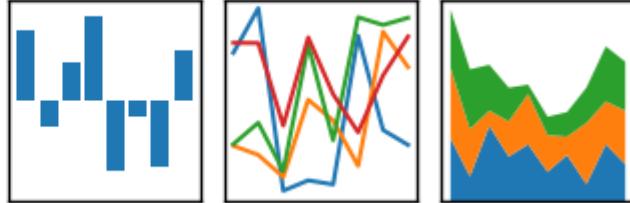
A Crash Course in Python

Introduction to Pandas

Pandas

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



- Pandas is Python library that allows high-performance, easy-to-use **data structures and data analysis tools**.
- It's an invaluable tool for **data scientist and analysts**
- The name stems from the term 'panel data' an econometrics term for multidimensional structured data sets.



Understanding Pandas

- What can Pandas do?
- Pandas allows us to manipulate data frames (think of excel sheets or tables of data) and produce useful data outputs



Simple Example – Imagine a 1000s of rows of data like the table below

Name	DOB	Subject	Exam Scores
lenord, robin	2001-02-22	Mathematics	71
lenord, robin		Physics	64
:	:	:	:
khan, imran	2002-08-19	Spanish	76

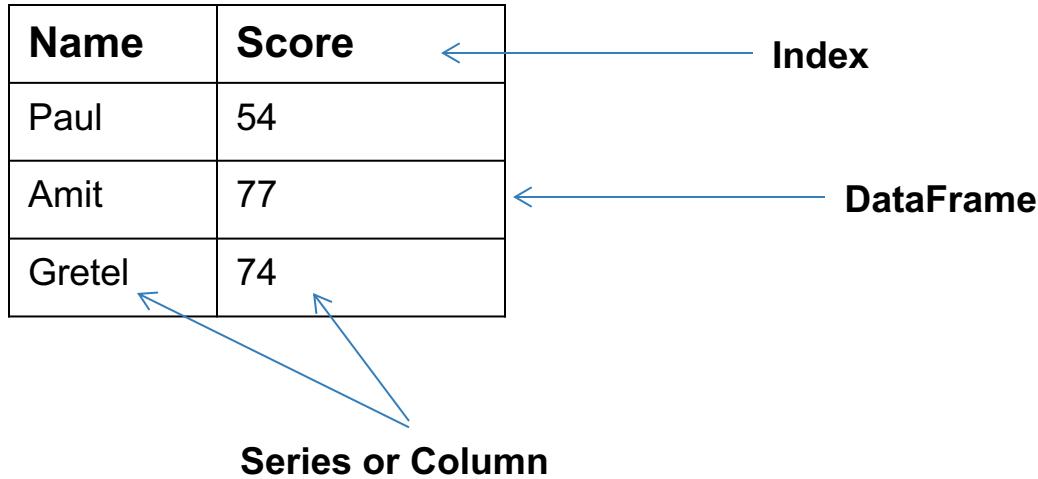


We can use pandas to produce this:

First Name	Last Name	Age	General Subject Area	Overall Grade	Average Mark
Robin	Lenord	18	Sciences	B+	65
Imran	Khan	17	Languages	B	62
:	:	:	:	:	:



Understanding Pandas DataFrames



Introduction to Pandas

What is pandas and why is it useful?

Statistics for Data Analysts and Scientists



Statistics – No one likes statistics

- Almost everyone seems to have a dislike their high school or college Statistics classes
- Most found it confusing, difficult and think it's useless real life.
- They couldn't be more wrong...



Statistics – Why is so important?

Everyone deals with statistics!

- Will it rain tomorrow?
- Who's expected to score the most goals in the next World Cup?
- Is Trump going to win the next election?

And in business....

- What month will I have the most sales, or what time of day?
- Should I take out insurance?
- Is the economy doing well?



Everything dealing with forecasting/predicting comes down to Statistics

Companies and Researchers leveraging statistical knowledge know:

- What products of movies you like (think Amazon or Netflix)
- When fraudulent activities are taking place
- Predicting customer demand
- Understanding the cause of certain illnesses
- Understanding what the best advertisements, medications, diets and more!

“

“Being a Statistician will be the sexiest job over the next decade”

Hal Varian, Chief Economist, Google



The Subfields of Statistics

- **Descriptive Statistics** – Measures or descriptions used to assess some performance or indicator e.g. Batting Averages, GPA
- **Inference** – Using knowledge from data to make informed inferences, e.g. answering the questions “How often do people get the common cold” or “How many people can afford to buy a house at the age of 25”



The Subfields of Statistics

- **Risk and Probability** – What's the likelihood of your rolling a 6 on a dice? Probability is an extensive and important field that is critical for many businesses such as Insurance, Casinos and Finance.
- **Correlation and Relationships** – How do we know smoking causes cancer? Extensive statistical studies have to be used for Hypothesis testing. This is an area that's often extremely difficult, but extremely useful in making impactful decisions.



The Subfields of Statistics

- **Modeling** – Many times in movies or documentaries, you'll hear scientists referring to "*Their model predicts X*". In the real world, especially in data science, modeling is the bread and butter of the job. Building good models that predict some outcome based on the inputs, is critical for many industries. E.g. predicting which customers will purchase Item A.

Descriptive Statistics



Descriptive Statistics

- Descriptive statistics are used to describe or summarize data in ways that are meaningful and useful, such that, for example, patterns might emerge from the data.

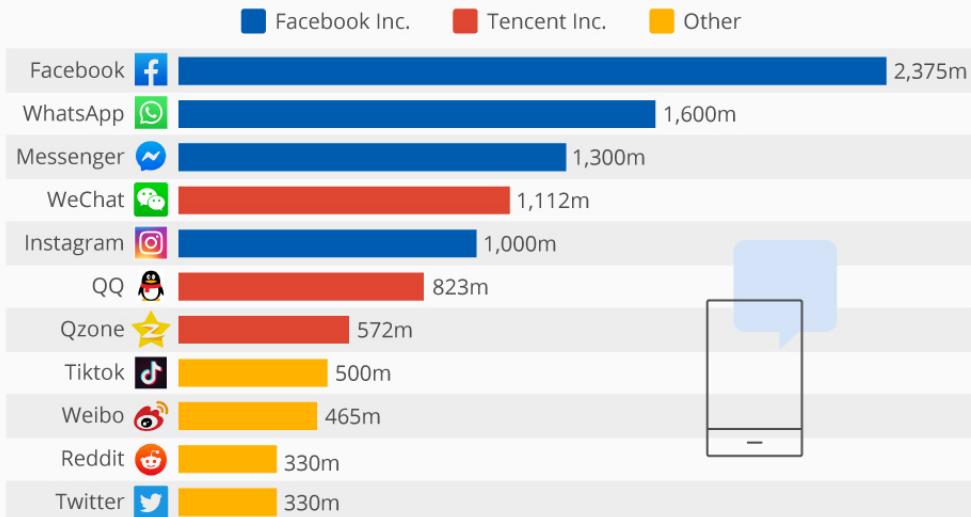


Examples of Descriptive Statistics

- Average heights and weights of males (5'9" and 184lbs for the UK)
- Average number of items sold per day at a store

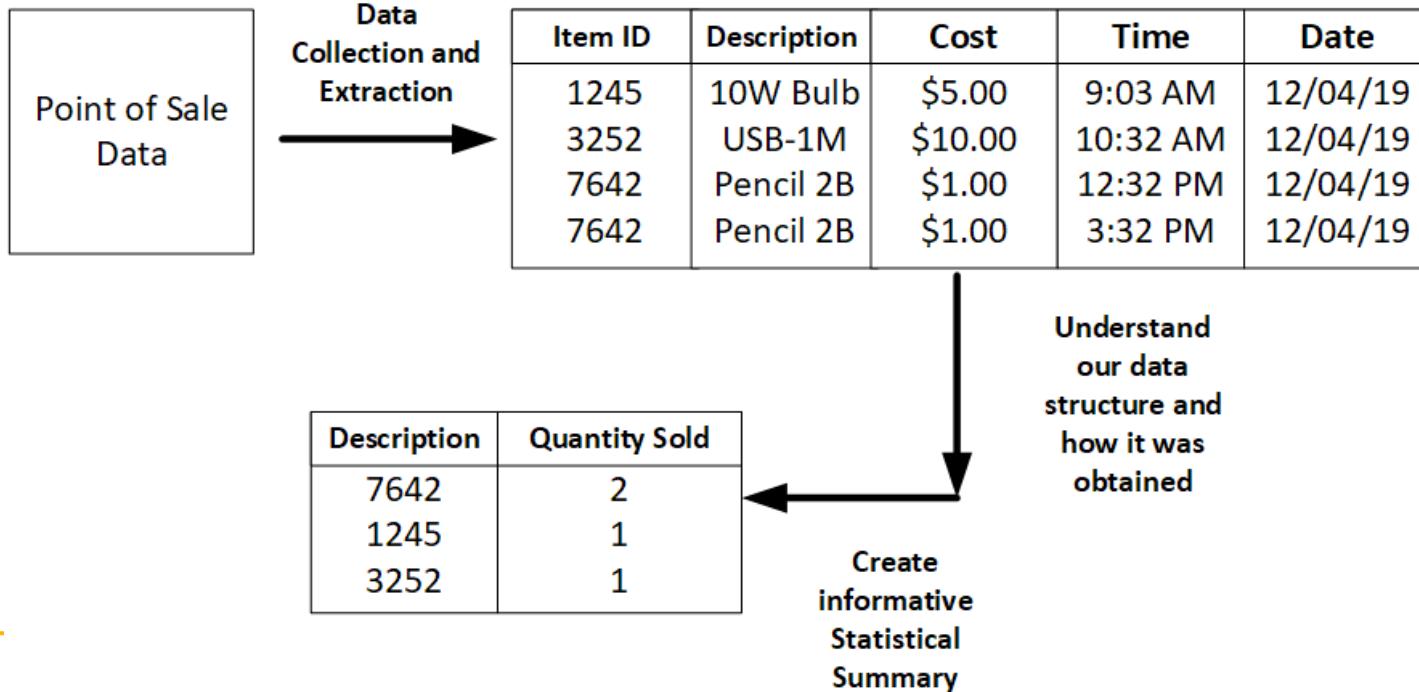
Facebook Inc. Dominates the Social Media Landscape

Monthly active users of selected social networks and messaging services worldwide*





What are good Descriptive Statistics?





Simple Visualizations Help Us Understand Descriptive Statistics

- In order to get a better understanding our data, we need to embark on an Exploratory Data Analysis.
- Simply taking data and computing descriptive statistics without examining the data is the recipe for BAD Data Analysis



Simple Exploratory Analysis

Subject	Mark/Score
English	77
Mathematics	94
Physics	88
Chemistry	91
Biology	0

- Looking at this report card, this student has 5 subjects, this student has scored a total of 350 marks out of 500. If we look at the average grade we get 70. However, they have a 0 in Biology. Is this a mistake? If we ignore the 0, their average is substantially better at 87.5%



We now see the need for actually examining our data

- This process where we visualize the data is called Exploratory Analysis.
- Remember a picture is worth 1000 words, and this certainly holds through for Statistical **Exploratory Data Analysis**

Exploratory Data Analysis

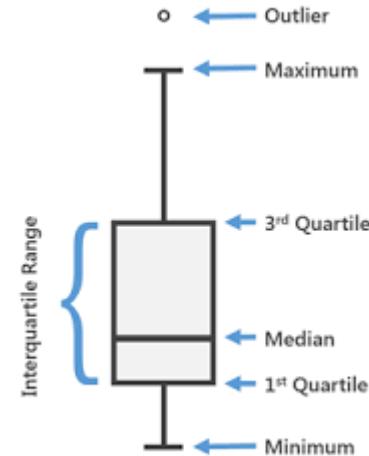
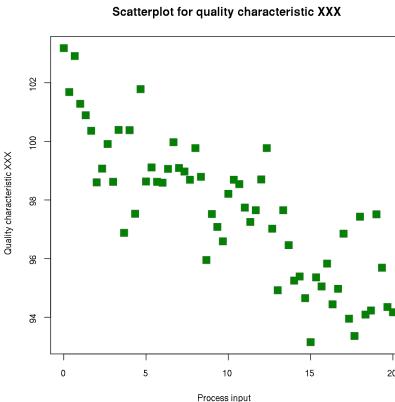
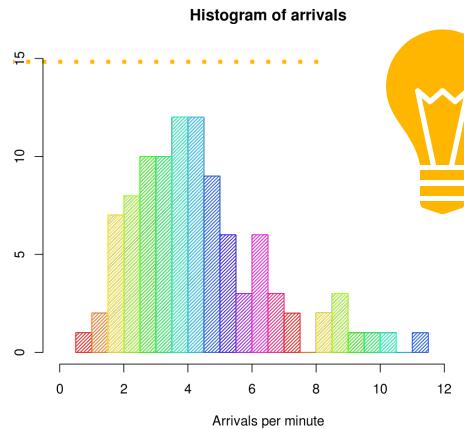
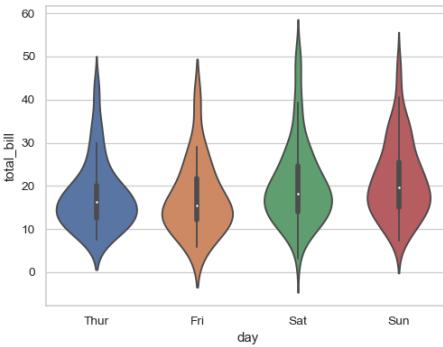


Exploratory Data Analysis (EDA)

- This is the process where we visualize, examine, organize and summarize our data.

Methods of EDA

- Histogram Plots
- Scatter Plots
- Violin Plots
- Boxplots
- Many more!



Sampling – How to Lie with Statistics

“

“There are lies, damn lies, and statistics!”

Mark Twain



How to Lie with Statistics?

- A Poll was conducted to show 70% of people are supporting Trump in the 2020 US Elections
- Trust worthy bit of information?



How to Lie with Statistics?

- How many people were surveyed?
 - 10 out of 350M
- Who was surveyed?
 - 10 retired persons living in Texas



More examples

"Scientists found that feeding beagles a high rice, low protein diet, correlates with high kidney cancer rates."

Headline: "Rice (component in 98% of dog food) causes kidney cancer in dogs!"



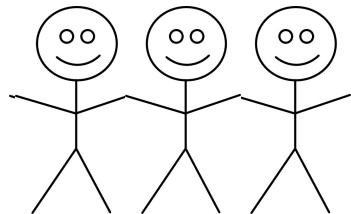
Trump Supporters prefer Beer to Wine!

- What if the general population prefers beer over wine?
- What if Trump supporters typically lived in warmer states where beer was more popular over wine?
- What if the actual statistics showed 64% of Trump Supporters preferred beer over wine, while 63% of Hillary Clinton's supporters held that same view?

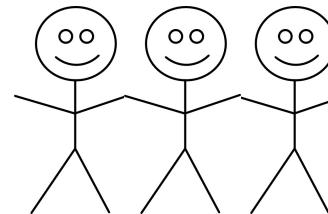


Statistical Goal?

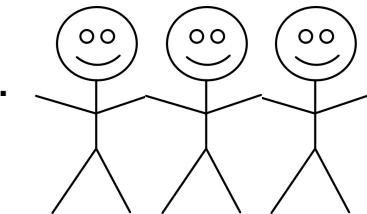
- How do I figure out what 350M people think, without asking everyone of them?



Use smaller population sample



To answer questions about larger population

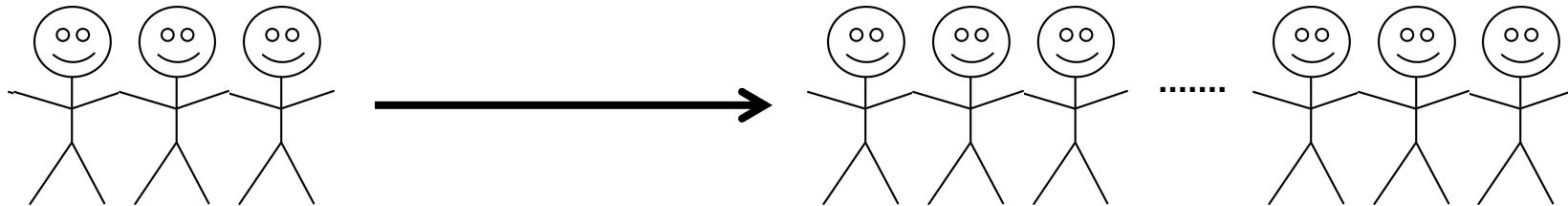


Sampling



What is Sampling?

- In reality we can't always survey the entire population to answer our questions



To answer questions about larger population

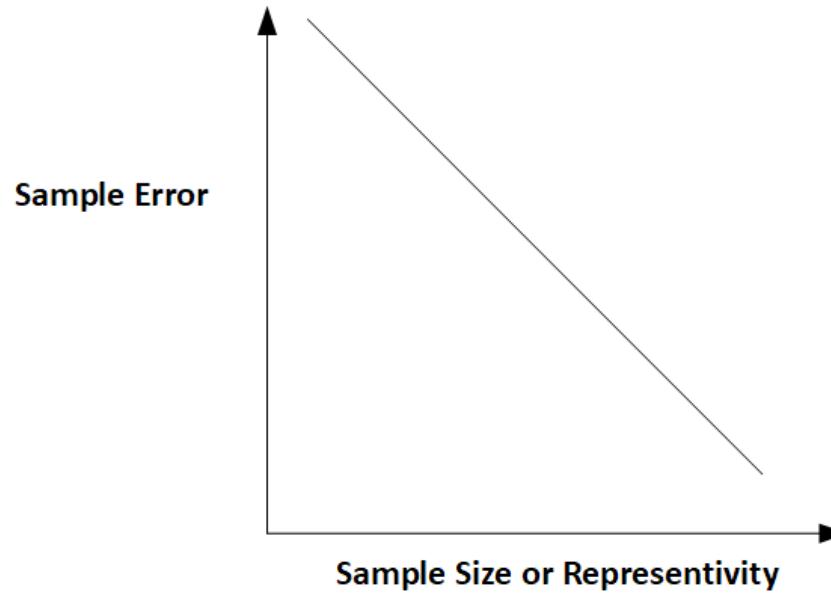


Sampling Example

- Take 5 of your female friends and get their heights
 - 5'2", 5'1", 5'3", 5'8", 5'1"
 - Average = 5'3"
- The actual average height of an adult woman is 5'4"
- My incredibly small sample size (compared to 3B women) ended up being fairly accurate.
- My error was 1", we call this the **Sampling Error**



Sampling Error reduces with Sample Size





Good Sampling

- We want our sample size to be as representative of the main population.
- This means, in our height example, we want women who are tall to be as likely to be chosen as women who are short.
- This requires the sample to be **Random**



Stratifying Data

- Think about our wine dataset – if we use the mean from the entire population of wines, does it accurately reflect the mean alcohol percentage for wines?
- Yes and No – Yes if we're thinking about all types of wines, but generally No because Red and Whites typically have different alcohol percentages and mixing them together skews this.
- We need to separate them into types and then sample from each.



Tips for Creating Good Strata

- Minimize variability in each stratum – in our wine example we should remove outliers (i.e. wines with too high alcohol content in each stratum)
- Good strata are different from each other
- Ensure the criteria we use to select strata are correlated to the property or value you're looking to measure – example for our wine, the different types of wine should correlate to the alcohol content (if not, then a better criterion should be chosen)



Why do we need Sampling?

- Imagine you're a data scientist working for a large national supermarket and you've been asked to determine customer buying habits for a meat products.
- You'll be working with 100M+ rows of data to perform this, in reality, you could randomly sample this dataset and perform this operation much more quickly with almost the same degree of accuracy



Sampling Summary

- A subset of a **population** (i.e. the total or all individuals in a set) is called a **sample**
- A good sample aims to be a good **representative** form of that population
- **Sampling Error** is the difference between the parameters or descriptive statistics (i.e. mean etc.) of the population. **Small Sampling Errors** are good
- **Types of Sampling** that we use to create good representations include:
 - Random sampling
 - Stratified sampling

Variables in Data



What are variables?

- Understanding the data we encounter is essential
- Think of data as the information we collect to describe something
- Data can come in two main forms:
 - Quantitative variables
 - Qualitative or Categorical variables



Understanding Variable Types

- Qualitative /Categorical variable

First Name	Last Name	Age	General Subject Area	Overall Grade	Average Mark
Robin	Leonard	18	Sciences	B+	65
Imran	Khan	17	Languages	B	62
Ryan	Martin	17	Modern Arts	C+	56

- Quantitative variable



Quantitative vs. Qualitative

	Quantitative	Qualitative/Categorical
Describe how much in numeric form (quantity)	YES	NO
Describe a quality or description	NO	YES
Number based (An ID number isn't quantitative since it doesn't measure something)	YES	YES
Number is a Quantity	YES	NO
String or text	YES	YES
Words to describe a quantity (high, low, medium)	YES	NO



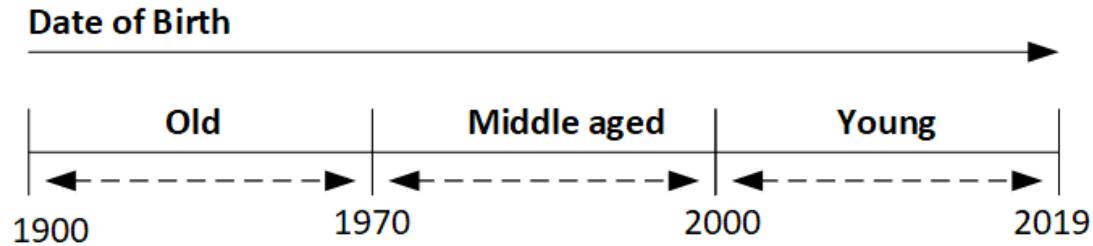
Nominal and Ordinal

- **Nominal** scale variables differentiate individual data points e.g. an ID variable or Name
- They say nothing about the value, direction, size or any quantitative measure. They are always qualitative.
- **Ordinal** scale variables can measure direction, size and values. However, e.g. for a high, low, medium measurements. We don't know exact values i.e. how high or how low.



Interval & Ratio

- **Interval scales** can tell us the size difference between categories, however they still can't tell us the exact value e.g. Date of Birth



- **Ration scales** are similar to interval, but they have the added property of having an inherent zero. E.g. height or weight.



Continuous and Discrete Variables

- Goals are **discrete**, there is no way a player can score a fraction of a goal. Discrete variables measure quantity and value, but have no interval measurement between adjacent values.
- Height however, is a continuous. Just because we give whole number integer values, that doesn't mean Player 1 and Player 3 are exactly the same height. Player 1 can be 177.3cm while Player 3 can be 176.9cm. You can perhaps never have exactly the same value in height of two people as there would be nanometer differences in height

Player	Goals	Height
Player 1	43	177cm
Player 2	25	180cm
Player 3	3	177cm

Frequency Distributions



Why do we collect data?

- **Science/Health**- to describe some observation we see in the world
- **Industries/Business** – make better decisions
- **Engineering** – improve systems
- **Social Sciences** – describe observations we see in society



The Data Collection Process

- 1. Collect Data**
- 2. Analyze Data**
- 3. Use analysis for decision making etc.**

We've taken a look at how we collect data, now let's explore how we analyze and summarize our data



Let's explore Frequency counts

- Imagine we have a table of 100 rows of data of student grades and subject categories

First Name	Last Name	Age	General Subject Area	Overall Grade	Average Mark
Robin	Leonard	18	Sciences	B+	65
Imran	Khan	17	Languages	B	62
Ryan	Martin	17	Modern Arts	C+	56



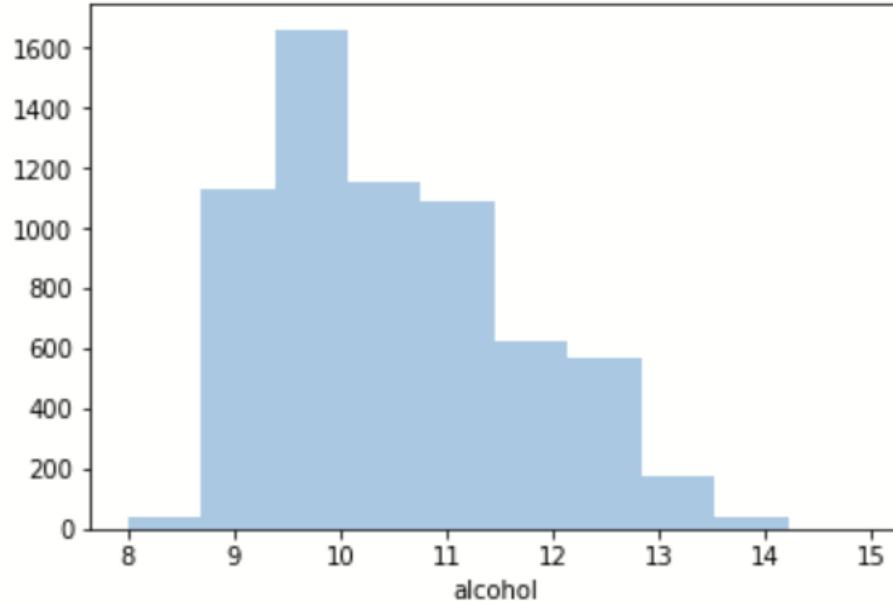
We can use Frequency Breakdown to Summarize the data

General Subject Area	Count/Frequency
Sciences	45
Languages	23
Modern Arts	32



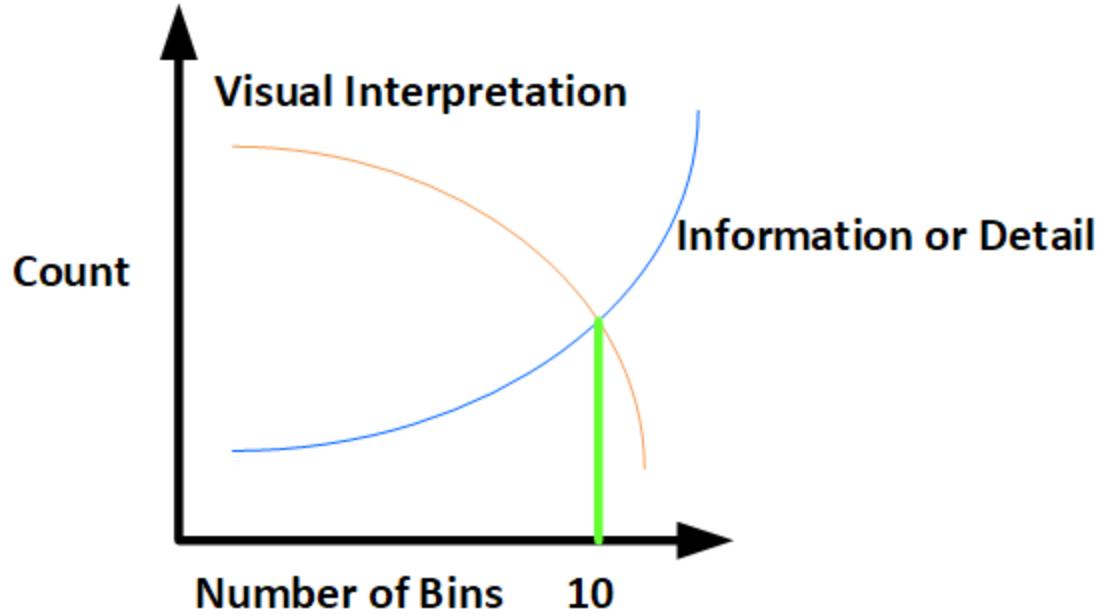
Histogram Plots

**As we saw previously,
Histogram plots
represent this
frequency count well.**



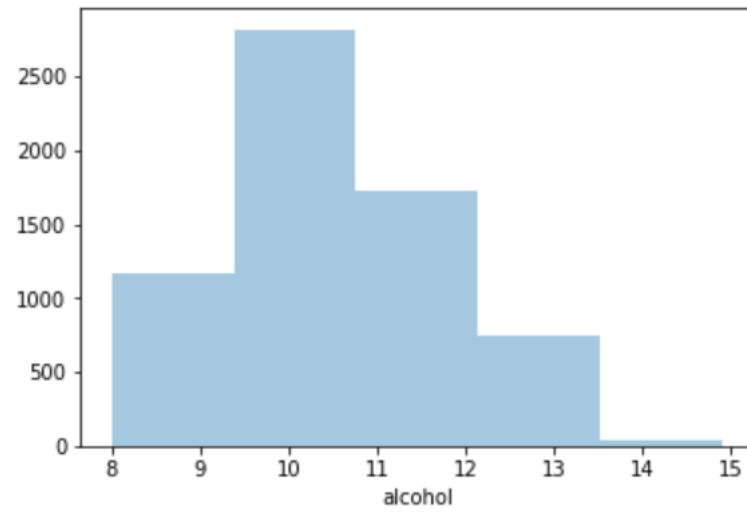
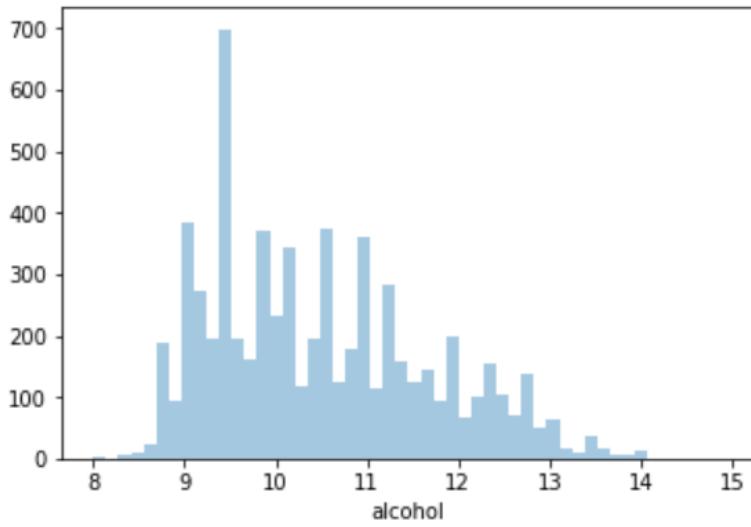


Making Good Histogram Plots



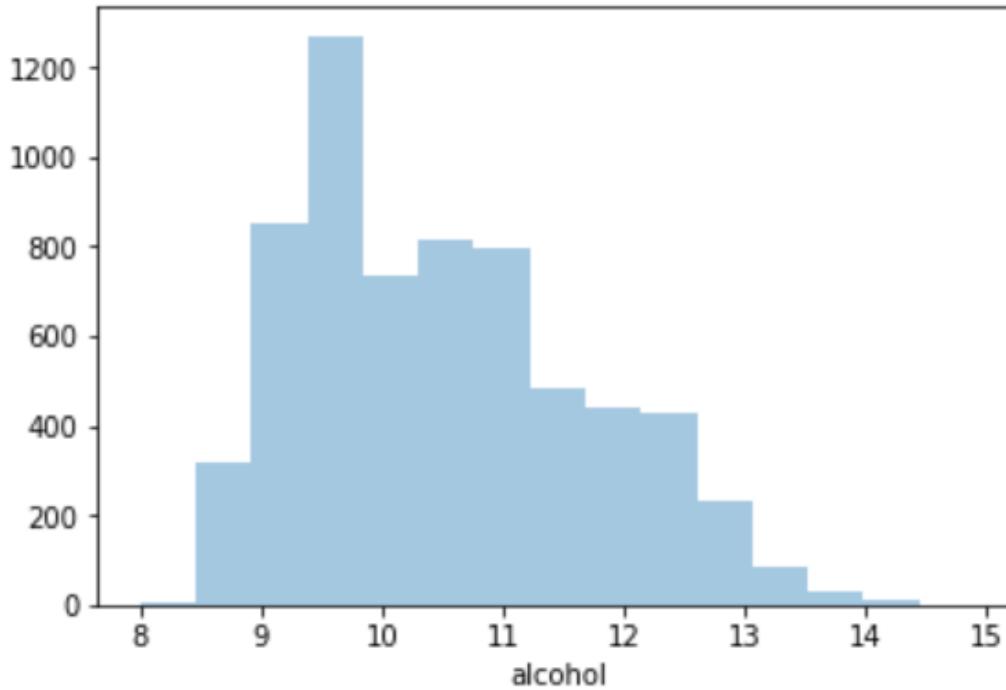


Bad or Confusing Histograms





Good Histograms





Rule of Thumbs for Good Plots

1. Remove irrelevant information – the plot with too many bins doesn't show the user that the most common alcohol percentage for wine is just under 10%

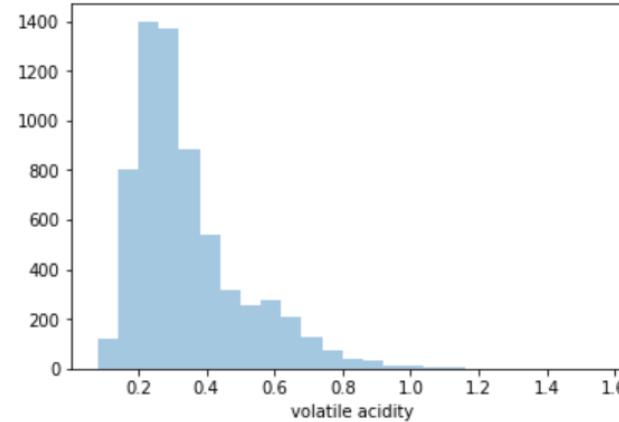
2. Allow your information to tell the story you want it to show easily without necessitating further questions

Frequency Distributions Shapes



Histogram Plots are Frequency Distributions

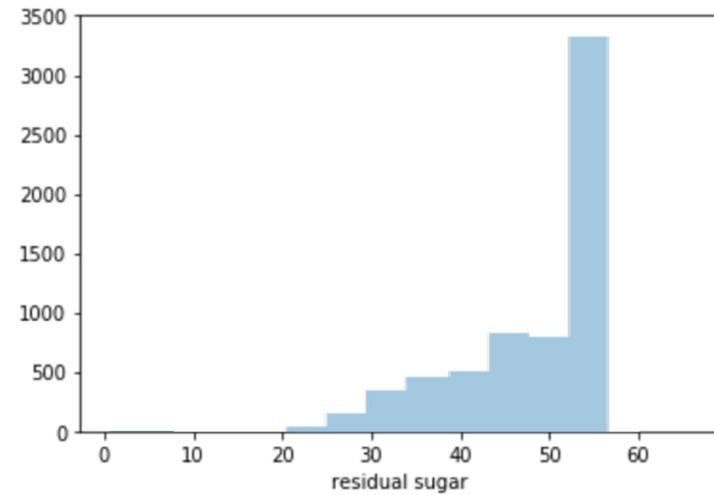
- Imagine a distribution where there is a long drag of data to the right.
- This right skew or positively skewed





Histogram Plots are Frequency Distributions

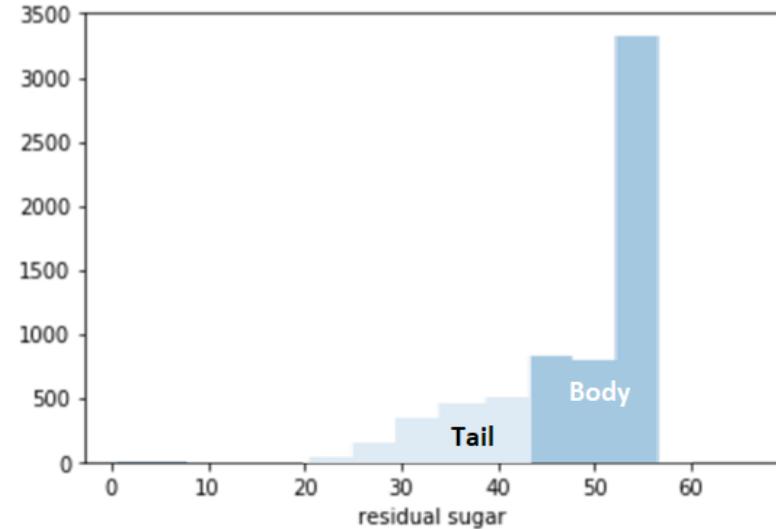
- Conversely we can have the opposite where it's skewed to the **left** or **negatively skewed**
- Or perhaps we have a **normal** looking (pun intended) distribution where the bulk of the data is in the **middle**.





Analyzing Frequency Distribution Plots

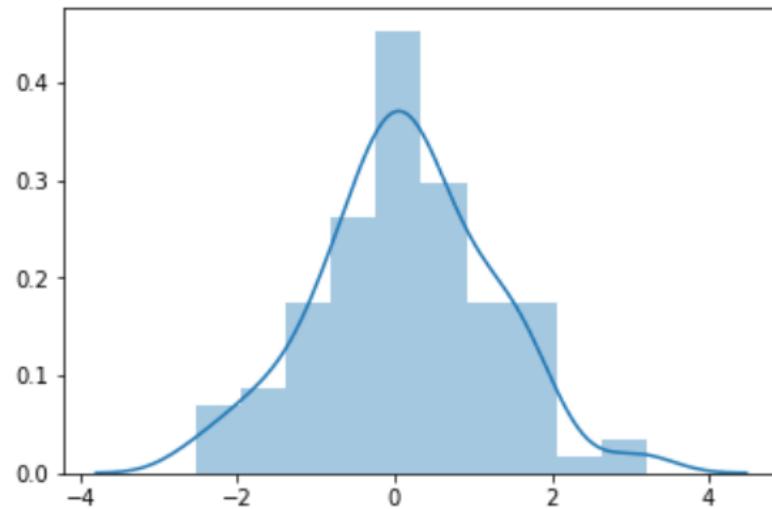
- Location of the Tail indicates the skewness



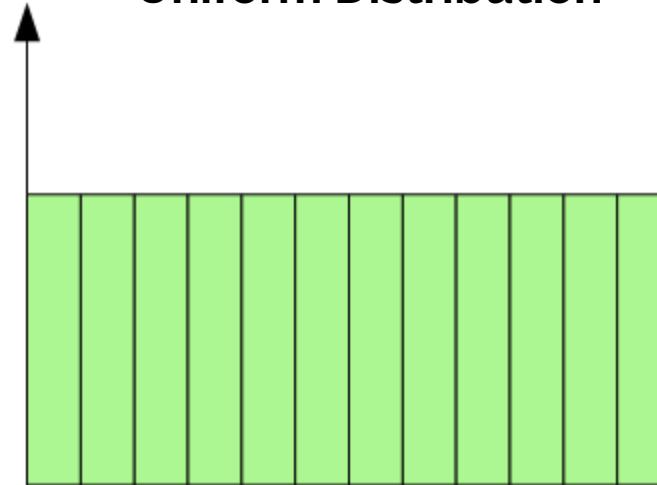


Normal & Uniform Distributions

Normal Distribution



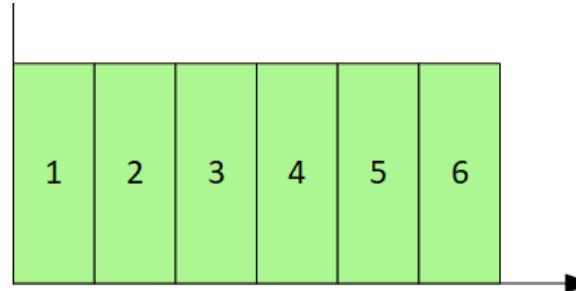
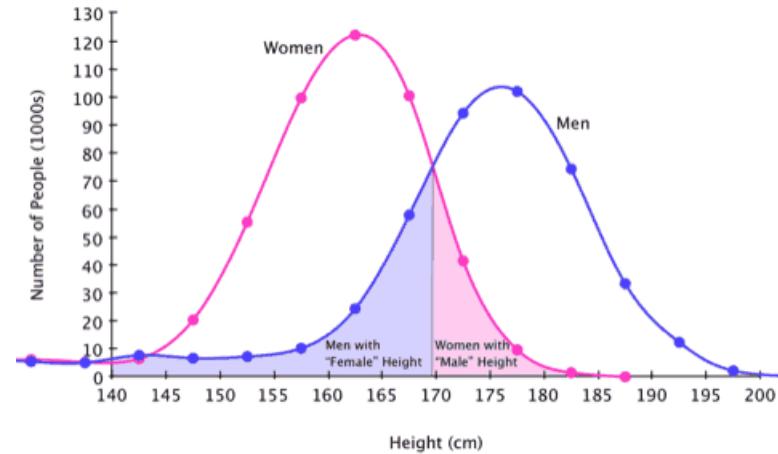
Uniform Distribution



Examples of Normal & Uniform Distributions



- An example of a Normal Distributions are human heights.
- An example of a uniform distribution is rolling a fair dice. After say 1000 rolls, we expect equal numbers of 1s, 2s, 3s, 4s, 5s & 6s



Analyzing Frequency Distributions



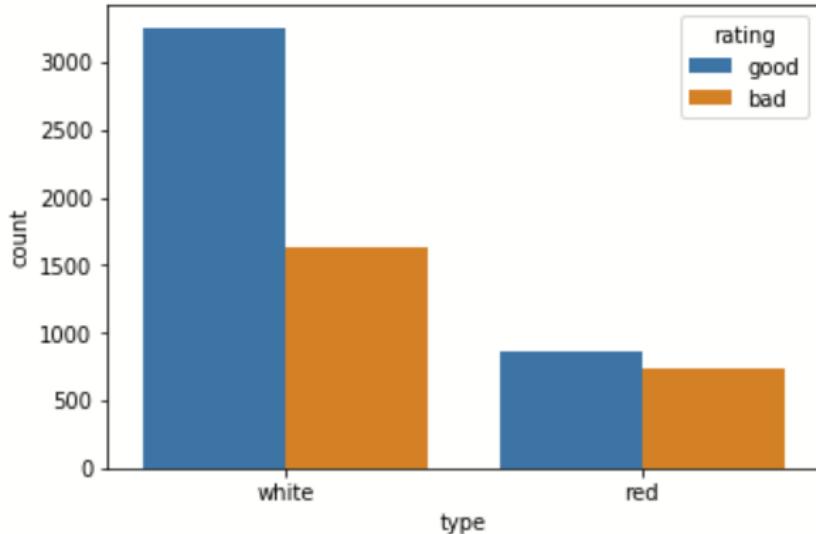
We've seen how useful Frequency Distributions are, but what else can we do?

- Here is a question we'll now answer:
 - When choosing wines randomly in a supermarket, do you have a better chance of choosing a good red or white wine?





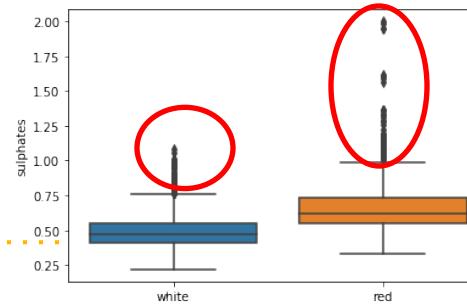
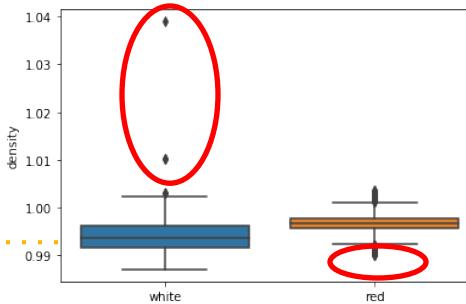
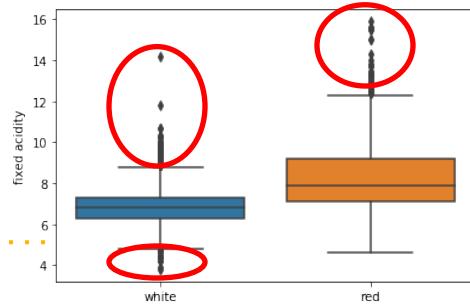
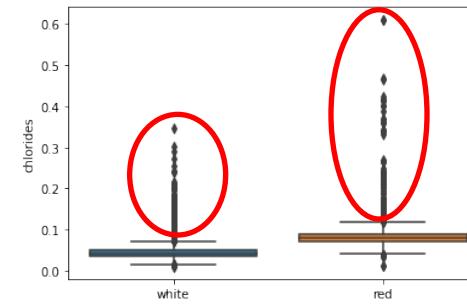
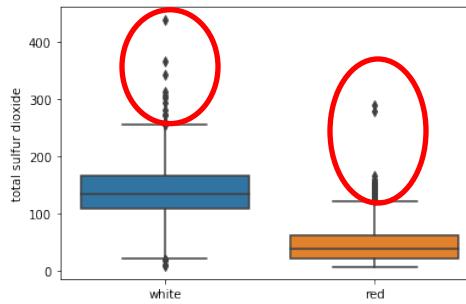
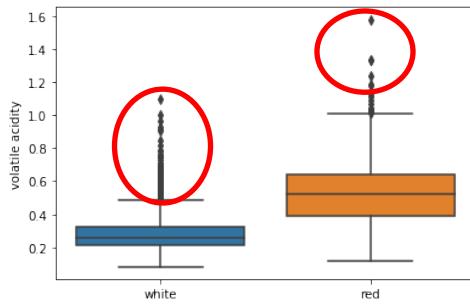
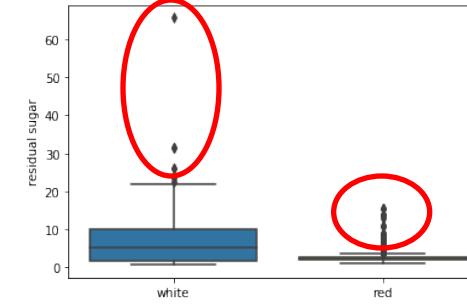
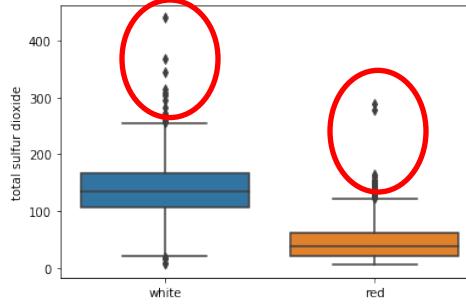
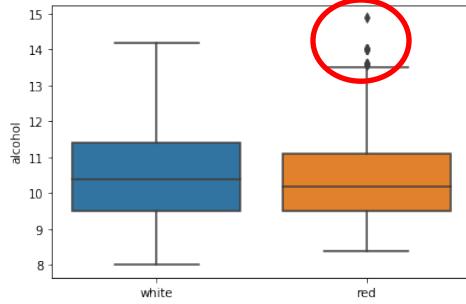
Comparing our Whites and Red



- We see that overall, we can see there are more good white wines than bad, whereas in red it's almost equal.



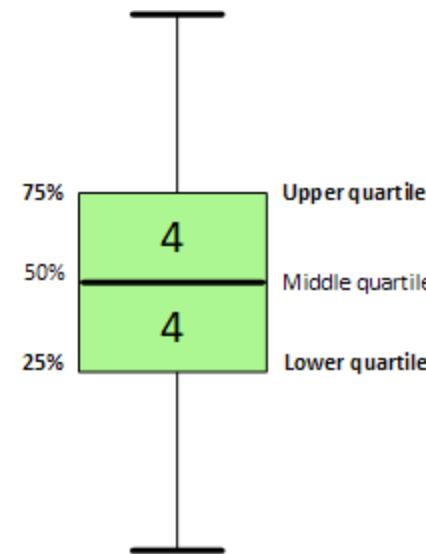
Analyze Frequency Distributions to Find Outliers





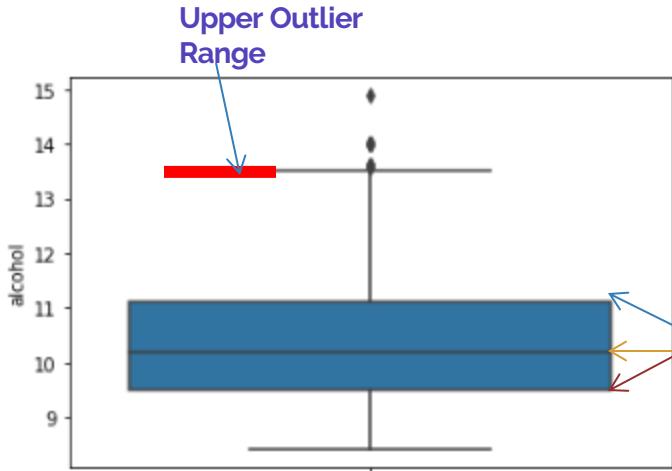
Outliers Rule of Thumb

- A value can be considered an outlier if it exceeds **1.5X the difference between the upper quartile and the lower quartile (the inter Quartile range)**.



Inter Quartile Range = Upper quartile - Lower quartile

Inter Quartile Ranges for Red Wines



```
# Showing the quartile ranges of alcohol content
```

```
df[df['type'] == 'red']['alcohol'].describe()
```

count	1599.000000
mean	10.422983
std	1.065668
min	8.400000
25%	9.500000
50%	10.200000
75%	11.100000
max	14.900000

Name: alcohol, dtype: float64

Upper Quartile = 11.1
Lower Quartile = 9.5

Values exceeding 13.5 or being less than 7.1 are outliers

Inter Quartile Range = $11.1 - 9.5 = 1.6$

Outlier Ranges = $1.6 * 1.5 = 2.4$

Upper Outlier Range = $11.1 + 2.4 = 13.5$

Lower Outlier Range = $9.5 - 2.4 = 7.1$



Understanding Outliers

- **Outliers** aren't necessarily bad, though in most data analysis, we often drop them because:
 - They were faulty or incorrect data
 - They are so far off the bulk/body of the distribution it led to misleading analysis
- Nevertheless, they do sometimes represent extreme statistics that we need to be aware of.
- Note: our definition of outlier thresholds can be changed according to our own intuition

Mean, Weighted Mean, Mode & Median



Summarizing Statistics

- We've seen tables and visual representations of our data thus far.
- Now, we're about to see how we can accurately summarize the statistics or information in our distribution by using:
 - Mean
 - Weighted Mean
 - Median
 - Mode



The Mean

- The mean is quite simple and we will have discussed it before so you intuitively know that mean is the same as average.

Person	Phones Owned
Nancy	6
Amy	5
Savi	7
MEAN	6

$$\text{Mean} = \frac{6 + 5 + 7}{3} = 6$$



Common Mean Misconceptions

- Many times, we consider mean to be the expected average i.e. the center or most common value. However, that is often a mistake.
- Outliers can skew our means
- Joan with her 26 phones, skews our mean

$$\text{Mean} = \frac{6 + 5 + 7 + 26}{4} = 11$$

Person	Phones Owned
Nancy	6
Amy	5
Savi	7
Joan	26
MEAN	6

Illustration of Mean Skewing

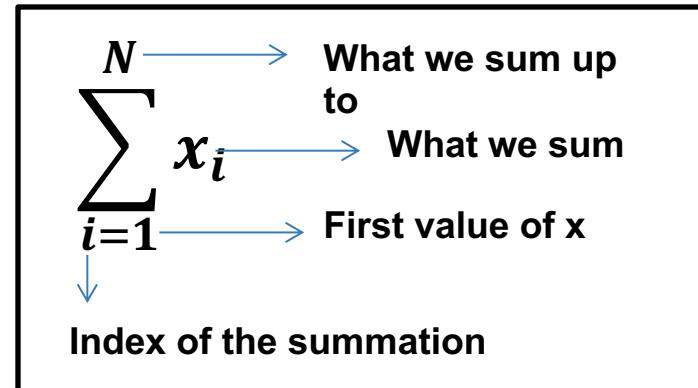


Mathematical Notation for Mean

$$\mu = \frac{x_1 + x_2 + x_3 \dots + x_n}{N} = \frac{\sum_{i=1}^N x_i}{N}$$

Where:

- μ = Mean
- x = each value in our dataset
- N = Number of data samples in our dataset



Sigma (capital sigma to be exact) is shorthand in mathematics for representing a series of sums. E.g. $1 + 2 + 3 + 4$ can be represented as:

$$\sum_{i=1}^4 x_i$$

The range stated by the bottom ($i=1$) and top numbers (4) next to the Sigma.



Weighted Mean

- Here's an issue you might encounter with means. If we're given just the summary data below, how can we calculate the mean of the cars sold across all years?

Year	Cars Sold	Mean Sale Price
2015	54	18,425
2016	51	19,352
2017	58	18,215
2018	33	17,942



Weighted Mean

- You might assume we can just find the mean of the “Mean Sale Price” and be done with it. That would be wrong.
- Think about this simple example. We have 2 boys and 3 girls
- Boy 1 weighs 101lbs, Boy 2: 115lbs, Girl 1: 90lbs, Girl 2: 77lbs, Girl 3: 99lbs.
- The average weight of all 5 of them is: 96.4 lbs
- However, if you were given the summarized the mean separately:
 - Mean Boy Weight = 108 lbs
 - Mean Girl Weight = 88.7 lbs
 - $(\text{Mean Boy Weight} + \text{Mean Girl Weight}) / 2 = 98.33 \text{ lbs}$
- The means aren't equal!



Calculating the Weighted Mean

Child	Weight
Boy 1	101
Boy 2	115
Girl 1	90
Girl 2	77
Girl 3	99
Mean	96.4

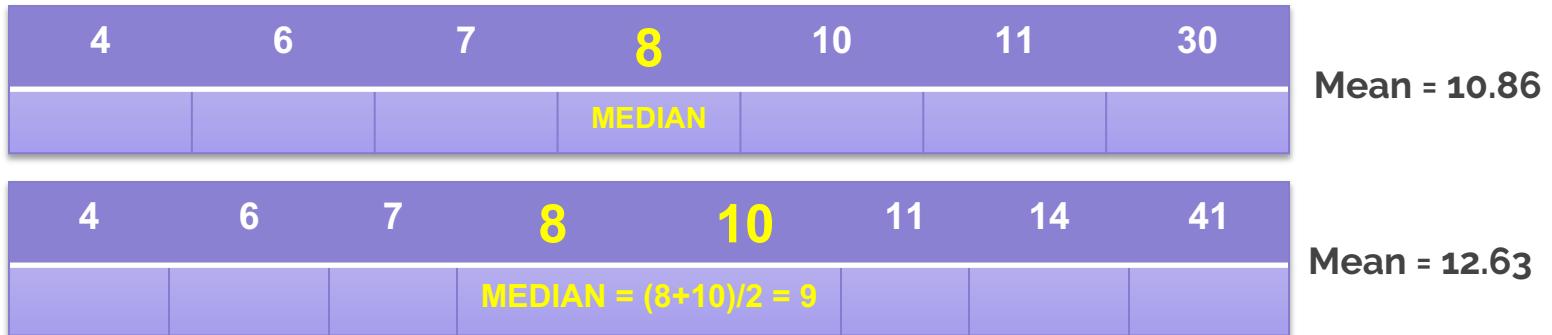
Child	Mean Weights	Quantity	Expanded Means	
Boys	108	2	$2 * 108 = 216$	
Girls	88.7	3	$3 * 88.7 = 266$	

Getting our accurate weighted mean is simple:
 $(216 + 266) / 5 = 96.4\text{lbs}$



Median

- Many people confuse Means and Medians.
- Remember while means are the average of all values, Median is average of the two middle values or the actual middle value itself (depending on if the quantity of data is odd or even)





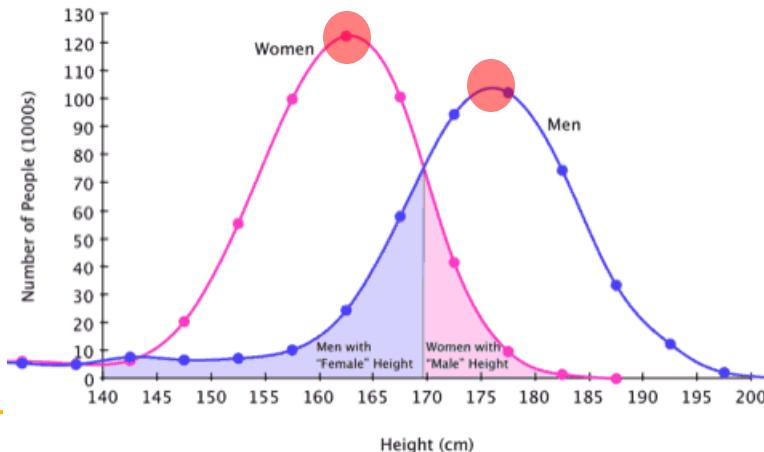
Median

- Medians are useful because they ignore the effects of outliers and give us a good idea of a general average of the data
- It's a **robust statistic**
- One way to lie with statistics is using means over medians. E.g. The average salary of an employee in company A is 100,000 per year. However, that average or mean can easily be skewed upward by a few executives making millions per year. In fact, 95% of the company can be making less the mean salary!



Mode

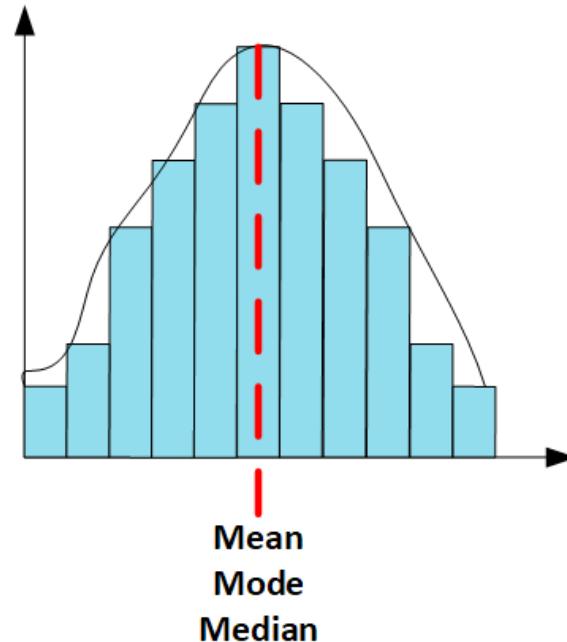
- The Mode is simply the most frequent item in distribution.
- In any Kernel Density plot (KDE in Seaborn), the mode is always the peak





When is Mean, Mode and Median the same?

- They are all the same when a distribution is **symmetrical**
- For a symmetrical distribution, we really only need to describe it by stating what the mean is and how wide or narrow the distribution is.



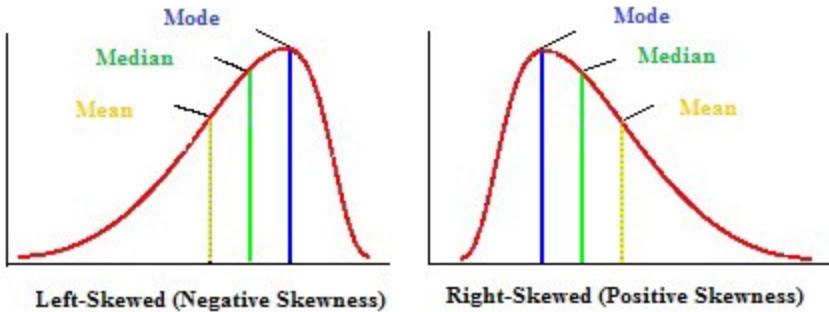


When to use Mean, Mode and Median?

- **Means** are best for numeric (continuous, discreet or numerically encoded ordinal data) and is good for summarizing the entire **population** size of a distribution.
- **Medians** can be used on the same data type, and are good for summarizing data when there are **outliers** (use **boxplots** to check)
- **Mode** can be used on both numeric or categorical data and are good for summarizing data for persons not well versed in the intricacies of mode and medians.



Pearson Mode Skewness



Source: <https://www.statisticshowto.datasciencecentral.com/pearson-mode-skewness/>

- **Positive or right Skewness** is when the mean > median > mode
 - Outliers are skewed to the right
- **Negative Skewness** is when the mean < median < mode
 - Outliers are skewed to the left

Variance, Standard Deviation and Bessel's Correction



Measure of Spread/Dispersion

- Mean, mode and median all give us a view of what's the most likely or common data point in a dataset.
- Think about human heights, we can use the mean, mode or median to illustrate the point that the average male human is 5'9".
- However, this tells us nothing about how common it is to find someone over 7'
- Or even more descriptive measure: 95% of men lie between what range of height?



The Range of our Data

- In our Wine dataset we can see the max and min alcohol parentages were 14.9% and 8.0%
- Therefore our range:
 - Range = max – min
 - $14.9 - 8.0 = 6.9$
- This is one way to measure the spread of the data but there is a flaw

```
print(df['alcohol'].max())
print(df['alcohol'].min())
```

14.9

8.0



The Short Comings of Range

- The weakness in using Range is that it only uses the max and min.
- What if we had a distribution that was:
 - **10, 10, 11, 12, 11, 10, 11, 12, 200**
 - This distribution has a range of $200 - 10 = 190$, which gives us the impression that our values swing widely up to 190, however, in reality our data is consistently varying between 10-12 with one major exception.



The lead up to Variance - Difference

- 10, 10, 11, 12, 11, 10, 11, 12, 200 Mean = 31.88

Values below mean

$x - \mu$	Distance
10-31.88	-21.88
10-31.88	-21.88
11-31.88	-20.88
12-31.88	-19.88
11-31.88	-20.88
10-31.88	-21.88
11-31.88	-20.88
12-31.88	-19.88
Total	-168.11

Values above mean

$x - \mu$	Distance
200-31.88	168.11
Total	168.11

- The Difference = (-168.11 + 168.11) / 9 = 0
- How do we solve this problem?



Use Mean Absolute Distance

- We treat all distances as positive i.e. take the modulus
- Mathematically written as $|x|$ e.g. $|-2| = +2$
- Mean Absolute Distance is written mathematically as:

$$\text{Mean absolute distance} = \frac{|x_1 - \mu| + |x_2 - \mu| + |x_3 - \mu| + \dots + |x_N - \mu|}{N} = \frac{\sum_{i=1}^N |x_i - \mu|}{N}$$

- So from our previous example, we get:
 - $(21.88 + 21.88 + 20.88 + 19.88 + 20.88 + 21.88 + 20.88 + 19.88 + 168.88) / 9 = 37.3$



Use Mean Squared Distance - Variance

- Mean Squared Distance is written mathematically as:

$$\text{Mean Squared Distance} = \frac{(x_1-\mu)^2 + (x_2-\mu)^2 + \dots + (x_N-\mu)^2}{N} = \frac{\sum_{i=1}^N (x_i-\mu)^2}{N}$$

- So from our previous example, we get:

- $(21.88^2 + 21.88^2 + 20.88^2 + 19.88^2 + 20.88^2 + 21.88^2 + 20.88^2 + 19.88^2 + 168.88^2) / 9 = 3530.22$

- Mean Squared Distance is commonly called the **Variance**
- Standard Deviation** is the square root of Variance =

- Standard Deviation = $\sqrt{\frac{(x_1-\mu)^2 + (x_2-\mu)^2 + \dots + (x_N-\mu)^2}{N}} = \sqrt{\frac{\sum_{i=1}^N (x_i-\mu)^2}{N}}$



Variance and Standard Deviation

- **Variance** measures how far is the sum of squared distances from each point to the mean i.e the dispersion around the mean.
- **Standard Deviation:** One weakness of using variance is that it suffers from unit difference. The square root of the variance is the standard deviation. It conveys to us, the concentration of the data around the mean of the data set.
● $\sigma = \sqrt{\sigma^2}$



Coefficient of Variation(CV)

- **Coefficient of Variance (CV) is also known as relative standard deviation and it is the:**
 - **The ratio of standard deviation to the population mean.**
- **The example on the right shows that the CV is the same, hence both methods are equally precise.**



Analyte: Glucose

Method: Hexokinase

Standard deviation = 4.8

Mean = 120



Analyte: Glucose

Method: Glucose oxidase

Standard deviation = 4.0

Mean = 100

Which method is more precise ?

Calculate the CV to see:

$$CV (\%) = \frac{\text{Standard Deviation}}{\text{Mean}} \times 100$$

$$\frac{4.8 \text{ (SD)}}{120 \text{ (Mean)}} \times 100 = 0.04\% \text{ (CV)}$$

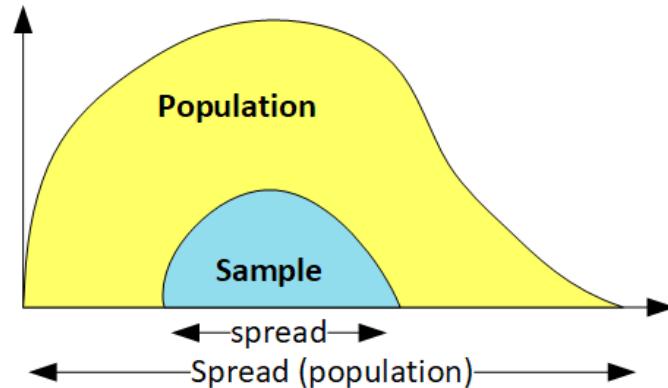
$$\frac{4.0 \text{ (SD)}}{100 \text{ (Mean)}} \times 100 = 0.04\% \text{ (CV)}$$



Sampling from a Population

Bessel's Correction

- One thing to note is that when we sample from a distribution, we are likely to **underestimate** the standard deviation.
- By decreasing the denominator, we are increasing the STD



$$\text{Standard Deviation} = \sqrt{\frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + \dots + (x_N - \mu)^2}{N-1}} = \sqrt{\frac{\sum_{i=1}^N (x_i - \mu)^2}{N-1}}$$

Covariance and Correlation

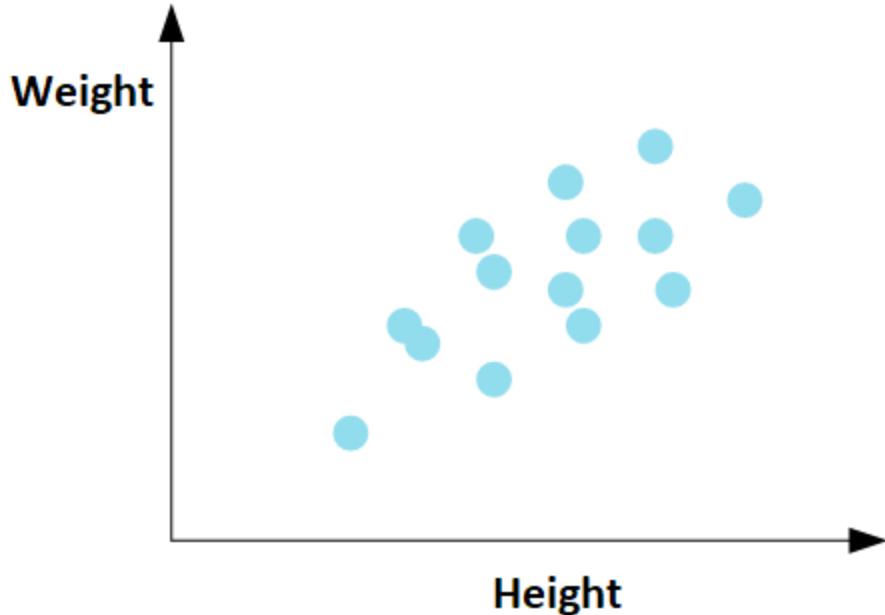


Covariance

- We know now that variance measures the spread around the mean. More accurately, it is the sum of the squared distances of each point from the mean.
- Covariance is a very important metric in statistical analysis
- It is a measure to show the relationship between the variability of 2 variables. In other words, we are looking to assess how changes in our variable affect the other



Covariance

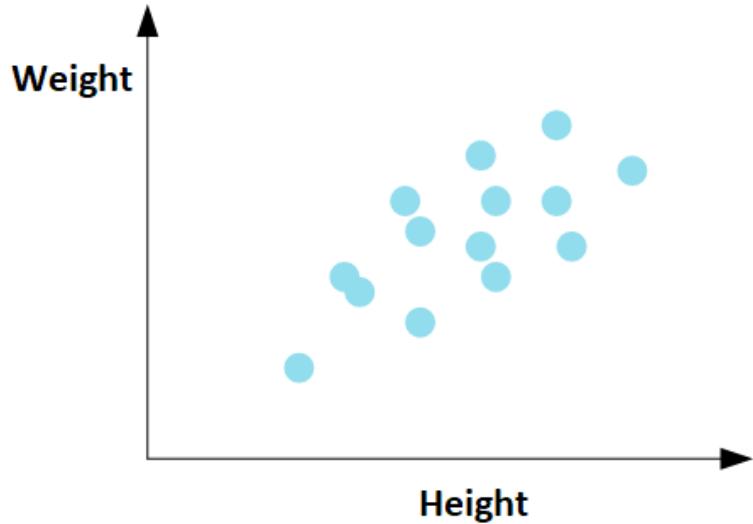


Subject	Height	Weight
1	174	136
:	184	175
N	173	120

- Let's think about some data collected about the heights and weights of people.



Covariance



- We can see that as height increases so does the weight, and vice versa.
- This means these two variables are positively covariance

$$Cov(x, y) = \sigma_{xy} = \sum_{i=1}^N \frac{(x_i - \mu_x)(y_i - \mu_y)}{N}$$



Working through the Covariance

X	1	4	5	7	Mean = 16 / 4 = 4.25
Y	0.3	0.4	0.8	0.9	Mean = 2.4 / 4 = 0.6

- $Cov(x, y) = \sigma_{xy} = \sum_{i=1}^N \frac{(x_i - \mu_x)(y_i - \mu_y)}{N}$
- $Cov(x, y) = \frac{(1-4.25)(0.3-0.6)+(4-4.25)(0.4-0.6)+(5-4.25)(0.8-0.6)+(7-4.25)(0.9-0.6)}{4}$
- $Cov(x, y) = \frac{(-3.25)(-0.3)+(-0.25)(-0.2)+(1.25)(0.2)+(3.75)(0.3)}{4} = \frac{0.975+0.05+0.25+1.125}{4} = +0.6$
- Our result is positive, which means the variables have a positive covariance



The Weakness of Covariance

- Covariance does not give effective information about the relation between 2 variables as it is not normalized.
- This means we can't compare variances over data sets with different scales (like pounds and inches).
- A weak covariance in one data set may be a strong one in a different data set with different scales.



Correlation

- **Correlation** provides a better understanding of the relationship between two variables because it is the **Normalized Covariance**.
- **Normalization** is the process of ensuring all variables are of the same scale.
 - E.g. imagine we had variable x in inches, ranging from 1 to 342 and y in weight, ranging from 50 to 100lbs. Normalization changes the scale or maps those variables on scale equal to both.
 - Normalization for instance would adjust both scales so that they both range from 0 to 1 (one of many normalization ranges)



Correlation Formula

- *Correlation* = $\rho = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$
- Correlation ranges from **-1** to **1**
- Negative **1 (-1)** indicates negative correlation
- Positive **1 (+1)** indicates positive correlation
- Zero (**0**) indicates no correlation between variables. i.e. they are independent of each other



X	Y	$x - \bar{x}$ $\bar{x} = 4.25$	$y - \bar{y}$ $\bar{y} = 0.6$	$(x - \bar{x})(y - \bar{y})$	$(x - \bar{x})^2$	$(y - \bar{y})^2$
1	0.3	-3.25	-0.3	0.975	10.5625	0.09
4	0.4	-0.25	-0.2	0.05	0.0625	0.04
5	0.8	1.25	0.2	0.25	1.5625	0.04
7	0.9	3.75	-0.3	-1.125	14.0625	0.09

- $\bar{x} = 4.25$ and $\bar{y} = 0.6$
- $\sum(x - \bar{x})(y - \bar{y}) = 0.15$
- $\sum(x - \bar{x})^2 = 26.25$
- $\sum(y - \bar{y})^2 = 0.26$
- $Correlation = \rho = \frac{\sum(x - \bar{x})(y - \bar{y})}{\sqrt{\sum(x - \bar{x})^2} \sqrt{\sum(y - \bar{y})^2}} = \frac{0.15}{\sqrt{26.25} \sqrt{0.26}} = \frac{0.15}{5.123 \times 0.51} = 0.057$

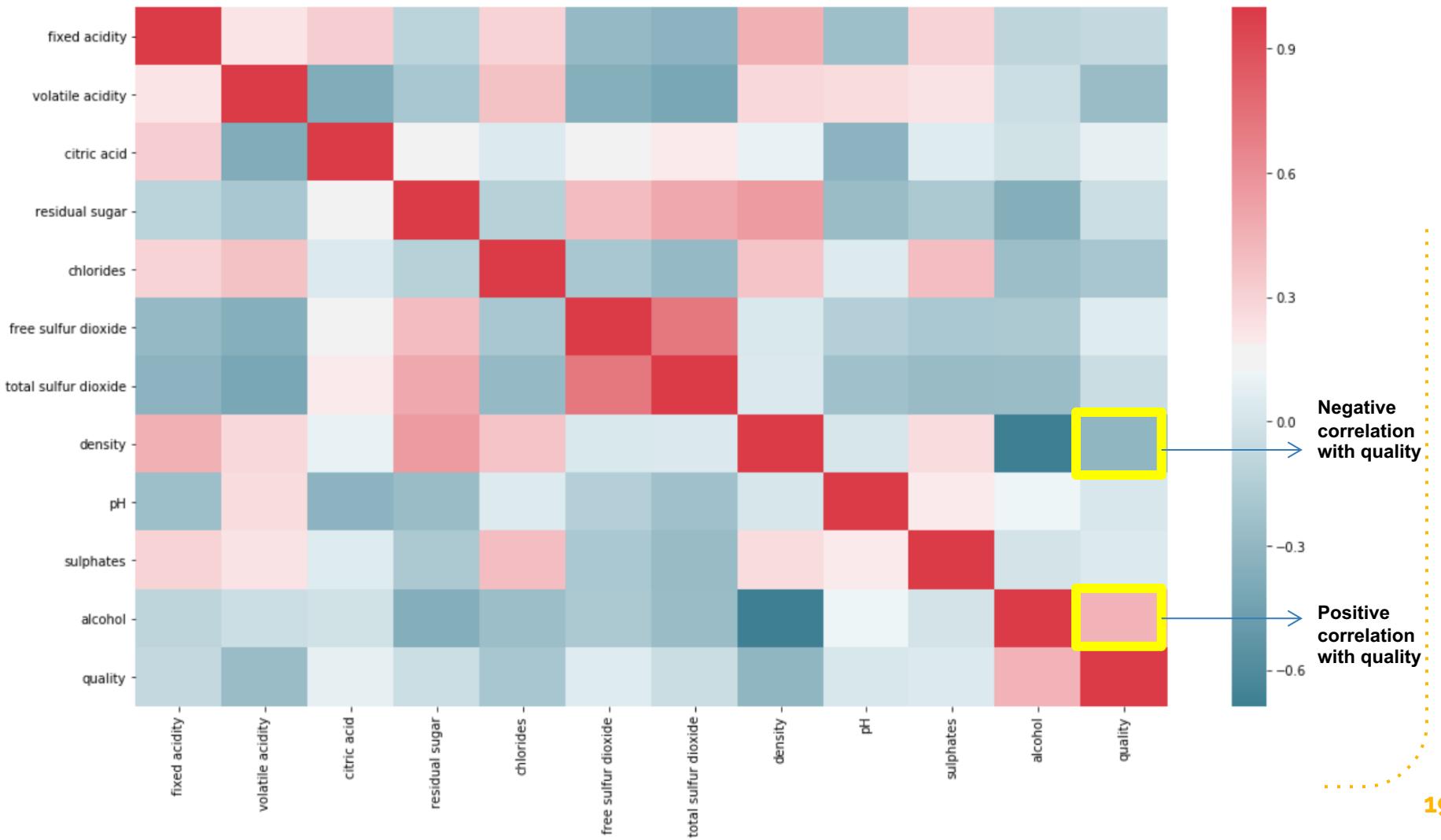
Correlation Matrix

- We can use Pandas and Seaborn to produce very informative correlation plots.

Correlation Matrix

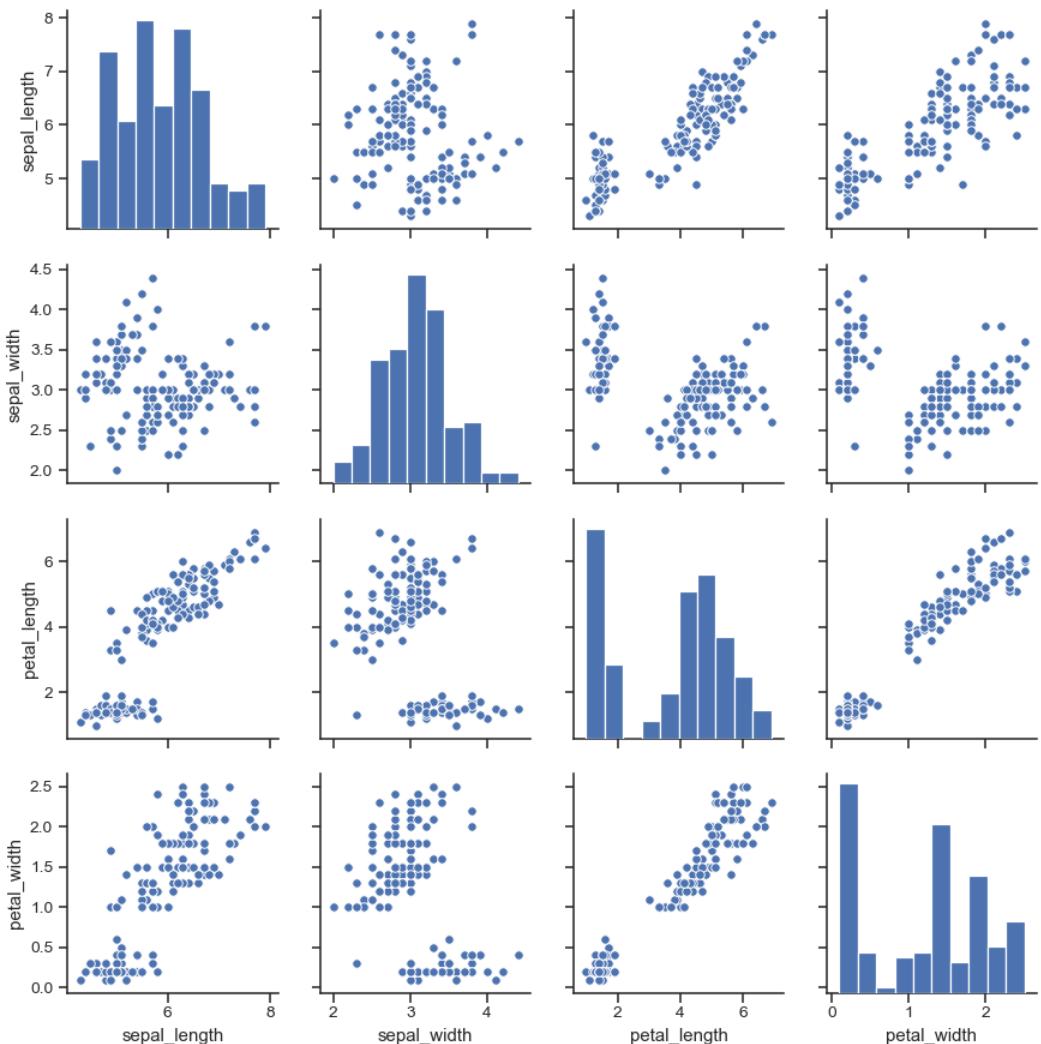
```
▶ # Calculate correlations
corr = df.corr()

# Heatmap
plt.figure(figsize=(18, 10))
sns.heatmap(corr, cmap=sns.diverging_palette(220, 10, as_cmap=True))
```



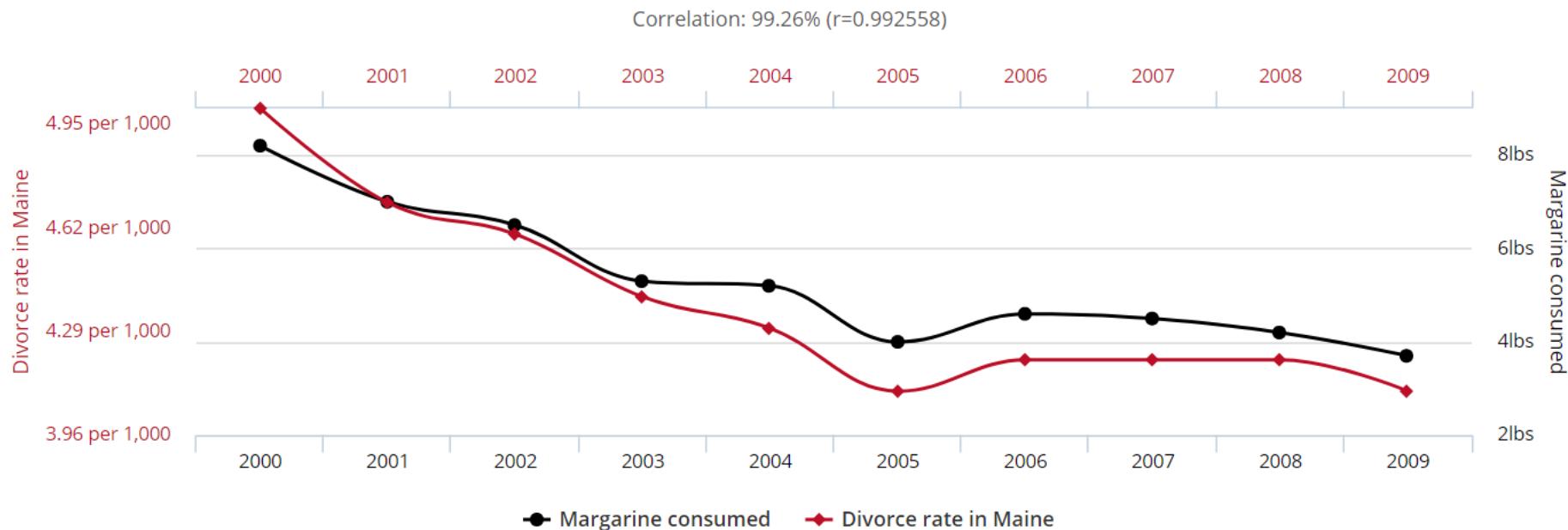
Pairwise Plots

- Pairwise plots create scatter plots showing the relation between each feature in our dataset



**Lying with Correlations – Divorce
Rates in Maine caused by
Margarine Consumption?**

Divorce rate in Maine correlates with Per capita consumption of margarine



tylervigen.com

Data sources: National Vital Statistics Reports and U.S. Department of Agriculture

Source: <http://www.tylervigen.com/spurious-correlations>



Lying and Misleading with Statistics

- Unfortunately, people who aren't well versed in statistics can often be misled with bogus claims.
- Correlation between two variables DOES NOT mean causation.
- Causation implies one variable is influencing another.

The Normal Distribution & the Central Limit Theorem

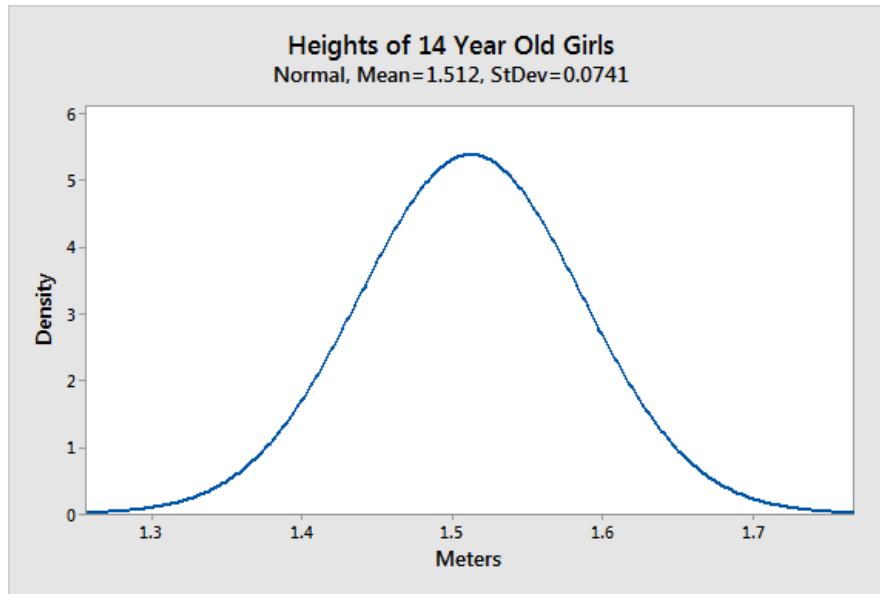


Normal or Gaussian Distributions

- We've discussed mean, mode, median, standard deviation and variance before.
- We previously defined Normal distributions are when the mean, mode and median are roughly the same.
- They're often call a natural phenomena as so many things in nature, geography and biology tend to follow normal distributions.

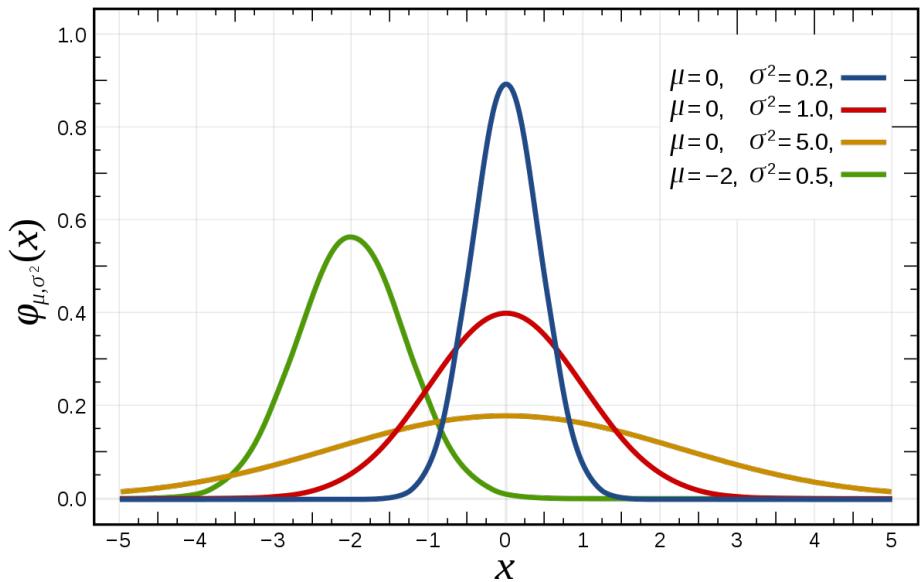


Normal Distributions Example





Describing Normal Distributions



Notation : $N(\mu, \sigma^2)$

- Normal Distributions can be easily described by two simple statistics, the mean and variance

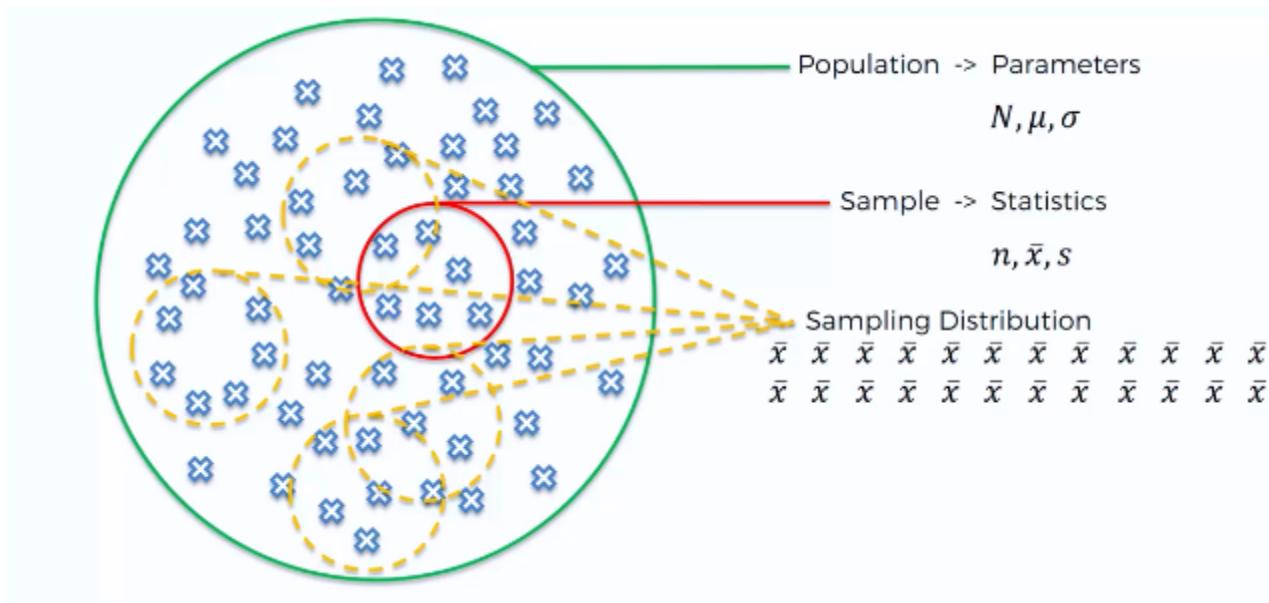


Central Limit Theorem

- One of the most important theorems in Statistics is said to be the Central Limit Theorem.
- The Central Limit Theorem states that the sampling distribution will look like a normal distribution regardless of the population we are analyzing



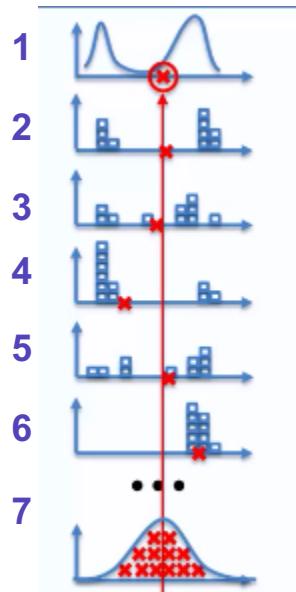
Sampling a Population



Source: <https://www.superdatascience.com/courses/statistics-business-analytics-a-z/>



Sampling a Population

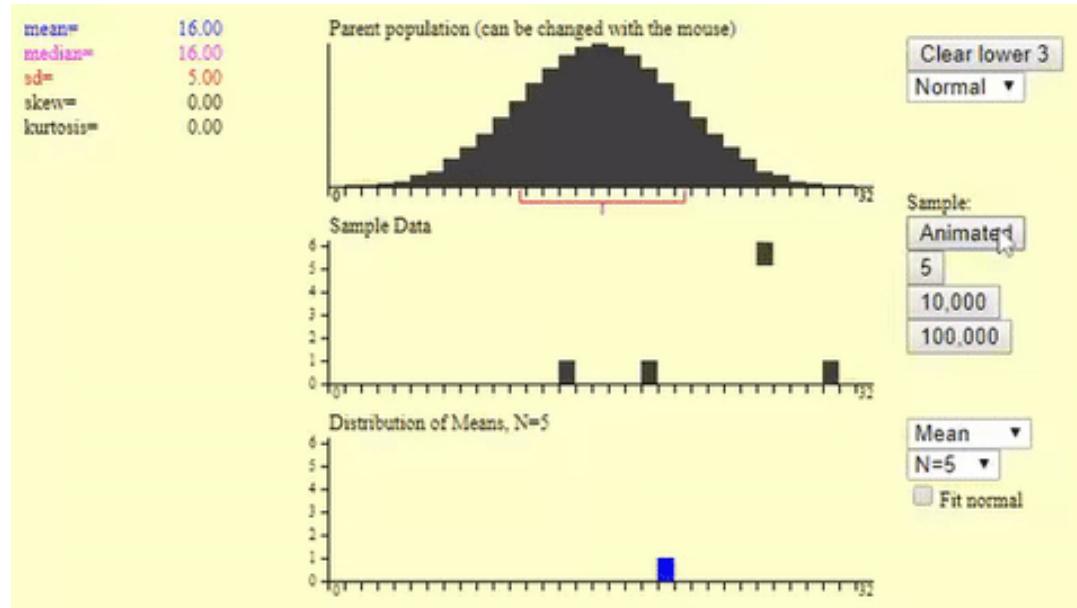


- Imagine we are sampling a distribution shown right.
- We take several samples (2 to 6)
- The red X in rows (2 to 6) represent the means of each sample
- Row 7, shows the distribution of our means taken from the samples.
- It follows a normal distribution!

Source: <https://www.superdatascience.com/courses/statistics-business-analytics-a-z/>

Try this experimental

http://onlinestatbook.com/stat_sim/sampling_dist/index.html?source=page-----a67a3199dcd4-----

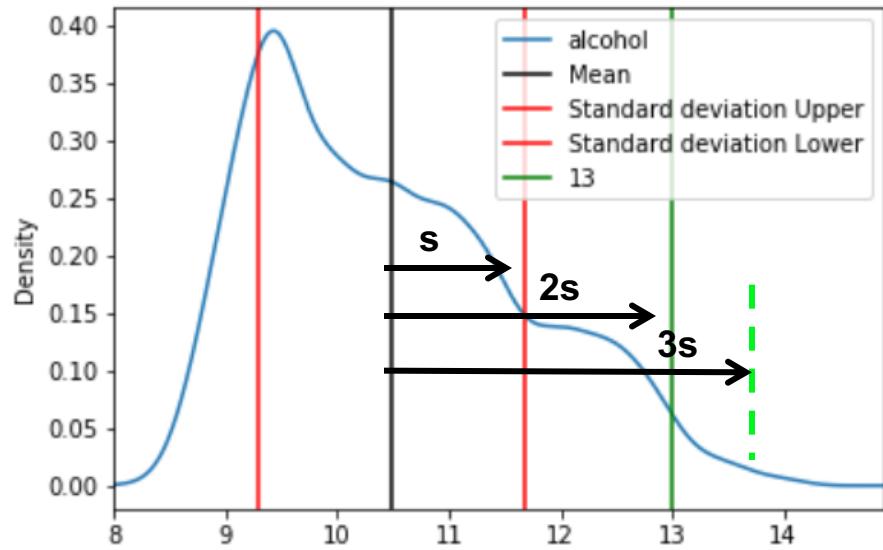


Z-Scores and Percentiles



Standard Deviations Revisited

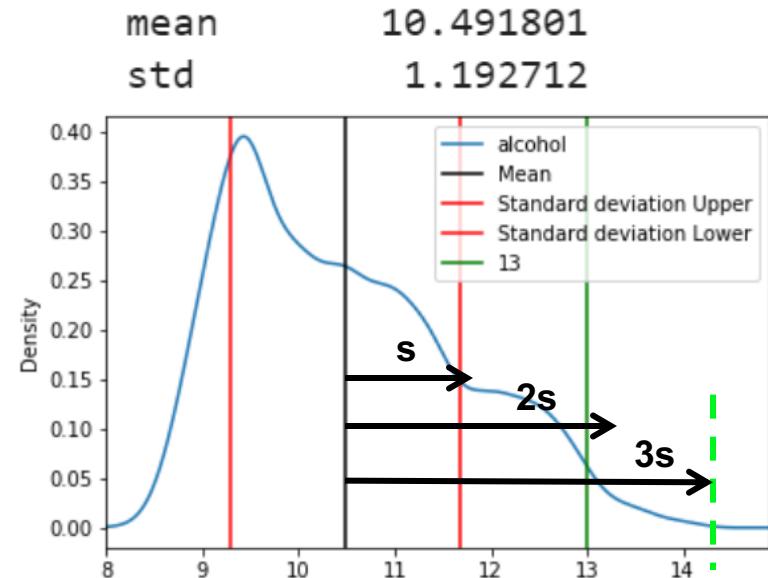
- Imagine we pull a random bottle of wine and see that its alcohol content is 13%
- From our STD plot we see that 13% exceeds the mean alcohol content.
- How different to the mean is 13% really?
- It's ~2 standard deviations away from the mean



Z-Score

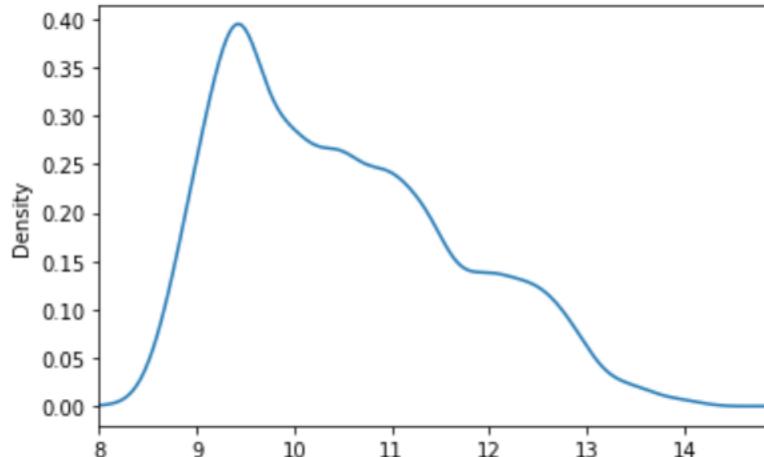


- Logically, what we did was take our 13% sampled value (x) and subtract the mean, then divide that by the standard deviation.
 - $$Z = \frac{x - \mu}{\sigma} = \frac{13 - 10.49}{1.19} = 2.11$$
 - This is what Z-Scores are, a distance measured in standard deviations.
 - + z-scores indicate the value is larger than the mean and negative z-scores imply the opposite

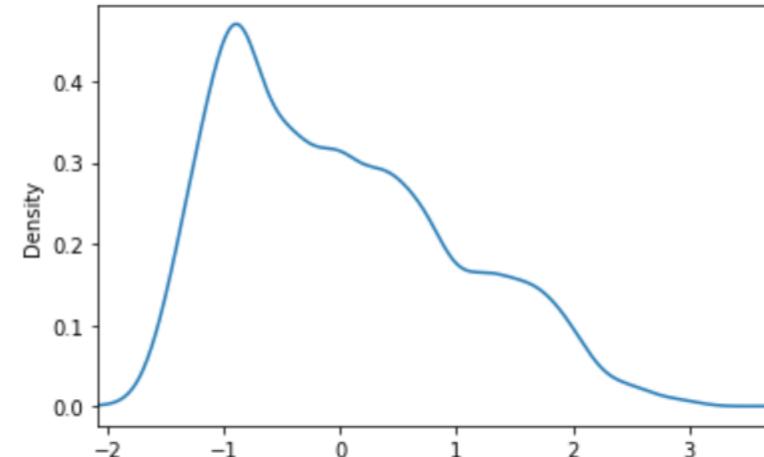


Transforming entire Distributions to Z-Scores

Original Distribution



Z-Transform



- Note the shape of distribution is entirely preserved!



Z-Score Means are always 0 & Standard Deviations always 1

```
z_mean_area = df['z_scores'].mean()  
z_stdev_area = df['z_scores'].std(ddof = 0)  
print(z_mean_area)  
print(z_stdev_area)
```

```
-4.905973358136756e-14  
1.000000000000064
```



Why do we use Z-Scores?

- Z-scores in general allow us to compare things that are NOT in the same scale, as long as they are NORMALLY distributed.

- Example, think of examination marks for two different subjects:
 - Ryan's mark is 24/40, the class mean was 22, STD = 10
 - Sara's mark is 39/50, the class mean was 34, STD = 15
 - Who performed better relative to their peers?
 - Ryan's z-score = $(24-22)/10 = 0.2$
 - Sara's z-score = $(39-34)/15 = 0.3$
 - Therefore, Sara performed marginally better!



Z-Scores and Percentiles

- This brings us to how we can use Z-Scores to determine your percentile. Let's go back to our previous example
- Ryan's Z-Score was 0.2 and Sara's 0.3
- Ryan is the 0.5793 percentile, meaning he's scored better than 57% than his

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549

Source: <http://i1.wp.com/www.sixsigmastudyguide.com/wp-content/uploads/2014/04/z-table.jpg>

Probability – An Introduction



Coin Flipping – What are the odds of a Heads?



- We know, for a fair coin, it's a 50:50 chance.\
- Meaning, there's\ an equal likeliness we get Heads or Tails
- **Probability** is a measure quantifying the likelihood that events will occur.



Why is Probability Important?

- In statistics, data science and Artificial Intelligence, obtaining the probabilities of events is essential to making a successful 'smart' or in the Business World, a calculated risk.
 - What is the probability of my Marketing Campaign succeeding?
 - What is the probability that Customer A will buy products X, Y & Z?
- In conclusion - Accurately Estimating Probability, given some information, is our goal.



Probability Scope

- In this section, we'll learn to estimate probabilities both theoretically and empirically
- The rules of Probability
 - The Addition Rule
 - The Product Rule
- Permutations and Combinations
- Bayes Theorem

Estimating Probability



Empirical or Experimental Estimates

- If we tossed a coin 100 times, we'd expect something like the following:
 - 45 heads
 - 55 tails
- This gives us a Probability of getting a heads, written as:
 - $P(\text{heads}) = \frac{45}{100} = 0.45$
- Empirically, probability of an event happening is:
 - $P(\text{Event}) = \frac{\text{number of times event occurred}}{\text{number of trials repeated}}$



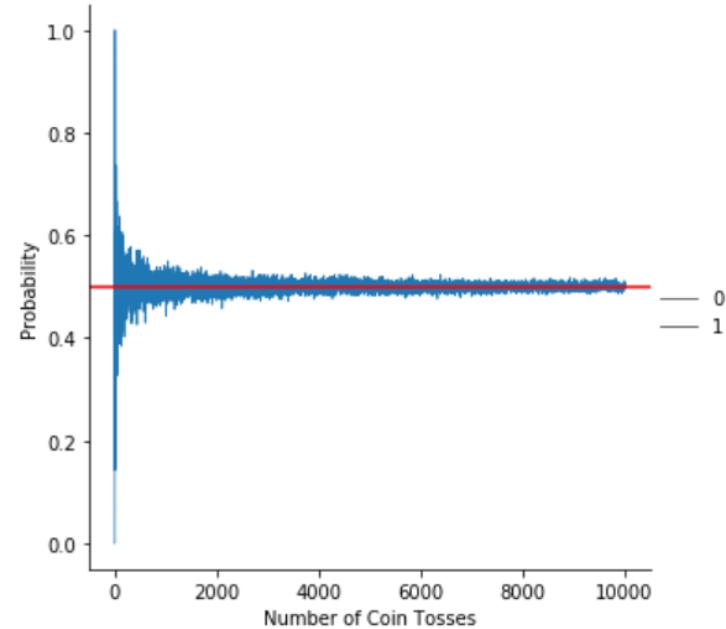
Empirical or Experimental Estimates

- Why isn't the probability after 100 experiments 0.5?
- Well it could have been, but due to the random nature of coin tosses it's possible we would have some degree of variance.
- However, if we performed 1000 coin tosses our results may look something like:
 - Heads – 490
 - Tails – 510
 - Probability (Heads) = 0.49



Let's attempt this in Python

- We'll make a function that generates random numbers of 0 and 1 to represent heads and tails.
- As we can see, the more trials we attempt the closer it gets to the actual value of 0.5





Probability as a Percent

- Often times you'll see probabilities represented as a percentage or a value between 0 and 1.
- NOTE: mathematically we work with the probability value always being between 0 and 1.
- $0 \leq P(Event) \geq 1$



Probabilities of All Events Must Sum to 1

- Let's consider a fair six sided dice.
- The probability of getting a 1 is $1/6$
- As such, the probability of each number is $1/6$





Simple Probability Question

- We have a bag of 30 marbles
 - 12 Green
 - 8 blue
 - 10 black
- What are the chances of pulling a black one out at random?
 - $P(\text{black}) = \frac{10}{12+8+10} = \frac{10}{30} = 0.3$



Probability – Addition Rule



Probability and Certainty

- From our random experiments, we get a very good idea of the chance or likeliness of an event occurring. However, it's still a prediction based on random experimentation and even if an event is 99% likely to occur, we can expect with some certainty that the 1% event can occur once in a 100 trials
- In a coin toss event, we have two outcomes: Heads or Tails
- In a dice roll we have 6 possible outcomes for each event.
- The Omega symbol Ω is used to signify the sample space i.e. all possible outcomes, this is known as the **Sample Space**:
 - $\Omega(\text{dice}) = \{1, 2, 3, 4, 5, 6\}$
 - $\Omega(\text{coin}) = \{\text{Heads}, \text{Tails}\}$



Probability Multiple Events

- Imagine we now have two coins being tossed simultaneously



- Our possible outcomes or sample space is now:

$$\bullet \quad \Omega(Coin_1, Coin_2) = \left\{ \begin{array}{l} (Heads, Heads), \\ (Heads, Tails), \\ (Tails, Tails), \\ (Tails, Heads) \end{array} \right\}$$



Outcomes with Two Coins

- We now have 4 possible outcomes and we can see each outcome has a 1 in 4 chance of happening:
 - $P(\text{Head, Tails}) = 0.25$
 - $P(\text{Heads, Heads}) = 0.25$
 - $P(\text{Tails, Tails}) = 0.25$
 - $P(\text{Tails, Heads}) = 0.25$
- Let's now take a look at the Probability Rules concerning multiple (two or more events) e.g. rolling two dice



The Addition Rule

- Let's roll a dice and answer the question:
 - What is the probability of getting a 1 or a 6?

- We know the probability of getting any digit in one roll is $\frac{1}{6}$
 - $P(1) = \frac{1}{6}$
 - $P(6) = \frac{1}{6}$
 - $P(1 \text{ or } 6) = \frac{1}{6} + \frac{1}{6} = \frac{2}{6} = \frac{1}{3}$





The Addition Rule

- $P(A \text{ or } B) = P(A) + P(B)$
- This is also written as:
 - $P(A \cup B) = P(A) + P(B)$
- As this rule works for two or more events, this also holds up:
 - $P(A \cup B \cup C) = P(A) + P(B) + P(C)$



Another Example

- We have a bag or 30 marbles
 - 12 Green
 - 8 blue
 - 10 black
- What are the chances of pulling a green or blue one out at random?

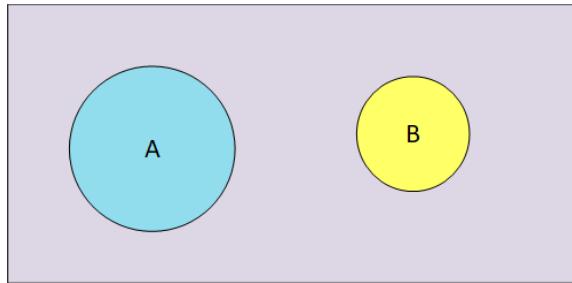


$$P(\text{green} \cup \text{blue}) = \frac{12}{12 + 8 + 10} + \frac{8}{12 + 8 + 10} = \frac{12}{30} + \frac{8}{30} = \frac{20}{30} = \frac{2}{3}$$



More on the Addition Rule

- Previously our events were **Mutually Exclusive**, that is both outcomes cannot both happen.
- We can illustrate this using a Venn diagram to show there is no intersection between the events.





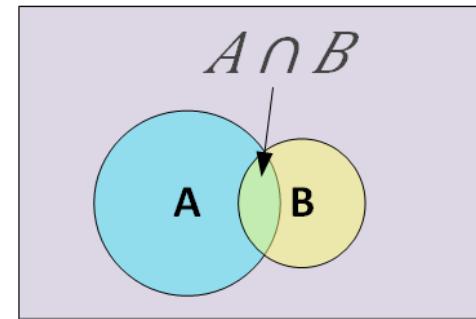
Example of Mutually Exclusive Events

- Let's say we have a class of 13 students, 7 female and 6 male.
- We're forming a small committee involving 3 of these students
- What is the probability that the committee has 2 or more female students?
- Let's consider the events that allow this possibility
 - Event A – 2 females and 1 male is chosen
 - Event B – 3 females are chosen
- These two events cannot occur together, meaning they're mutually exclusive.



Example of Non-Mutually Exclusive Events

- In a deck of cards, what is the Probability we pull at random, a card that is either a King or a Hearts?
- $P(\text{King}) = P(A) = \frac{4}{52}$
- $P(\text{Hearts}) = P(B) = \frac{13}{52}$
- $P(A \text{ and } B) = P(A \cap B) = \frac{1}{52}$
- $P(A \text{ or } B) = P(A \cup B) = \frac{4}{52} + \frac{13}{52} - \frac{1}{52} = \frac{16}{52}$
- $P(\text{King or Heart}) = \frac{4}{13} = 0.31$





Addition Rule for Non-Mutually Exclusive Events

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$



Other Probability Problems

- What is the probability of pulling 3 Aces in row?
- There are 4 aces in a deck of 52 cards
- So, the probability of getting one ace is $\frac{4}{52}$
- So the probability of getting a second ace would be the same?
- Nope, there are now 3 aces left and 51 cards left.
- $P(3 \text{ Aces}) = \frac{4}{52} \times \frac{3}{51} \times \frac{2}{50} = \frac{1}{5525} = 0.000181 \text{ or } 0.0181\%$
- This is an example of **sampling without replacement**, if we were to place the Ace back into the deck, this would be **sampling with replacement**.

Probability – Permutations & Combinations



Permutations and Combinations

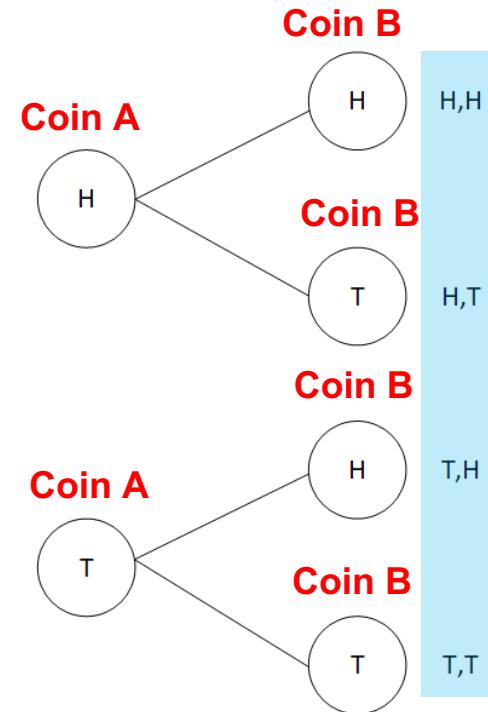
- At this point we have a good grasp of working out basically probabilities and understanding mutually exclusive events along with use of the Addition rule.
- But how do we go about working out the odds of winning your local Lottery? Or cracking the code in a 4-digit combination lock.





Determining the Number of Outcomes

- Let's explore tossing two coins:
- A – Coin 1
- B – Coin 2
- $\Omega(\text{coin}) = \{\text{Heads}, \text{Tails}\}$
- Number of Outcomes for A = 2
- Number of Outcomes for B = 2
- Total number of Outcomes = $a \times b = 2 \times 2 = 4$
- This is known as the Product Rule





Number of Outcomes

- Exploring this concept further, let's consider a combination lock with a 2-Digit code.
- What the odds of guessing the code in your first guess?

- $P(E) = \frac{\text{Number of Successes}}{\text{Total Number of Outcomes}}$

- $\Omega_1 = \{0,1,2,3,4,5,6,7,8,9\}$

- $\Omega_2 = \{0,1,2,3,4,5,6,7,8,9\}$

- $P(E) = \frac{1}{10 \times 10} = \frac{1}{100} = 0.01$

- We can extend this to any combination of locks:

- 4 Digit = $\frac{1}{10,000} = 0.0001$





Permutations

- As we just saw, we explored an ordered set numbers in order to get the probability of guessing the combination lock.
- An ordered combination of elements is called a Permutation
- Order matters because, let's say the unlock code was 1234, reordering the sequence to 4321 or 2341 would not work. Order is important!



Should really be called a
“Permutation Lock”



Two Types of Permutations

- **Permutations with repetition** - In our previous example with the combination lock, we can see digits can be repeated.
 - Therefore, calculating the number of possibilities is quite easy.
 - Total Number of Outcomes for N choices = $\Omega = n_1 \times n_2 \times n_3$
- **Permutations without repetition** – These are instances where once an outcome occurs, it is no longer replaced.
 - Let's look at 16 numbered pool balls
 - When randomly choosing our first ball, there are 16 possibilities.
 - After choosing this ball, it is removed, so we're now left with 15
 - To calculate the number of possible combinations we do:
 $16 \times 15 \times 14 \times 13 \times 12 \dots \times 1$ this is known as 16! Or 16 Factorial
 -





Permutations Without Repetition

- Let's say we wanted to choose 3 random pool balls from our set of 16
- How many different combinations of balls can there be? E.g. 5,3,12 or 15,2,6 etc.
- Simple $\frac{16!}{13!} = 3360$ permutations
- The formula we apply is written as:



- $$\frac{n!}{(n-r)!}$$
- Where 'n' is the number of possible things to choose from and r is the number of repetitions



Combinations

- Before, we just looked at the two types of Permutation (with and without repetition). In Permutations order mattered.
- In the scenario where it doesn't matter is called **Combinations**

Order does matter	Order doesn't matter
1 2 3	1 2 3
1 3 2	
2 1 3	
2 3 1	
3 1 2	
3 2 1	



Calculating Combinations Example

- If we don't care about the order, the number of Combinations is given by:

- $$\frac{n!}{r!(n-r)!}$$

- $$\frac{16!}{3!(16-3)!} = \frac{16!}{3! \times 13!} = \frac{20,922,789,888,000}{6 \times 6,227,020,800} = 560$$



Combinations with Repetition

- Now this brings us to the last bit of this chapter where we look at Combinations with Repetition.
- In our last slide, the values were not repeated e.g. in the case of combinations of 1,2,3 we could only use one digit once.
- In the situation where they can be repetition the formula is as follows:
 - $\frac{(r+n-1)!}{r!(n-1)!}$

Bayes Theorem



Introduction

- At this point, if you're familiar with Statistics and Probability you would have realized we skipped topics involving:
 - Joint Probability
 - Marginal Probability
 - Conditional Probability
- These are probability theorems are used for two or more dependent/independent events.
- However, to teach this properly will require a couple hours and practice on your part!



Introduction to Joint, Marginal, and Conditional Probability

- **Joint probability** is the probability of two events occurring simultaneously.
 - $P(A \text{ and } B)$
- **Marginal probability** is the probability of an event irrespective of the outcome of another variable. E.g. Probability of a card being a Red Queen = $P(\text{red and queen}) = 2/52$
 - $P(\text{Event } X = A \text{ for all outcomes of event } Y)$
- **Conditional probability** is the probability of one event occurring in the presence of a second event. E.g. you pulled a black card from a deck, what is the probability of it being a ten. $P(\text{ten}|\text{black}) = 2/26 = 1/13$
 - $P(A \text{ given } B) \text{ or } P(A|B)$



Independence and Exclusivity

- As we mentioned previously, it's possible for Events A and B to be either dependent on each other or completely independent.
- If the events cannot occur simultaneously, this is called **Exclusivity**.

For Independent Events:

- The **Joint Probability** of two independent events is given by:
 - $P(A \text{ and } B) = P(A) * P(B)$
- For events that are independent, the **Marginal Probability** is given by:
 - $P(A) = P(A)$
- Similarly, for the **Conditional Probability**, it's given by:
 - $P(A|B) = P(A)$



Independence and Exclusivity

For Exclusive Events:

- The **Joint Probability** of two exclusive events, i.e. both of them occurring together is **NOT possible**, and therefore it's equal to zero:
 - $P(A \text{ and } B) = 0$
- For Probabilities of the either event occurring though is given by their sums :
 - $P(A \text{ or } B) = P(A) + P(B)$
- If the events are **NOT** mutually exclusive, meaning they can occur together
 - $P(A|B) = P(A)$



Bayes Theorem

"In probability theory and statistics, Bayes' theorem (alternatively Bayes' law or Bayes' rule) describes the probability of an event, based on prior knowledge of conditions that might be related to the event. For example, if cancer is related to age, then, using Bayes' theorem, a person's age can be used to more accurately assess the probability that they have cancer than can be done without knowledge of the person's age." Wikipedia

- $P(A|B) = \frac{P(A \text{ and } B)}{P(B)}$



Bayes Theorem Demonstrated

Let's explore a practical example:

- 1% of women have breast cancer (99% do not)
- 80% of mammograms detect cancer correctly, so 20% of them are wrong
- 9.6% of mammograms report cancer being detected when it is in fact not present (False Positive)

	Cancer (1%)	No Cancer (99%)
Positive Test	80%	9.6%
Negative Test	20%	90.4%



Bayes Theorem Demonstrated

- So let's suppose you or someone you know unfortunately, tests positive for breast cancer.
- This means got top row outcome.
- Probability of a **True Positive** (meaning we have cancer and it was a positive result) = **1% x 80% = 0.008**
- Probability of a **False Positive** (meaning we don't have cancer and it was a positive result) = **99% x 9.6% = 0.09504**

	Cancer (1%)	Cancer (99%)
Positive Test	80%	9.6%
Negative Test	20%	90.4%



Bayes Theorem Demonstrated

- Remember, Probability is equal to:
- $P(\text{Event}) = \frac{\text{Event}}{\text{all possibly outcomes}}$
- $P(\text{Having Cancer}|\text{Positive Test}) = \frac{0.008}{0.008+0.09504} = \frac{0.008}{0.10304} = 0.0776 \text{ or } 7.8\%$

	Cancer (1%)	Cancer (99%)
Positive Test	80% 1% x 80% = 0.008	9.6% 99% x 9.6% = 0.09504
Negative Test	20%	90.4%



Bayes Theorem Problem Summary

$$P(\text{Cancer}|\text{Positive}) = P(C|P) = \frac{P(P|C)P(C)}{P(P|C)P(C) + P(P|\sim C)P(\sim P)}$$

$P(C|P)$ = Probability of having Cancer and having a Positive test = $\frac{7}{8}\%$

$P(P|C)$ = Probability of a Positive test given that you had Cancer = 80%

$P(C)$ = Probability of having Cancer = 1%

$P(\sim C)$ = Probability of not having Cancer = 99%

$P(\sim P)$ = Probability of Test being Negative

$P(P|\sim C)$ = Chance of Positive test given you did not have Cancer = 9.6%

Hypothesis Testing Introduction



What is a Hypothesis?

- A hypothesis is a proposed explanation (**unproven**) for a phenomenon.
- Examples of Hypothesis are:
 - If I eat less sugar, I will lose weight faster
 - If I drink more water, I'll feel more energized
 - If we use this advertising slogan, we'll increase sales
- **Testing or proving a Hypothesis** is a very important field of Statistics and is immensely helpful in the Business world.



Testing a Hypothesis

- Take the hypothesis “**If we change the color of the *Add to Cart* button on our e-commerce website, we'll increase sales**”.
- To test this, we can foresee many problems.
 - What if the change was carried out when the end of the month was approaching (i.e. sales were naturally going to increase)?
 - What if, the company increased ads around that same time?
 - What if there was some random event that resulted in a chance in sales?
- As you can see, there are many difficulties when testing Hypothesis, in reality this is one of the controversial use of Statistics.



Framing of Hypothesis

Null Hypothesis

- This describes the existing conditions or present state of affairs.

Examples:

1. All Lily's have 3 petals
2. The number of children in a household is unrelated to the amount of televisions owned
3. Changing the *Add to Cart* button color on our e-commerce website, will have no effect on our sales



Framing of Hypothesis

Alternative Hypothesis

- This is used to compare or contrast against the Null Hypothesis
- Example: *If we change the color of the Add to Cart button on our e-commerce website, we'll increase sales.*

Null Hypothesis – Users exposed to the new *Add to Cart* button did not change their tendency to make a purchase

Alternative Hypothesis – Users exposed to the new button *Add to Cart* button resulted in increased sales.

Statistical Significance



Research Design – Blind Experiment

A common practice used in drug testing by pharmaceutical companies is drug testing. Let's design an experiment to investigate whether a new flu medication reduces the length of time flu symptoms were experienced.

Setup:

1. We need a group of persons, let's get about 80 volunteers
2. Separate the population sample into 2 equal groups of 40
 - **Group 1 (control group)** – 40 Volunteers given a placebo or fake pill
 - **Group 2 (treatment or experimental group)** – 40 Volunteers given the real pill





Formalize our Hypothesis

Null Hypothesis

Subjects taking the new flu medication did not change the duration of the flu symptoms compared to those who took the placebo.

Alternative Hypothesis

Subjects taking the new flu medication had the duration of their flu symptoms reduced compared to those who took the placebo



Research Design – Blind Experiment

- In blind experiments, users are kept **uninformed** about the pill they are taking so that we avoid changes in the user behavior. Example those given the fake pill may attempt other remedies to their flu, affecting the final results
- Other caveats and issues with such a test is that assessing the participants on whether they really have the flu or is it some other strain of the common cold?
- Persons who know they're taking pill make take less rest assuming the pill will help, thus prolonging or worsening symptoms.



Statistical Significance

- Once we obtain the results from the two groups, how do we interpret the results with any degree of certainty?

Take a look at these results:

- Group 1 (control group) had flu symptoms for – 5.4 days
- Group 2 (experiment group) had flu symptoms for – 4.8 days



Comparing our Means

$$\bar{x}_1 = 5.4$$

$$\bar{x}_2 = 4.8$$

Our Null Hypothesis stated:

- $\bar{x}_1 - \bar{x}_2 = 0$

Our Alternative Hypothesis stated:

- $\bar{x}_1 - \bar{x}_2 > 0$
- $5.4 - 4.8 = 0.6 > 0$

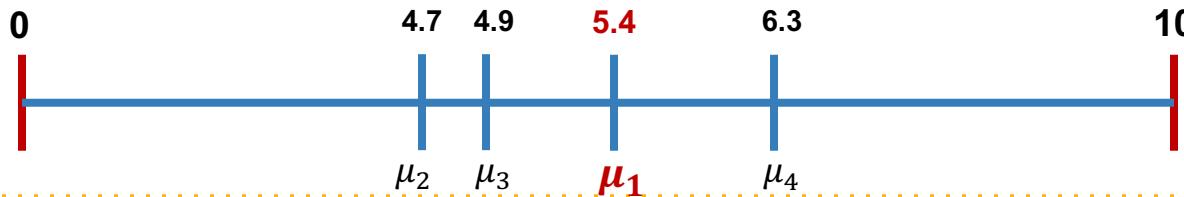
Therefore, 5.4 is greater than 4.8.

- **Should our Null Hypothesis be rejected?**



Would this Happen Every Time?

- **Thought Experiment** – We got our means to be:
 - Group 1 (control group) had flu symptoms for – **5.4 days**
 - Group 2 (experiment group) had flu symptoms for – **4.8 days**
- Ask yourself, if you were to repeat the experiment another time? Would the means be exactly the same?
- Our sample size was relatively small at 40 persons each, meaning things can change
- Imagine if we run this experiment on our Placebo/Control group several times, each time we get a new mean.





Permutation Test

- Each time we redo this experiment is called a **Permutation Test**.
- We do this so we can calculate a distribution of the test statistics for each trial or iteration of this experiment
- We call this distribution the **Sampling Distribution**



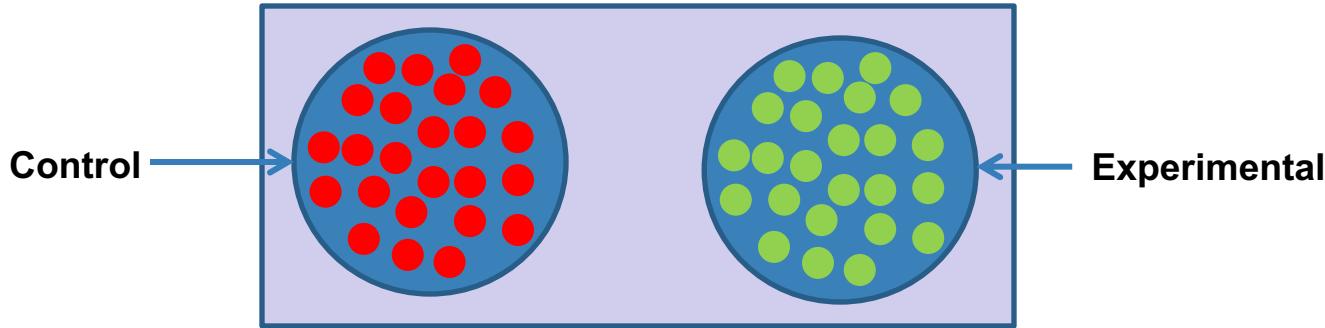
Sampling Distribution

- Our sampling distribution approximates a full range of possible test statistics for the null hypothesis.
- We then compare our control group mean (5.4) to see how likely it is to observe this mean.
- We do this by re-running our control group test several times and observe whether a mean of 5.4 is likely or rare.



Simulation of Re-trails

- In the real world we can't re-run our control group study several times, due to time/money and effort constraints.
- However, we don't need to. Here's the trick.





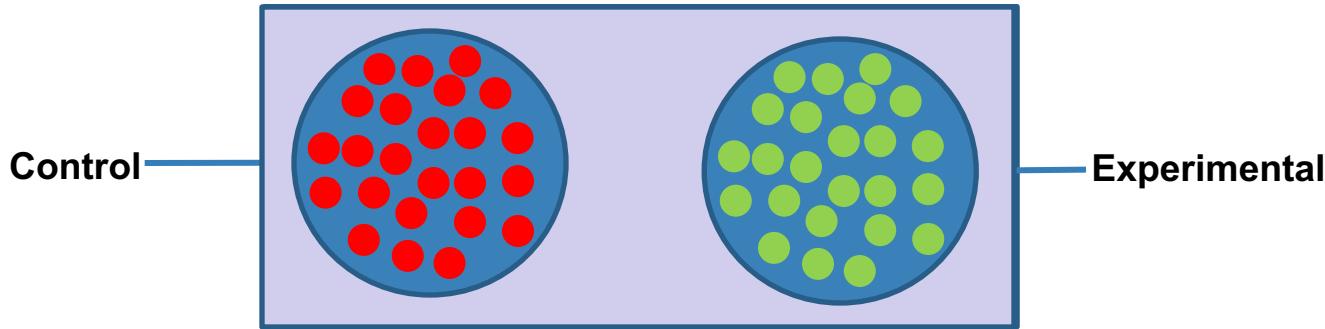
Recall Our Results

Subject	Group (Randomly Assigned)	Flu Duration
1	Control	4 Days
2	Control	3 Days
3	Experimental	3 Days
4	Control	5 Days
:		
80	Experimental	5 Days



Simulation of Re-trails

- We now take some of the individual data points and randomly assign them to the other group.
- We then calculate the new mean after this randomization





Our Randomized Assignments

- Total in each group remains the same (i.e. 40 each)

Subject	Group (Randomly Assigned)	Flu Duration
1	Control → Experimental	4 Days
2	Control	3 Days
3	Experimental → Control	3 Days
4	Control → Control	5 Days
:	:	:
80	Experimental → Experimental	5 Days



Record the Means of Each Simulated Trial

- For each iteration or trial, we log the mean and the Mean Difference
- The Mean Difference is our initial mean of 5.4 (Control) minus the mean of randomized trail re-assignments.

Simulation Number	Mean of Control Group	Mean Diff
1	5.3	$5.4 - 5.3 = 0.1$
:	:	
10,000	5.1	$5.4 - 5.1 = 0.3$

- We use these means to create our **Sample Distribution**

Hypothesis Testing – P Value

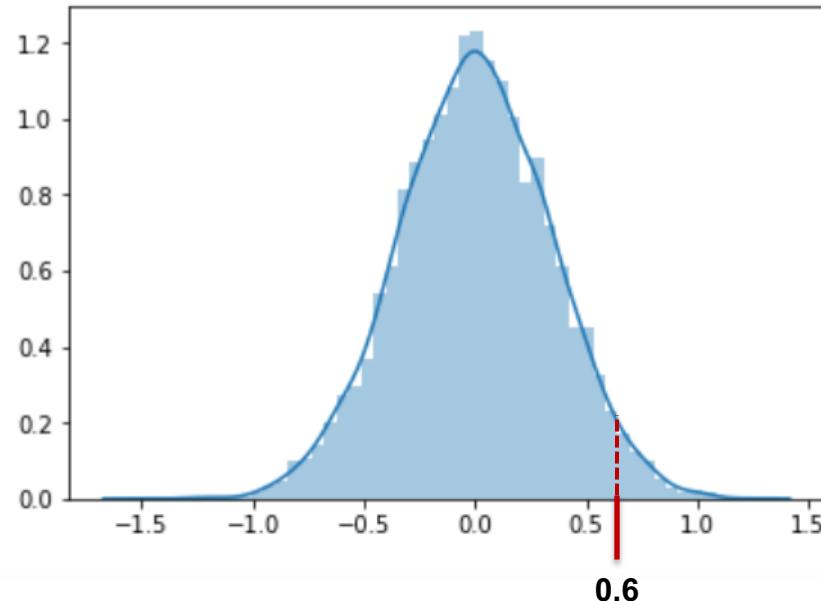


\

Let's run these Simulations in Python



Let's run these Simulations in Python



- This plot shows the Normal Distribution of our Sample Distribution of Mean Differences.
- Remember our mean difference was $5.4 - 4.8 = 0.6$



What can we take away from this?

- We see that our real life mean difference of 0.6 lies a bit to the far right.
- We see that the mean in our experimental iterations is almost zero.
 - This means that mean difference between groups is almost purely random and there is no real difference between groups
- But how do we show this statistically?
- What is the probability of a mean difference of 0.6 occurring?
 - Lets check our mean difference values and see how many of these values were great or equal to 0.6



P-Value

- The Probability of exceeding our original Mean Difference ($5.4 - 4.8 = 0.6$) is known as our **P-Value**
- If our P-Value is **less than a pre-defined threshold** then we can reject the Null Hypothesis. This means that the difference in means between the control and experimental groups was **Statistically Significant**. In our example, this would mean that our medication works
- Common P-Value thresholds are usually **5% or 0.05**



Let's Determine our P-Value in Python

- In our experiment our P-Value was **0.0381** or **3.81%**
- This is below our P-Value Threshold, as such, our new flu medication **does appear work!**
- A P-Value threshold of **0.5** means that there's a **5%** chance the results can be attributed to random coincidence
- This means our difference in means i.e. **5.4** vs. **4.8** **was statistically significant.**



More on P-Values

- We've shown that our results **support our Alternative Hypothesis** – i.e. our new flu medication reduces the length of the flu.
- However, always remember P-values are the **probability** that what you measured is **the result of some random fluke**.
- Saying our P-Value is **3.81%** means our results have a **3.81% chance of being attributed to random coincidence**.
- However **3.81%** is small and is lower than the typical **5%** threshold used for P-Values. Therefore, **Statistically Significant**.

Hypothesis Testing – Pearson Correlation



Pearson Correlation Coefficient

- Pearson's correlation coefficient, r , tells us about the strength of the Linear Relationship between x and y points on a regression plot.
- What exactly do we mean by Linear Relationship?



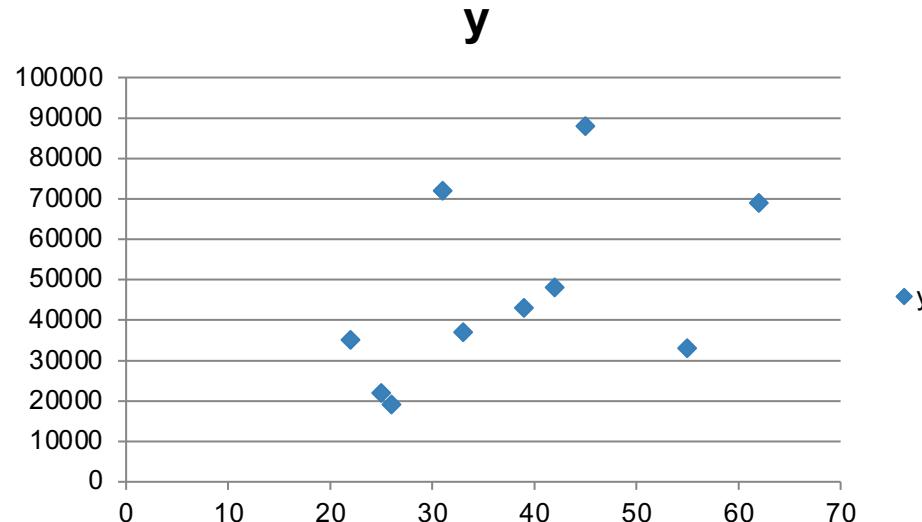
Pearson Correlation Coefficient

Subject	Age	Annual Income
1	22	35000
2	25	22000
3	45	88000
4	31	72000
5	33	37000
6	62	69000
7	42	48000
8	39	43000
9	26	19000



Pearson Correlation Coefficient

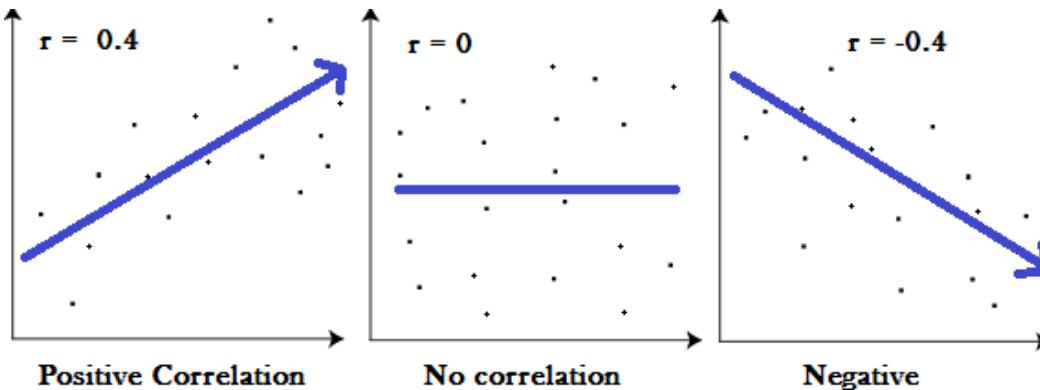
Subject	x	y
1	22	35000
2	25	22000
3	45	88000
4	31	72000
5	33	37000
6	62	69000
7	42	48000
8	39	43000
9	26	19000





Pearson Correlation Coefficient

- Hypothesis Testing with Pearson tells us whether we can conclude two variables correlate or influence each other in a way that is Statistically Significant
- r ranges from -1 to +1
 - Values close to 0 indicates no correlation
 - 1 indicates an inverse relationship and strong correlation
 - +1 indicates a positive relationship and strong correlation



r measures the strength and direction of a linear relationship between two variables on a scatterplot



Calculating the Pearson Correlation Coefficient

$$r = \frac{\sum x_i y_i - \frac{\sum x_i \sum y_i}{n}}{\sqrt{\left(\sum x_i^2 - \frac{(\sum x_i)^2}{n}\right)} \sqrt{\left(\sum y_i^2 - \frac{(\sum y_i)^2}{n}\right)}}$$

i = number of paired samples

$\sum xy$ = sum of products between paired scores

$\sum x$ = sum of x scores

$\sum x^2$ = sum of squared x scores

$\sum y^2$ = sum of squared y scores



Calculating our Pearson Correlation Coefficient

1. Null Hypothesis: No correlation between income and age
2. Define Alpha (our measure of significance strength, lower is stronger) – We'll use 0.05
3. Find Degrees of Freedom – Our sample has 10 subjects and the formula finding degrees of freedom is $DF = n - 2 = 8$ (in our case)
4. Use an r-Table to find the Critical Value of r (i.e. the threshold value we use to reject our Null Hypothesis). In our case using an Alpha of 0.05 we get a critical $r = 0.632$ from our r-tables. If our r is greater than our critical r , we reject the Null Hypothesis
5. Apply Formula:

$$r = \frac{\sum x_i y_i - \frac{\sum x_i \sum y_i}{n}}{\sqrt{(\sum x_i^2 - \frac{(\sum x_i)^2}{n})} \sqrt{(\sum y_i^2 - \frac{(\sum y_i)^2}{n})}}$$



r-Table

Numbers in the left column are degrees of freedom. Numbers in the top row are significance levels (alpha).

df(n-2)	0.1	0.05	0.02	0.01
1	0.988	0.997	1.000	1.000
2	0.900	0.950	0.980	0.990
3	0.805	0.878	0.934	0.959
4	0.729	0.811	0.882	0.917
5	0.669	0.754	0.833	0.874
6	0.622	0.707	0.789	0.834
7	0.582	0.666	0.750	0.798
8	0.549	0.632	0.716	0.765
9	0.521	0.602	0.685	0.735
10	0.497	0.576	0.658	0.708
11	0.476	0.553	0.634	0.684
12	0.458	0.532	0.612	0.661
13	0.441	0.514	0.592	0.641

<http://statisticslectures.com/tables/rtable/>



Calculating our Pearson Correlation Coefficient

Subject	x	y	x^2	y^2	xy
1	22	35000	484	1225000000	770000
2	25	22000	625	484000000	550000
3	45	88000	2025	7744000000	3960000
4	31	72000	961	5184000000	2232000
5	33	37000	1089	1369000000	1221000
6	62	69000	3844	4761000000	4278000
7	42	48000	1764	2304000000	2016000
8	39	43000	1521	1849000000	1677000
9	26	19000	676	361000000	494000
10	55	33000	3025	1089000000	1815000
SUM	380	466000	16014	2.637E+10	19013000



Calculating our Pearson Correlation Coefficient

$$r = \frac{\sum x_i y_i - \frac{\sum x_i \sum y_i}{n}}{\sqrt{(\sum x_i^2 - \frac{(\sum x_i)^2}{n})} \sqrt{(\sum y_i^2 - \frac{(\sum y_i)^2}{n})}}$$

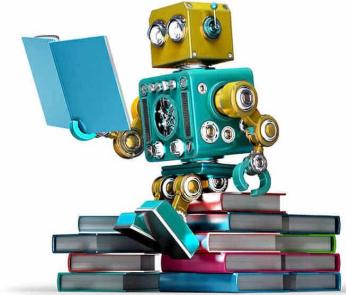
$$r = \frac{1305000}{(39.67)(68223.16)} = 0.482$$

- Does it exceed our critical r?
- No it does not, $0.482 < 0.632$. Therefore we accept the Null Hypothesis that there is no correlation.

Introduction to Machine Learning

Machine Learning

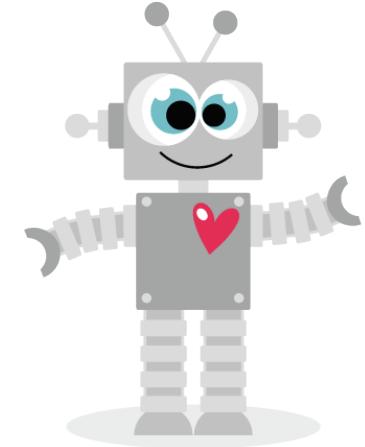
- Machine Learning is almost synonymous to **Artificial Intelligence (AI)** because it entails the study of how software can learn.
- It is a sub-field of AI that uses statistical techniques to give computer systems the ability to **"learn" from data, without being explicitly programmed.**
- It has seen **a rapid explosion in growth in the last 5-10 years** due to the combination of incredible breakthroughs in new algorithms such as Deep Learning, combined with almost exponential increases in CPU power, especially in parallel operations (GPU and TPU) which allowed for huge improvements in training Deep Learning networks.





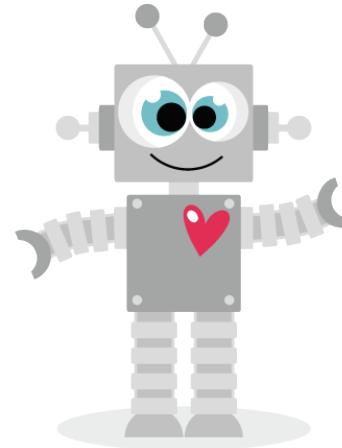
Explicit Programming

- We said Machine Learning allows software to learn without **explicitly programming** the information. But what do we mean by that exactly?
- If I asked you to teach this robot or computer to know which customer is most likely to buy an American football, how would you do it?
- You'd probably set up some predefined rules on finding the most likely customer. Something like young males under 25.

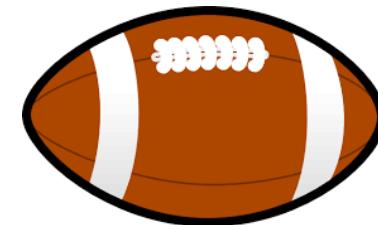




Imagine you're selling these footballs at the Entrance of a large General Store. You can't approach everyone, so who do you approach?

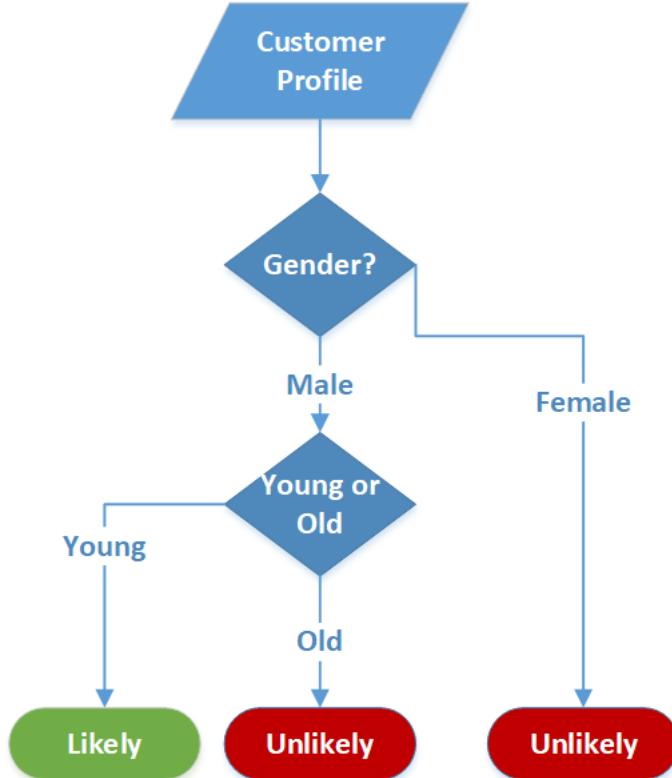


You





We would **create some Explicit** rules to determine which customer will buy the American football





Explicit Programming is difficult

- Now, imagine we had to program these rules into a computer system, and imagine we had access to a lot more information:
 - Gender
 - Age
 - Education
 - Income
 - Location
 - Past Purchases
- Creating an Explicit Rule Based system for each product will take extremely long, be prone to mistakes and generally not feasible.



Explicit Programming is difficult – Trust me!

If male then

 If age is between 18 and 25 years then:

 If location is City A then:

 If income between 10000 and 20000 then:





We Need a Better Solution!

- Machine Learning involves using algorithms that can learn from data without any explicit programming by the user.
- Let's understand how this works

How Machine Learning enables Computers to Learn



Machine Learning to the Rescue!

- Machine Learning allows us to create a **feasible method of teaching a program/algorithm to learn from data**.
- So what do we **need** to make this work?
- Imagine we had the customer profiles and their past purchase history
- Using some machine learning algorithm, we can feed this data to the computer and create a model that can tell us if a customer is likely to purchase a specific product!



Our input data

Customer ID	Gender	Age	Location	Income	Purchase History
00000001	M	19	Florida	8,000	Items: A, AC, Y, AB...
00000002	M	22	New York	60,000	Items: FR, W, V, EB...
00000003	F	18	Maine	12,000	Items: A, CA
00000004	M	54	Illinois	120,000	Items: U, C, YT, FR
00000005	F	36	Washington	90,000	Items: ZG, B, Y, OI
:	:	:	:	:	:
0000XXXX	F	26	California	34,000	Item: YU



Teaching a Machine Learning Algorithm

- Based on a customer's purchase history and other attributes, we can use this data to teach computer how to learn.
- **But How?** In simple terms, we use statistical methods to understand what attributes were associated with customers who purchased footballs.
- For example, our machine learning model can find that customers who purchased footballs were mostly young males who lived in warmer climates and had previously purchased other athletic equipment





How does Machine Learning Work?

- Generally we can say Machine Learning algorithms work by looking at **examples**, or what we appropriately call **Training Data**
- This is very much the way humans learn too!





Human Learning Example

- An ice cream salesman over time will know who the best target customers are (hint: children)
- Thus he/she would use things like balloons or musical ice cream trucks to get their attention





Another Human Learning Example

- Babies learn the names of animals by viewing pictures of them
- We train our machine learning algorithms much the same way





Example of Training Data

Gender	Age	Credit Rating	Declared Bankruptcy
Male	34	650	Yes
Female	45	900	No
Male	23	850	No
Male	29	890	No
Female	44	790	Yes
:	:	:	:
Female	39	954	No



We can then assess the Performance of our Machine Learning **Model** on **Test Data**. Basically it's just like Tests or Examinations we did in school

Gender	Age	Credit Rating	Declared Bankruptcy	
			Answer from our trained Machine Learning Model	What actually Happened (True Answer)
Female	54	754	No	Yes
Male	25	897	No	No
Female	63	861	Yes	Yes
Male	39	808	No	No
Male	24	690	Yes	Yes
:	:	:	:	
Male	59	859	No	No

What is a Machine Learning Model?



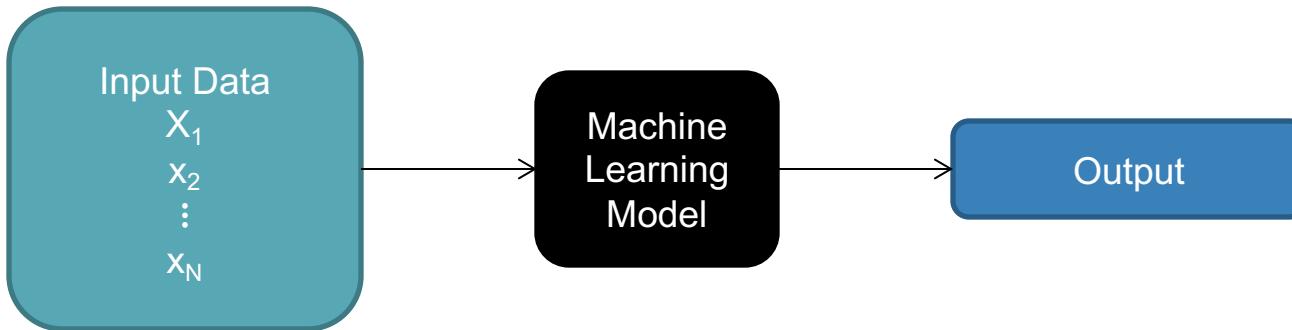
What is a Machine Learning Model?





Machine Learning Models are Equations!

- Machine Learning models are 'just' **mathematical equations** that transform our input data into the output we desire!



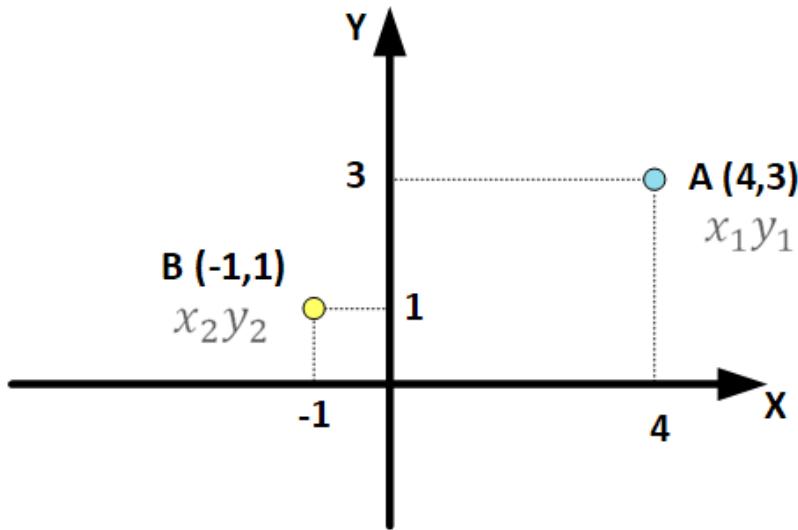


What do these Equations look like?

- **Model** = $w_1x_1 + w_2x_2 + w_3x_3 + b = \mathbf{w}^T \mathbf{x} + b$
- **What is this equation?**
 - x represents our input data
 - w represents our weights
 - b represents our bias weight
- **Still confused? Don't worry, let's go through a simple example**



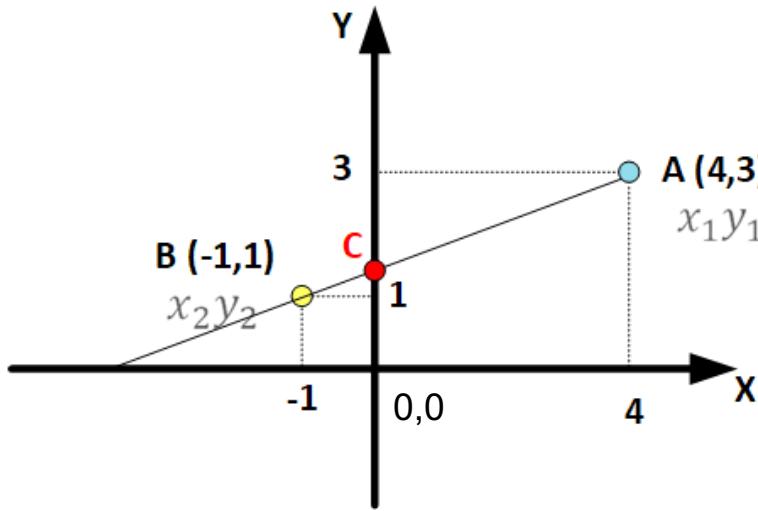
Finding the Gradient of Line given two points



- $m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta Y}{\Delta X}$
- $m = \frac{1-3}{-1-4} = \frac{-2}{-5} = \frac{2}{5}$
- The equation for a straight line is:
 - $y = mx + c$
- We need to find c , which is the y intercept (i.e. when $x = 0$)



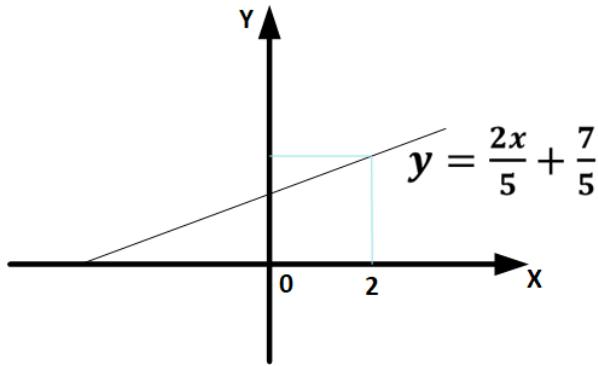
Finding the Equation of line given two points



- $y = mx + c$
- $1 = \frac{2}{5}(-1) + c$
- $1 + \frac{2}{5} = +c$
- $c = \frac{7}{5}$
- $y = \frac{2x}{5} + \frac{7}{5}$
- $5y = 2x + 7$



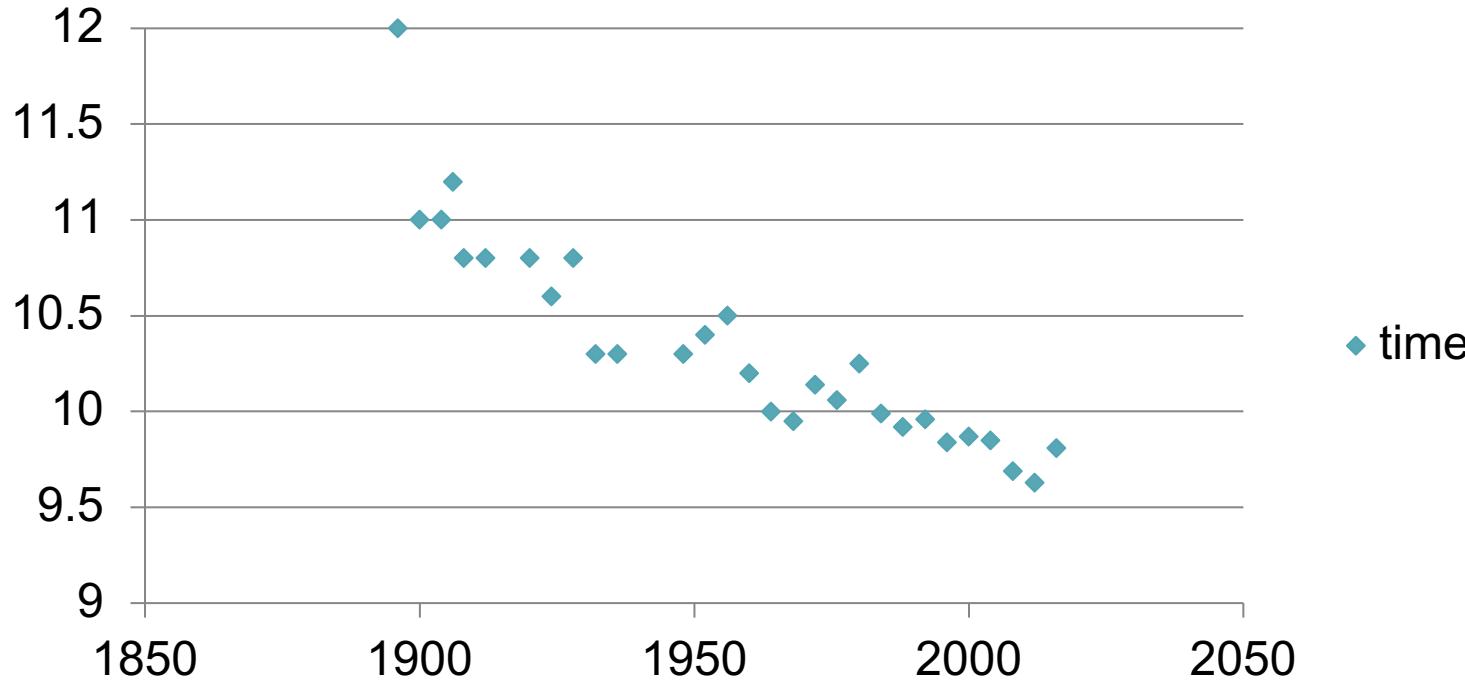
Why did we do this?



- We can use our equation now to predict values!
- If we consider **x** to be our **input** and **y** to be our **output**, we've just created a very simplistic model.
- Let's say we want to know the value of **y** when **x** = 2:
- $y = \frac{2(2)}{5} + \frac{7}{5} = \frac{11}{5} = 2.2$



Olympic 100m Gold Times Over Time

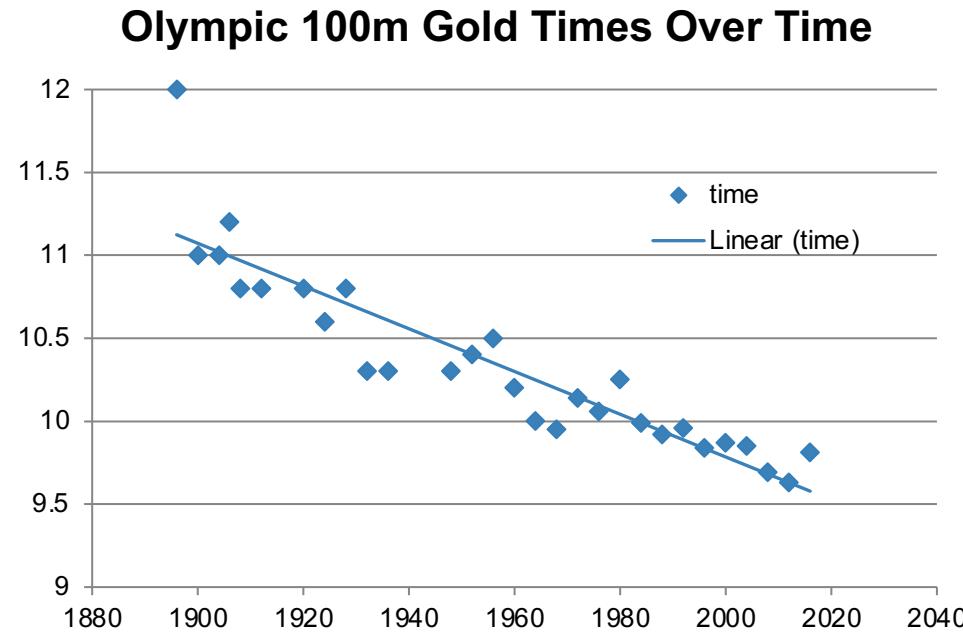




We'll use Least Squares Method to get Equation of the line

Predict 2020 Olympic time

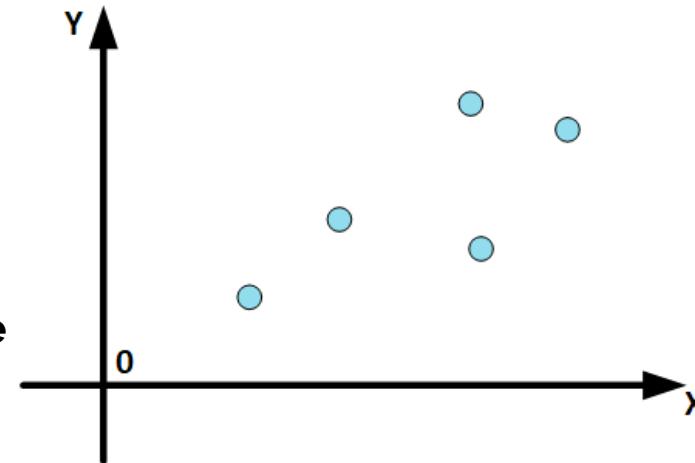
- $y = -0.01289x + 35.554$
- $y = -0.01289(2020) + 35.554$
- $y = 9.53$





Least Squares Method

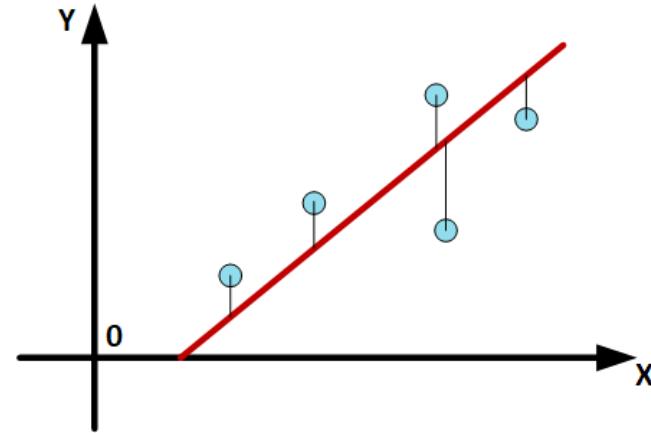
- Previously we used a simple formula to calculate the line equation when we had two points.
 - $m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta Y}{\Delta X}$
- The Least Squares Regression Line method is simply a way we apply the same method using several points.





Least Squares Method

- We use this method to find a **line** that makes the **vertical distance** from the data points as small as possible.
- It's called a "least squares" because the best line of fit is one that **minimizes the variance** (the sum of squares of the errors).
- The final equation that fits the points as closely as possible.



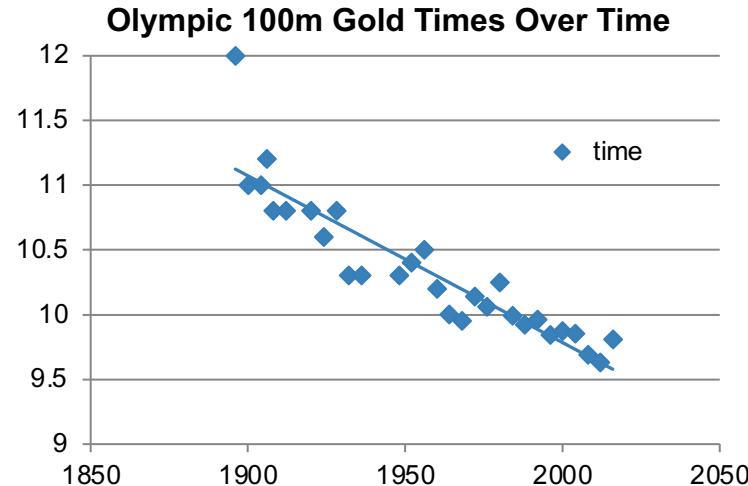
$$m = \frac{N \sum xy - \sum xy}{N \sum(x)^2 - (\sum x)^2}$$

$$c = \frac{\sum y - m \sum x}{N}$$



Recap – What is a Model

- A Machine Learning model is simply a **mathematical function** that transforms the inputs into an output relevant to our objective.
- Trying to predict the 2020 Olympic 100m winning Gold medal time?
- Use the inputs from past races as to create our prediction function 'y', then use $x = 2020$ to get our predicted winning time



$$y = -0.01289x + 35.554$$

$$y = -0.01289(2020) + 35.554$$

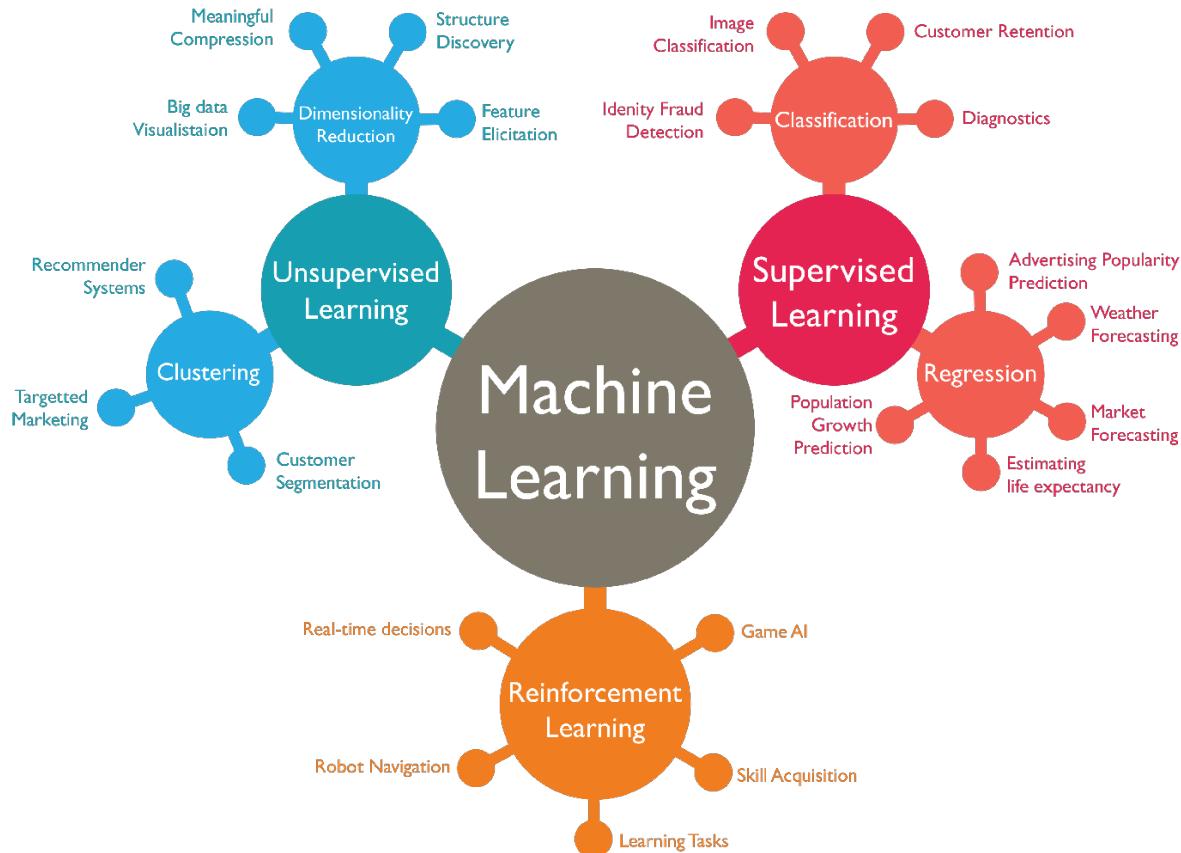
$$y = 9.53 \text{ seconds}$$

Types of Machine Learning



Types of Machine Learning

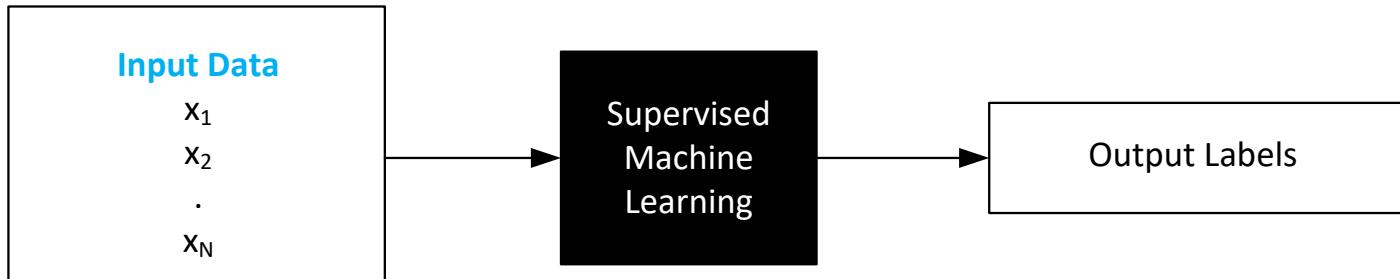
- There are **3 main types** of Machine Learning, these types are:
 1. Supervised Learning
 2. Unsupervised Learning
 3. Reinforcement Learning





Supervised Learning

- Supervised learning is by far the most popular form of AI and ML used today.
- We take a set **of labeled data** (called a dataset) and we feed it in to some ML learning algorithm that then creates a model to fit this data to some outputs
- E.g. let's say we give our ML algorithm a set of 10k spam emails and 10k non spam. Our model figures out what texts or sequences of text indicate spam and thus can now be used as a spam filter in the real world.



Supervised Learning In Business

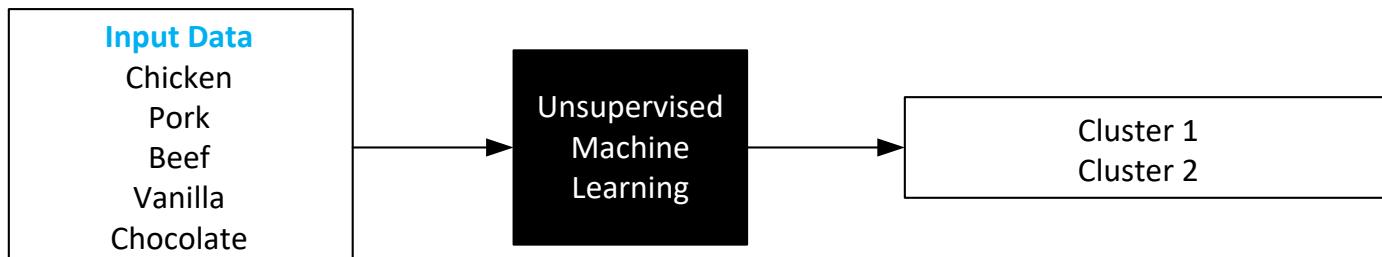
- In the Business world a lot of Machine Learning Models fall under this type. Some examples are:
 - Predicting which customers are most likely to leave our business (churn)
 - Predicting who might default on a loan
 - Predicting prices based on a set of information, e.g. predicting Airbnb prices based on location, apartment size, number of persons it can accommodate etc.





Unsupervised Learning

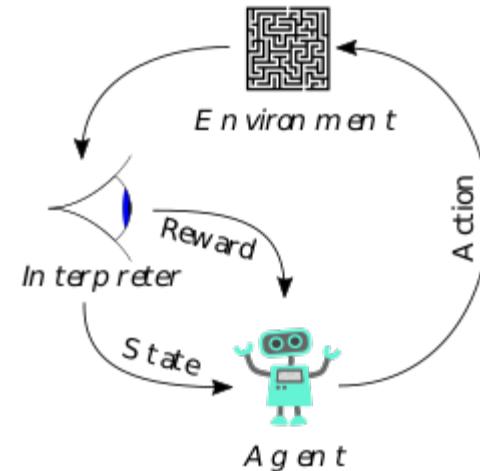
- Unsupervised learning is concerned with finding interesting clusters of input data. It does so **without any help of data labeling**.
- It does this by creating interesting transformations of the input data
- It is very important in data analytics when trying to understand data
- Examples in the Business world:
 - Customer Segmentation





Reinforcement Learning

- Reinforcement learning is a type of learning where an agent learns by receiving rewards and penalties.
- Unlike Supervised Learning, it isn't given the correct label or answer. It is taught based on experience
- Usually applications are AI playing games (e.g. DeepMind's Go AI) but it can be applied to Trading Bots (we'll be making one later!)





ML Supervised Learning Process

- **Step 1 – Obtain a labeled data set**
- **Step 2 – Split dataset into a training portion and validation or test portion.**
- **Step 3 – Fit model to training dataset**
- **Step 4 – Evaluate models performance on the test dataset**



ML Terminology

- **Target** – Ground truth labels
- **Prediction** – The output of your trained model given some input data
- **Classes** – The categories that exist in your dataset e.g. a model that outputs Gender has two classes
- **Regression** – Refers to continuous values, e.g. a model that outputs predictions of someone's height and weight is a regression model
- **Validation/Test** – They can be different, but generally refers to unseen data that we test our trained model on.



Machine Learning Algorithms

- **Supervised Learning**
 - **Regressions** – Linear Regression, Support Vector Machines, KNN, Naïve Bayes, Decision Trees and Random Forests
 - **Classifiers** – Logistic Regression, Neural Networks & Deep Learning
- **Unsupervised Learning**
 - Clustering – K-Means and many more
- **Reinforcement Learning**

Linear Regression – Introduction to Cost Functions and Gradient Descent



Linear Regression

- A Linear Regression is a statistical approach to modeling the relationship between a **dependent variable (y)** and one or more **independent variables (x)**.
- Basically, we want to know the **regression equation that can be used to make predictions**.
- Our model uses **linear predictor functions** whose parameters (similar to the **m** and **c** in ' $y=mx+c$ ') are estimated using the training data.
- Linear Regressions are one of the most popular ML algorithms due to it's simplicity and ease of implementation.

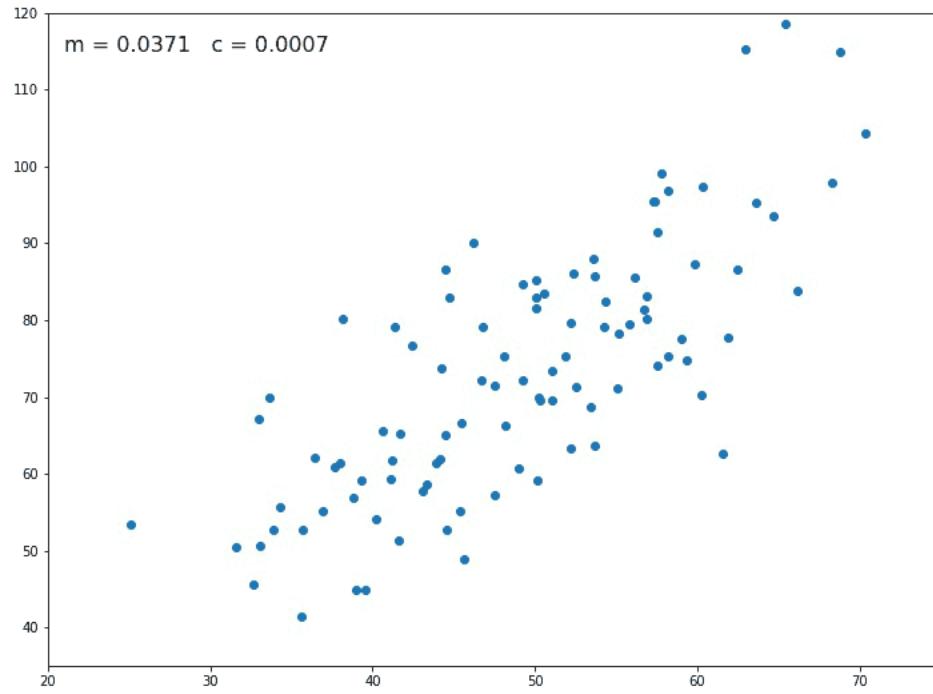


Linear Regression for one Independent Variable

- $y = f(x) = mx + b$
- Note we used 'c' before, but in most Machine Learning text 'b' is used, most likely as it refers to **bias** (Note often m is also replaced by w)
- Our goal would be to find the best values of **m** and **b** that provide the most accurate predictions?



Linear Regression for one Independent Variable



Look at how our line changes
when **m** and **c** change



Loss Functions

$$y = f(x) = mx + b$$

- How do we find the most appropriate values of **m** and **b**?
- We can **measure the accuracy or goodness of Linear Regression model by finding error difference between the predicted outputs and the actual outputs (ground truth).**

Remember our Least Square Regression!

- Least Square Regression is a method which minimizes the error in such a way that the sum of all square error is minimized.



Loss Functions: Mean Squared Error (MSE)

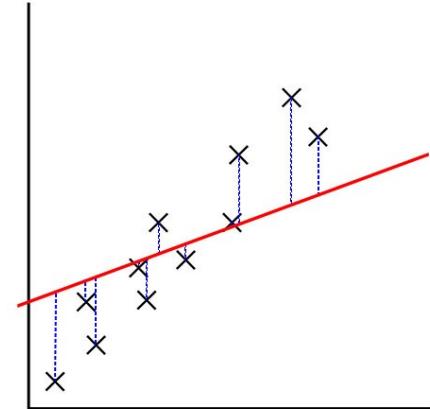
$$\text{Mean Squared Error}(m, b) = \frac{1}{N} \sum_{i=1}^N (\text{Actual Output} - \text{Predicted Output})^2$$

- Before we get into this, let's look at how we find the values for **m** and **b**



Finding the values of m and b

- Imagine we had a set of points and we're trying to fit a line of best fit too.
- Initializing with **random** values of m and b will give us a line that will not be ideal. However, it allows us to find the **MSE**
- Imagine now our goal is to **keep trying values of m and b** that produce the lowest value of **MSE**.
- Trying random values will be exhaustive and time consuming, so how do we do this?





Introducing Gradient Descent

- We use a process called Gradient Descent to minimize the **cost or error function**:
 - $Error(m, b) = \frac{1}{N} \sum_{i=1}^N (Actual\ Output - Predicted\ Output)^2$
 - Cost Function = $J(\theta_0, \theta_1) = \frac{1}{2} \sum_{i=1}^m (h_\theta(x_i) - y_i)^2$
- We treat **m** and **b** as θ_0 and θ_1
- The above equation simply tells us how **wrong** the line is by measuring how far the **predicted value of $h(x_i)$** is from the **actual value y_i**
- We **square** the error so that we don't end up with **negative** values
- We **divide by 2** to make updating our parameters easier



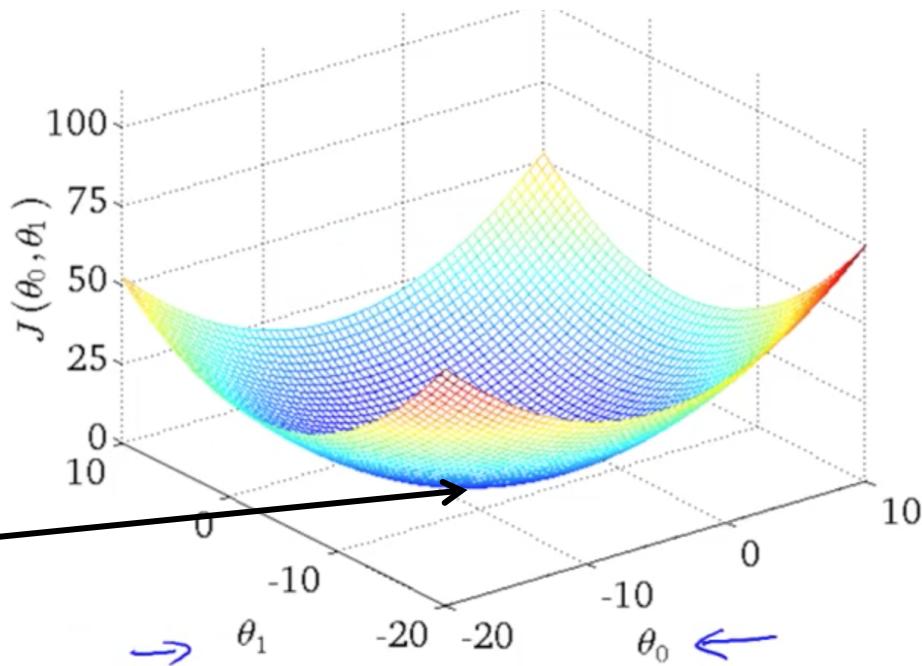
What is Gradient Descent

- $J(\theta_0, \theta_1) = \frac{1}{2} \sum_{i=1}^m (h_\theta(x_i) - y_i)^2$
- Recall **our Cost Function** equation (J), remember it gives us the error based on whatever values of θ_0 and θ_1 we use.
- How do we know what values to use that gives us the lowest cost?



The bowl shaped cost function for different values of θ_0 and θ_1

We want the θ_0 and θ_1 at this point



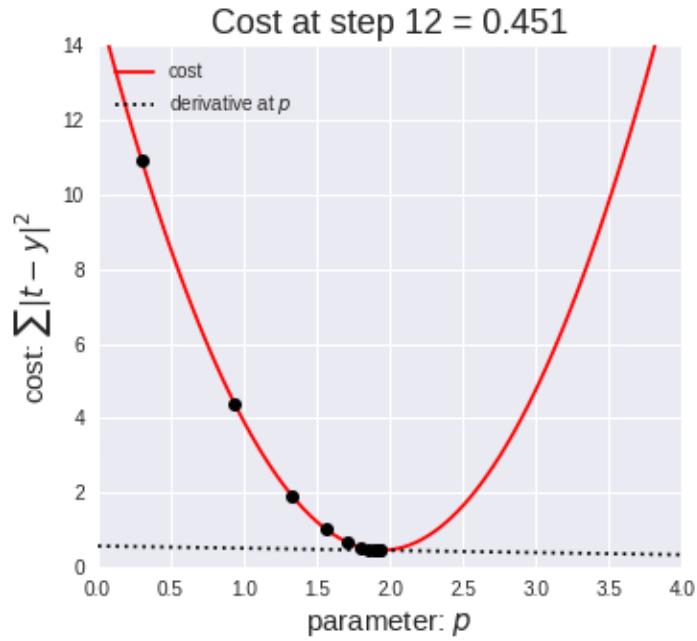


Gradient Descent Method

- Gradient Descent is the method by which we find this point where the gradient is zero
- Imagine you're walking down a valley with poor visibility, but you wish to get to the bottom of the valley. As you walk down, when it's steep you take large steps, however as you get lower and slope gets more gradual you take smaller steps, that way you don't overstep the lowest point

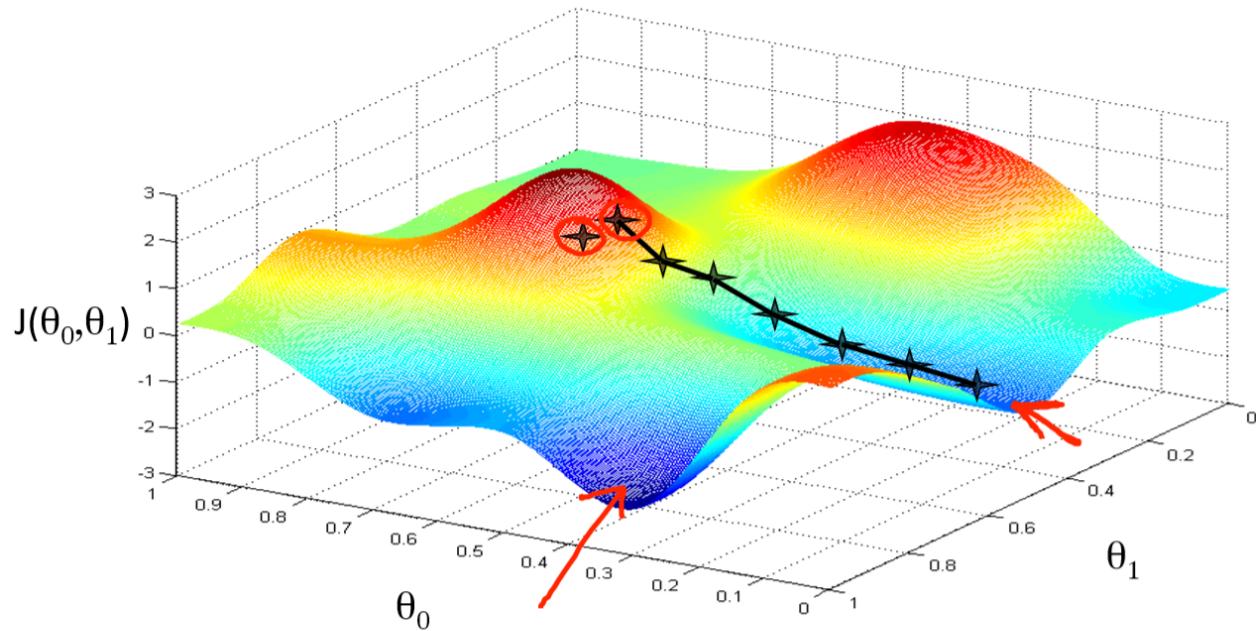


Gradient Descent Method Visualized



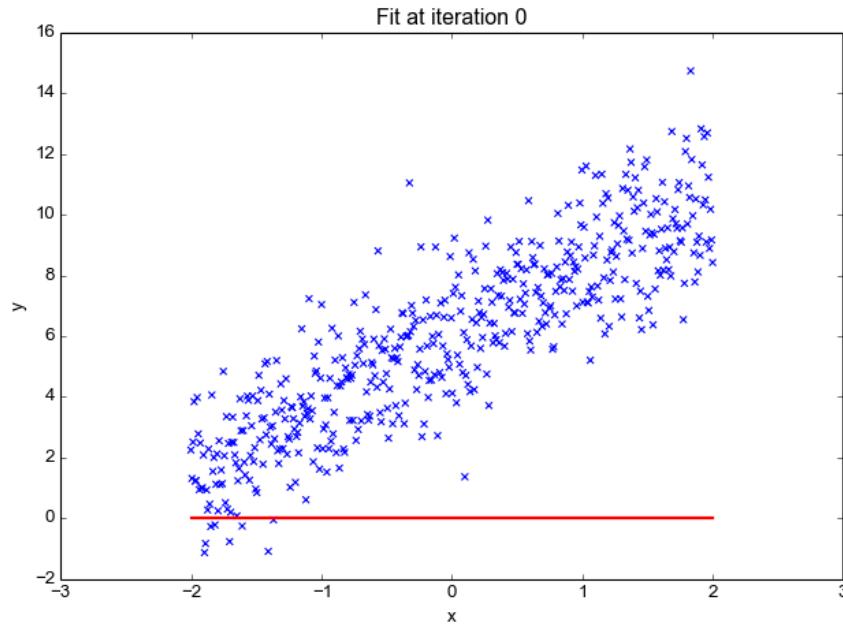


Gradient Descent Method Visualized



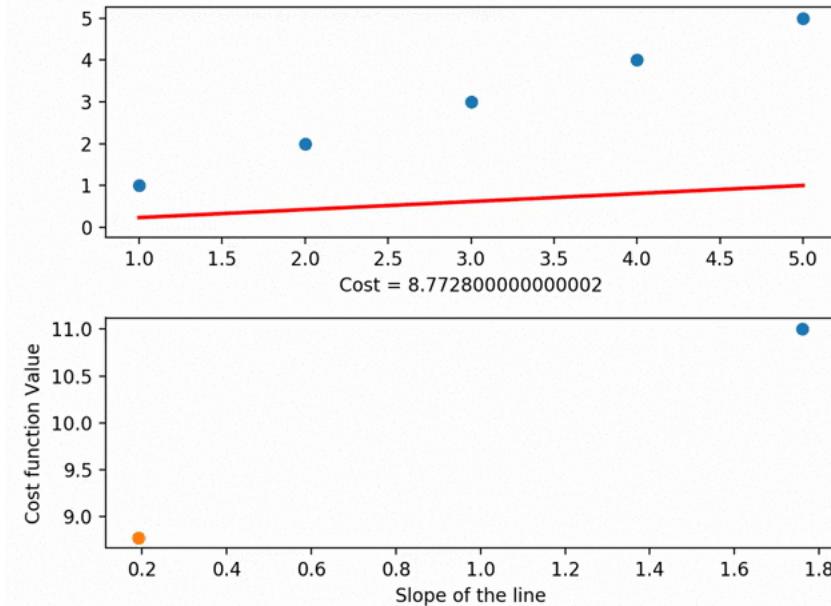


Line Fitting Visualized - Iteratively





Gradient Descent Method Visualized





Linear Algebra Representation

- In many tutorials, particularly math based tutorials, you'll see linear regression equations written in vector form:
- $f_n = f(x^{(n)}; w, b) = w^T x^{(n)} + b$



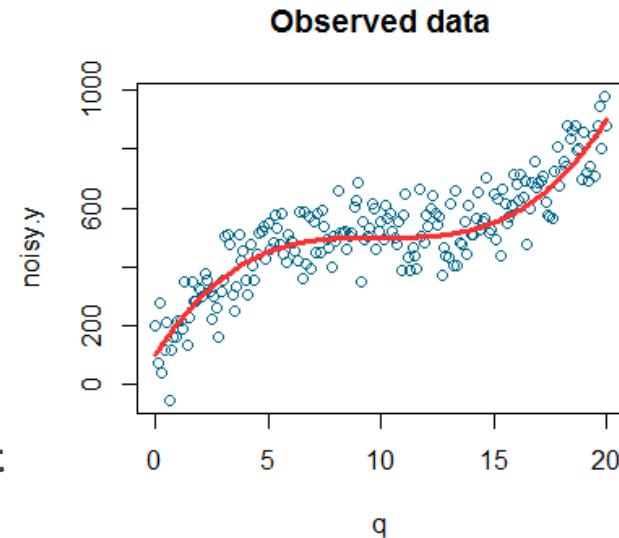
Let's Do Some Linear Regressions in Python

Polynomial and Multivariate Linear Regressions



A Polynomial Regression

- A **Polynomial Regression** is a type of linear regression in which the relationship between the independent variable x and dependent variable y is modeled as an **n th degree polynomial**.
- Polynomial regression fits a **nonlinear relationship** between the values of x and the corresponding outputs or dependent variable y .

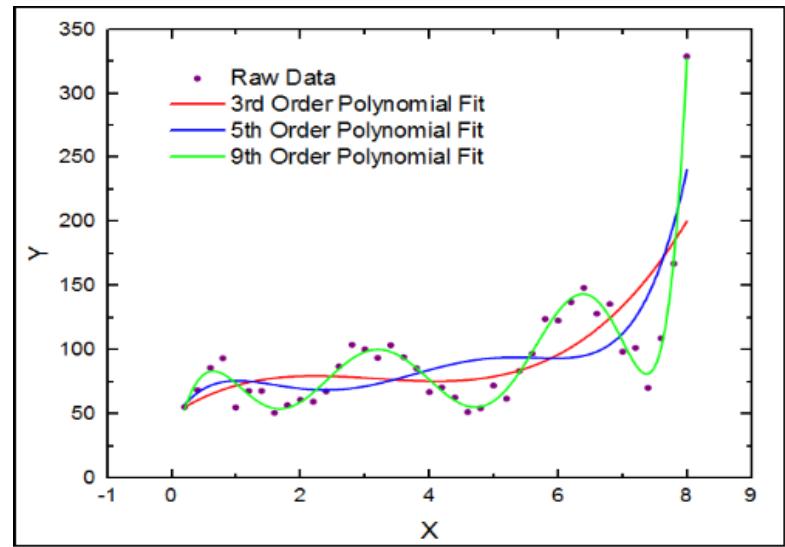




Polynomial Regression

$$y = m_1x + m_2x^2 + m_3x^3 + \dots + m_nx^n + b$$

'n' represents the order of the polynomial





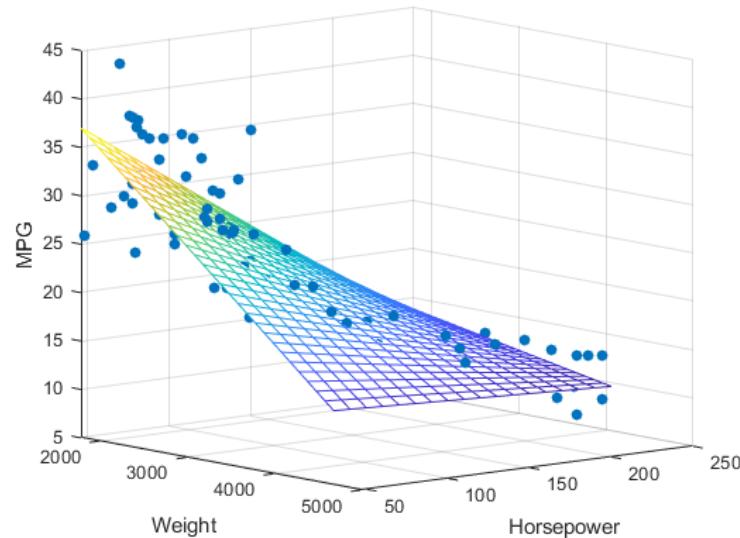
Polynomial Regression In Python



Multivariate Linear Regression

- What if we had multiple columns for our input?
- That would mean our output y would now be dependent upon more than one independent variable.

x1	x2	Y
342	32	32
235	36	23





Multivariate Linear Regression

- This is represented simply by:
- $y = f(x) = b + w_1x_1 + w_2x_2 \dots + w_nx_n = \sum_{i=1}^n w_i x_i$
- This doesn't change cost function which remains the same



Multivariate Regression In Python

Logistic Regression



Logistic Regression

- Previously with **Linear Regressions** we were predicting a **continuous variable**, like our 2020 Olympic 100m Gold time of 9.53 seconds.
- However, what if wanted to predict a **Yes** or **No** type answer, then we can't use a Linear Regression type model.
- We need something that takes our inputs and identifies whether something is **True** or **False**. Example, predicting whether:
 - a person is Male or Female
 - an email is Spam or not
 - a person is likely to say Yes or No



Binary Classification

- Predicting whether an input belongs to which class in a **two class** situation is called **Binary Classification**.
- The problem is phrased like our **Hypothesis testing**. Imagine we're trying to predict someone's gender based on their height and weight.
- Our Logistic Regression Classifier will look at the input and treat it as if it were responding to our Hypothesis, example "Does this input data belong to a male" or "This is a male sample". The response output of our **binary classifier** is typically:
 - **0** for False or No
 - **1** for True or Yes



Classification into Classes is Very Useful

- Predicting what class an input belongs to is extremely useful in many problems such as determining whether someone can be approved for a loan, or if student should be accepted into a competitive university etc.
- **Logistic Regressions** aren't limited to binary classification either, it can in fact be extended to **multiple classes** e.g.
 - Predicting handwritten digits (Computer Vision Application)
 - Predicting article categories e.g. Sports, Politics, Health etc. (NLP)
 - Or levels of risk involved in an investment



Theory behind Logistic Regressions

Remember our Linear Regressions linear algebraic representation:

- $f_n = f(x^{(n)}; w, b) = w^T x^{(n)} + b$
- $f = w^T x$

Logistic Regression is simply **the transformation of a linear function with a logistic sigmoid.**

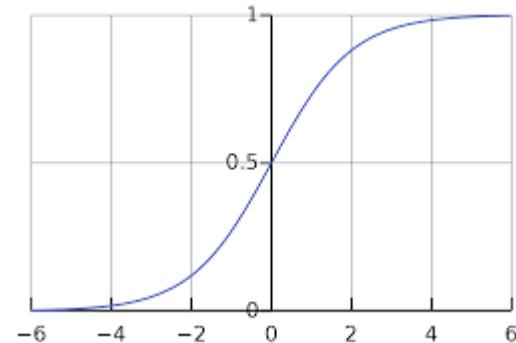
- $g(z) = \frac{1}{1+e^{-z}}$



Theory behind Logistic Regressions

The result is:

- $f(x; w) = \sigma(w^T x) = \frac{1}{1+e^{-w^T x}}$
- We do this so that we force the output of our Logistic transformed Linear Regression to be between **0 and 1** (note the y intercept is 0.5)

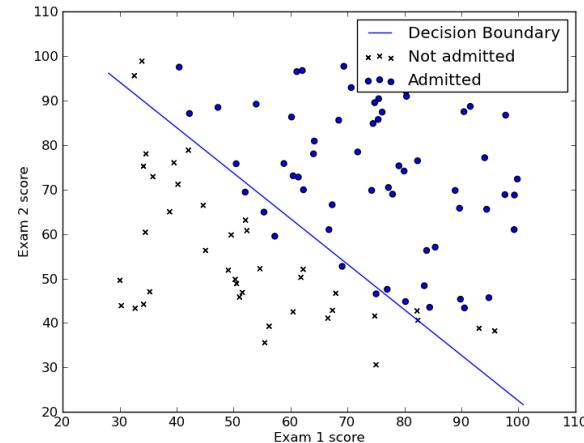


Sigmoid Function



Logistic Regressions Create Decision Boundaries

- We can visually interpret the Logistic Regression threshold by plotting it's **Decision Boundary**.
- This decision boundary line is given by **solving the linear term for values that evaluate to 0.5**





Logistic Regression – Cost or Loss Function

- As we saw before in Linear Regressions, the parameters of the Logistic Regression are chosen by **minimizing the loss or cost function**.
- $f(x; w) = \sigma(w^T x) = \frac{1}{1+e^{-w^T x}} = \frac{1}{1+e^{-\theta^T x}}$

Note: the parameters of the logistic regression are written as θ

- $Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$

This can be written

- $Cost(h_\theta(x), y) = -y[\log(h_\theta(x)) - (1 - y)\log(1 - h_\theta(x))]$

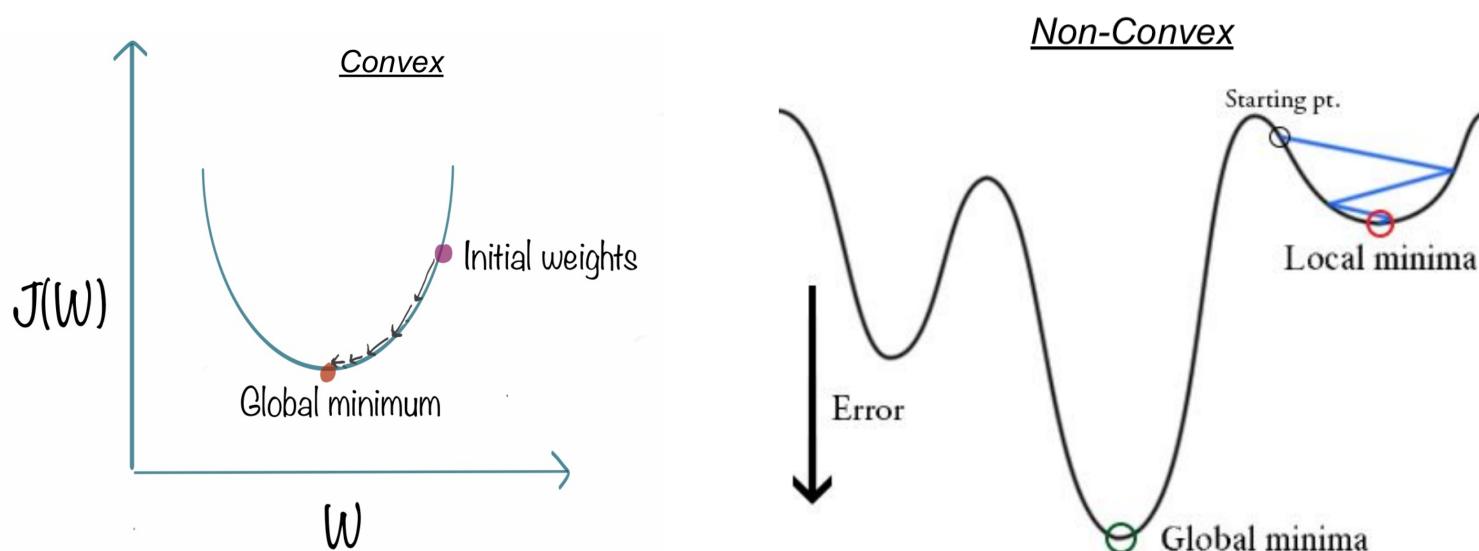


Minimizing our Cost or Loss Function

- The logistic regression cost function is **convex** which means in order to find the parameters (θ) we need to solve an **unconstrained optimization problem** i.e. finding the values of θ_n that minimize the **Cost function**.
- $Cost(h_{\theta}(x), y) = -y[\log(h_{\theta}(x)) - (1 - y)\log(1 - h_{\theta}(x))]$
- This can be done by **Gradient Descent** or **Newton's Method**
 - **Newton's Method** – uses both the gradient and the Hessian (square matrix of second-order partial derivatives) of the logistic regression cost function



Convex vs. Non-Convex



source: <https://medium.freecodecamp.org/understanding-gradient-descent-the-most-popular-ml-algorithm-a66c0d97307f>; https://www.cs.ubc.ca/labs/lci/mlrg/slides/non_convex_optimization.pdf



Other Machine Learning Classification Algorithms

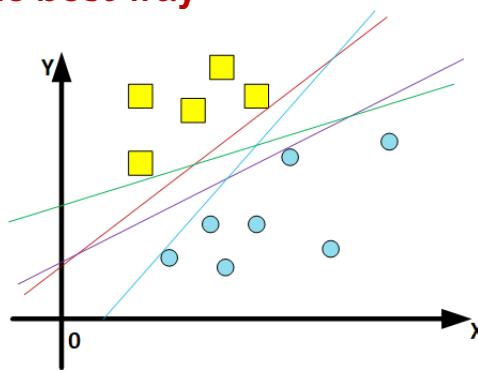
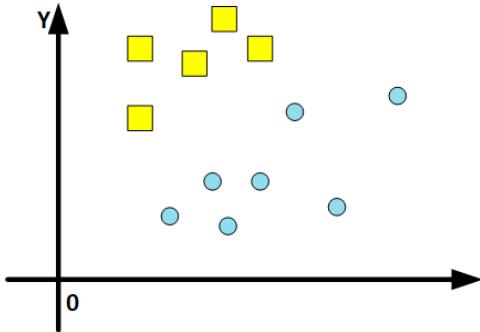
- Logistic Regressions aren't the only class predictors, there are many useful algorithms in Machine Learning such as:
 - Support Vector Machines
 - Random Forests
 - Neural Networks / Deep Learning

Support Vector Machines (SVMs)



Support Vector Machines (SVMs)

- SVMs are another ML classifier (not regression, and yes Logistic Regressions are so badly named!)
- As with Logistic Regressions, our aim is to create a hyper plane decision boundary between classes in the best way

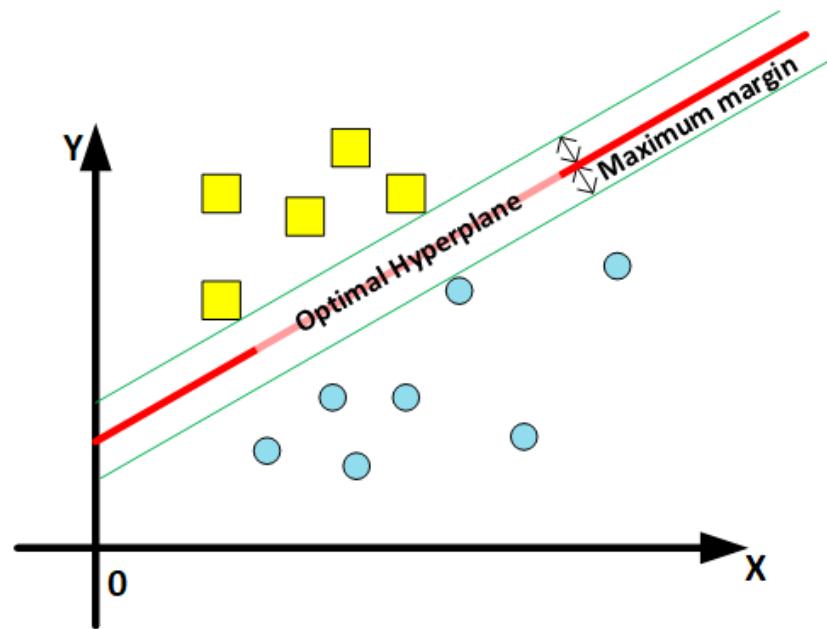


How do we know which decision boundary is best?



Optimal Hyperplane

- The goal of SVMs are to **maximize** the **margin** between the data points of both classes. The plane or decision boundary is then called the Hyperplane
- Points of each class are separated by our hyperplane

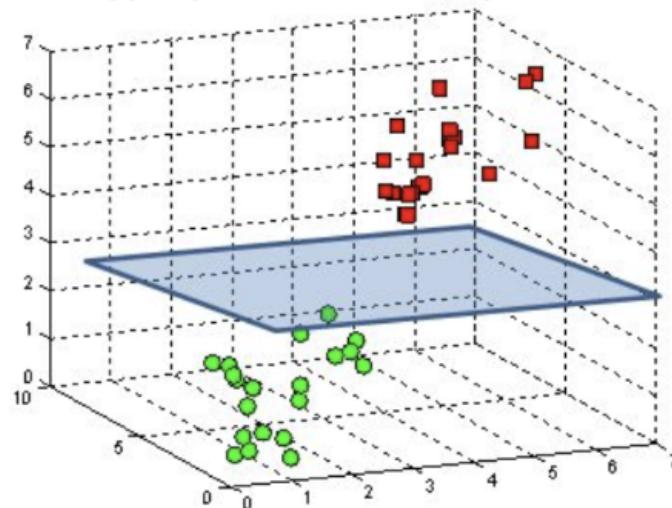




Optimal Hyperplane in Multiple Dimensions

- Hyperplanes can be in multiple dimensions, just like Logistic Regressions.
- On the right we see a 3D hyperplane, this is where we have two input features (x_1 and x_2)

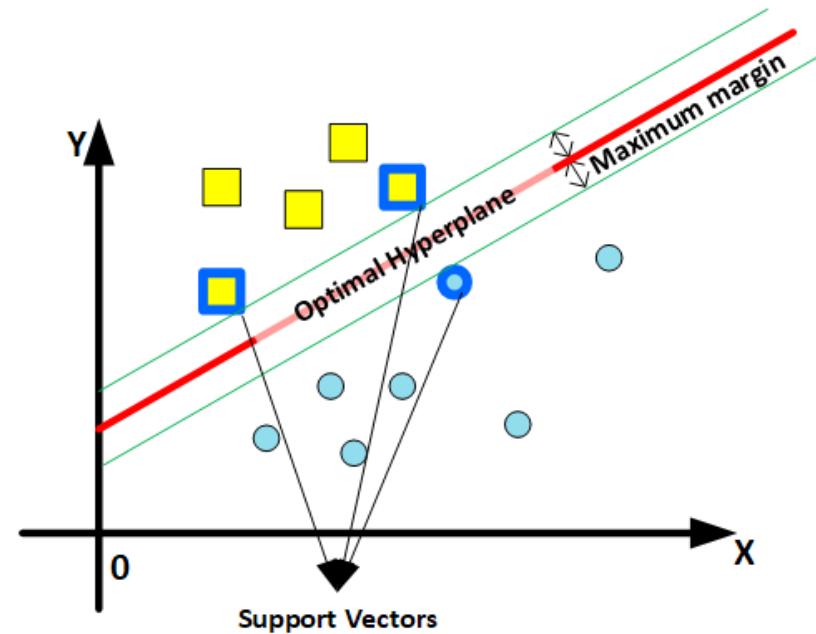
A hyperplane in \mathbb{R}^3 is a plane





Support Vectors

- **Support vectors** are the points that reside closest to the hyperplane
- The hyperplanes separate the classes, by 'looking' at the points on the **class extremes** (i.e. closest to the margin).
- These points influence the position and orientation of the hyperplane. NOTE removing these points will have large impacts on the hyperplane

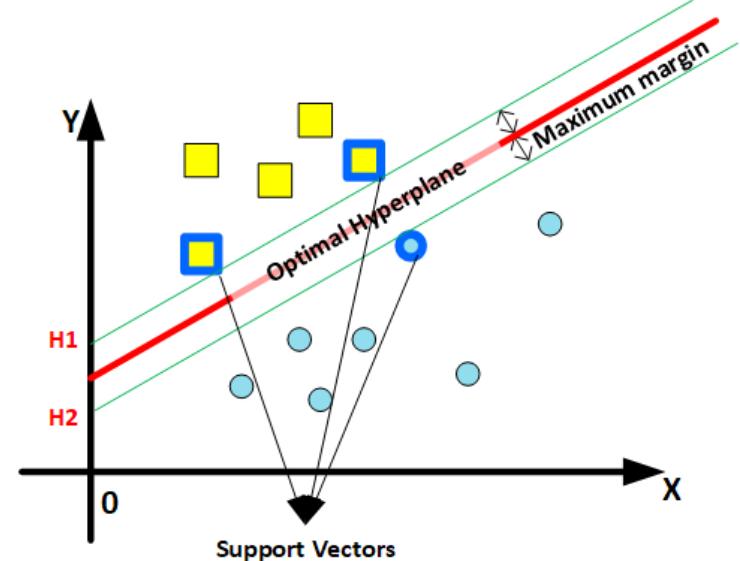




Hyperplane Intuition

- In Logistic Regressions we took a **linear** function and used a **sigmoid** function to squash the output range from 0 to 1 (probability type result), where our discriminator threshold was 0.5
- In SVMs we take the output of our linear function and for values **greater than 1** we label it as one class and values **less than -1** the other class
- Our SVM algorithm seeks to maximize the margin between our support vectors by finding the weights of the hyperplane that minimize our Cost function

$$\min_w \lambda \|w\|^2 + \sum_{i=1}^n (1 - y_i \langle x_i, w \rangle)_+$$



$$w \cdot x_i + b \geq 1 \quad \text{for } y_i = +1$$

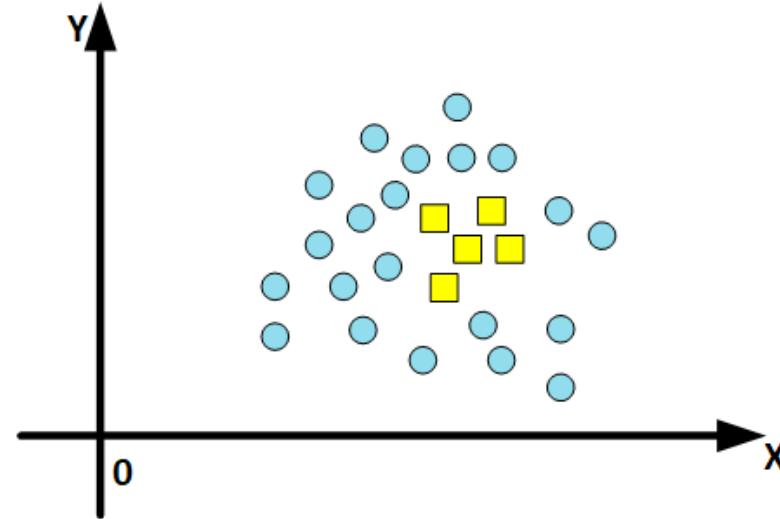
$$w \cdot x_i + b \leq -1 \quad \text{for } y_i = -1$$

combining above two equation , it can be written as
 $y_i(w \cdot x_i + b) - 1 \geq 0 \quad \text{for } y_i = +1, -1$



The Kernel Trick – Handling Non-Linear Data

- For **non-linear data** (see right) it is **impossible** for a hyperplane in this dimensional space to separate our data. So what do we do?
- We employ the **kernel trick**, which takes a low dimensional input space and transforms it into a **higher dimensional space**.

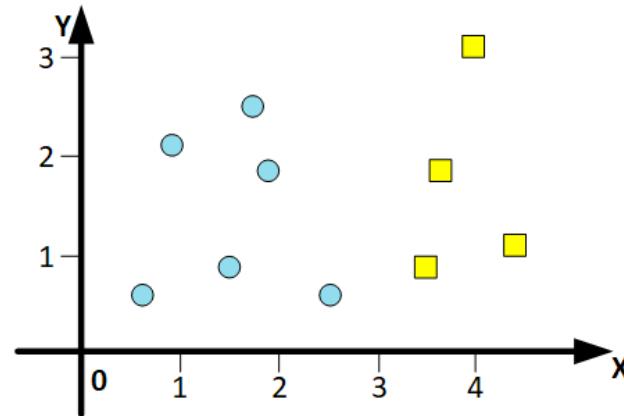


Decision Trees and Random Forests



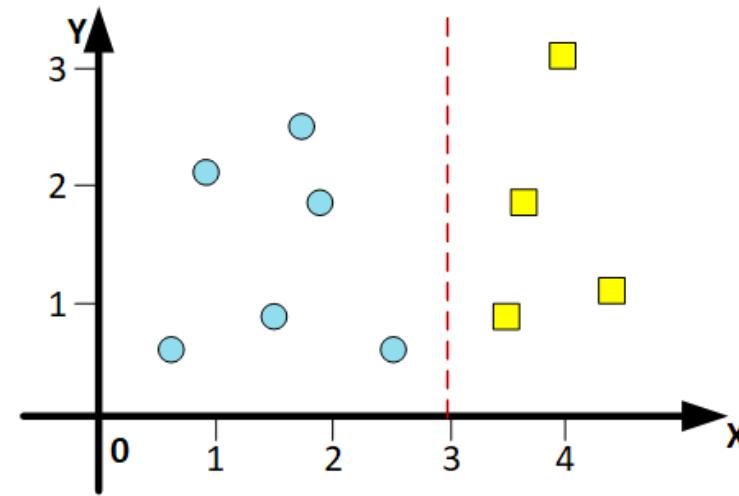
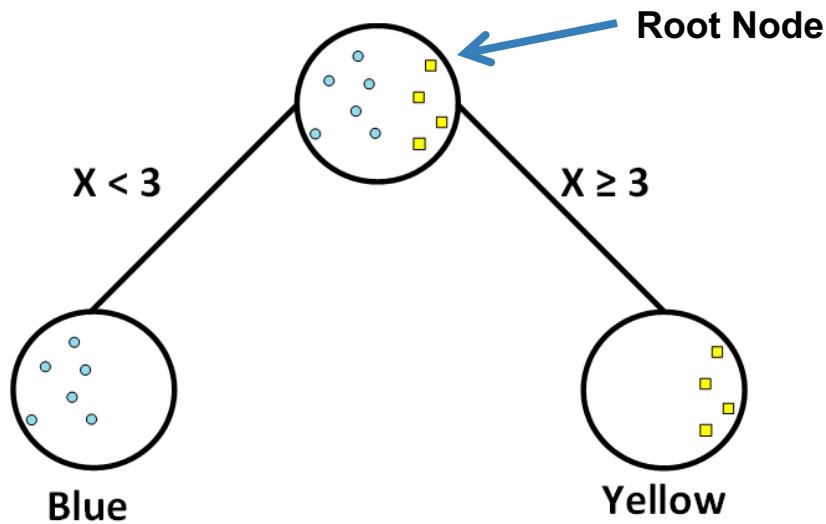
Decision Trees

- Let's look at the data on the right
- To separate the classes we don't necessarily need a fancy hyperplane.
- Visibly, we can see that when x is greater than 3, it belongs to the yellow class and if it's less than 3, it belongs to the blue class.
- Let's represent this as a tree



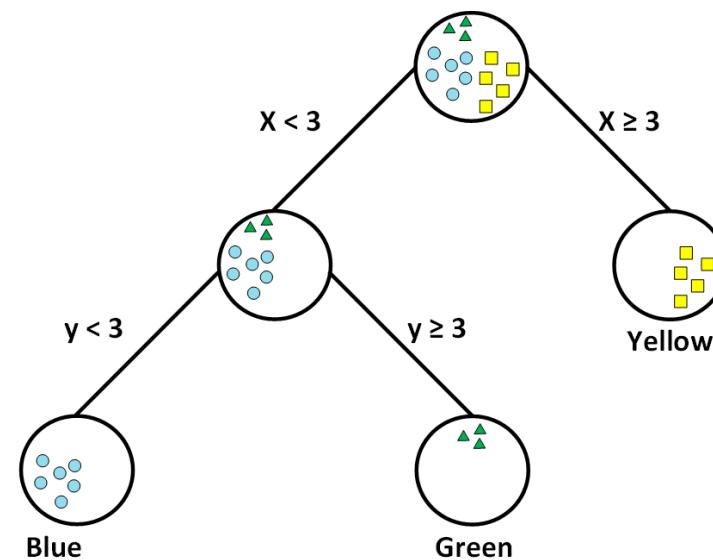
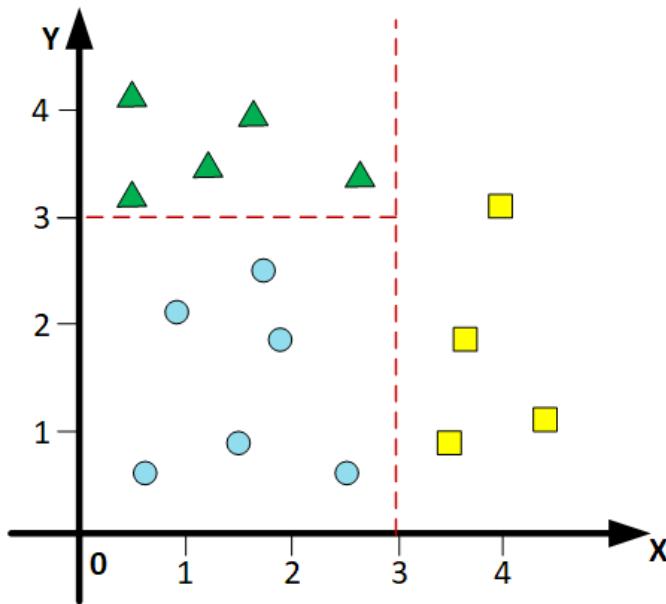


Decision Trees





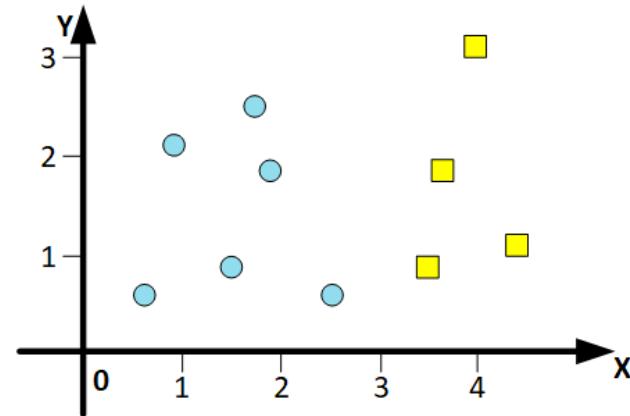
Decision Trees – 3 Classes





The Decision Tree Algorithm

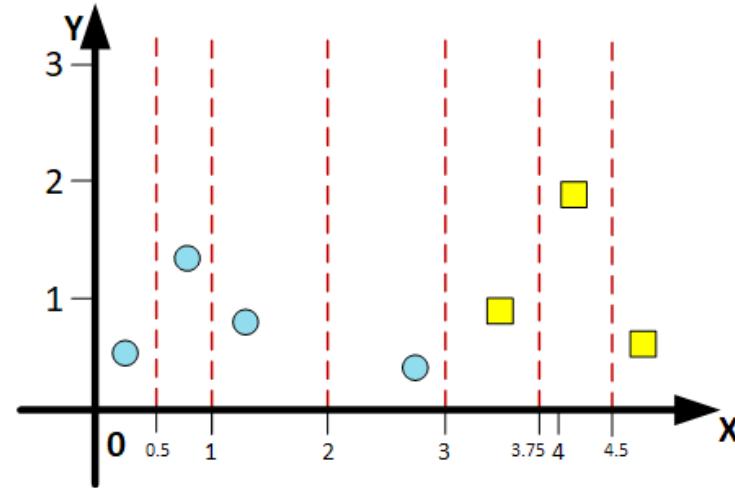
- As we saw in our multi-class decision tree, we need to find a way to define these '**splits**' so that we find the best splits to identify our classes.
- Secondly, there's the added problem of finding the first split or **root**.
- Should we try **every possible split** to determine which is best?
- That can be exhaustive, but let's try it.





Defining and Evaluating our splits

- We've defined six (6) possible splits given our data that separate each point.
- How do we know which splits are best?
- We use the **Gini Gain Calculation**



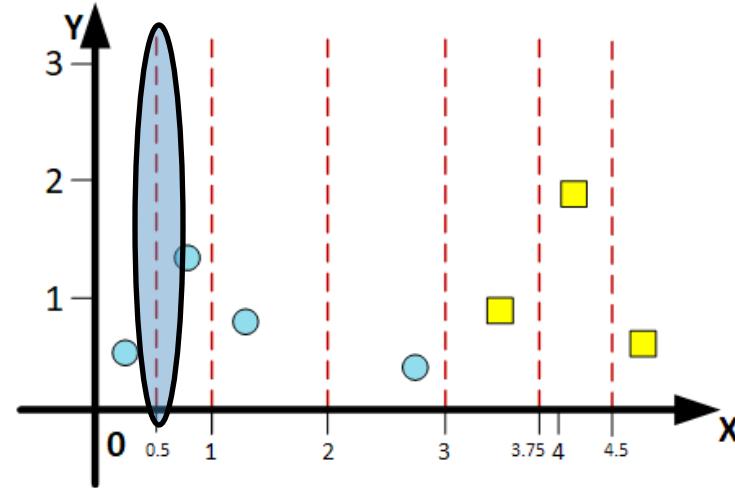


Gini Gain Calculation

- Firstly we calculate the **Gini Gain** (called **impurity**) for the whole dataset.
- This is the probability of incorrectly classifying a random selected element in the dataset.

$$G_{initial} = \sum_{i=1}^C p_i (1 - p_i)$$

- C – number of classes
- p_i – probability of randomly choosing element of class i





Gini Gain Calculation

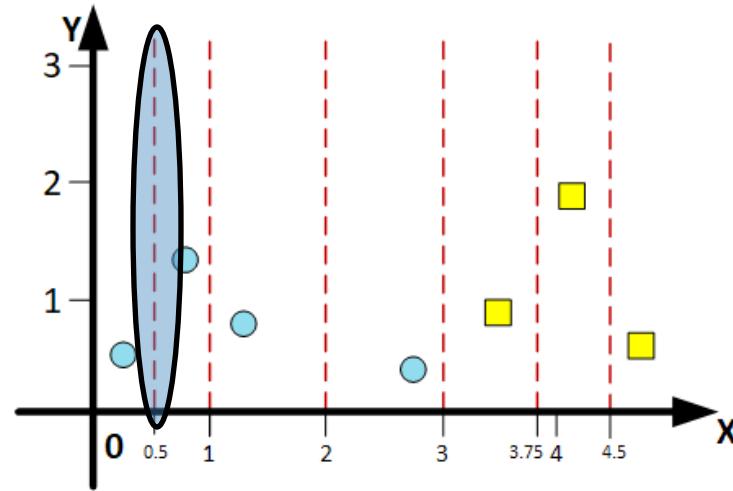
$$G_{initial} = \sum_{i=1}^c p_i (1 - p_i)$$

Where $c = 2$ and $p(1) = \frac{4}{7}$ and $p(2) = \frac{3}{7}$

$$G = p(1) \times (1 - p(1)) + p(2) \times (1 - p(2))$$

$$G = \frac{4}{7} \times \left(1 - \frac{4}{7}\right) + \frac{3}{7} \times \left(1 - \frac{3}{7}\right)$$

$$G = \frac{24}{49} = 0.49$$



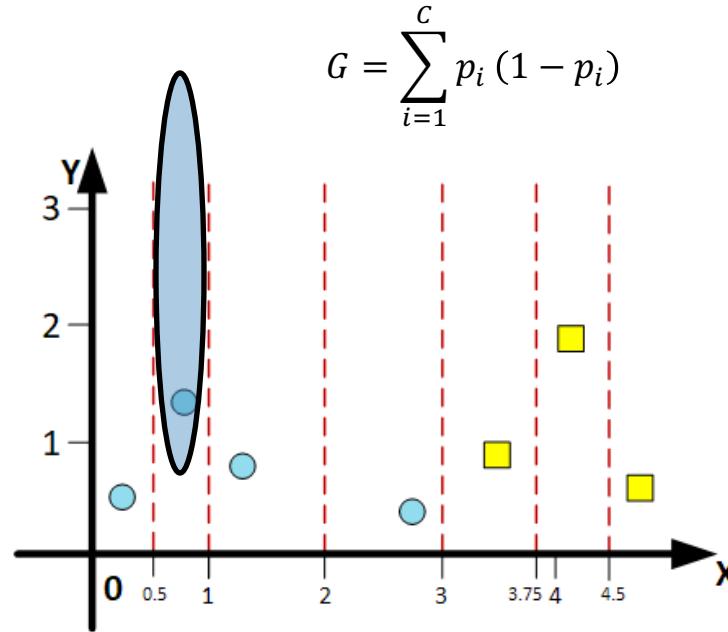


Gini Gain Calculation for $x = 0.5$

- Let's split the branches into **two** sections, **left and right**.
- Our **left** branch only has one blue ball, therefore $G_{Left} = 0$
- Our **right** branch has 3 blue and 3 yellow, therefore our Gini impurity is:

$$G_{Right} = \frac{3}{6} \times \left(1 - \frac{3}{6}\right) + \frac{3}{6} \times \left(1 - \frac{3}{6}\right)$$

$$G_{Right} = \frac{3}{6} \times \frac{3}{6} + \frac{3}{6} \times \frac{3}{6} = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$





Gini Gain Calculation for $x = 0.5$

$$G_{initial} = 0.49 \quad G_{Left} = 0 \quad G_{Right} = 0.5$$

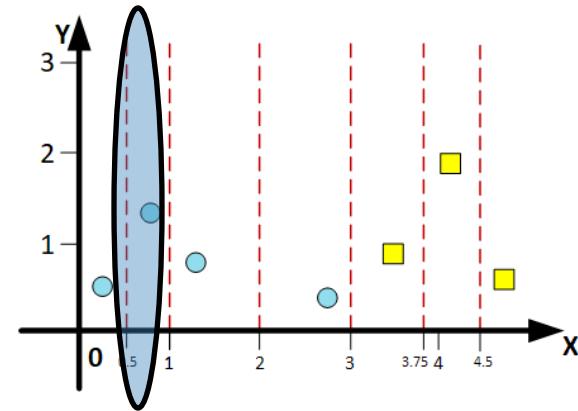
We can now determine the quality of this split by weighting the impurity of each branch by the number of elements it contains (left = 1, right = 6)

$$\left(\frac{1}{7} \times 0\right) + \left(\frac{6}{7} \times 0.5\right) = \frac{3}{7} = 0.43$$

Total impurity we removed with this split is:

$$Gini\ Gain_{x=0.5} = 0.49 - 0.43 = 0.06$$

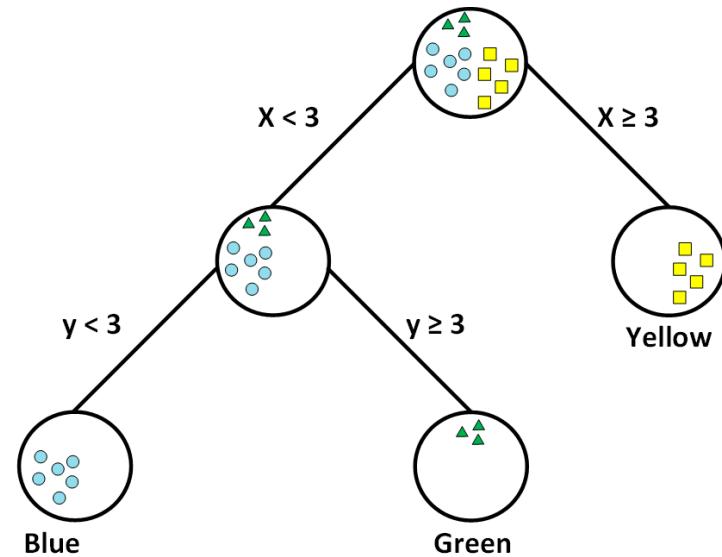
We do this for each split and use the one with the Highest Gini Gain





Gini Gains for Multiple Classes

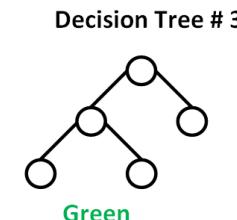
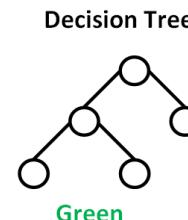
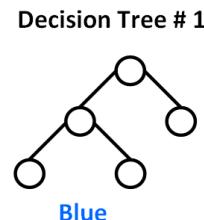
- Calculating Gini Gains is how we construct our Decision Tree.
- If we have 3 or more classes we do the same with the other classes, to form our second level of nodes. The first is our **root** and each node is called a **leaf**.
- We only stop when our Gini Gain (i.e. G impurity minus the impurity of our branches is zero). This is because in our last leaves there won't be any more classes to separate.





Random Forests

- Once we understand Decision Trees the intuition for Random Forests makes sense.
- Random forests are simply a **the output of multiple decision tree classifiers**.
- For Random Forests we **sample with replacement** from our training dataset, then train our decision tree on n set of samples and repeat this t times. Note some samples will be used multiple times in a single tree (hence the random part of Random forests)
- This process of using multiple classifiers and finding the **most common output (majority vote)** or **average** (if regression) is called **Bagging** or **Bostrapping**.



Green

K-Nearest Neighbors (KNN)



K-Nearest Neighbors (KNNs)

- KNNs are one of the most popular machine learning algorithms as it gives somewhat decent performance and is also quite easy to understand and implement.
- It's called a **lazy learner** (as opposed to eager learners) because it doesn't have a training phase, what it does instead is compute its classification using the training data (can be slow). This also makes it a **non-parametric algorithm**.



The KNN Algorithm – Step by Step

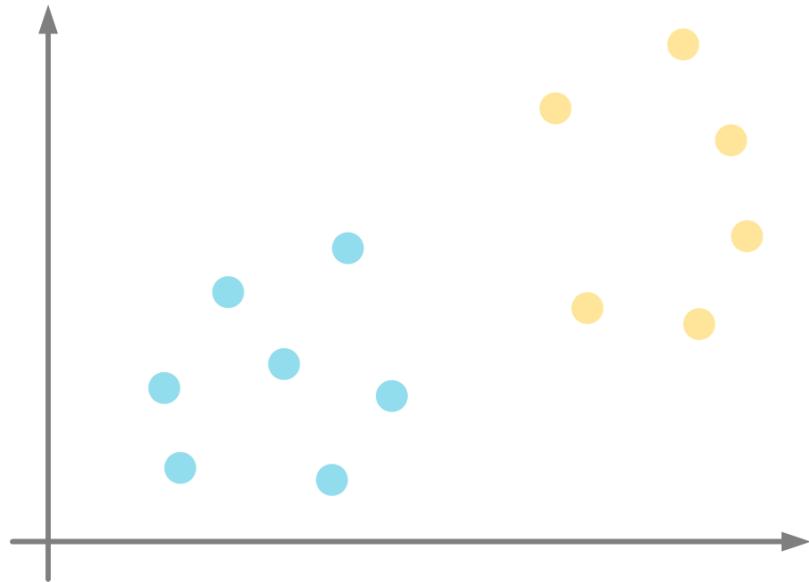
How KNNs classify a new data input:

1. Load all training data
2. Use a pre-set value of k (integer only)
3. Calculate the distance between the test data input and each value of the training data
4. Sort on our distances (ascending i.e. smallest distance to largest distance)
5. Choose the top K rows of our sorted data
6. Assign class to our input data to the most frequent class in our K rows

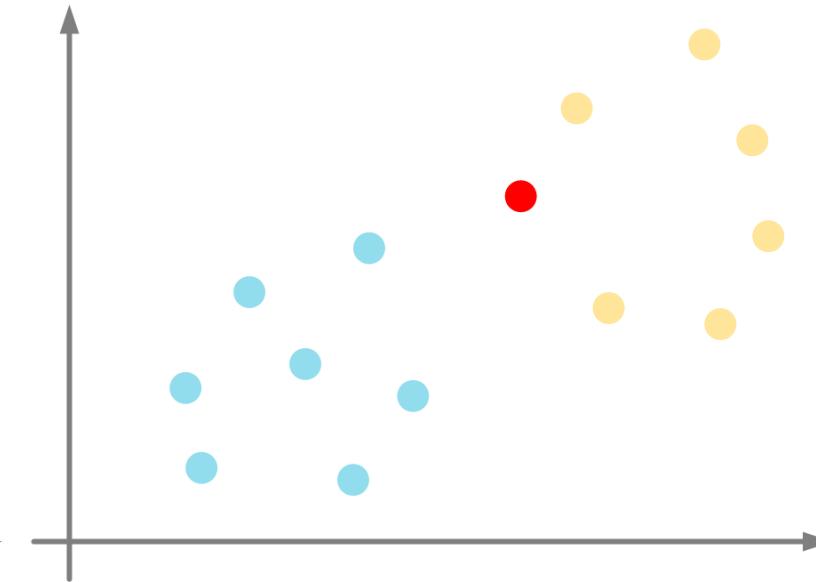


The KNN Algorithm – Visually

Scatter plot of our training data



Let's classify our new red input

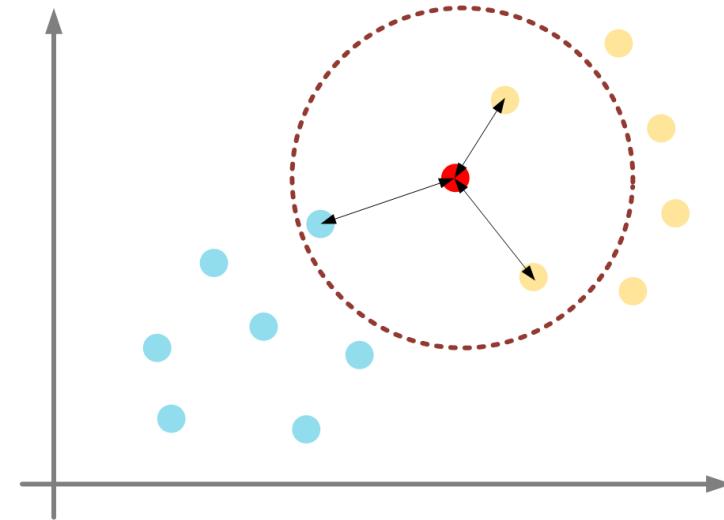




The KNN Algorithm – Using $k = 3$

- As $K = 3$, let's draw a circle enclosing the 3 nearest points to our input (red circle)
- Two out our 3 points were yellow when sorted by distance, therefore the class KNN will assign the red point to will be the yellow class

Point	Distance	Class
1	10	Yellow
2	12	Yellow
3	15	Blue

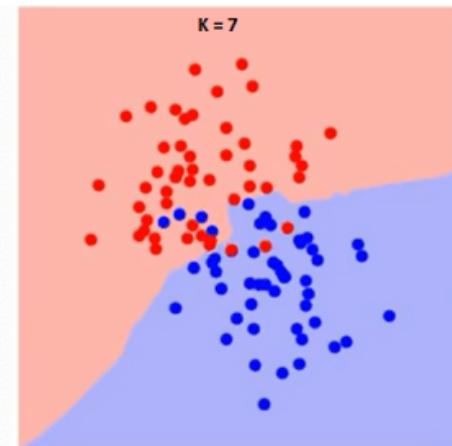
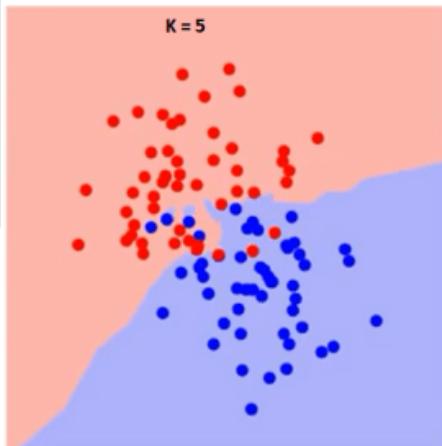
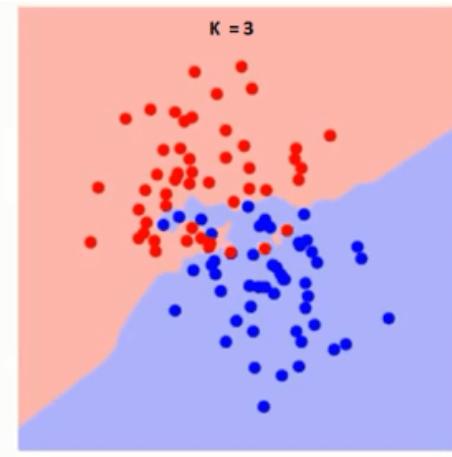
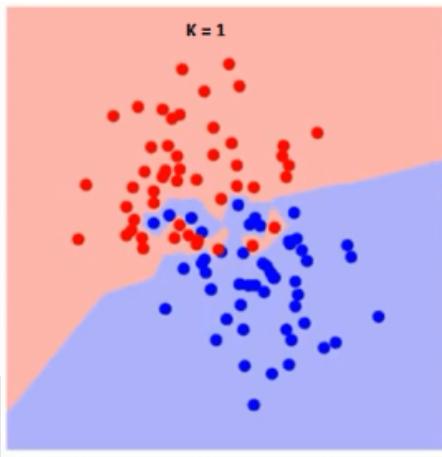
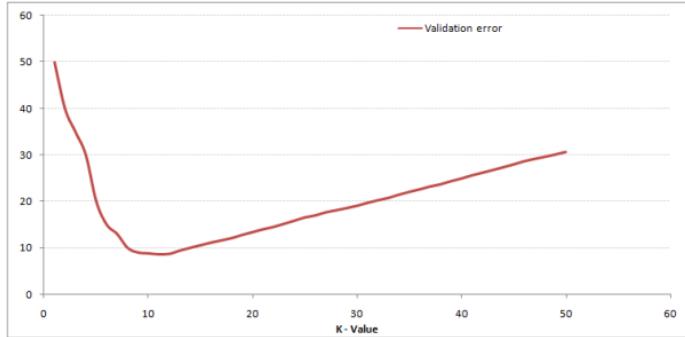




Choosing K

- As we can see from our last diagram, the larger k, the more points it considers. But how do we go about knowing which k to choose?
- If $k = 1$, we will only assign classes considering the closest point, this can lead to a model that **overfits!**
- However, as k goes up, we begin to simplify the model too much thus leading to a model that **is under-fitting** (high bias and low variance)
- Let's visualize this!

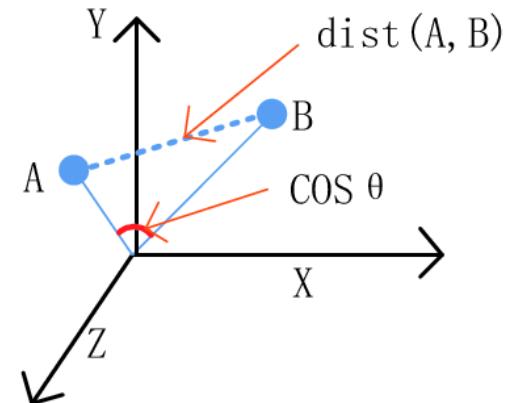
- Notice how our decision boundary becomes smoother as k increases (less overfitting) but only up to a point!





Disadvantages of KNNs

- As you may have noticed, every time we wish to calculate a new input, we need **to load all our training data**, then calculate the distances to every point! This is exhaustive and negatively impacts performance, not to mention the high memory usage it requires.
- Also of note, for KNN to work well, we need to normalize or scale all the input data so that we can calculate the distances fairly. Common distance metrics used are Euclidean distance or Cosine Distance.
- Datasets with a large number of features (i.e. dimensions) will impact KNN performance. To avoid overfitting we thus need even more data for KNN which isn't always possible. This is known as the Curse of Dimensionality.



Assessing Performance – Accuracy, Confusion Matrix, Precision and Recall



Assessing Model Accuracy

- Accuracy is simply a measure of how much of our training data did our model classify correctly
 - $$Accuracy = \frac{\text{Correct Classifications}}{\text{Total Number of Classifications}}$$



Is Accuracy the only way to assess a model's performance?

- While very important, accuracy alone doesn't tell us the whole story.
- Imagine we're using a Model to predict whether a person has a life threatening disease based on a blood test.
- There are now 4 possible scenarios.
 1. **TRUE POSITIVE** - Test Predicts Positive for the disease and the person has the disease
 2. **TRUE NEGATIVE** - Test Predicts Negative for the disease and the person does NOT have the disease
 3. **FALSE POSITIVE** - Test Predicts Positive for the disease but the person does NOT have the disease
 4. **FALSE NEGATIVE** - Test Predicts Negative for the disease but the person actually has the disease



For a 2 or Binary Class Classification Problem

- **Recall** – How much of the positive classes did we get correct

- $$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

- **Precision** – When predicting positive, how many of our positive predictions were right?

- $$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

- **F-Score** – Is a metric that attempts to measure both Recall & Precision

- $$F - Score = \frac{2 \times Recall \times Precision}{Precision + Recall}$$



Confusion Matrix Real Example

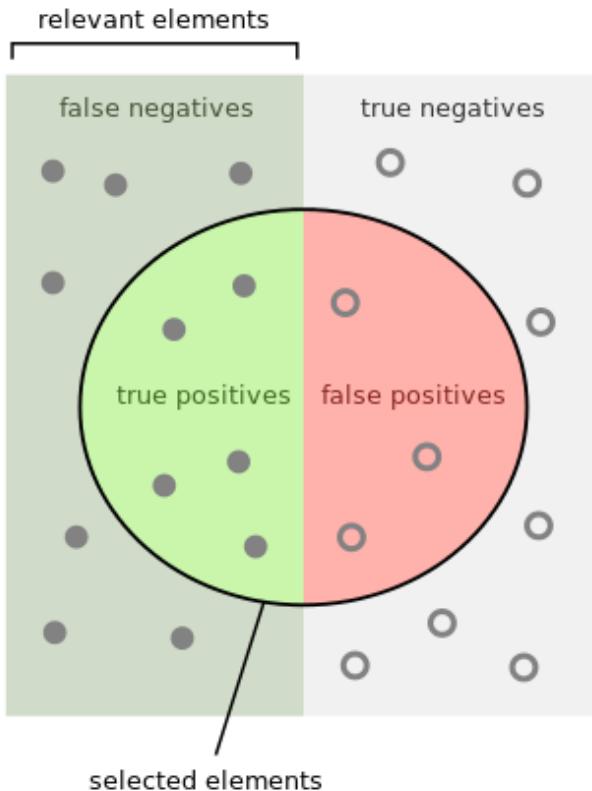
Let's say we've built a classifier to identify Male Faces in an image, there are:

- 10 Male Faces & 5 Female Faces in the image
- Our classifier identifies 6 male faces
- Out of the 6 male faces - 4 were male and 2 female.

$$Precision = \frac{TP}{TP + FP}$$

	Male	Not Male
Predicted Male	TP = 4	FP = 2
Predicted Not Male	FN = 6	TN = 3

- Our Recall is $4 / (4 + 6)$ (6 male faces were missed) or 0.4 or 40%
- Our Precision is $4 / (4 + 2)$ or 0.66 or 66%
- Our F-Score is $\frac{2 \times Recall \times Precision}{Precision + Recall} = \frac{2 \times 0.4 \times 0.66}{0.4 + 0.66} = \frac{0.528}{1.06} = 0.498$



How many selected items are relevant?

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

https://en.wikipedia.org/wiki/Precision_and_recall



Confusion Matrix for Multiple Classes

- We can use scikit-learn to generate our confusion matrix.
- Let's analyze our results on our MNIST dataset

```
[[ 977    0    0    0    0    0    1    1    1    0
   0 1130    3    1    0    0    1    0    0    0
   1    0 1029    0    1    0    0    1    0    0    0
   0    0    4 1003    0    1    0    1    1    0
   0    0    0    0  976    0    0    0    2    4
   3    0    0    5    0  881    3    0    0    0
   6    2    0    0    1    1  948    0    0    0
   1    2   11    2    0    0    0    0 1010    1
   4    0    3    0    0    0    0    2  963    2
   1    2    0    1    5    3    0    2    5 990]]
```



Confusion Matrix for Multiple Classes

Predicted	0	1	2	3	4	5	6	7	8	9	True Values
[977	0	0	0	0	0	1	1	1	0	0
[0	1130	3	1	0	0	1	0	0	0	1
[1	0	1029	0	1	0	0	1	0	0	2
[0	0	4	1003	0	1	0	1	1	0	3
[0	0	0	0	976	0	0	0	2	4	4
[3	0	0	5	0	881	3	0	0	0	5
[6	2	0	0	1	1	948	0	0	0	6
[1	2	11	2	0	0	0	1010	1	1	7
[4	0	3	0	0	0	0	2	963	2	8
[1	2	0	1	5	3	0	2	5	990	9



Confusion Matrix Analysis

- Classifying 7s as 2s, 6s as 0s, 9s as 4s and 9s as 8s.

Predicted	0	1	2	3	4	5	6	7	8	9	True Values
[977	0	0	0	0	0	1	1	1	0	0
[0	1130	3	1	0	0	1	0	0	0	1
[1	0	1029	0	1	0	0	1	0	0	2
[0	0	4	1003	0	1	0	1	1	0	3
[0	0	0	0	976	0	0	0	2	4	4
[3	0	0	5	0	881	3	0	0	0	5
[6	2	0	0	1	1	948	0	0	0	6
[1	2	11	2	0	0	0	1010	1	1	7
[4	0	3	0	0	0	0	2	963	2	8
[1	2	0	1	5	3	0	2	5	990]	9



Recall

- **Recall** – Actual true positives over how many times the classifier predicted that class
- Let's look at the number 7:
 - TP = 1010 and FN = 18
 - $1010 / 1028 = 98.24\%$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

0	1	2	3	4	5	6	7	8	9	
977	0	0	0	0	0	1	1	1	0	0
0	1130	3	1	0	0	1	0	0	0	0
1	0	1029	0	1	0	0	1	0	0	0
0	0	4	1003	0	1	0	1	1	0	0
0	0	0	0	976	0	0	0	2	4	0
3	0	0	5	0	881	3	0	0	0	0
6	2	0	0	1	1	948	0	0	0	0
1	2	11	2	0	0	0	1010	1	1	7
4	0	3	0	0	0	0	2	963	2	8
1	2	0	1	5	3	0	2	5	990	9



Precision

- **Precision** – Number of correct predictions over how many occurrences of that class were in the test dataset.
- Let's look at the number 7:
 - TP = 1010 and FP = 7
 - $1010 / 1017 = 99.31\%$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

0	1	2	3	4	5	6	7	8	9	
977	0	0	0	0	0	1	1	1	0	0
0	1130	3	1	0	0	1	0	0	0	0
1	0	1029	0	1	0	0	1	0	0	0
0	0	4	1003	0	1	0	1	1	0	0
0	0	0	0	976	0	0	0	2	4	0
3	0	0	5	0	881	3	0	0	0	0
6	2	0	0	1	1	948	0	0	0	0
1	2	11	2	0	0	0	1010	1	1	0
4	0	3	0	0	0	0	2	963	2	0
1	2	0	1	5	3	0	2	5	990	1



Classification Report

Using scikit-learn we can automatically generate a **Classification Report** that gives us **Recall, Precision, F1 and Support**.

	precision	recall	f1-score	support
0	0.98	1.00	0.99	980
1	0.99	1.00	1.00	1135
2	0.98	1.00	0.99	1032
3	0.99	0.99	0.99	1010
4	0.99	0.99	0.99	982
5	0.99	0.99	0.99	892
6	0.99	0.99	0.99	958
7	0.99	0.98	0.99	1028
8	0.99	0.99	0.99	974
9	0.99	0.98	0.99	1009



F1-Score and Support

- **F1 Score** is the weighted average of Precision and Recall. As a result, this score takes both false positives and false negatives into account. It is useful if you have an **uneven class distribution** as accuracy works best if false positives and false negatives have similar cost.
 - The **f1**-scores corresponding to every class will tell you the accuracy of the classifier in classifying the data points in that particular class compared to all other classes
- The **support** is the number of samples of the true response that lie in that class.



More on Recall vs. Precision

- **High recall (or sensitivity) with low precision.**
 - This tells us that most of the positive examples are correctly recognized (low False Negatives) but there are a lot of false positives i.e. other classes being predicted as our class in question.
- **Low recall (or sensitivity) with high precision.**
 - Our classifier is missing a lot of positive examples (high FN) but those we predict as positive are indeed positive (low False Positives)



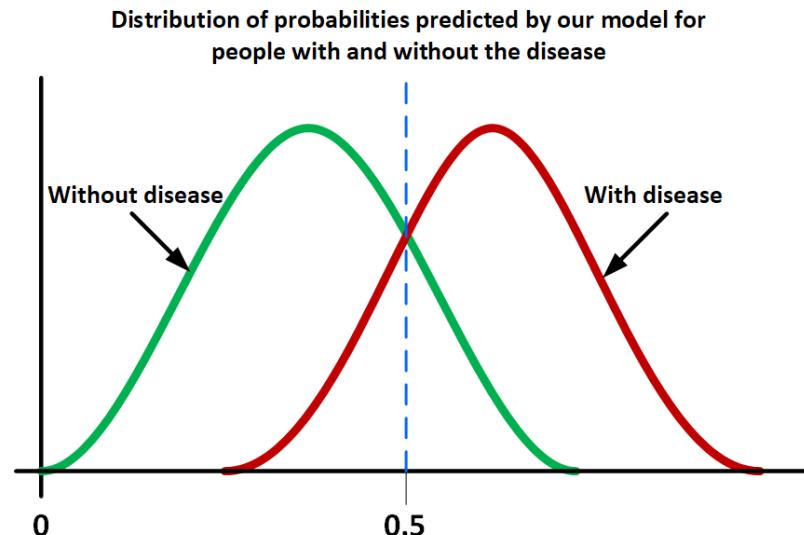
**Let's compare the
performance between
Logistic Regressions, SVMs
and KNN Classifiers in Python**

Understanding the ROC and AUC Curve



ROC (Receiver Operating Characteristic)

- The ROC curve is very important as it tells us how good our model is at **distinguishing between two classes**.
- Imagine we have a model that predicts whether someone has a disease or not.
- In the diagram on the right we use 0.5 as our threshold to identify someone as having the disease.
- However, look at the area around 0.5, there can be a lot of ambiguity if a sample point lies in that region.





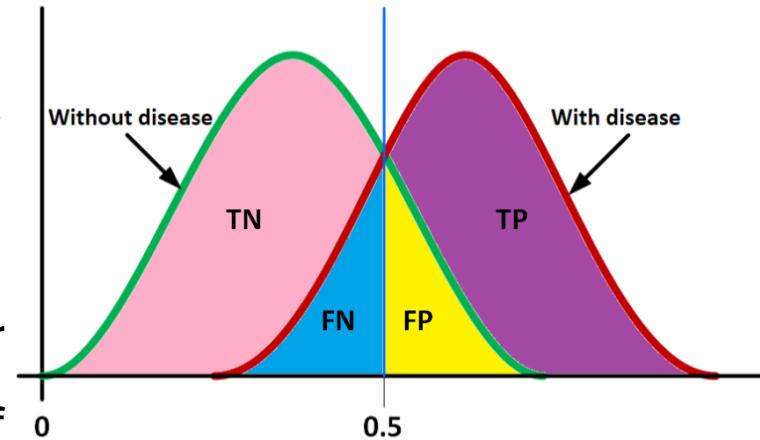
Remember our Recall and Precision

- **Recall** - in this example, Recall would be the number of patients who our model identified as having the disease (TP) over the total number of patients that actually have the disease.

- $$\circ \quad Recall = \frac{TP}{TP+FN}$$

- **Precision** - in this example would be the number of patients who our model identified as NOT having the disease (TN) over the total number of patients that actually did NOT have the disease.

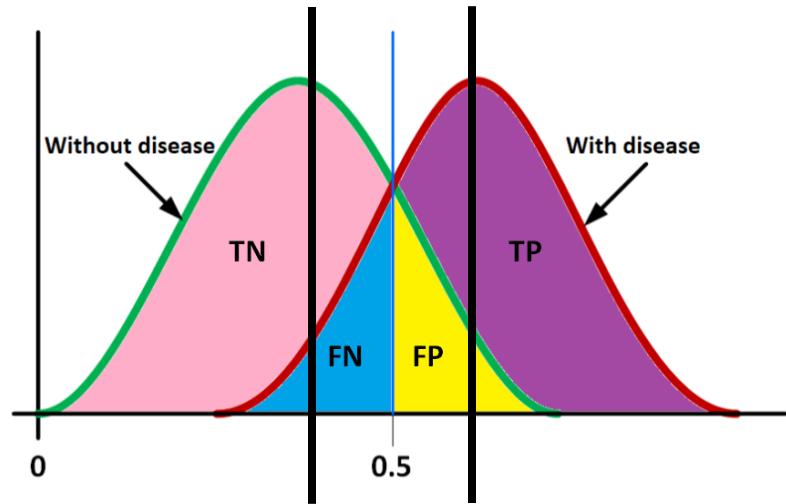
- $$\circ \quad Precision = \frac{TN}{TN+FP}$$





How Thresholds affect Recall and Precision

- Let's make a **lower** threshold moving it from **0.5** to **0.4**.
 - We get **more positive** predictions of a patient having the disease
 - Increase in FP
 - Decrease in FN
- This will **decrease the Precision** and **increase Recall**
- If we did the reverse and increased it **to 0.6** we get:
 - More negative predictions (i.e. less classifications saying we found the disease)
 - Decrease in FP
 - Increase in FN
- This will **increase the Precision** and **reduce Recall**.

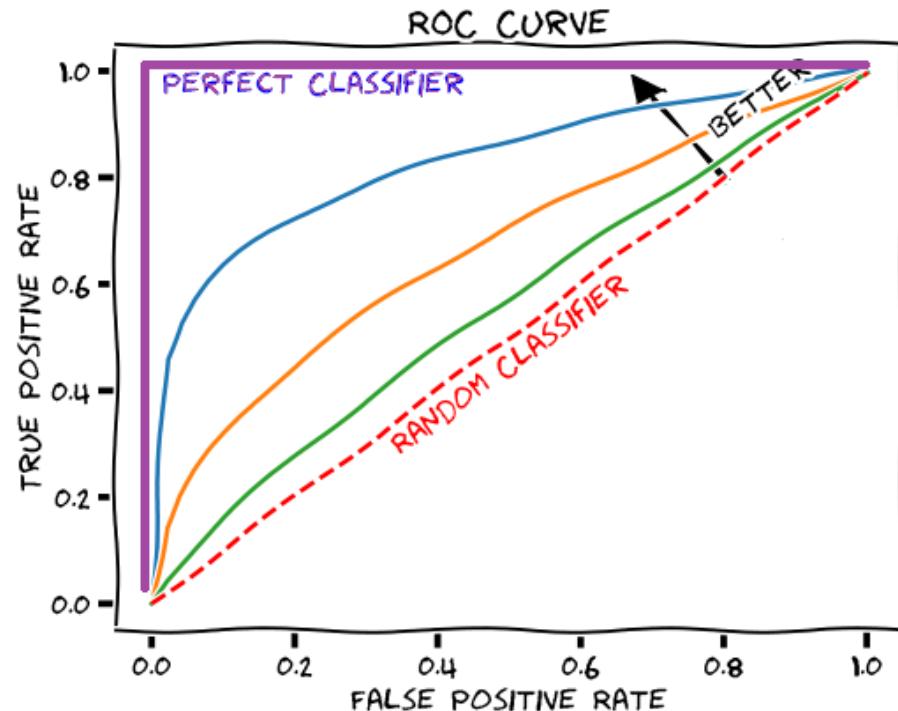


As Recall decreases Precision increases
As Precision decreases Recall increases



The ROC Curve

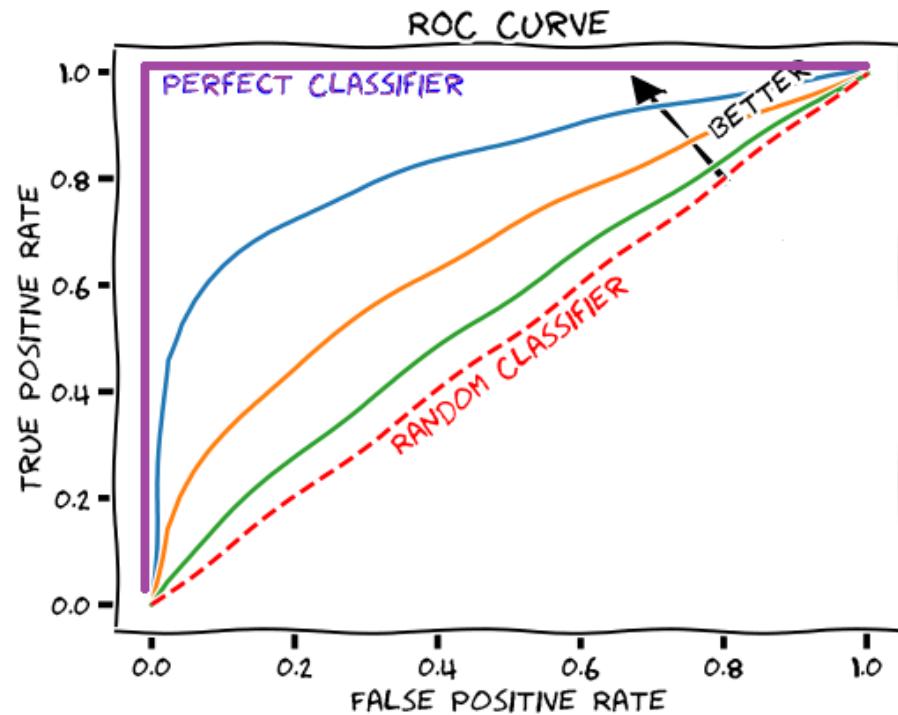
- The ROC Curve is the plot of **Recall** or our **True Positive Rate (TPR)** against (**1-Precision**) or our **False Positive Rate (FPR)**.
- **TPR** is the proportion of people with the disease that were correctly identified by our model.
- The **FPR** is the proportion of people identified by our model to not have the disease that were false positives.
- The **AUC** or **Area under the Curve** is an overall measure of performance, the greater area the better the classifier.





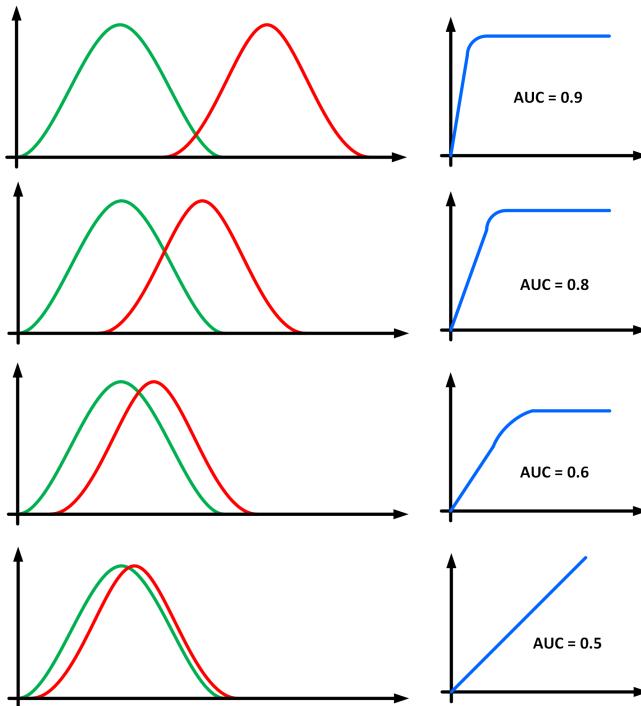
Creating the ROC Curve

- The Area under the ROC Curve is very important when comparing models. Generally the model with the greater AUC is better.
- However, how is this curve generated?
- We simply use different thresholds in our classifier to get our TPR and FPR values and plot them on a graph, connecting the lines to gives us this curve.





How we use the ROC Curve and AUC



Overfitting – Regularization, Generalization and Outliers



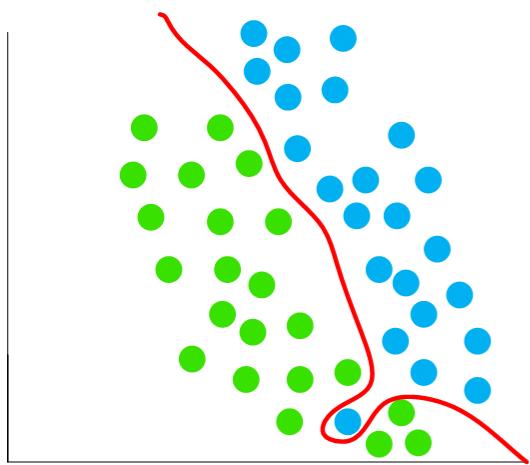
What Makes a Good Model?

- A good model is accurate
- Generalizes well
- Does not overfit
- But what do these things mean?

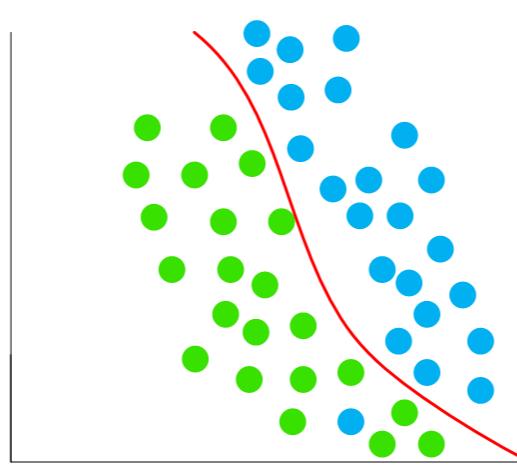


Examine these Models and the Decision Boundary

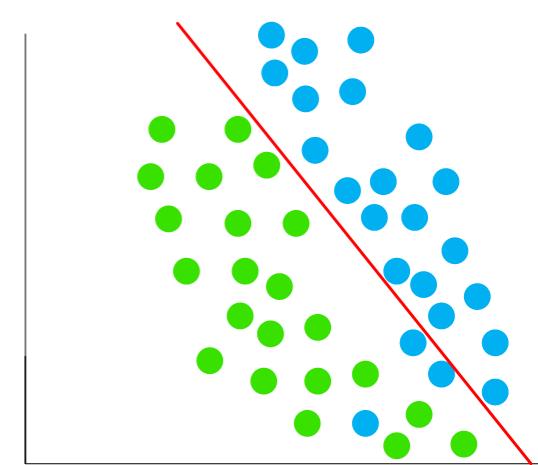
● Model A



● Model B



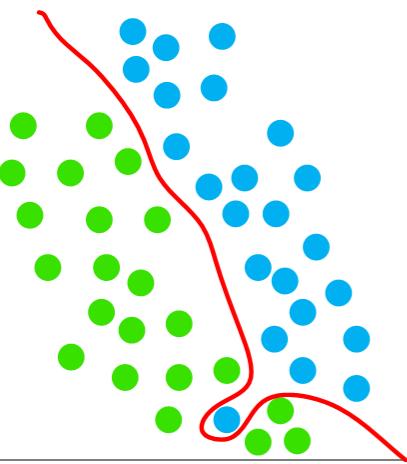
● Model C



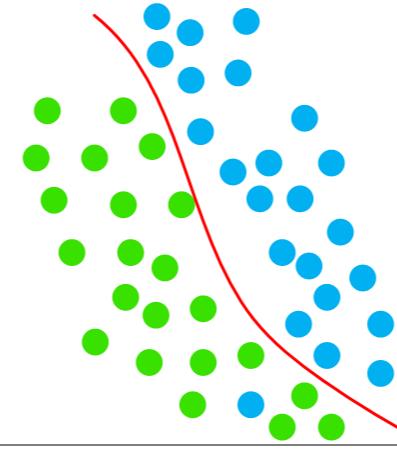


Let's look at Each

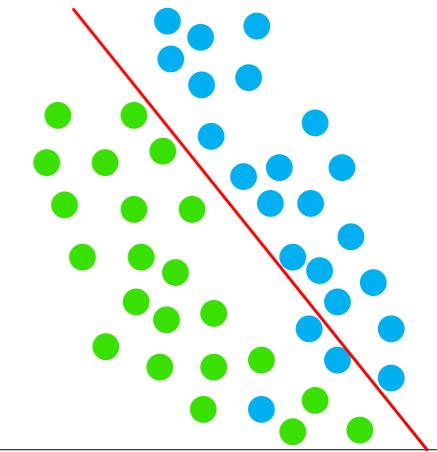
- Overfitting



- Ideal or Balanced



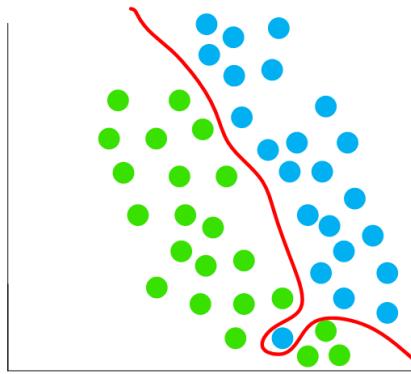
- Underfitting





Overfitting and Underfitting

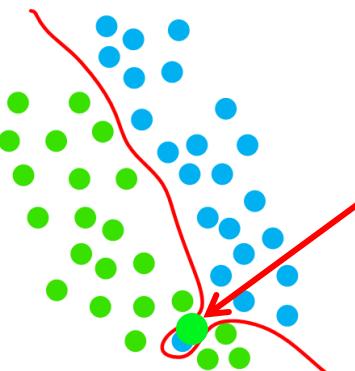
- Overfitting occurs when a statistical model or machine learning algorithm captures the noise of the data. Overfitting occurs if the model or algorithm shows **low bias** but **high variance**
- Underfitting occurs when a statistical model or machine learning algorithm cannot capture the underlying trend of the data





Overfitting

- Overfitting leads to poor models and is one of the most common problems faced developing in AI/Machine Learning/Neural Nets.
- Overfitting occurs when our Model fits near perfectly to our training data, as we saw in the previous slide with Model A. However, fitting too closely to training data isn't always a good thing.

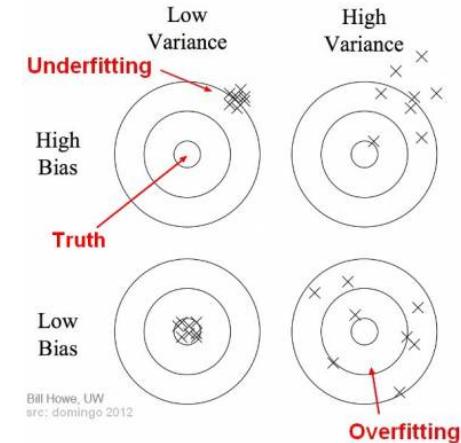


- What happens if we try to classify a brand new point that occurs at the position shown on the left? (who's true color is green)
- It will be misclassified because our model has overfit the test data
- Models don't need to be complex to be good



Overfitting – Bias and Variance

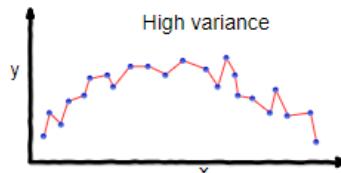
- **Bias Error** – Bias is the difference between the average prediction and the true value we're attempting to predict. Bias error results from simplifying assumptions made by a model to make the target function easier to learn. Common low bias models are Decision Trees, KNN and SVMs. **Parametric** models like Linear Regression and Logistic Regression have high-bias (i.e. more assumptions). They're often fast to train, but less flexible.
- **Variance Error** – Variance error originates from how much would your target model change if different training data were used. Normally, we do expect some variance in models for different training data, however, the underlying model should be generally the same. High variance models (Decision Trees, KNN and SVMs) are very sensitive to this whereas low variance models (Linear Regression and Logistic Regression) are not.
- Parametric or linear models have high bias and low variance
- Non-parametric or non-linear have low bias and high variance



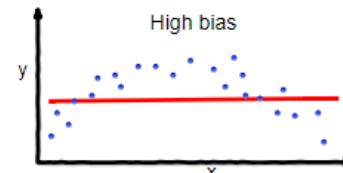


The Bias and Variance Tradeoff

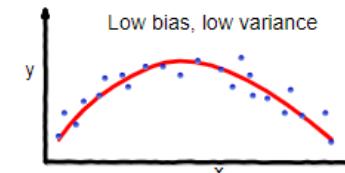
- **Trade-Off** – We want both low bias and low variance, however as we increase bias we decrease variance and vice versa. But how does this happen?
- In linear regression, if we increase the degrees in our polynomial, we are lowering the bias but increasing the variance, conversely if we reduce the complexity we increase bias but reduce variance
- In K-nearest neighbors, increasing the number of clusters (k) increases bias but reduces variance



overfitting



underfitting



Good balance



How do we know if we've Overfit?

Test on your model on.....Test Data!

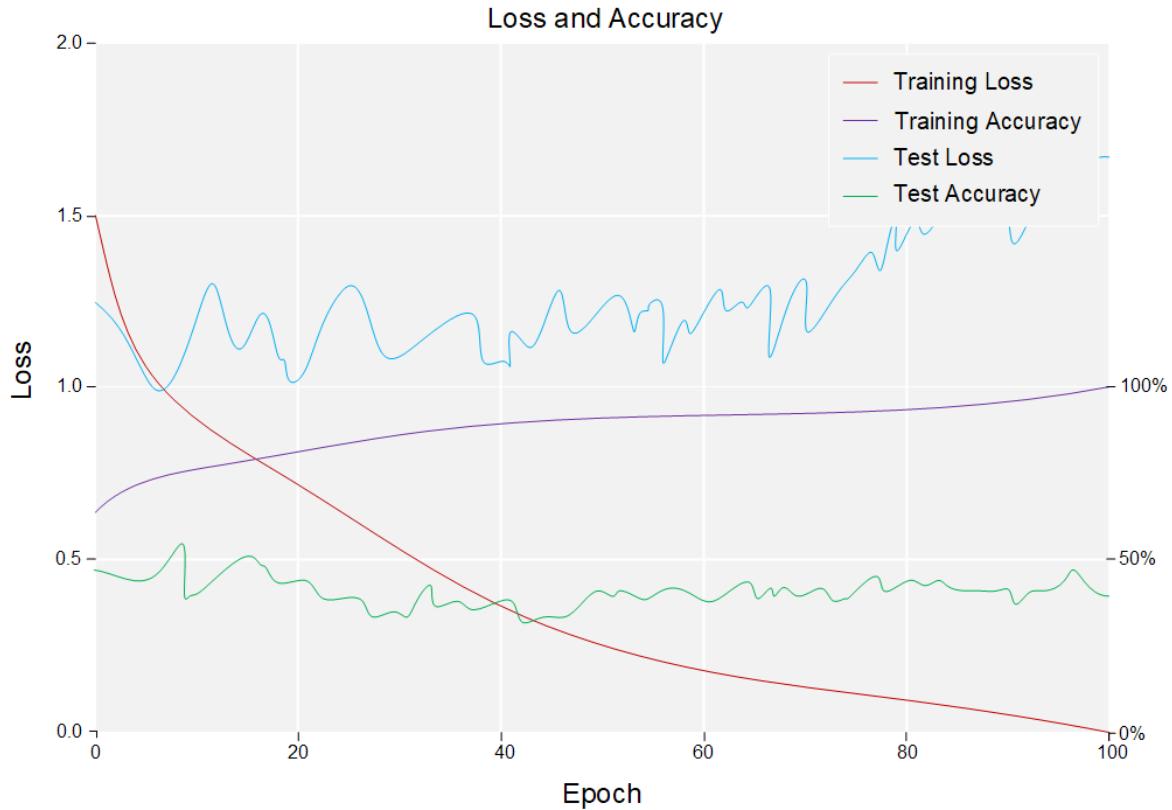
- In all Machine Learning it is extremely important we hold back a portion of our data (10-30%) as pure untouched test data.



- Untouched meaning that this data is NEVER seen by the training algorithm. It is used purely to test the performance of our model to assess its accuracy in classifying new never before seen data.
- Many times when Overfitting we can achieve high accuracy 95%+ on our **training data**, but then get abysmal (~70%) results on the test data. That is a perfect example of Overfitting.



Overfitting Illustrated Graphically

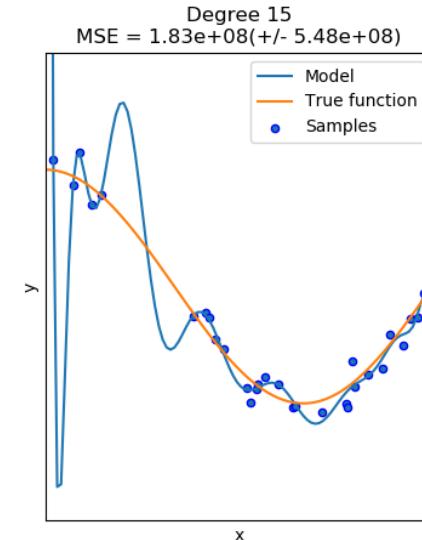
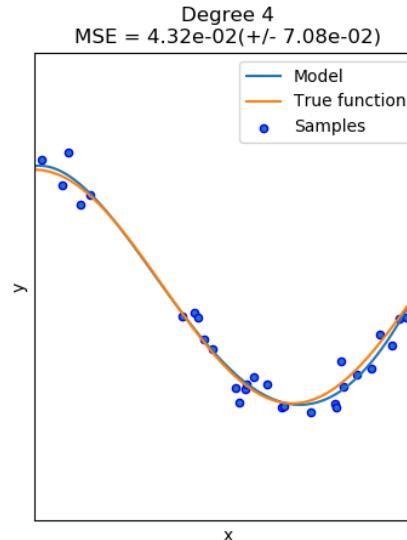
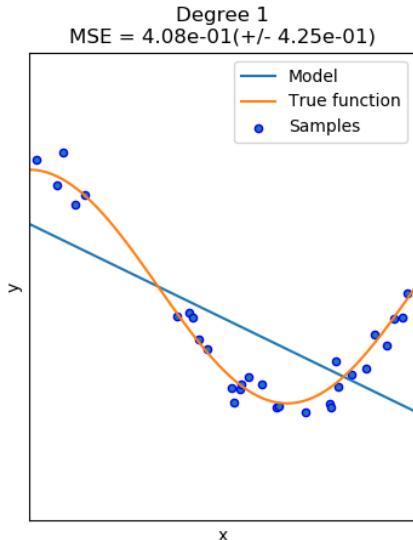


- Examine our Training Loss and Accuracy. They're both heading the right directions!
- But, what's happening to the loss and accuracy on our Test data?
- This is classic example of Overfitting to our training data



How do we avoid overfitting?

- Rule of thumb is to reduce the complexity of your model





How do we avoid overfitting?

- Why do less complex models overfit less?
- Overly complex (i.e. less degrees in the polynomial or in Deep Learning, shallower networks) can sometimes find features or interpret noise to be important in data, due to their abilities to memorize more features (called memorization capacity)

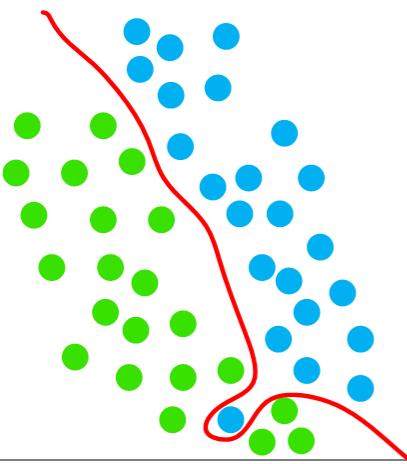
Another method is to use **Regularization!**

- It is better practice to regularize than reduce our model complexity.

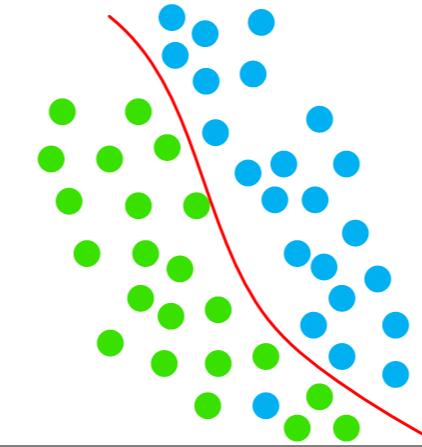


What is Regularization?

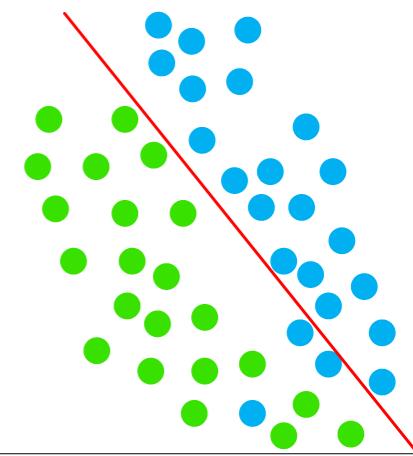
- It is a method of making our model more general to our dataset.
- Overfitting



- Ideal or Balanced



- Underfitting





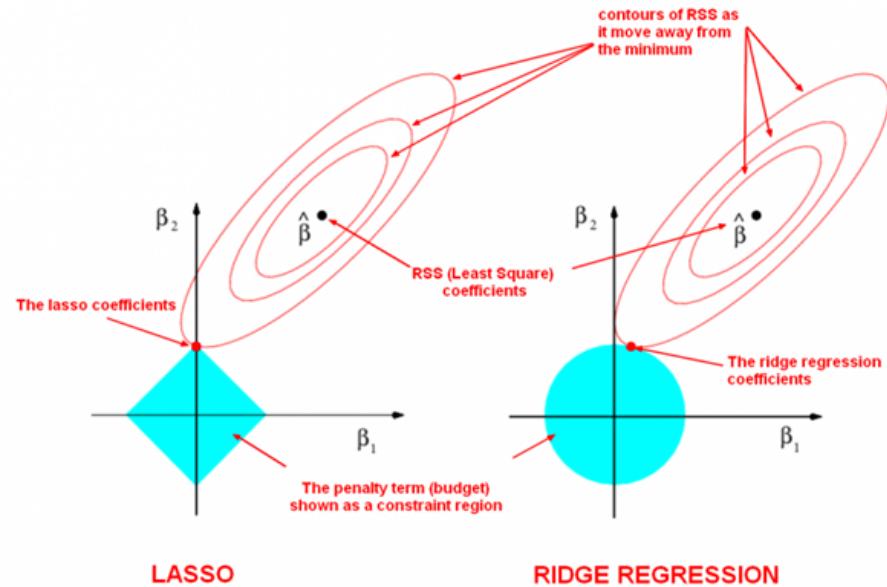
Types of Regularization

- In most Machine Learning Algorithms we can use:
 - L1 & L2 Regularization
 - Cross Validation
- In Deep Learning we can use:
 - Early Stopping
 - Drop Out
 - Dataset Augmentation



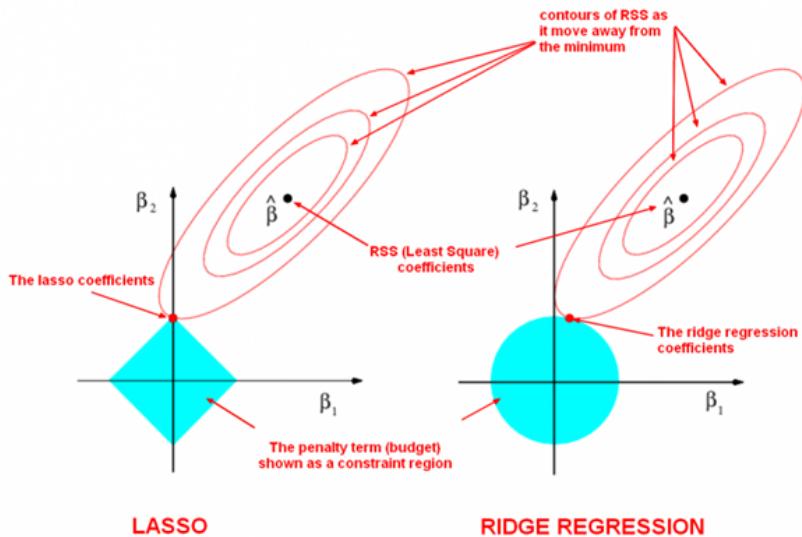
L1 And L2 Regularization

- L1 & L2 regularization are techniques we use to penalize large weights. Large weights or gradients manifest as abrupt changes in our model's decision boundary. By penalizing, we are really making them smaller.
- L2 also known as Ridge Regression
 - $Error = \frac{1}{2}(\text{target}_{01} - \text{out}_1)^2 + \frac{\lambda}{2} \sum w_i^2$
- L1 also known as Lasso Regression
 - $Error = \frac{1}{2}(\text{target}_{01} - \text{out}_1)^2 + \frac{\lambda}{2} \sum |w_i|$
- λ controls the degree of penalty we apply.
- The difference between them is that L1 brings the weights of the unimportant features to 0, thus acting as feature selection algorithm (known as sparse models or models with reduced parameters.)





L1 And L2 Regularization



On the left: LASSO regression (you can see that the coefficients, represented by the red rings, can equal zero when they cross the y-axis). On the right: Ridge regression (you can see that the coefficients approach, but never equal zero, because they never cross the y-axis).

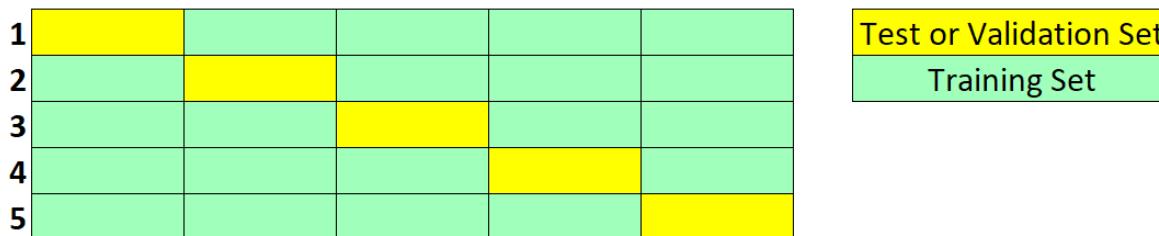
Meta-credit: "[Regularization in Machine Learning](#)" by Prashant Gupta



Cross Validation

- Cross validation or also known as k-fold cross validation is a method of training where we split our dataset into k folds instead of a typical training and test split.
- For example, let's say we're using 5 folds. We train on 4, and use the 5th final fold as our test. We then train on the other 4 folds, and test on another.
- We then use the average weights across coming out of each cycle.
- Cross Validation reduces overfitting but slows the training process

k-folds (5 shown)



Introduction to Neural Networks

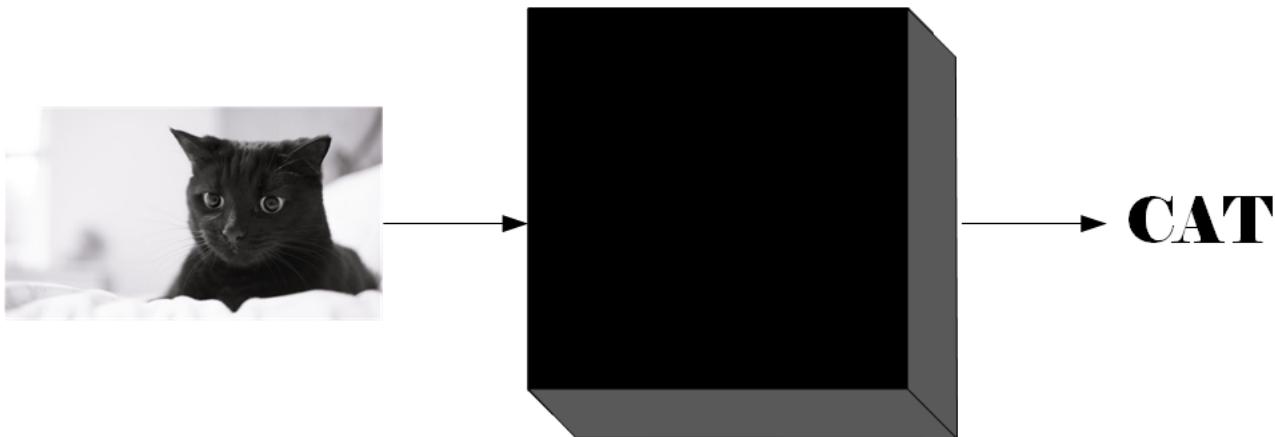


What are Neural Networks

- Neural Networks act as a '**black box**' or brain that takes inputs and predicts an output.
- It's different and 'better' than most traditional Machine Learning algorithms because it **learns complex non-linear mappings** to produce far more accurate output classification results.



The Mysterious 'Black Box' Brain





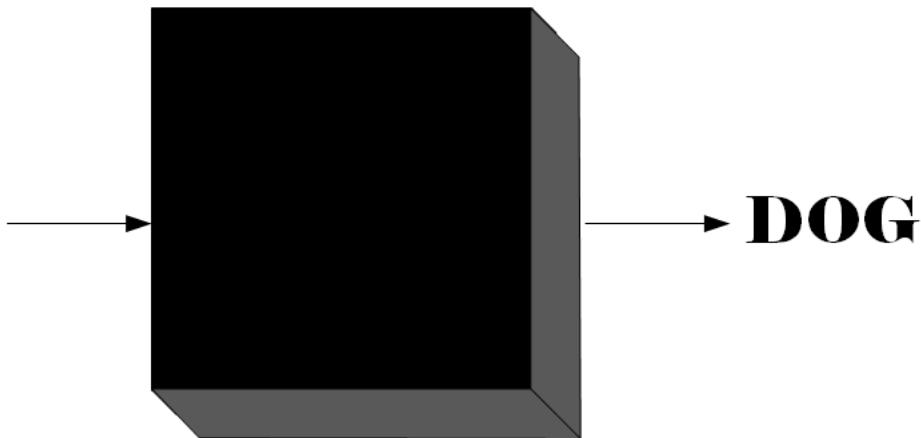
The Mysterious 'Black Box' Brain



GORILLA



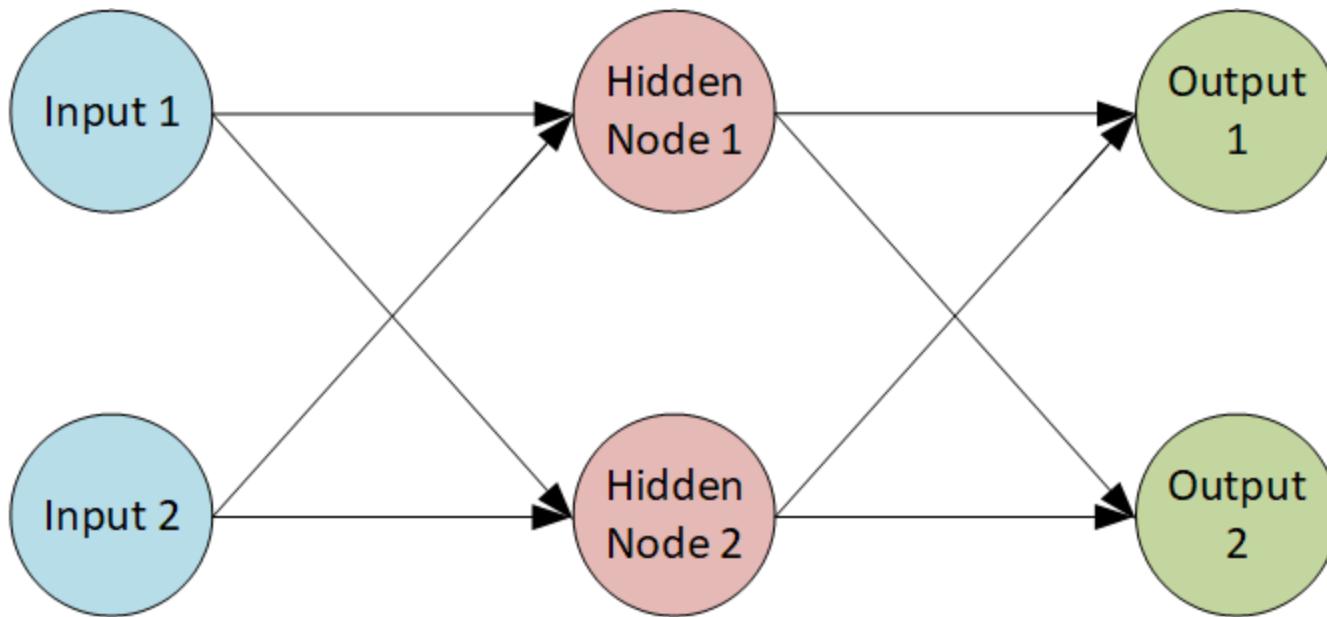
The Mysterious 'Black Box' Brain





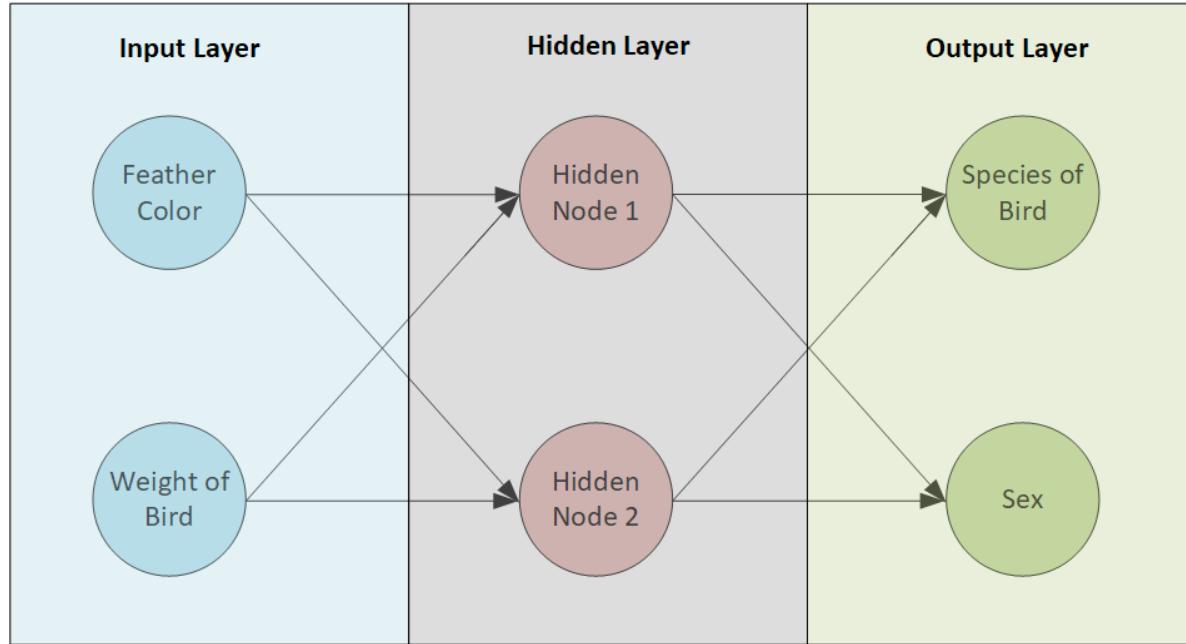
How NNs 'look'

A Simple Neural Network with 1 hidden layer





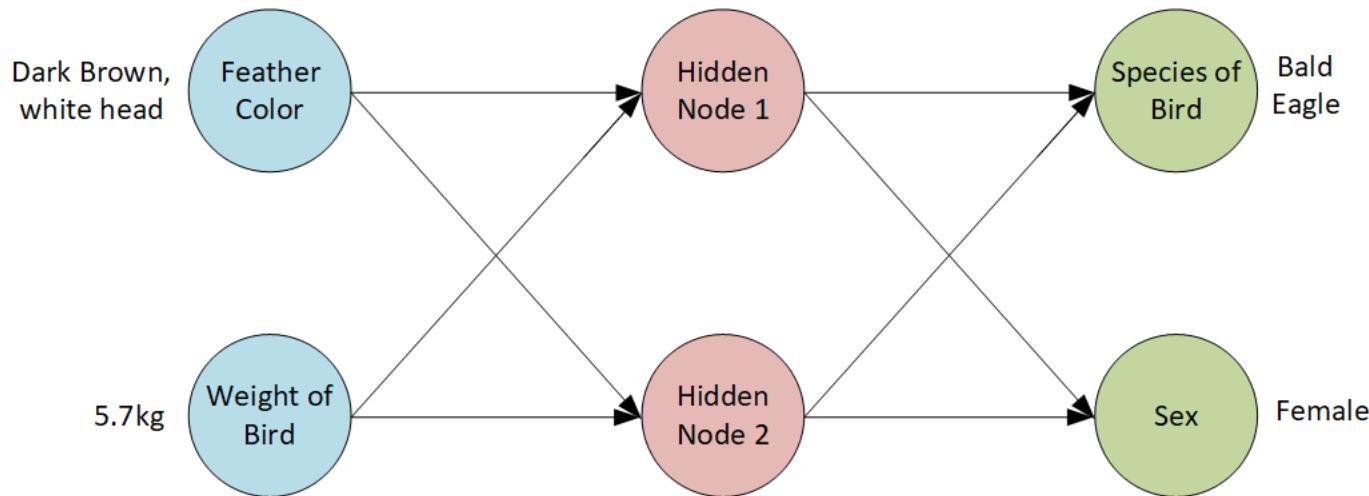
Example Neural Network





How do we get a prediction?

Pass the inputs into our NN to receive an output





How we predict example 2

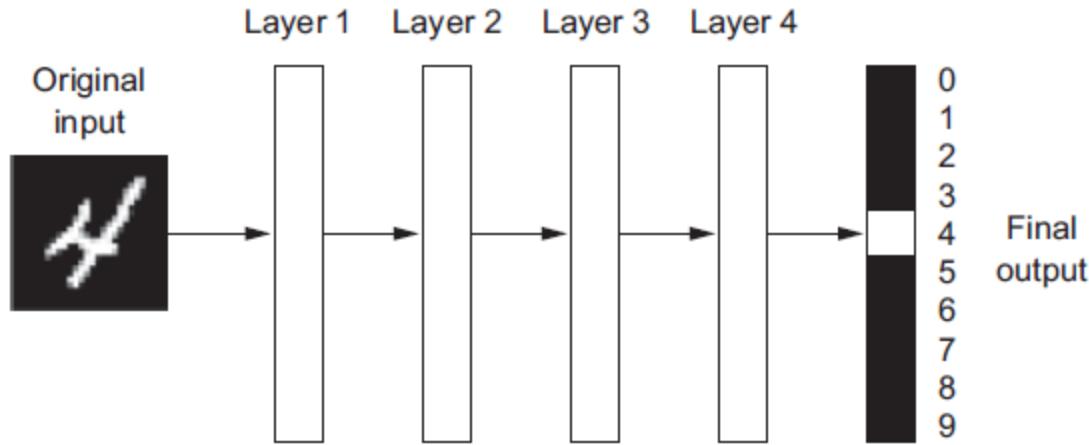


Figure 1.5 A deep neural network for digit classification

Types of Deep Learning Models – Feed Forward, CNNs, RNNs & LSTMs



Deep Learning has Spawned Dozens of Model Types

- With the advent of Deep Learning algorithms obtaining incredible performance, tweaking and designing intricate elements to these Neural Networks has spawned dozens and dozens different models.
- However, despite the dozens of variations they mostly fall into the following categories:
 - Feed Forward Neural Networks
 - Convolutional Neural Networks (**CNN**)
 - Recurrent Neural Networks (**RNN**)
 - Long Short Term Memory Networks (**LSTM**)

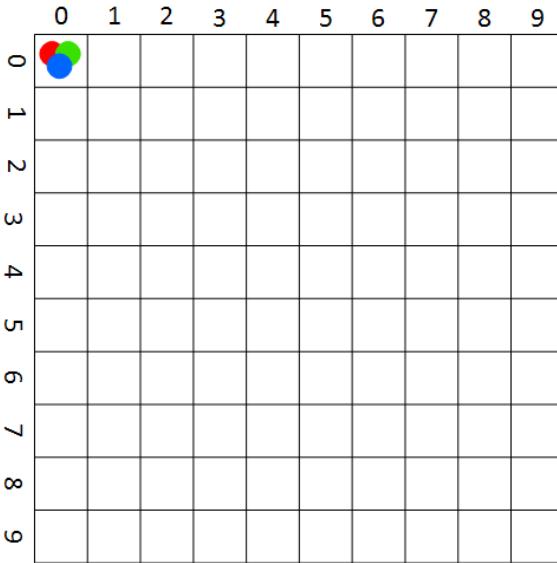


Convolutional Neural Networks (CNNs) – Why are they needed?

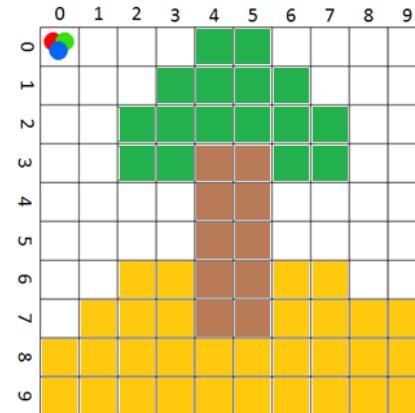
- Because Neural Networks don't scale well to image data



How do Computers Store Images?



- Each pixel coordinate (x , y) contains 3 values ranging for intensities of 0 to 255 (8-bit).
- Red
- Green
- Blue

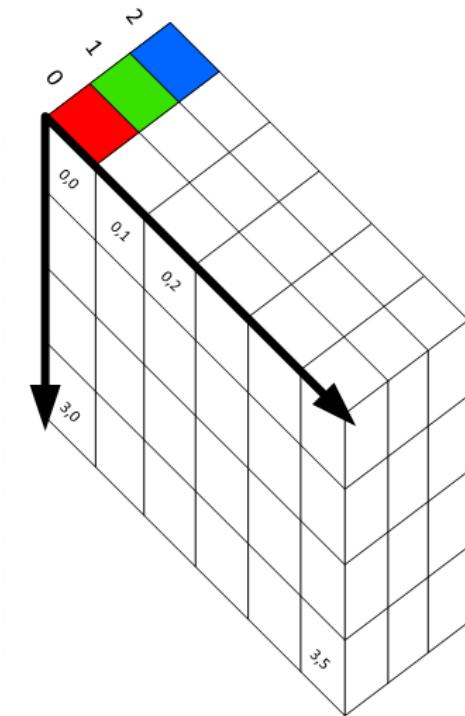


How do Computers Store Images?



- A Color image would consist of 3 channels (RGB) - Right
- And a Grayscale image would consist of one channel - Below

255	255	255	255	255	255	220	146	237	255	255	255	255	255	255
255	255	255	255	255	255	91	10	170	255	255	255	255	255	255
255	255	255	255	255	255	253	68	10	220	255	255	255	255	255
255	255	255	255	255	255	238	30	62	241	255	255	255	255	255
255	255	255	255	255	255	218	64	241	255	255	255	255	255	255
255	255	255	255	255	255	145	64	241	255	255	255	255	255	255
255	255	255	255	255	255	81	72	244	255	255	255	255	255	255
255	255	255	255	255	255	72	128	255	255	255	255	255	255	255
255	255	255	255	255	255	73	146	255	255	255	255	255	255	255
255	255	255	255	255	255	64	7	184	255	255	255	255	255	255
255	255	255	255	255	255	63	10	188	255	255	255	255	255	255
255	255	255	255	255	255	71	9	187	255	255	255	255	255	255
255	255	255	255	255	255	68	9	190	255	255	255	255	255	255
255	255	255	255	255	255	83	160	255	255	255	255	255	255	255
255	255	255	255	255	255	181	48	184	255	255	255	255	255	255



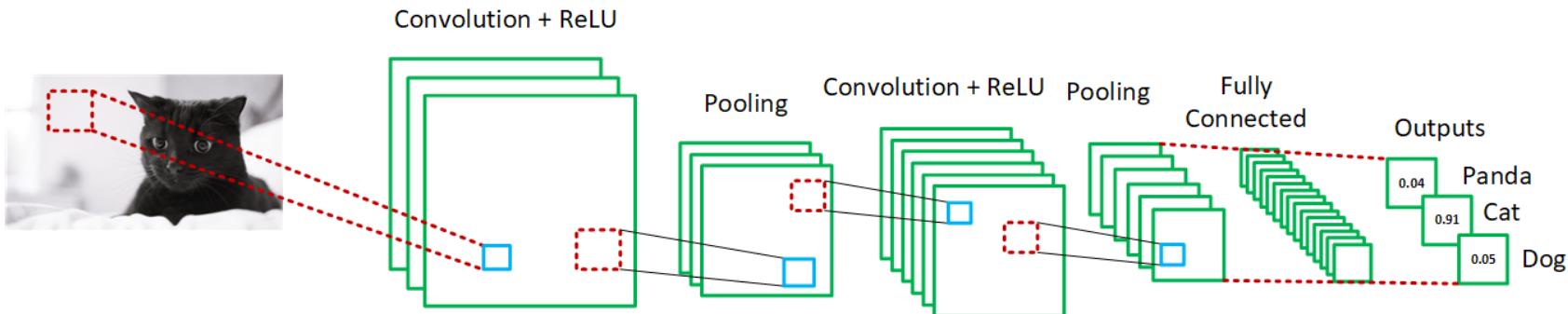


Why NNs Don't Scale Well to Image Data

- Imagine a simple image classifier that takes color images of size 64×64 (height, width).
- The Input size to our NN would be $64 \times 64 \times 3 = 12,288$
- Therefore, our input layer will thus have 12,288 weights. While not an insanely large amount, imagine using input images of 640×480 ? You'd have 921,600 weights! Add some more hidden layers and you'll see how fast this can grow out of control. Leading to very long training times
- However, our input is image data, consists of patterns and correlated inputs. There must be a way to take advantage of this.



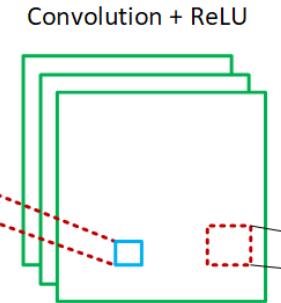
CNNs are perfectly suited for Image Classifications





CNN's use a 3D Volume Arrangement for it's Neurons

- Because our input data is an image, we can constrain or design our Neural Network to better suit this type of data
- Thus, we arrange our layers in **3 Dimensions.**
Why 3?
- Because of image data consists of:
 - Height
 - Width
 - Depth (RGB) our colors components





It's all in the name

- The **Convolution Layer** is the most significant part of a CNN as it is this layer that learns image features which aid our classifier.
- But what exactly is a **Convolution**?

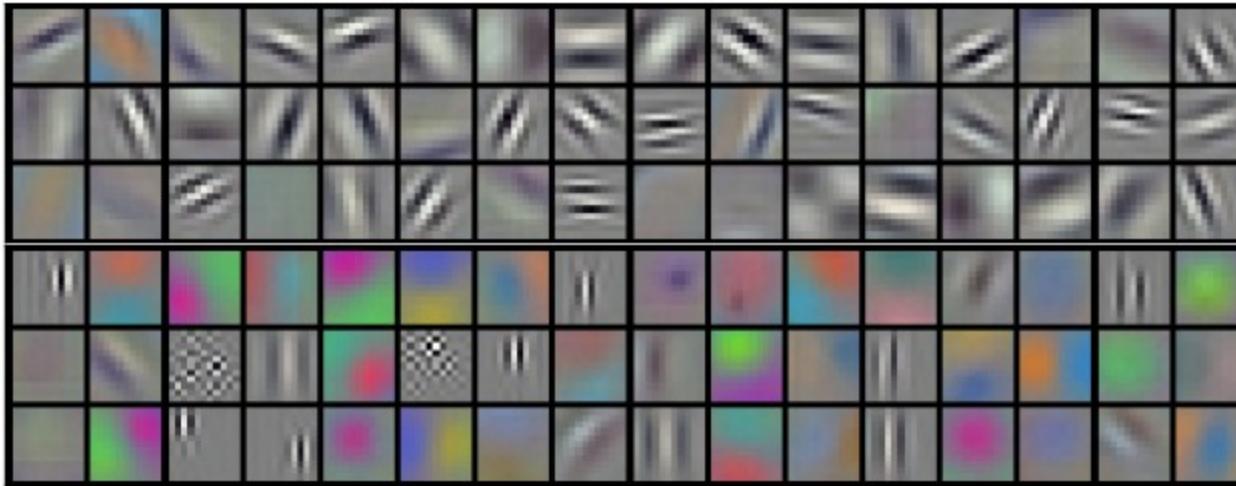


Convolutions

- Convolution is a mathematical term to describe the process of combining **two functions** to produce a **third function**.
- This **third function** or the output is called a **Feature Map**
- A convolution is the action of using a **filter** or **kernel** that is applied to the input. In our case, the input being our image.



Examples of Image Features

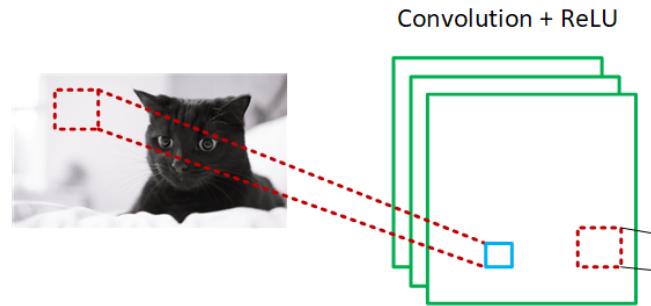
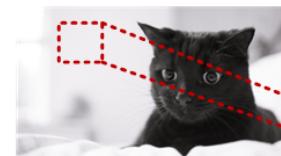


- Example filters learned by Krizhevsky et al.



The Convolution Process

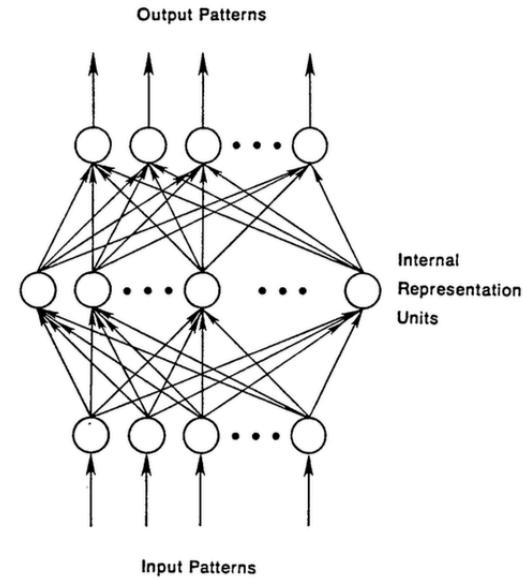
- **Convolutions are executed by sliding the filter or kernel over the input image.**
- This sliding process is a simple matrix multiplication or dot product.





Recurrent Neural Networks

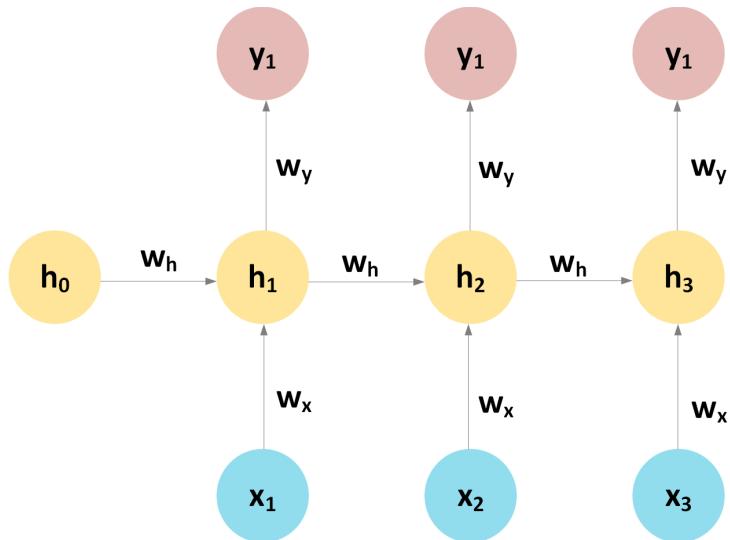
- Feed Forward Neural Networks have no concept of time, each classification is done only on the current inputs
- Recurrent networks**, take as their input not just the current input example they see, but also what they have perceived previously in time.
- Think about them as having two inputs, the present and past inputs.





Recurrent Neural Networks

- RNNs can take the same input but produce **different outputs** depending on the **past inputs**.
- They are influenced not just by weights applied on inputs like a regular NN, but also by a “**hidden state vector (h)** representing the context based on prior input(s)/output(s).





RNN Uses and Weaknesses

- RNNs have been very successfully applied to speech recognition, language modeling and image captioning.
- For RNNs, memory or 'context' awareness is very important. For example, saying the "The sky is...." we don't need much context to know the next word is most likely blue.
- However, saying, "I grew up in Brazil and I speak fluent....." knowing the next word is Portuguese isn't as easy and hence requires the ability to know deeper context. This is a situation where the gap between relevant information and the point where it's needed is very large.



Introducing Long Short Term Memory Networks (LSTM)

- RNNs suffer with **vanishing gradient** and **exploding gradient**, which is some situations make it unusable.
- LSTMs solved this by introducing a memory unit (cell) into the network.
- LSTMs are a type of RNN, capable of learning long-term dependencies.
- LSTMs were designed to avoid the long-term dependency problem.
- They are excellent at remembering information for long periods of time
- LSTMs have the ability of having long chains of information that it decides whether to keep or not by using Gates.

6.0

Neural Networks Explained



Neural Networks Explained

\

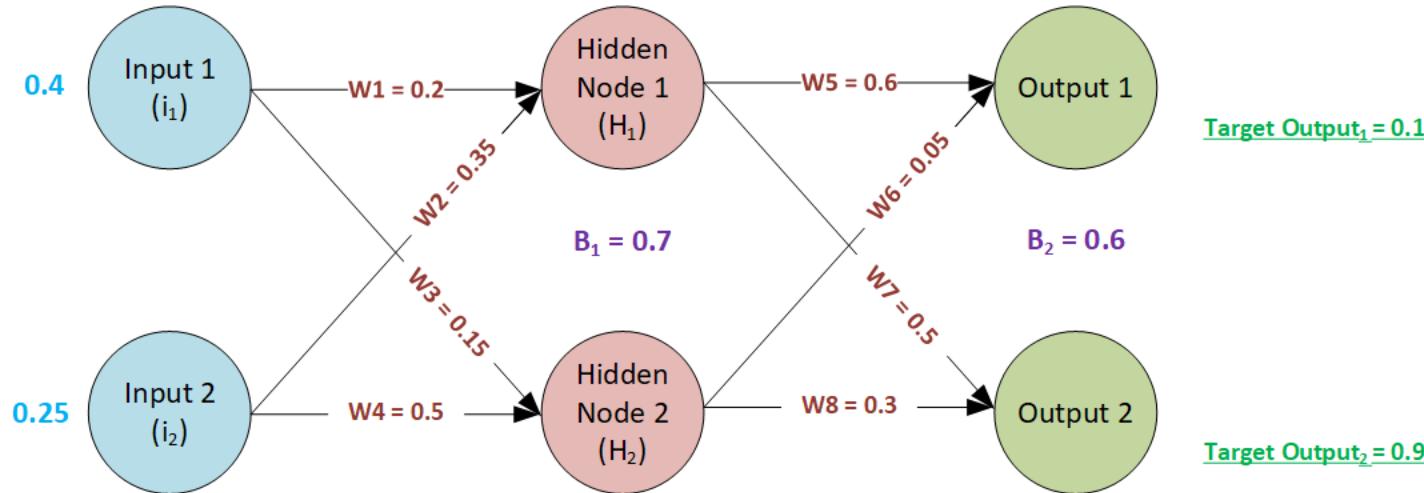
- 6.3 Forward Propagation
- 6.4 Activation Functions
- 6.5 Training Part 1 – Loss Functions
- 6.6 Training Part 2 – Backpropagation and Gradient Descent
- 6.7. Backpropagation & Learning Rates – A Worked Example
- 6.8 Regularization, Overfitting, Generalization and Test Datasets
- 6.9 Epochs, Iterations and Batch Sizes
- 6.10 Measuring Performance and the Confusion Matrix
- 6.11 Review and Best Practices

Forward Propagation

How Neural Networks process their inputs to produce an output

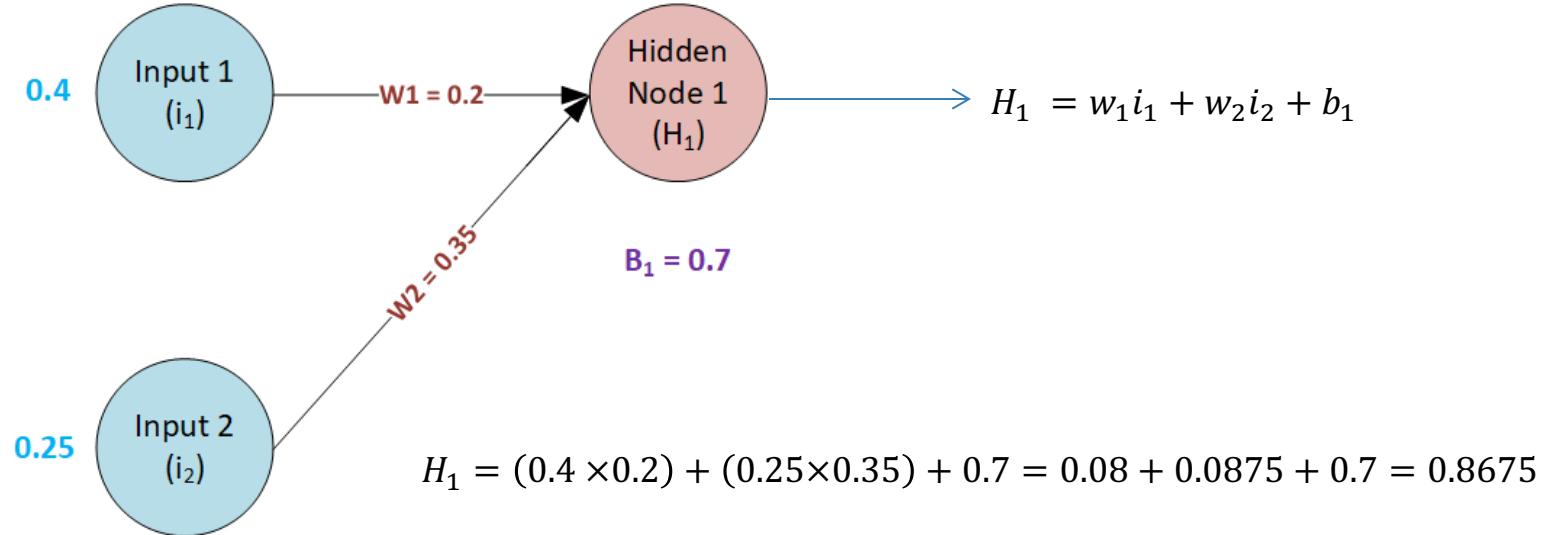


Using actual values (random)



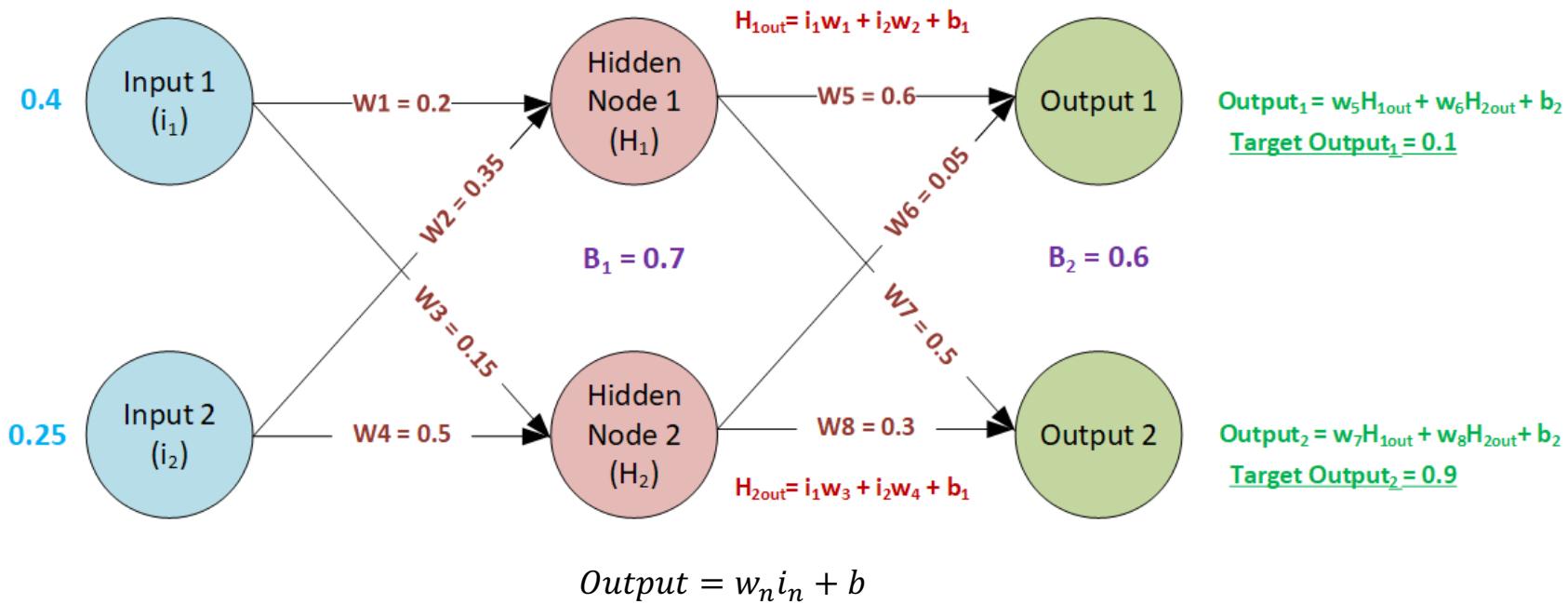


Looking at a single Node/Neuron



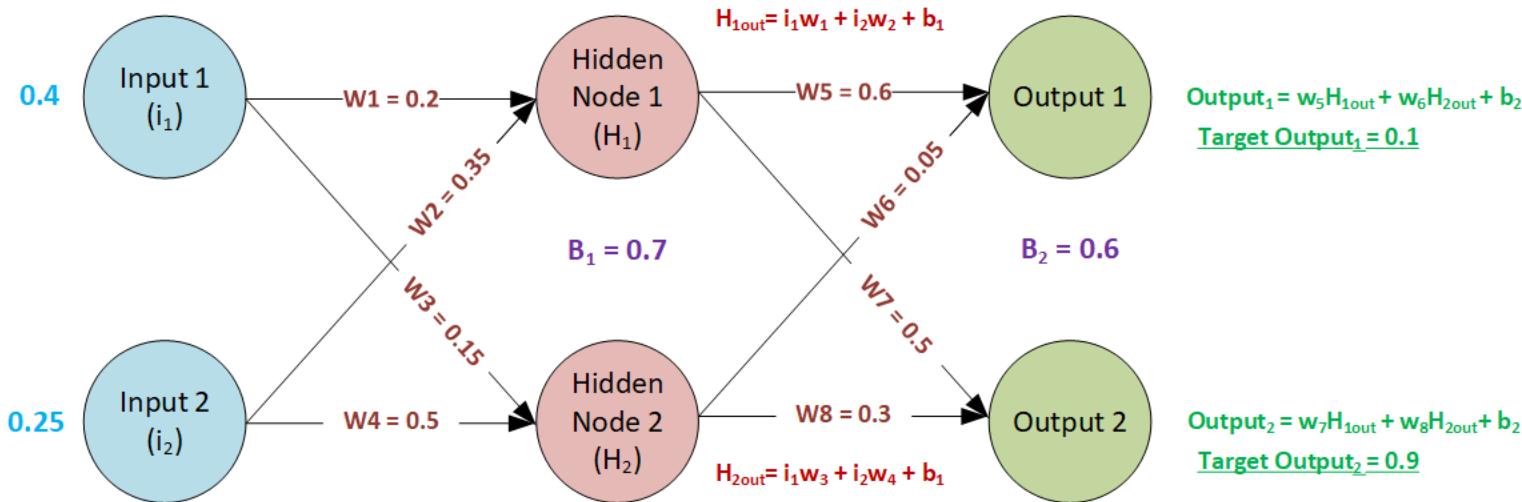


Steps for all output values





In reality these connections are simply formulas that pass values from one layer to the next



$$\text{Output of Nodes} = w_n i_n + b$$

$$H_1 = i_1 w_1 + i_2 w_2 + b_1$$

$$\text{Output}_1 = w_5 H_1 + H_2 w_6 + b_2$$



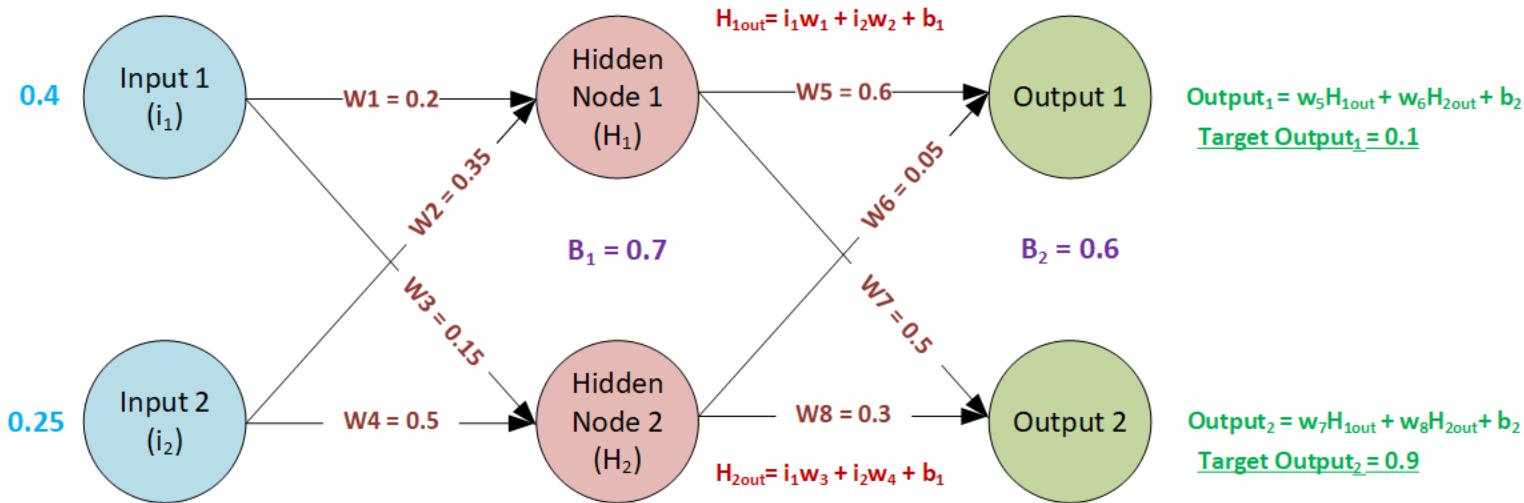
Calculating H_2

$$H_2 = i_1w_3 + i_2w_4 + b_1$$

$$H_2 = (0.4 \times 0.15) + (0.25 \times 0.5) + 0.7 = 0.06 + 0.125 + 0.7 = 0.885$$



Getting our final outputs



$$Output_1 = w_5 H_1 + H_2 w_6 + b_2$$

$$Output_1 = (0.6 \times 0.8675) + (0.05 \times 0.885) + 0.6 = 0.5205 + 0.04425 + 0.6 = 1.16475$$

$$Output_2 = 0.43375 + 0.2655 + 0.6 = 1.29925$$



What does this tell us?

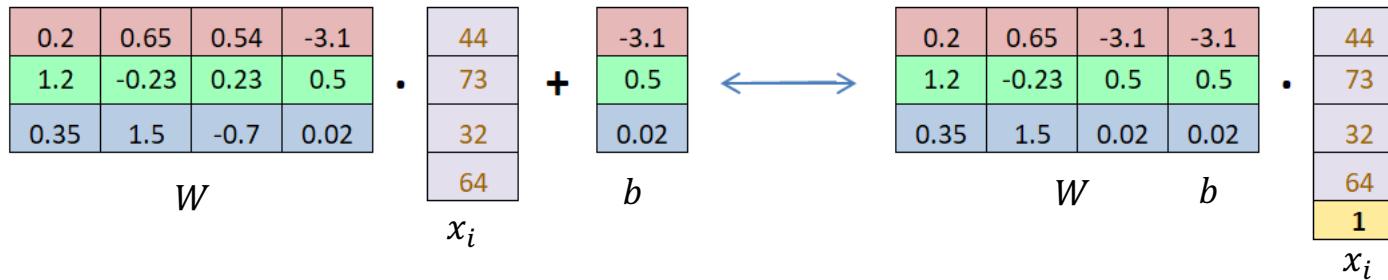
- Our initial default random weights (w and b) produced very incorrect results
- Feeding numbers through a neural network is a matter of simple matrix multiplication
- Our neural network is still just a series of linear equations



The Bias Trick

– Making our weights and biases as one

- Now is a good time as any to show you the bias trick that is used to simplify our calculations.



- x_i is our input values, instead of doing a multiplication then adding of our biases, we can simply add the biases to our weight matrix and add an addition element to our input data as 1.
- This simplifies our calculation operations as we treat the biases and weights as one.
- NOTE:** This makes our input vector size bigger by one i.e if x_i had 32 values, it will now have 33.

Activation Functions & What ‘Deep’ really means

An introduction to activation functions and their usefulness



Introducing Activation Functions

- In reality each Hidden Node also passes through an activation function.
- In the simplest terms an activation function changes the output of that function. For example, let's look at a simple activation function where values below 0 are clamped to 0. Meaning negative values are changed to 0.

Output = Activation ($w_i x_i + b$)

$$f(x) = \max(0, x)$$

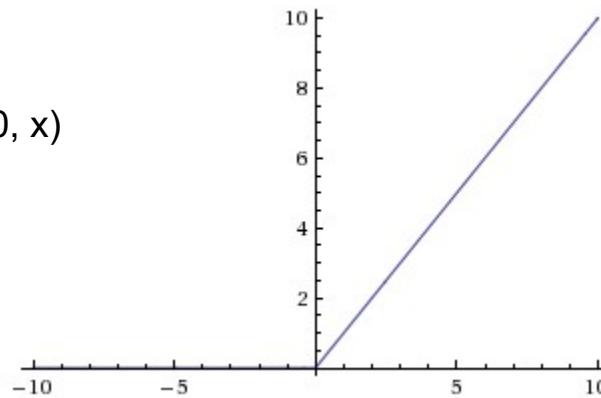
- Therefore, if $w_i x_i + b = 0.75$
- $f(x) = 0.75$
- However, if $w_i x_i + b = -0.5$ then $f(x) = 0$



The ReLU Activation Function

- This activation function is called the ReLU (Rectified Linear Unit) function and is one of the most commonly used activation functions in training Neural Networks.
- It takes the following appearance. The clamping values at 0 accounts for its non linear behavior.

$$f(x) = \max(0, x)$$





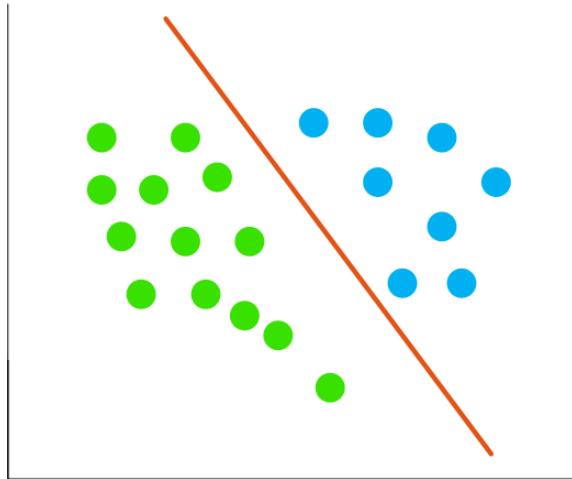
Why use Activation Functions? For Non Linearity

- Most ML algorithms find non linear data extremely hard to model
- The huge advantage of deep learning is the ability to understand nonlinear models

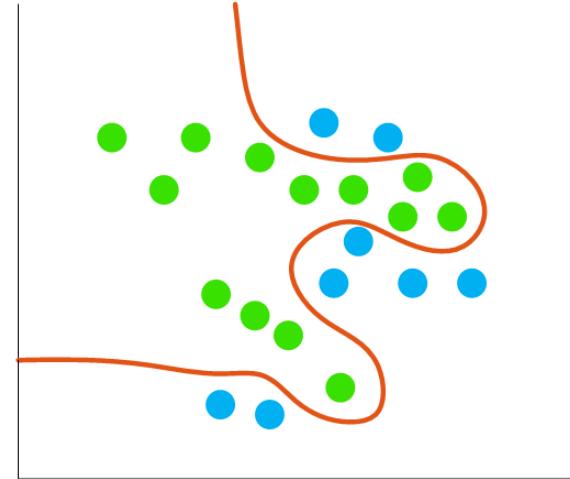


Example of Non Linear Data

Linearly Separable



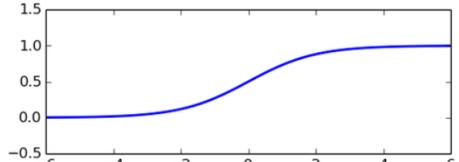
Non-Linearly Separable



NOTE: This shows just 2 Dimensions, imagine separating data with multiple dimensions

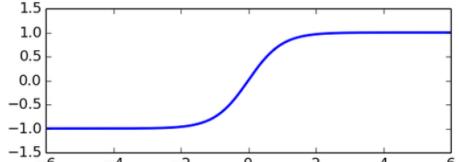


Types of Activation Functions



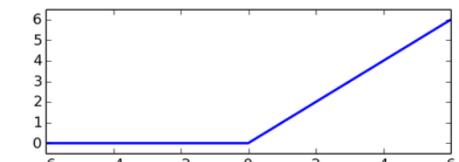
Sigmoid

$$\phi(z) = \frac{1}{1 + e^{-z}}$$



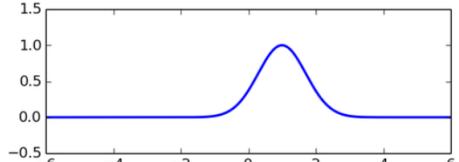
Hyperbolic Tangent

$$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$



Rectified Linear

$$\phi(z) = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{if } z \geq 0 \end{cases}$$



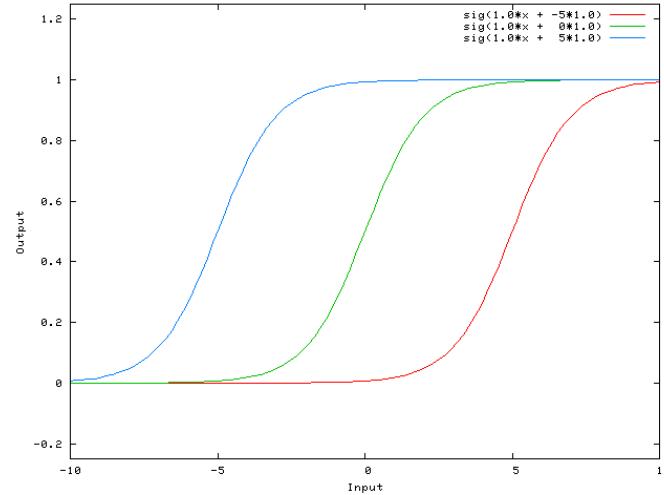
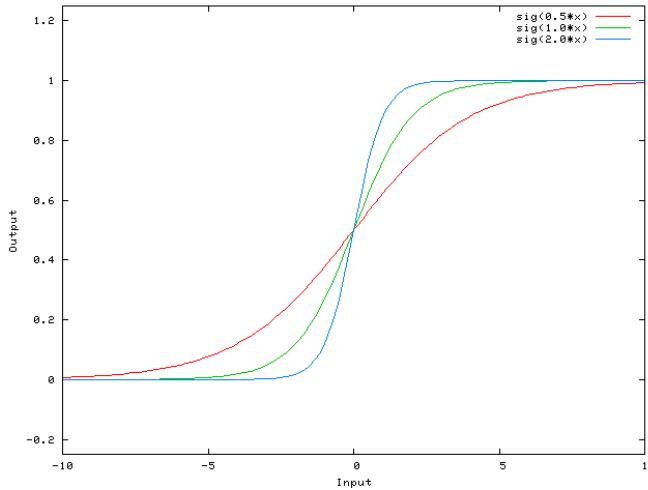
Radial Basis Function

$$\phi(z, c) = e^{-(\epsilon \|z - c\|)^2}$$



Why do we have biases in the first place?

- Biases provide every node/neuron with a trainable constant value.
- This allows us to shift the activation function output left or right.

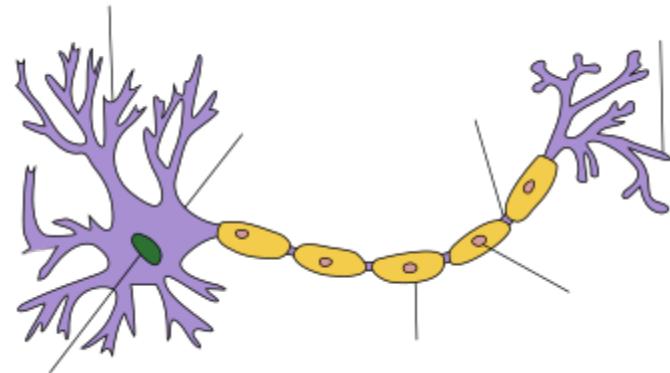


- Changes in weights simply change the gradient or steepness of the output, if we needed shift our function left or right , we need a bias.



Neuron Inspiration

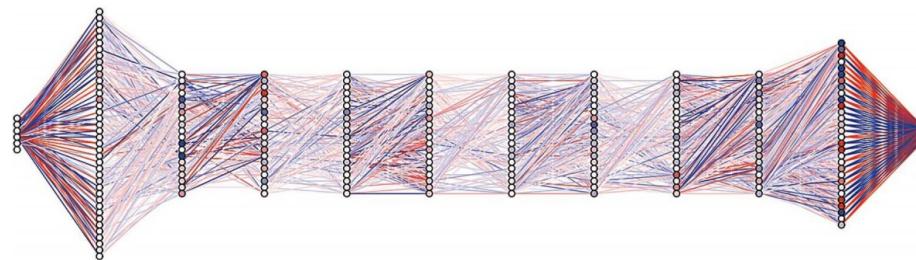
- Neuron only fires when an input threshold is reached
- Neural Networks follow that same principle





The ‘Deep’ in Deep Learning: Hidden Layers

- Depth refers to the number of hidden layers
- The deeper the network the better it learns non-linear mappings
- Deeper is always better, however there becomes a point of diminishing returns and overly long training time.
- Deeper Networks can lead to over fitting



Source: A visualization of Meade's neural network for predicting earthquakes

<https://harvardsmagazine.com/2017/11/earthquakes-around-the-world>



The Real Magic Happens in Training

- We've seen Neural Networks are simple to execute, just matrix multiplications
- How do we determine those weights and biases?

Training Part 1: Loss Functions

The first step in determining the best weights for our Neural Network



Learning the Weights

- As you saw previously, our random default weights produced some very bad results.
- What we need is a way to figure out how to change the weights so that our results are more accurate.
- This is where the brilliance of Loss Functions , Gradient Descent and Backpropagation show their worth.



How do we begin in training a NN? What exactly do we need?

- Some (*more than some*) accurately labeled data, that we'll call a dataset
- A Neural Network Library/Framework (*Keras*)
- Patience and a decently fast computer

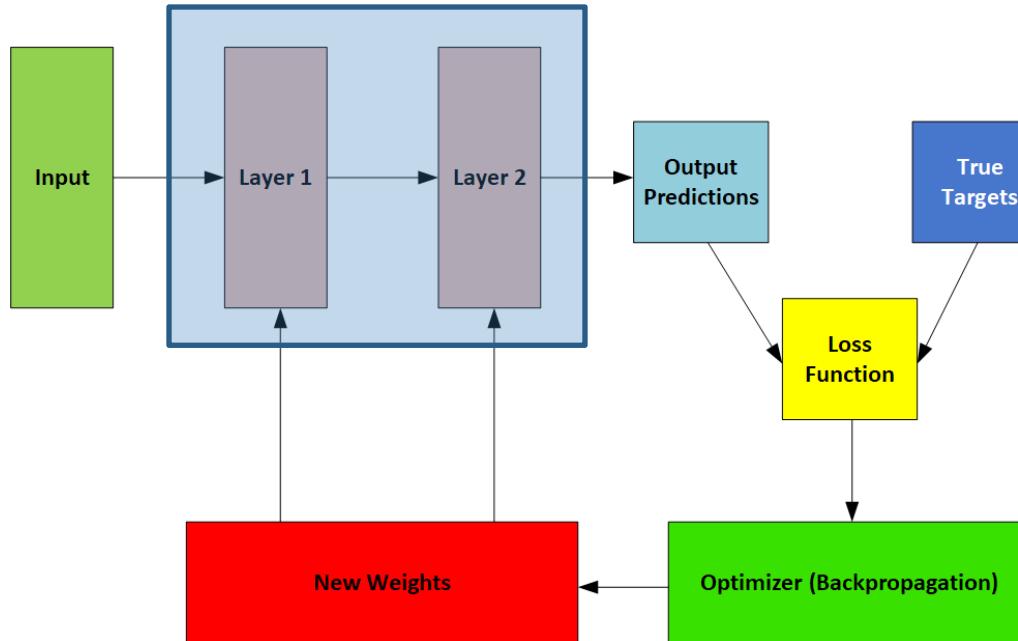


Training step by step

1. Initialize some random values for our weights and bias
2. Input a single sample of our data
3. Compare our output with the actual value it was supposed to be, we'll be calling this our Target values.
4. Quantify how 'bad' these random weights were, we'll call this our Loss.
5. Adjust weights so that the Loss lower
6. Keep doing this for each sample in our dataset
7. Then send the entire dataset through this weight 'optimization' program to see if we get an even lower loss
8. Stop training when the loss stops decreasing.



Training Process Visualized





Quantifying Loss with Loss Functions

- In our previous example our Neural Network produced some very 'bad' results, but how do we measure how bad they are?

Outputs	Predicted Results	Target Values	Difference (P-T)
1	1.16475	0.1	1.06475
2	1.29925	0.9	0.39925

Loss Functions



- Loss functions are integral in training Neural Nets as they measure the inconsistency or difference between the predicted results & the actual target results.
- They are always positive and penalize big errors well
- The lower the loss the 'better' the model
- There are many loss functions, Mean Squared Error (MSE) is popular
- $\text{MSE} = (\text{Target} - \text{Predicted})^2$

Outputs	Predicted Results	Target Values	Error (T-P)	MSE
1	1.16475	0.1	-1.06475	1.1336925625
2	1.29925	0.9	-0.39925	0.1594005625



Types of Loss Functions

- There are many types of loss functions such as:
 - L1
 - L2
 - Cross Entropy – Used in binary classifications
 - Hinge Loss
 - Mean Absolute Error (MAE)

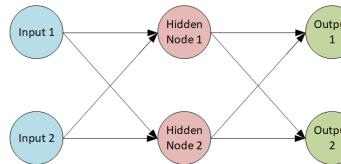
In practice, MSE is always a good safe choice. We'll discuss using different loss functions later on.

NOTE: A low loss goes hand in hand with accuracy. However, there is more to a good model than good **training** accuracy and low loss. We'll learn more about this soon.



Using the Loss to Correct the Weights

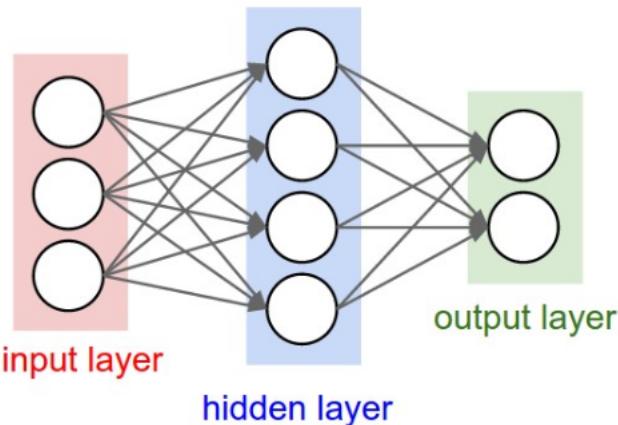
- Getting the best weights for our classifying our data is not a trivial task, especially with large image data which can contain thousands of inputs from a single image.



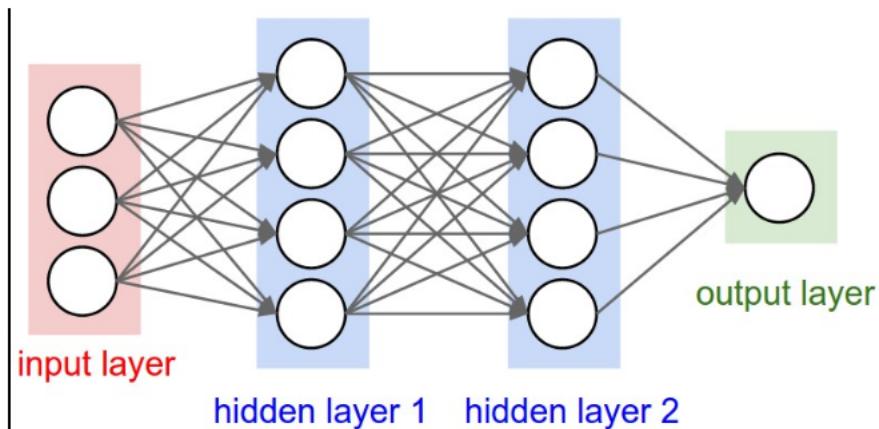
- Our simple 2 input, 2 output and 1 hidden layer network has X parameters.
 - Input Nodes X Hidden Layers + Hidden Layers x Output + Biases
 - In our case this is: $(2 \times 2) + (2 \times 2) + 4 = 12$ Learnable Parameters



Calculating the Number of Parameters



$$(3 \times 4) + (4 \times 2) + (4+2) = 26$$



$$(3 \times 4) + (4 \times 4) + (4 \times 1) + (4+4+1) = 41$$

Training Part 2: Backpropagation & Gradient Descent

Determining the best weights efficiently



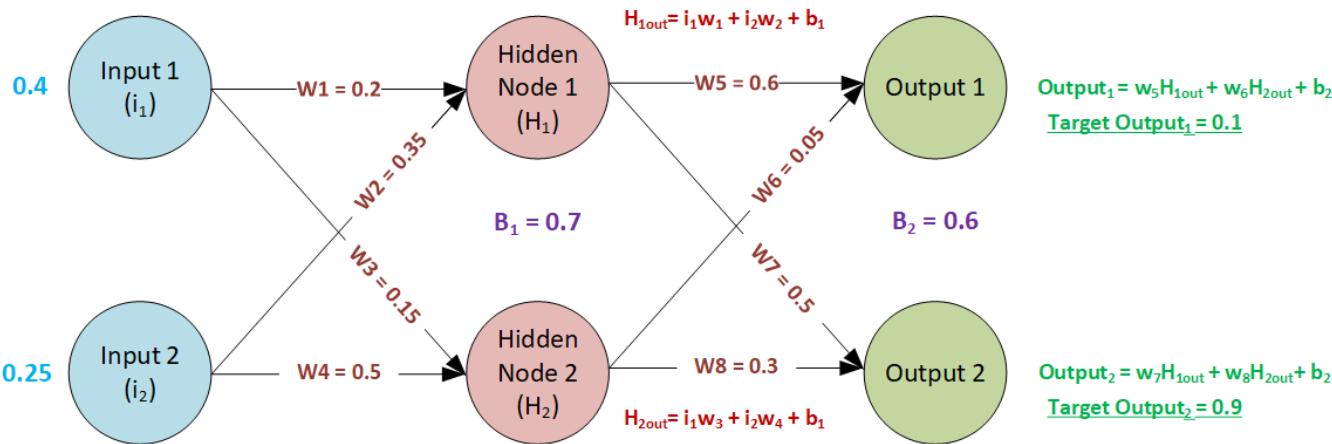
Introducing Backpropagation

- What if there was a way we could use the loss to determine how to adjust the weights.
- That is the brilliance of Backpropagation

Backpropagation tells us how much would a change in each weight affect the overall loss.



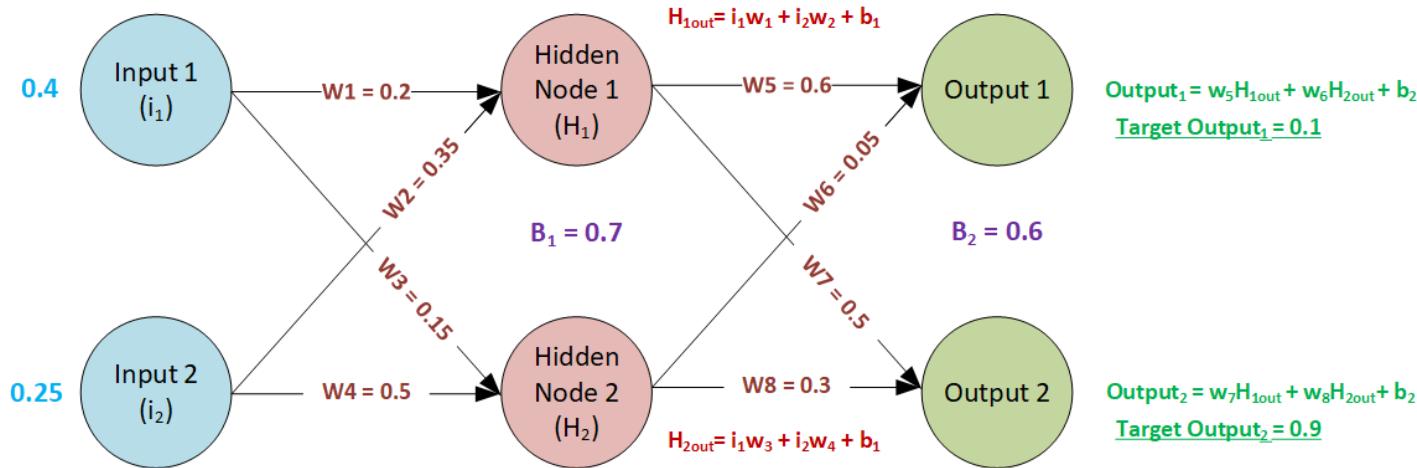
Backpropagation: Revisiting our Neural Net



- Using the MSE obtained from Output 1, Backpropagation allows us know:
 - If changing w_5 from 0.6 by a small amount, say to 0.6001 or 0.5999,
 - Whether our overall Error or Loss has increased or decreased.



Backpropagation: Revisiting our Neural Net



- We then backpropagate this loss to each node (Right to Left) to determine which direction the weight should move (negative or positive)
- We do this for all nodes in the Neural Network



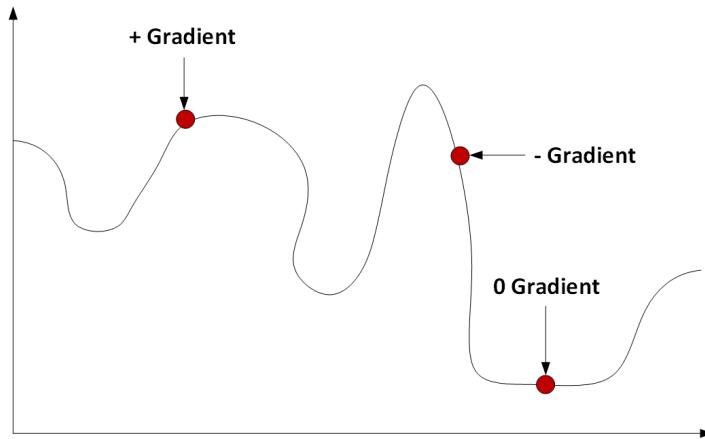
Backpropagation – The full cycle

- Therefore, by simply passing one set of inputs of a single piece of our training data, we can adjust all weights to reduce the loss or error.
- However, this tunes the weights for that specific input data. How do make our Neural Network generalize?
- We do this for each training samples in our training data (called an Epoch or Iteration).



Introducing Gradient Descent

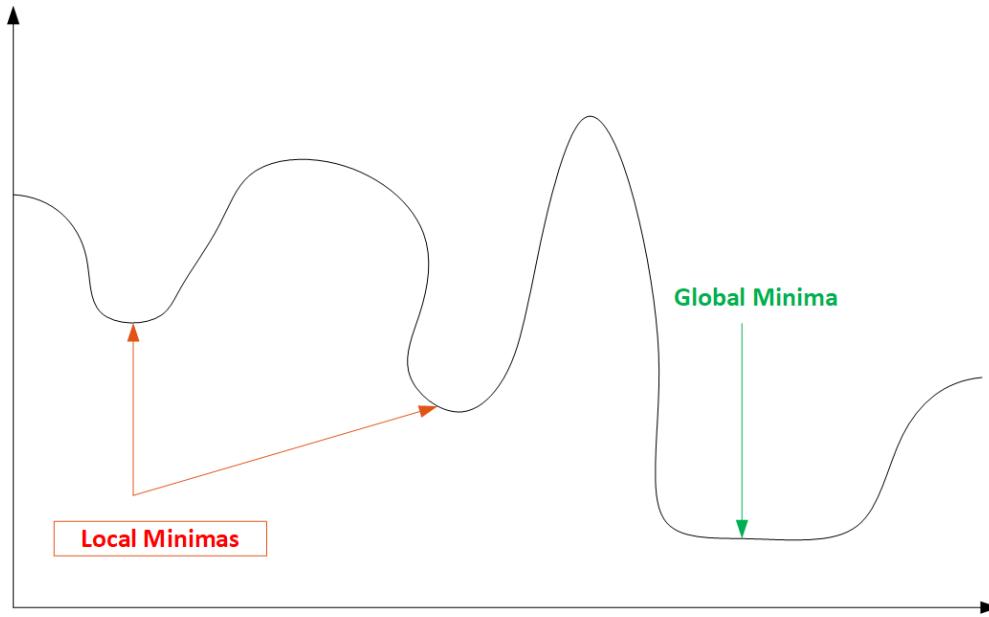
- By adjusting the weights to lower the loss, we are performing gradient descent. This is an 'optimization' problem.
- Backpropagation is simply the method by which we execute gradient descent.
- Gradients (also called slope) are the direction of a function at a point, it's magnitude signifies how much the function is changing at that point.



- By adjusting the weights to lower the loss, we are performing gradient descent.
- Gradients are the direction of a function at a point



Gradient Descent



Imagine our global minima is the bottom of this rough bowl. We need to traverse through many peaks and valleys before we find it



Stochastic Gradient Descent

- Naïve Gradient Decent is very computationally expensive/slow as it requires exposure to the entire dataset, then updates the gradient.
- Stochastic Gradient Descent (SGD) does the updates after every input sample. This produces noisy or fluctuating loss outputs. However, again this method can be slow.
- Mini Batch Gradient Descent is a combination of the two. It takes a batch of input samples and updates the gradient after that batch is processed (batches are typical 20-500, though no clear rule exists). It leads to much faster training (i.e. faster convergence to the global minima)



Overview

We learned:

- That Loss Functions (such as MSE) quantify how much error our current weights produce.
- That Backpropagation can be used to determine how to change the weights so that our loss is lower.
- This process of optimizing or lowering the weights is called Gradient Descent, and an efficient method of doing this is the Mini Batch Gradient Descent algorithm.

Back Propagation & Learning Rates: A Worked Example

A worked example of Back Propagation.



Backpropagation

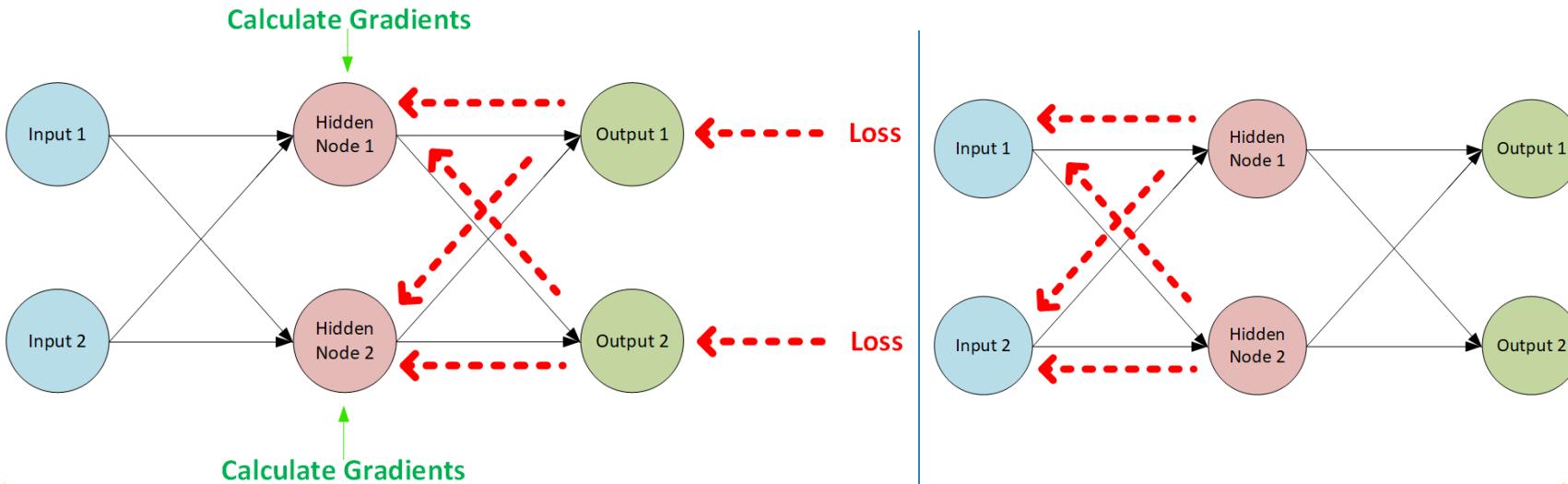
- From the previous section you have an idea of what we achieve with Backpropagation.
- It's a method of executing gradient descent or weight optimization so that we have an efficient method of getting the lowest possible loss.

How does this 'black magic' actually work?



Backpropagation Simplified

- We obtain the total error at the output nodes and then propagate these errors back through the network using Backpropagation to obtain the new and better gradients or weights.





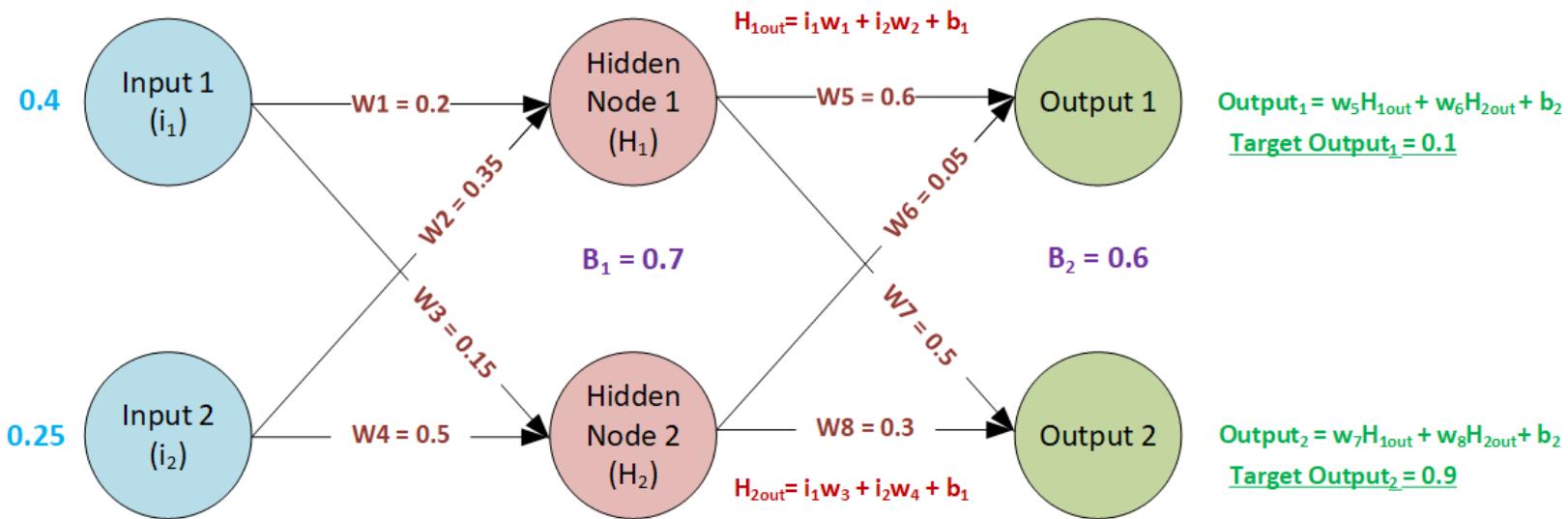
The Chain Rule

- Backpropagation is made possible by the *Chain Rule*.
- What is the Chain Rule? Without over complicating things, it's defined as:
 - If we have two functions: $y = f(u)$ and $u = g(x)$ then the derivative of y is:

$$\frac{dy}{dx} = \frac{dy}{du} \times \frac{du}{dx}$$

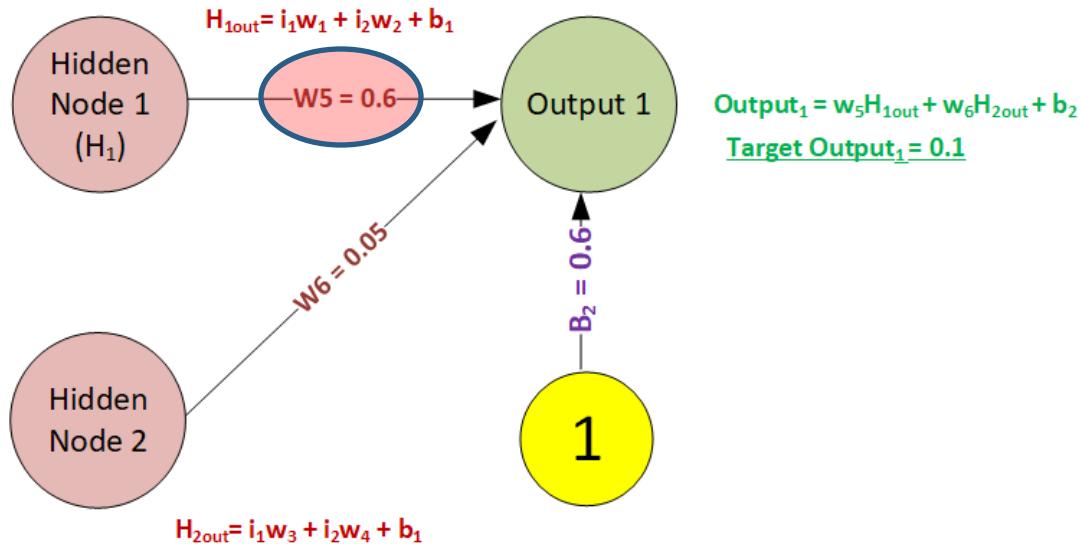


Let's take a look at our previous basic NN





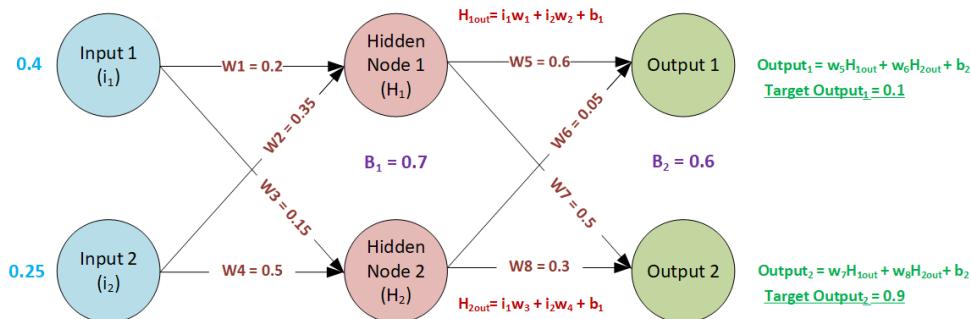
We use the Chain Rule to Determine the Direction the Weights Should Take



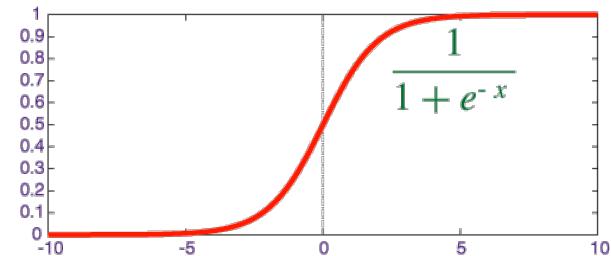
- Let's take a look at W_5 , how does a change in W_5 affect the Error at Output?
- Should W_5 be increased or decreased?



Our Calculated Forward Propagation and Loss Values



Logistic Activation Function
Squashes the output between 0 and 1



- Using a Logistic Activation Function at each node, our Forward Propagation values become:

- $H_1 = \mathbf{0.704225}$
- $H_2 = \mathbf{0.707857}$

- $\text{Output } 1 = \mathbf{0.742294}$
- $\text{Output } 2 = \mathbf{0.762144}$

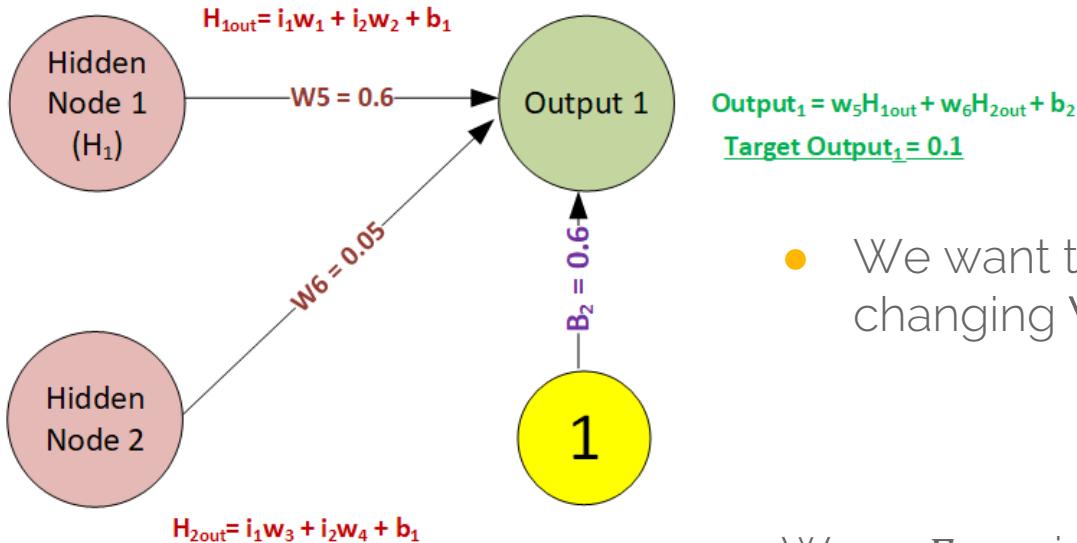


Slight Change in Our MSE Loss Function

- Let's define the MSE as
 - = Error = $\frac{1}{2}(target - output)^2$
- The $\frac{1}{2}$ is included to cancel the exponent when differentiated (looks better)
- Output 1 Loss = **0.206271039**
- Output 2 Loss = **0.009502145**



Exploring W_5



- We want to know how much changing W_5 changes to total Error.

$$\frac{dE_{total}}{dw_5}$$

Where E_{total} is the sum of the Error from Output 1 and Output 2



Using the Chain Rule to Calculate W_5

- 1 2 3
 $\frac{dE_{total}}{dw_5} = \frac{dE_{total}}{dout_1} \times \frac{dout_1}{dNetOutput_1} \times \frac{dNetOutput_1}{dw_5}$
- $E_{total} = \frac{1}{2}(target_{o1} - out_1)^2 + \frac{1}{2}(target_{o2} - out_2)^2$

Therefore differentiating E_{total} with respect to out_1 gives us

$$\frac{dE_{total}}{dout_1} = 2 \times \frac{1}{2}(target_{o1} - out_1)^{2-1} \times (-1) + 0$$

$$\frac{dE_{total}}{dout_1} = out_1 - target_{o1}$$

1 $\frac{dE_{total}}{dout_1} = (0.742294385 - 0.1) = 0.642294385$



Let's get $\frac{dOut_1}{dNetOutput_1}$

- This is how output 1 changes with respect to its input
- $dOut_1 = \frac{1}{1+e^{-neto_1}}$
- $\frac{dOut_1}{dneto_1} = out_{o1}(1 - out_{o1}) = 0.742294385(1 - 0.742294385)$
- 2 • $\frac{dOut_1}{dneto_1} = -0.191293$



Let's get $\frac{dnetO_1}{dw_5}$

- $netO_1 = (w_5 \times outH_1) + (w_6 \times outH_2) + (b_2 \times 1)$
- $\frac{dnetO_1}{dw_5} = 1 \times outH_1 \times w_5^{(1-1)} + 0 + 0$
- 3 • $\frac{dnetO_1}{dw_5} = outH_1 = 0.704225234$



We now have all the pieces to get $\frac{dE_{total}}{dw_5}$

- $\frac{dE_{total}}{dw_5} = \frac{dE_{total}}{dOut_1} \times \frac{dOut_1}{dNetOutput_1} \times \frac{dNetOutput_1}{dw_5}$
- $\frac{dE_{total}}{dw_5} = 0.642294385 \times 0.191293431 \times 0.704225234$
- $\frac{dE_{total}}{dw_5} = 0.086526$



So what's the new weight for W_5 ?

- $New w_5 = w_5 - \eta \times \frac{dE_{total}}{dw_5}$
- $New w_5 = 0.6 - (0.5 \times 0.086526) = 0.556737$

Learning Rate

- Notice we introduced a new parameter ' η ' and gave it a value of 0.5
- Look carefully at the first formula. The learning rate simply controls how a big a magnitude jump we take in the direction of $\frac{dE_{total}}{dw_5}$
- Learning rates are always positive and range from $\gamma > 0 \leq 1$
- A large learning rate will allow faster training, but can overshoot the global minimum (getting trapped in a local minima instead). A small learning will take longer to train but will more reliably find the global minimum.



Check your answers

- $\text{new } w_6 = 0.400981$
- $\text{new } w_7 = 0.508845$
- $\text{new } w_8 = 0.558799$



You've just used Backpropagation to Calculate the new W₅

- You can now calculate the new updates for W₆, W₇ and W₈ similarly.
- W₁, W₂, W₃ and W₄ are similar:

$$\frac{dE_{total}}{dw_1} = \frac{dE_{total}}{dOut_{H1}} \times \frac{dOut_{H1}}{dNetOutput_{H1}} \times \frac{dNetOutput_{H1}}{dw_1}$$

- I'll leave this as an exercise for you.

Regularization, Overfitting, Generalization and Test Datasets

How do we know our model is good?



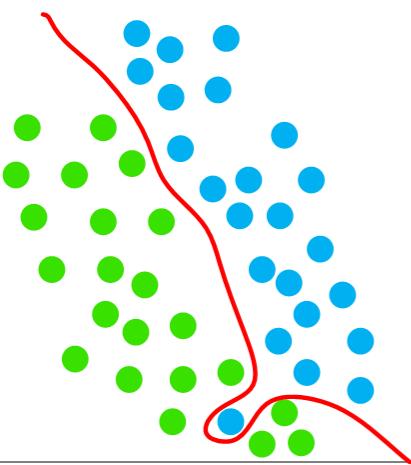
What Makes a Good Model?

- A good model is accurate
- Generalizes well
- Does not overfit

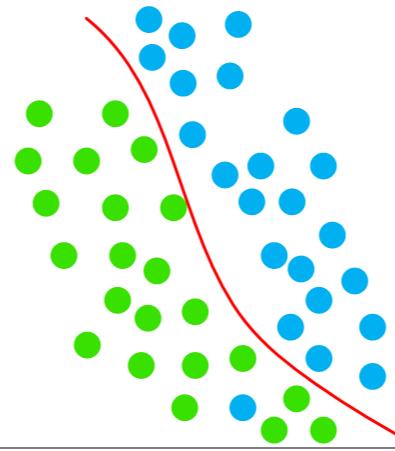


What Makes a Good Model?

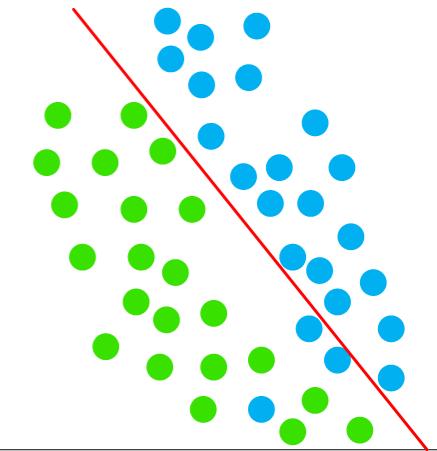
● Model A



● Model B



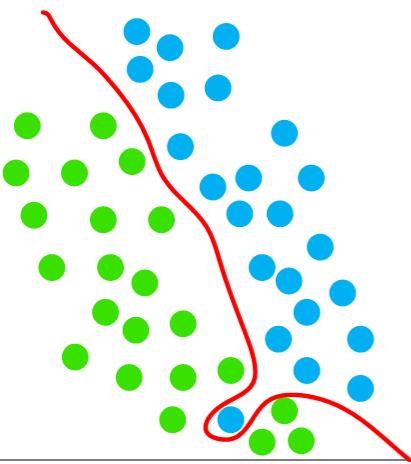
● Model C



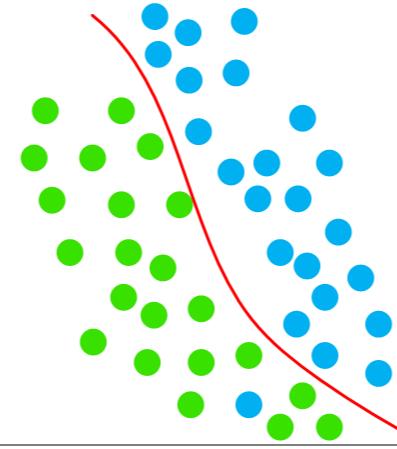


What Makes a Good Model?

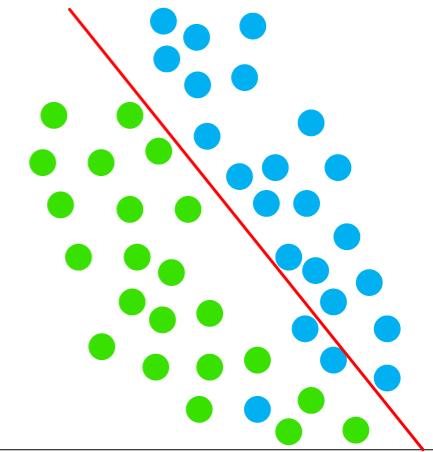
- Overfitting



- Ideal or Balanced



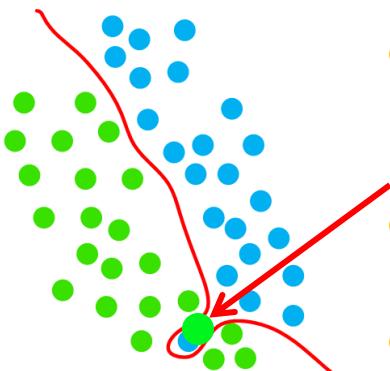
- Underfitting





Overfitting

- Overfitting leads to *poor models* and is one of the most common problems faced developing in AI/Machine Learning/Neural Nets.
- Overfitting occurs when our Model fits near perfectly to our training data, as we saw in the previous slide with Model A. However, fitting too closely to training data isn't always a good thing.



- What happens if we try to classify a brand new point that occurs at the position shown on the left? (whose true color is green)
- It will be misclassified because our model has overfitted the test data
- Models don't need to be complex to be good



How do we know if we've Overfit?

Test on your model on.....Test Data!

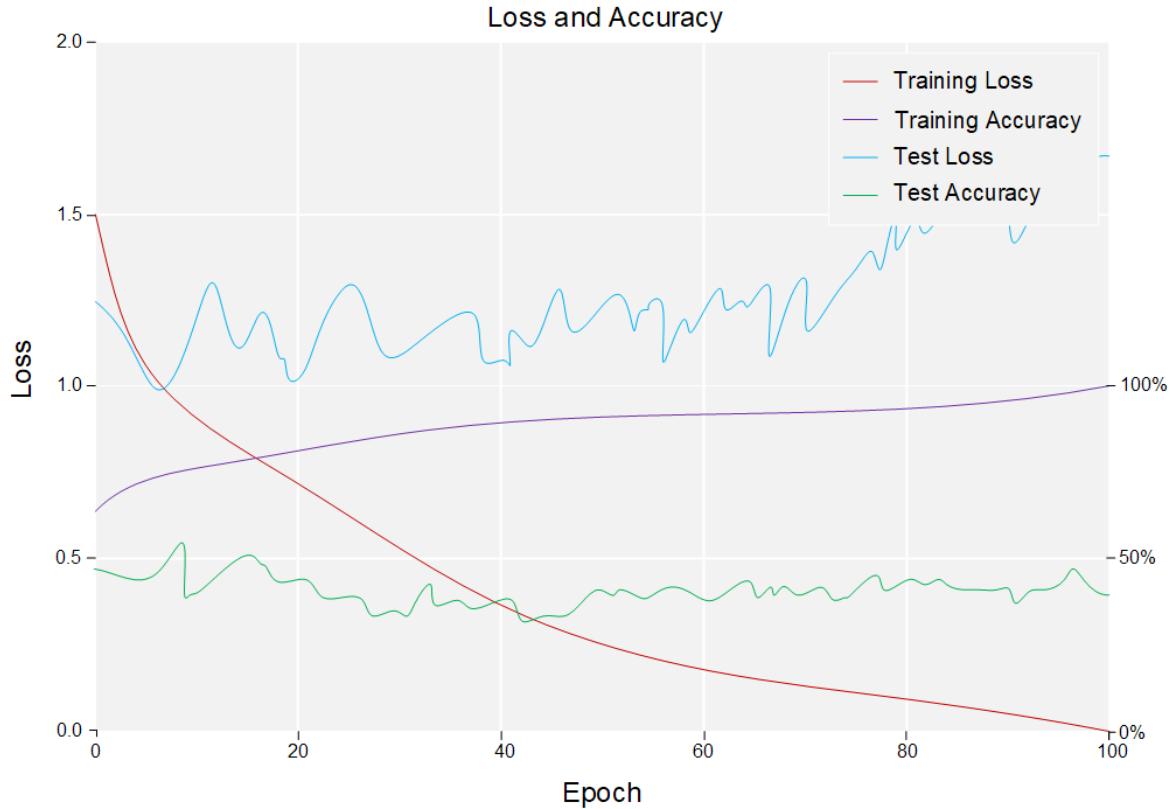
- In all Machine Learning it is extremely important we hold back a portion of our data (10-30%) as pure untouched test data.

Training Data	Test Data
---------------	-----------

- Untouched meaning that this data is NEVER seen by the training algorithm. It is used purely to test the performance of our model to assess its accuracy in classifying new never before seen data.
- Many times when Overfitting we can achieve high accuracy 95%+ on our test data, but then get abysmal (~70%) results on the test data. That is a perfect example of Overfitting.



Overfitting Illustrated Graphically

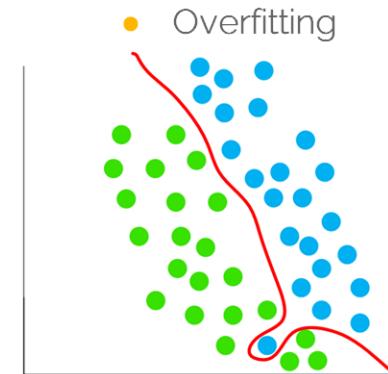


- Examine our Training Loss and Accuracy. They're both heading the right directions!
- But, what's happening to the loss and accuracy on our Test data?
- This is classic example of Overfitting to our training data



How do we avoid overfitting?

- Overfitting is a consequence of our weights. Our weights have been tuned to fit our Training Data well but due to this 'over tuning' it performs poorly on unseen Test Data.
- We know our weights are a decent model, just too sensitively tuned. If only there were a way to fix this?





How do we avoid overfitting?

- We can use less weights to get smaller/less deep Neural Networks
 - Deeper models can sometimes find features or interpret noise to be important in data, due to their abilities to memorize more features (called memorization capacity)



How do we avoid overfitting?

- We can use less weights to get smaller/less deep Neural Networks
 - Deeper models can sometimes find features or interpret noise to be important in data, due to their abilities to memorize more features (called memorization capacity)

Or Regularization!

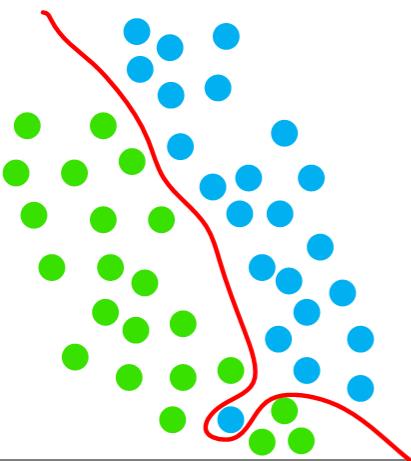
- It is better practice to regularize than reduce our model complexity.



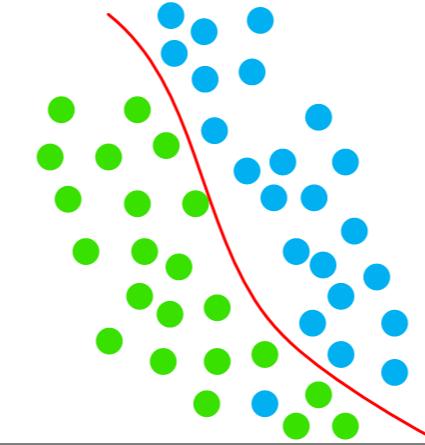
What is Regularization?

- It is a method of making our model more general to our dataset.

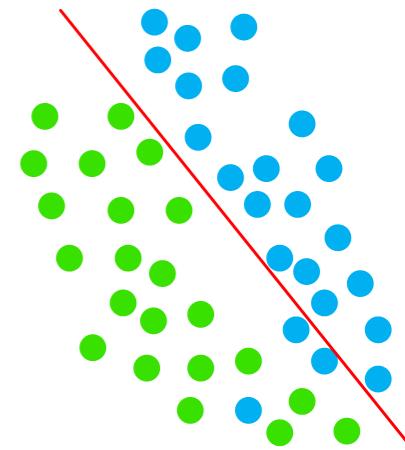
- Overfitting



- Ideal or Balanced



- Underfitting





Types of Regularization

- L1 & L2 Regularization
- Cross Validation
- Early Stopping
- Drop Out
- Dataset Augmentation



L1 And L2 Regularization

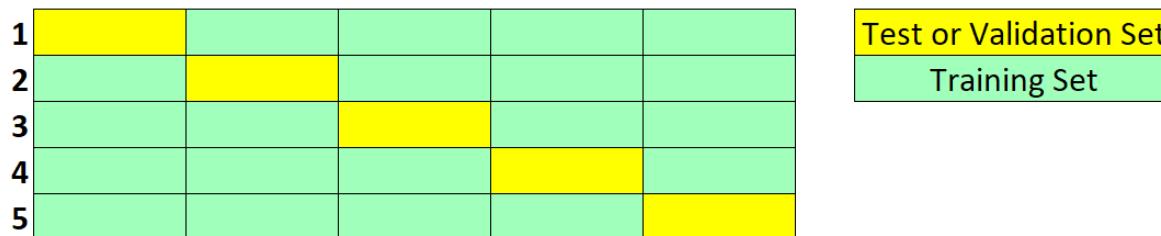
- L1 & L2 regularization are techniques we use to penalize large weights. Large weights or gradients manifest as abrupt changes in our model's decision boundary. By penalizing, we're really making them smaller.
- L2 also known as Ridge Regression
 - $Error = \frac{1}{2}(target_{01} - out_1)^2 + \frac{\lambda}{2} \sum w_i^2$
- L1 also known as Lasso Regression
 - $Error = \frac{1}{2}(target_{01} - out_1)^2 + \frac{\lambda}{2} \sum |w_i|$
- λ controls the degree of penalty we apply.
- Via Backpropagation, the penalty on the weights is applied to the weight updates
- The difference between them is that L1 brings the weights of the unimportant features to 0, thus acting as feature selection algorithm (known as sparse models or models with reduced parameters.)



Cross Validation

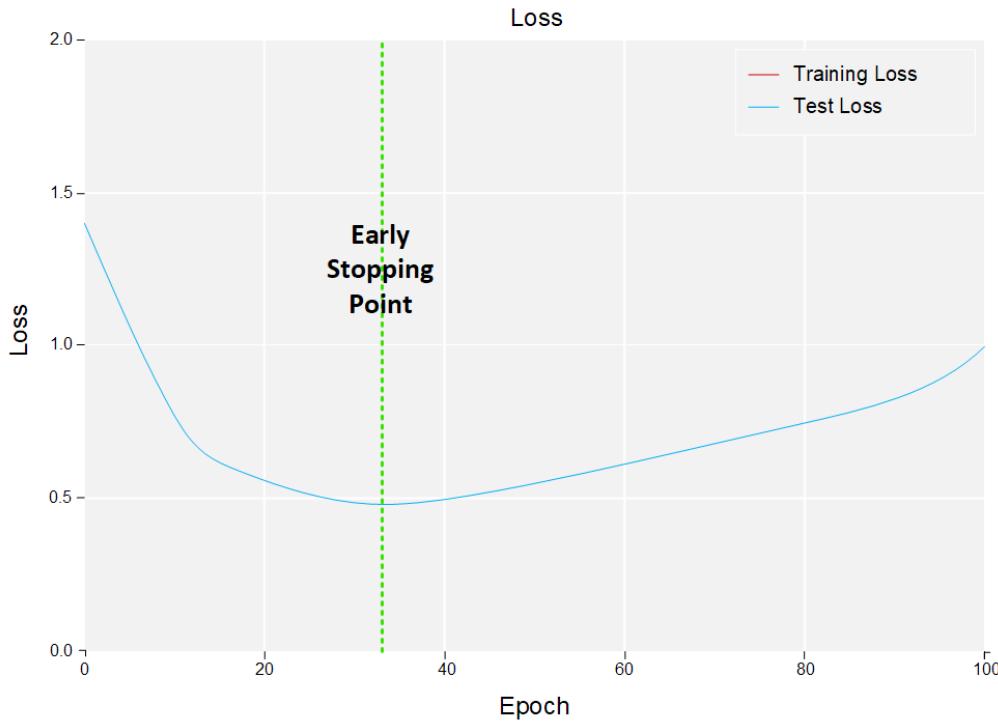
- Cross validation or also known as k-fold cross validation is a method of training where we split our dataset into k folds instead of a typical training and test split.
- For example, let's say we're using 5 folds. We train on 4, and use the 5th final fold as our test. We then train on the other 4 folds, and test on another.
- We then use the average weights across coming out of each cycle.
- Cross Validation reduces overfitting but slows the training process

k-folds (5 shown)





Early Stopping



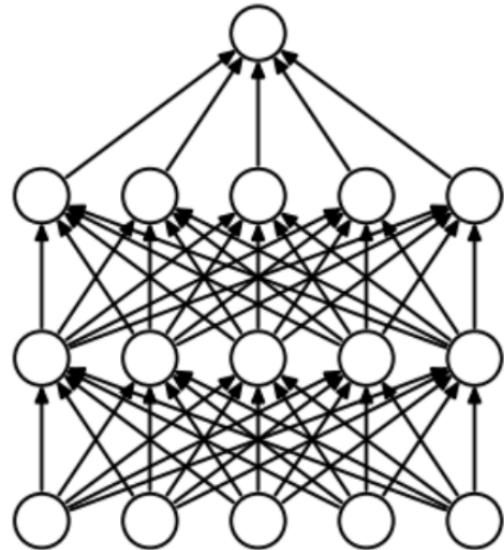


Dropout

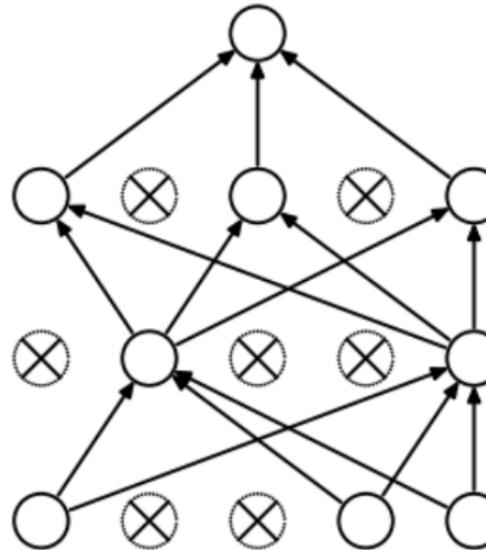
- Dropout refers to dropping nodes (both hidden and visible) in a neural network with the aim of reducing overfitting.
- In training certain parts of the neural network are ignored during some forward and backward propagations.
- Dropout is an approach to regularization in neural networks which helps reducing interdependent learning amongst the neurons. Thus the NN learns more robust or meaningful features.
- In Dropout we set a parameter ' P ' that sets the probability of which nodes are kept or $(1-p)$ for those that are dropped.
- Dropout almost doubles the time to converge in training



Dropout Illustrated



(a) Standard Neural Net



(b) After applying dropout.



Data Augmentation

- Data Augmentation is one of the easiest ways to improve our models.
- It's simply taking our input dataset and making slight variations to it in order to improve the amount of data we have for training. Examples below.
- This allows us to build more robust models that don't overfit.



Epochs, Iterations and Batch Sizes

Understanding some Neural Network Training Terminology



Epochs

- You may have seen or heard me mention Epochs in the training process, so what exactly is an Epoch?
 - An Epoch occurs when the full set of our training data is passed/forward propagated and then backpropagated through our neural network.
 - After the first Epoch, we will have a decent set of weights, however, by feeding our training data again and again into our Neural Network, we can further improve the weights. This is why we train for several iterations/epochs (50+ usually)



Batches

- Unless we had huge volumes of RAM, we can't simply pass all our training data to our Neural Network in training. We need to split the data up into segments or **Batches**.
- Batch Size is the number of training samples we use in a single batch.
- Example, say we had 1000 samples of data, and specified a batch size of 100. In training, we'd take 100 samples of that data and use it in the forward/backward pass then update our weights. If the batch size is 1, we're simply doing Stochastic Gradient Descent.



Iterations

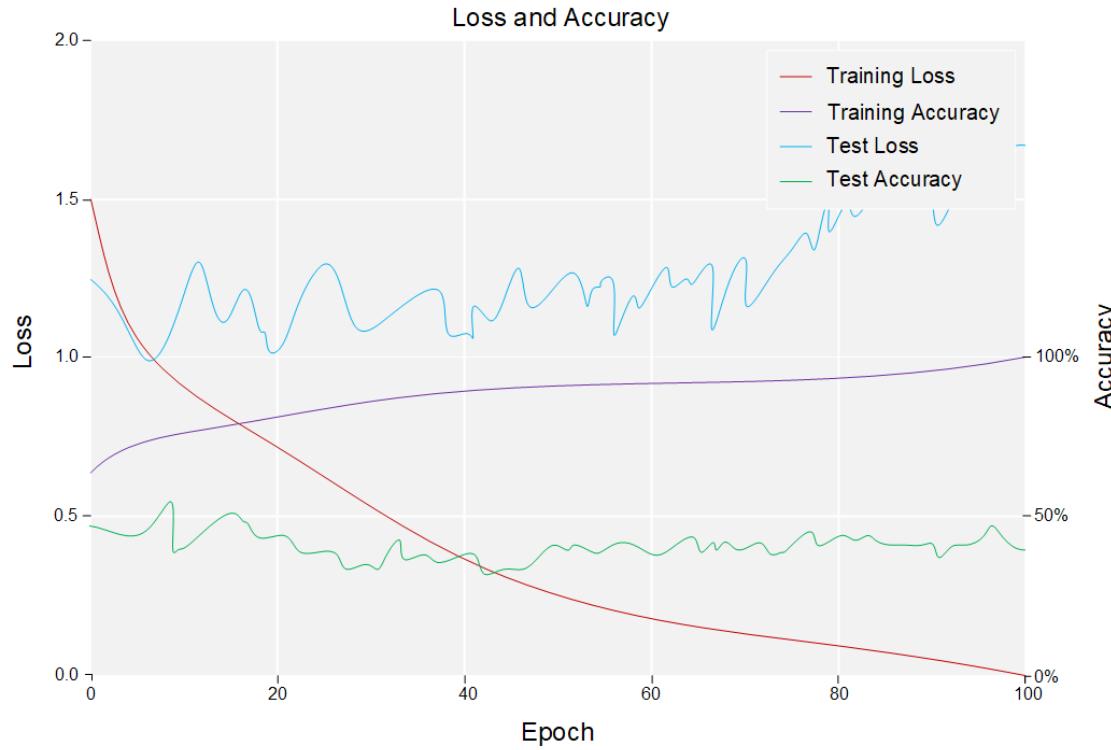
- Many confuse iterations and Epochs (I was one of them)
- However, the difference is quite simple, Iterations are the number of batches we need to complete one Epoch.
- In our previous example, we had 1000 items in our dataset, and set a batch size of 100. Therefore, we'll need 10 iterations (100×10) to complete one Epoch.

Measuring Performance

How we measure the performance of our Neural Network



Loss and Accuracy





Loss and Accuracy

- It is important to realize while Loss and Accuracy represent different things, they are essentially measuring the same thing. The performance of our NN on our training Data.
- Accuracy is simply a measure of how much of our training data did our model classify correctly
 - $$Accuracy = \frac{\text{Correct Classifications}}{\text{Total Number of Classifications}}$$
- Loss values go up when classifications are incorrect i.e. different to the expected Target values. As such, Loss and Accuracy on the Training dataset WILL correlate.



Is Accuracy the only way to assess a model's performance?

- While very important, accuracy alone doesn't tell us the whole story.
- Imagine we're using a NN to predict whether a person has a life threatening disease based on a blood test.
- There are now 4 possible scenarios.
 1. TRUE POSITIVE
 - Test Predicts Positive for the disease and the person **has** the disease
 2. TRUE NEGATIVE
 - Test Predicts Negative for the disease and the person **does NOT** have the disease
 3. FALSE POSITIVE
 - Test Predicts Positive for the disease but the person **does NOT** have the disease
 4. FALSE NEGATIVE
 - Test Predicts Negative for the disease but the person actually **has** the disease



For a 2 or Binary Class Classification Problem

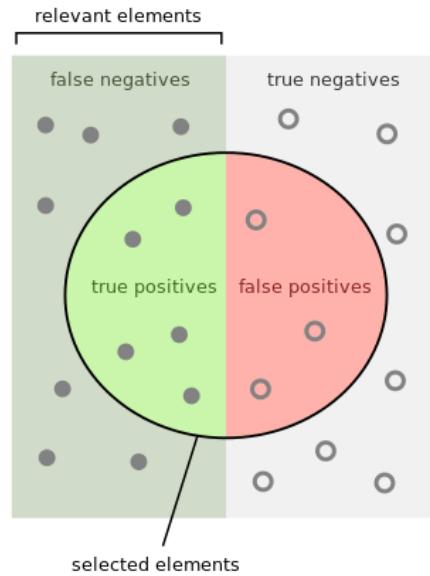
- Recall – How much of the positive classes did we get correct
 - $$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$
- Precision – Out of all the samples how much did the classifier get right
 - $$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$
- F-Score – Is a metric that attempts to measure both Recall & Precision
 - $$F - Score = \frac{2 \times Recall \times Precision}{Precision + Recall}$$



An Example

Let's say we've built a classifier to identify gender (male vs female) in an image where there are:

- 10 Male Faces & 5 Female Faces
- Our classifier identifies 6 male faces
- Out of the 6 male faces - 4 were male and 2 female.
- Our Precision is $4 / 6$ or 0.66 or 66%
- Our Recall is $4 / (4 + 6)$ (**6 male faces were missed**) or 0.4 or 40%
- Our F-Score is
$$\frac{2 \times \text{Recall} \times \text{Precision}}{\text{Precision} + \text{Recall}} = \frac{2 \times 0.4 \times 0.66}{0.4 + 0.66} = \frac{0.528}{1.06} = 0.498$$



How many selected items are relevant?

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

https://en.wikipedia.org/wiki/Precision_and_recall

Review and Best Practices

Review on the entire training process and some general guidelines to designing your own Neural Nets

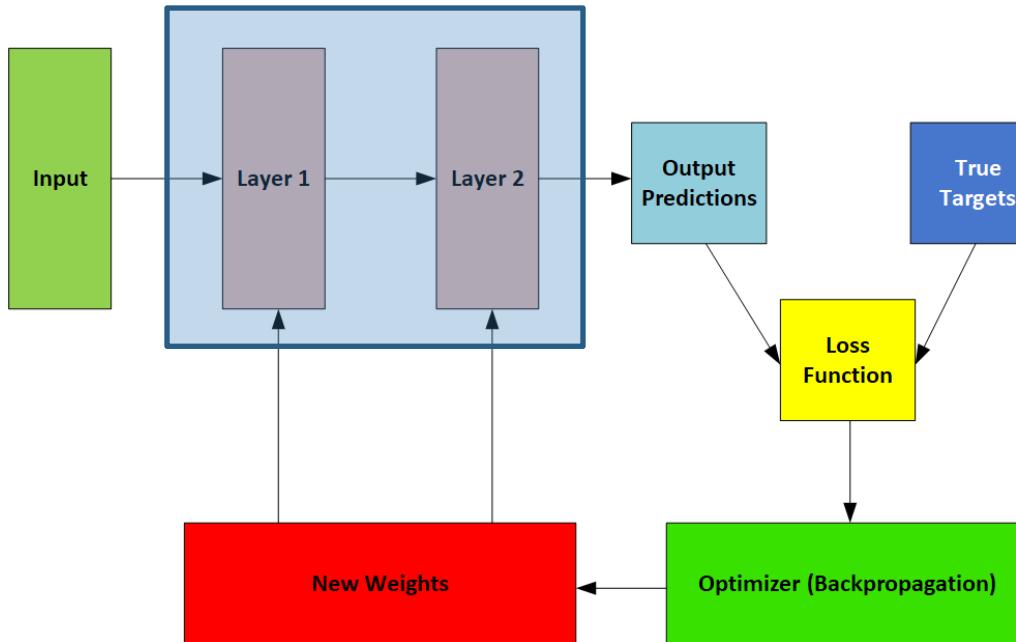


Training step by step

1. Initialize some random values for our weights and bias
2. Input a single sample of our data or batch of samples
3. Compare our output with the actual value target values.
4. Use our loss function to put a value to our loss.
5. Use Backpropagation to perform mini batch gradient descent to update our weights
6. Keep doing this for each batch of data in our dataset (iteration) until we complete an Epoch
7. Complete several Epochs and stop training when the Accuracy on our test dataset



Training Process Visualized





Best Practices

- Activation Functions - Use ReLU or Leaky ReLU
- Loss Function - MSE
- Regularization
 - Use L2 and start small and increase accordingly (0.01, 0.02, 0.05, 0.1.....)
 - Use Dropout and set P between 0.2 to 0.5
- Learning Rate – 0.001
- Number of Hidden Layers – As deep as your machine's performance will allow
- Number of Epochs – Try 10-100 (ideally at least 50)

A/B Testing A Fun Theoretical Example



A/B Testing Introduction

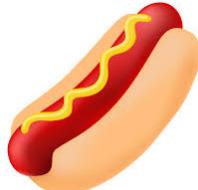
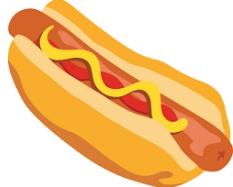
- A/B Testing has become almost a buzz word given how much it's thrown around in marketing circles and amongst web & UX/UI interfaces.
- But what exactly is it? Is it just comparing the results of A versus B? That doesn't sound too hard does it?
- It's not, but, there're a lot of things to consider when designing and analyzing your A/B Test
- Let's conduct a simple real world example to better understand



Our Real Life A/B Test Example

- So now, let's test our understanding with looking at our real world example

Hot Dogs!



- We want to test which hot meat sausage is better between two meats.





Our Hot Dogs

- Hot Dog Sausage (**A**) is what we've been using for years and it's worked fairly well.
- Along comes another brand of sausage meat (**B**) that claims to test better.





Our Experiment Design

- So how do we go about testing which of these two hot dogs is best?
- Someone said A/B Testing? Correct! So how do we setup our experiment?

Step 1 – Define our Hypothesis

Step 2 – How do we evaluate our experiment

Step 3 – How much of the effect do we want?

Step 4 – How many people or test subjects do we need?





Defining our Hypothesis

Null Hypothesis

- There is no difference between the two hot dog sausage brands (A and B)

Alternative Hypothesis

- The new hot dog sausage brand (B) is better than A.

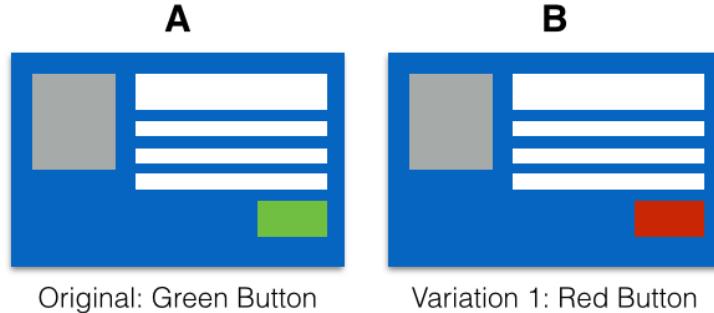




Our Evaluation Metric

We need a way to define what our **test metric** is, in most cases this is quite simple. Did button A result in more clicks than button B, with click through rate being our metric. However, in our hotdog experiment what should our metric be?

We can ask the tasters if they'd like **a second hotdog from the same batch?** This is an effective **metric**. It removes some subjectivity of rating taste. Instead, a simple yes or no would suffice and it allows us to compare figures easily.





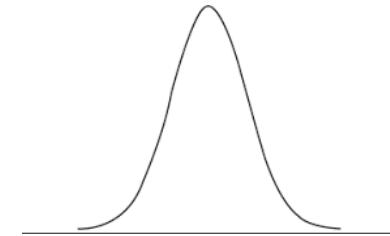
Desired Effect

- Now, in our example the two hot dog sausages cost roughly the same so the cost impact is **negligible**.
- An example of understanding how desired effect comes into play, let's assume we can use Kobe beef sausages. Those are perhaps \$50 a sausage (if it even exists).
- We need to know if the cost investment of using B over A is worth the investment of B. B can perform 5% better than A but cost double. In a situation like that we need to work out the economics of whether our added benefit to determine if it's a feasible investment.



Experiment Size

- This is where a lot A/B tests go wrong.
- Imagine we run an experiment on 10 people with our hotdog A only. In this example, we ask them the same question, "Would like another hotdog from the same batch"



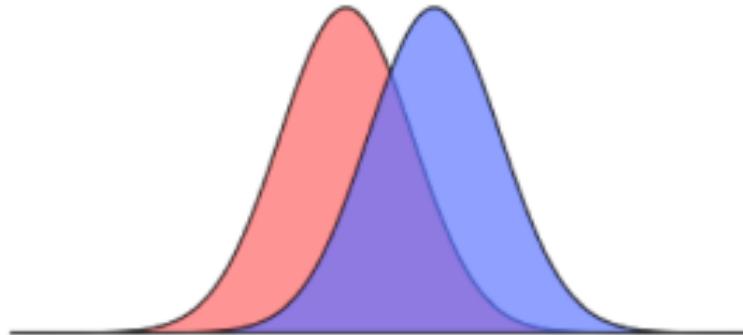
Group	Yes to another hotdog
1	4
2	5
3	6
4	5

- Every time we run an experiment we can get different results. These results typically form a normal distribution curve.
- When testing for two different cases (A & B) how do we know if our results are real and not due to random variation?



Experiment Size – Sample Size

- This is why we need to choose a sufficiently large sample size that allows us to know whether the difference in results is Statistically Significant





Statistical Power

- When choosing sample size, we need to decide what the Statistical Power of our test will be.
- Statistical Power is the **probability of correctly rejecting the Null Hypothesis**
- Statistical Power is often referred to **1-Beta** as it's inversely proportional to making **Type II Error** (Beta Errors).
- Type II error is the probability of failing to reject the Null Hypothesis when you should have.
- We typically use a **statistical power of 80%** and don't worry, I'll explain what that 80% means shortly.
- Statistical power represents the **probability that you'll get a false negative**. A power of 0.80 means that there is an 80% chance that if there was an effect, we would detect it (or a 20% chance that we'd miss the effect)



Statistical Significance Level

- Statistical Significance is how likely it is that the difference between your experiment's control version and test version **isn't due to error or random chance.**
- Basically it means, if we run an experiment test with a 95% significance level - you can be 95% confident that the differences are real and not attributed to chance.



Baseline Metric and Effect Size

- Before moving ahead with testing variations of hotdogs, we need to establish our **baseline metric**.
- In our experiment, our **evaluation metric** was the number of people who wanted another hotdog after having one.
- But by what amount of change are looking to get from our new hotdogs? In our example let's look at getting a 25% increase.



Calculating our Sample Size

```
▶ from scipy.stats import norm, zscore

def sample_power_probttest(p1, p2, power=0.8, sig=0.05):
    #two-sided t test
    z = norm.isf([sig/2])
    zp = -1 * norm.isf([power])
    d = (p1-p2)
    s = 2*((p1+p2) / 2)*(1-((p1+p2) / 2))
    n = s * ((zp + z)**2) / (d**2)
    return int(round(n[0]))

sample_power_probttest(p1=0.5, p2=0.75, power=0.8, sig=0.05)
```

➡ 59

- Online Calculator - <http://www.evanmiller.org/ab-testing/sample-size.html>

P1 (baseline) = 50%
P2 (p1+effect size) = 75%
Statistical Power = 80%
Significance Level = 5%

Sample Size N = 59



Experiment Length

- In our example this doesn't apply, however one should be very careful with experiment lengths.
- In the business world there are many other factors that could mess with our results such as seasonality, trends etc.
- Example don't run a test during Christmas week or right before Mothers Day and expect your results to be relevant.
- However, one advantage of doing it during a peak time would be to get larger N over a shorter space of time.



Back to our Hotdog Experiment

- So we ran our test on two groups of 60 persons (chosen at random)
- Null Hypothesis – Our new hotdog B does not make a difference.
- $H_0: \hat{d} = 0$ (*d 'hat' is the difference between the two groups*)
- $N_{control} = N_{experiment} = 60$

Results

- $X_{control} = 33$
- $X_{experiment} = 42$
- $\hat{d} = \hat{p}_{experiment} - \hat{p}_{control} = \frac{42}{60} - \frac{33}{60} = 0.15$



Discussing our Results

- $\hat{d} = \hat{p}_{experiment} - \hat{p}_{control} = \frac{42}{60} - \frac{33}{60} = 0.15$
- So we have a **0.15** improvement with our Hotdog B, but was this by random chance or is this statistically significant?
- In order to test this, we need to calculate our **confidence interval** for the difference of the results of the two groups.
- This **interval** tells us the range of values the difference between the two groups can have.



Calculating our Confidence Intervals

- We firstly need to calculate the pooled probability, which is really just the combined probability of the two samples.
- $\hat{p}_{pooled} = \frac{X_{control} + X_{experiment}}{N_{control} + N_{experiment}} = \frac{33 + 42}{120} = \frac{75}{120} = 0.625$
- The second and last step to get the Confidence Interval is getting the **Standard Error**. It estimates of how much variation the obtained results will have. This means how widespread the values in the distribution of the sample will be. We'll calculate the **Pooled Standard Error** which combines the standard error of both samples.
- $SE_{pooled} = \sqrt{\hat{p}_{pooled} \times (1 - \hat{p}_{pooled}) \times \left(\frac{1}{N_{control} + N_{experiment}}\right)}$
- $SE_{pooled} = \sqrt{0.625 \times (1 - 0.625) \times \left(\frac{1}{120}\right)} = 0.044$

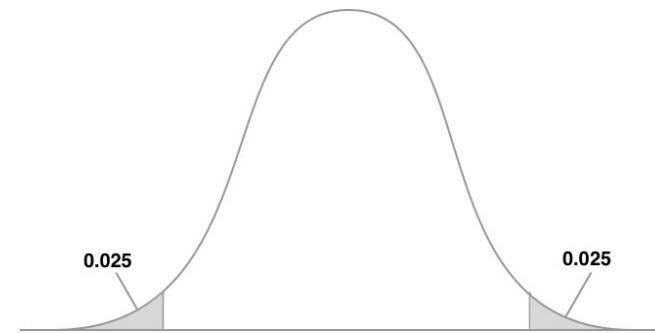


Final Analysis

So now we have everything to get out confidence intervals and enough information to determine if we should reject our Null Hypothesis.

- $H_0: \hat{d} = 0$
- $H_1: \hat{d} > 1.96 \times StandardError_{pooled}$

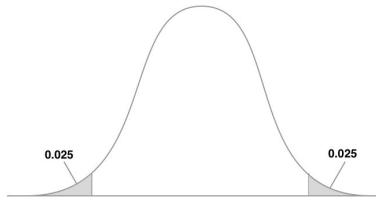
Remember we chose to use a 5% significance level? What we do now is split the 5% into the two ends of our normal distribution curve (we assume normally distributed data).





Final Analysis – Obtaining our Z-Score

<i>z</i>	.00	.01	.02	.03	.04	.05	.06	.07
-3.4	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0003
-3.3	.0005	.0005	.0005	.0004	.0004	.0004	.0004	.0004
-3.2	.0007	.0007	.0006	.0006	.0006	.0006	.0006	.0005
-3.1	.0010	.0009	.0009	.0009	.0008	.0008	.0008	.0008
-3.0	.0013	.0013	.0013	.0012	.0012	.0011	.0011	.0011
-2.9	.0019	.0018	.0018	.0017	.0016	.0016	.0015	.0015
-2.8	.0026	.0025	.0024	.0023	.0023	.0022	.0021	.0021
-2.7	.0035	.0034	.0033	.0032	.0031	.0030	.0029	.0028
-2.6	.0047	.0045	.0044	.0043	.0041	.0040	.0039	.0038
-2.5	.0062	.0060	.0059	.0057	.0055	.0054	.0052	.0051
-2.4	.0082	.0080	.0078	.0075	.0073	.0071	.0069	.0068
-2.3	.0107	.0104	.0102	.0099	.0096	.0094	.0091	.0089
-2.2	.0139	.0136	.0132	.0129	.0125	.0122	.0119	.0116
-2.1	.0179	.0174	.0170	.0166	.0162	.0158	.0154	.0150
-2.0	.0228	.0222	.0217	.0212	.0207	.0202	.0197	.0192
-1.9	.0287	.0281	.0274	.0268	.0262	.0256	.0250	.0244
-1.8	.0359	.0351	.0344	.0336	.0329	.0322	.0314	.0307
-1.7	.0441	.0435	.0427	.0419	.0410	.0401	.0392	.0384



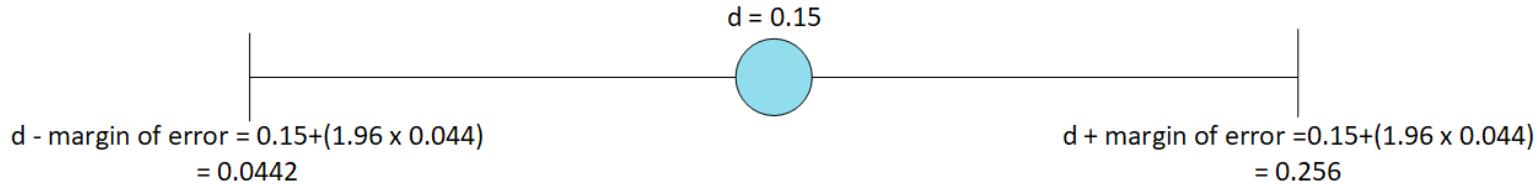
$$H_1: \hat{d} > 1.96 \times \text{StandardError}_{\text{pooled}}$$

- Notice the 1.96 in our Alternative Hypothesis?
- We used the Z-Table to get the value for 0.025
- Therefore, in order to reject the Null Hypothesis, we want to prove that the difference observed, is great than a certain interval around the value that refers to 5% of the results being due to chance.
- Hence why we use the z-score and the standard error.



So are our new Hotdogs better?

- $H_1: 0.15 > 1.96 \times 0.044 \equiv 0.15 > 0.08624$
- So yes, we have statistically proven with our AB test that the **new hotdogs (B)** is better than A.
- Before we say an emphatic yes, to using our new hotdogs, let's dig into the interpretation of our results, given our specified parameters.



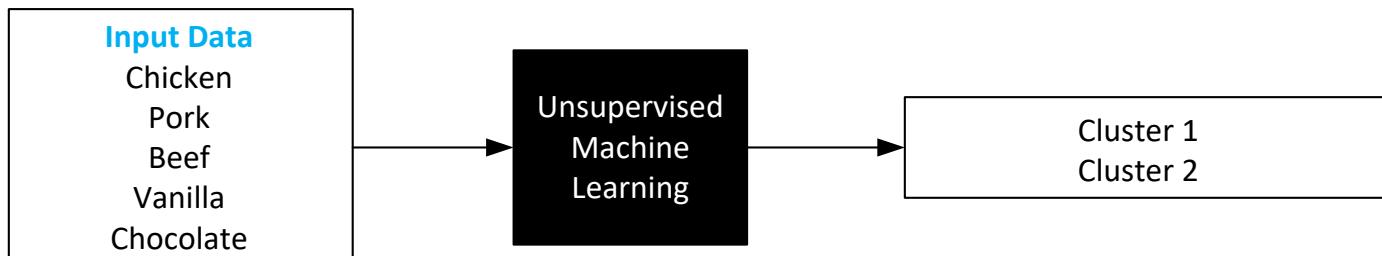
- Remember we chose a minimum effect size of 25%, as we can see given our Standard Error, the lower bound (left) is a situation where the minimum effect is not guaranteed as it's significantly less than 0.15
- We can therefore conclude that our new hotdogs have a strong possibility of resulting in 25% more sales

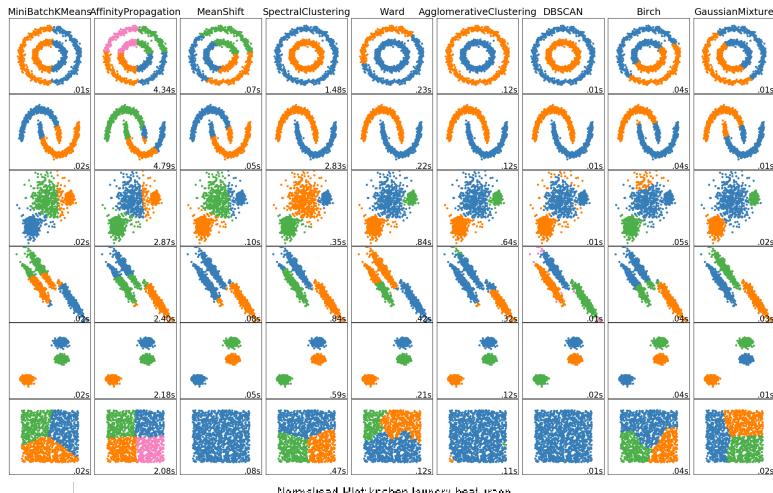
Clustering – Unsupervised Learning



Unsupervised Learning

- Unsupervised learning is concerned with finding interesting clusters of input data. It does so **without any help of data labeling**.
- It does this by creating interesting transformations of the input data
- It is very important in data analytics when trying to understand data
- Examples in the Business world:
 - Customer Segmentation







Goal of Clustering

- The goal of clustering is if given a set of data points, we can **classify each data point into a specific group or cluster.**
- We can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into naturally by using our clustering algorithms.
- *A cluster refers to a collection of data points aggregated together because of certain similarities.*
- There are several types of clustering methods which we'll now discuss.



Types of Clustering Algorithms

There are many types of clustering algorithms, some far more widely used than others, however we will discuss 5 main types:

- K-Means Clustering
- Agglomerative Hierarchical Clustering
- Mean-Shift Clustering
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
- Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)

K-Means Clustering



K-Means Clustering Algorithm

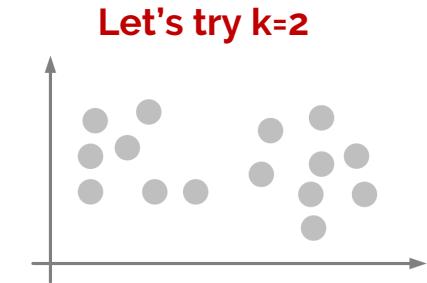
- K-Means is perhaps the most popular clustering algorithm in existence.
- It is extensively used in real world applications
- It's relatively simple, intuitive and great for beginners to conceptualize some machine learning concepts.
- Let's take a look at how it works!



K-Means Clustering – Step 1

Choose the number of clusters you wish to identify by choosing k

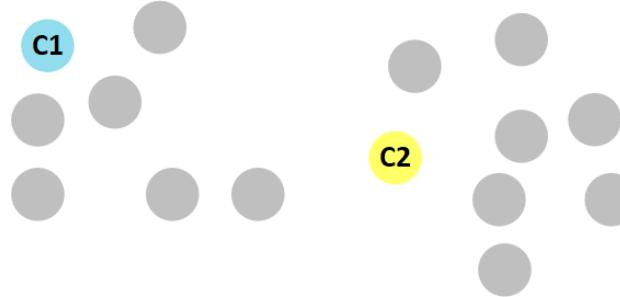
- Choosing K can be done either intuitively or via the **Silhouette Method or the Elbow method** (we'll discuss this next)
- You can choose K intuitively by:
 - Understanding the data domain** – e.g. if you're trying to cluster amongst newspaper articles you might have quite a few types, whereas if you're clustering on customer types, it might be better to start with a smaller k (less than 5)
 - Exploring it Visually** to see if we can spot natural clusters





K-Means Clustering – Step 2

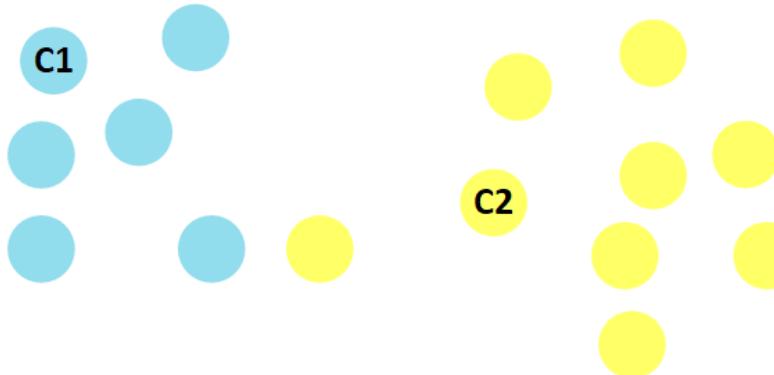
- Once we have k (which was equal to 2), we select k random points from our dataset and use these are centroids.
- Here we have c_1 (blue) and c_2 (yellow) that represent the centroid of these two clusters





K-Means Clustering – Step 3

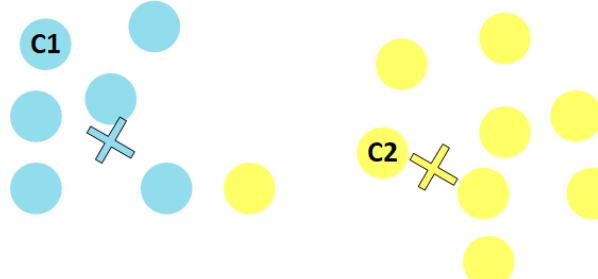
- Assign all the points to the closest cluster centroid
- So all points closest to C₁ get assigned to the blue cluster and all points closest to C₂ get assigned to the yellow cluster.





K-Means Clustering – Step 4

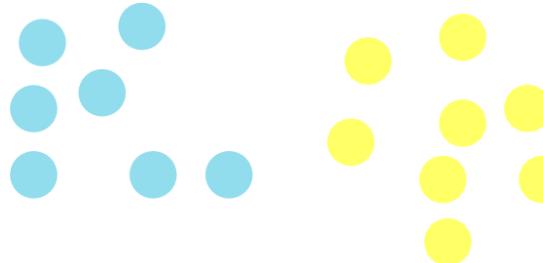
- We now compute the centroid of the newly formed clusters.
- The blue and yellow crosses represent the center of the newly formed clusters





K-Means Clustering – Step 5

- **Repeat Step 3 & 4** – We now use the new centroids (the yellow and blue crosses) as the cluster centers and then assign the closest points to each centroid's cluster.
- We keep repeating this step until the newly formed clusters stop changing, all points remain in the same cluster and/or the number of specified iterations is reached.





K-Means Clustering Algorithm Advantages

- Relatively simple to implement.
- Scales to large data sets.
- Guarantees convergence.
- Can warm-start the positions of centroids.
- Easily adapts to new examples.
- Generalizes to clusters of different shapes and sizes, such as elliptical clusters.



K-Means Clustering Algorithm Disadvantages

- We still need to choose K manually
- It is dependent on initial values
- Can run into problems when clustering varying sizes and density
- Sensitive to outliers
- Doesn't scale well with large number of dimensions (can be mitigated by using PCA)
- Only works for numeric values, as such categorical values will either have to be translated into some numerical meaning (e.g. high, medium, low can be mapped to 3,2,1) this can't always work for categories like types of fruit. Alternatively, we can use K-Medians, or K-Modes to alleviate this issue..

Choosing K – Elbow Method & Silhouette Analysis



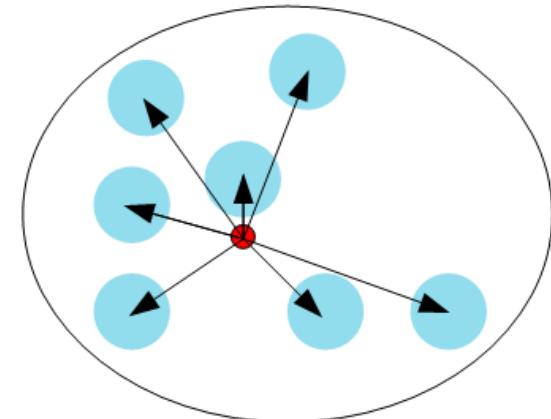
Choosing K

- As I stated in the previous section, choosing K is very important and should be guided by your knowledge of the dataset together with some guidance by some more scientific methods.
- We can get a rough idea on what k is good by using the two following methods:
 - Elbow Method
 - Silhouette Analysis



Elbow Method

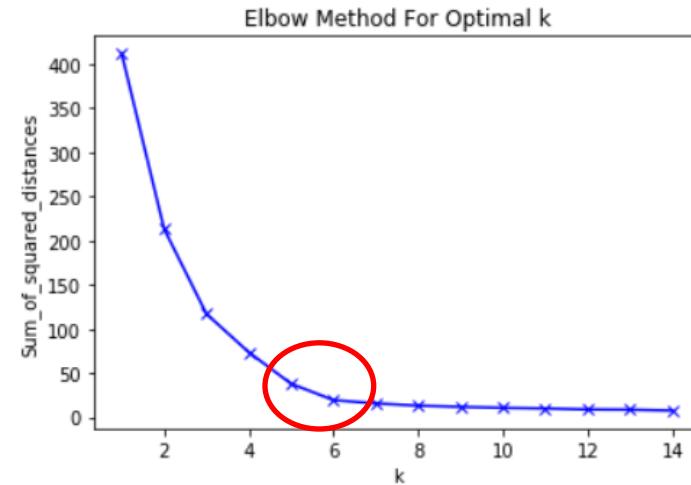
- The Elbow method works by running the clustering algorithm with a preset of cluster values (say 2 to 10).
- It then **computes the cost function** (or any other evaluation metric)
- Typically we use the **Inertia** (the within-cluster sum of distances from the centroid) and plot in a graph with the x-axis being the number of clusters and the y-axis being our evaluation metric. Small inertia is good.





Elbow Method

- The ideal number of clusters is obtained when the addition of a new cluster doesn't significantly increases the cost function.
- This can be visually illustrated by looking on the 'elbow' of the line.
- In our plot we can see the elbow lies at **k = 5 and k=6**.





Silhouette Analysis

- The Silhouette Analysis is a method of validating the consistency within clusters of data.
- The technique provides a succinct graphical representation of how well each object has been classified
- In this method a graph is plotted measuring how close the points in some cluster are to points of the other clusters.
- When the Silhouette coefficient is **near +1** this indicates that the points are far way from the other clusters
- When Silhouette coefficient is **near 0**, it indicates that the points are very close or intersecting other clusters.



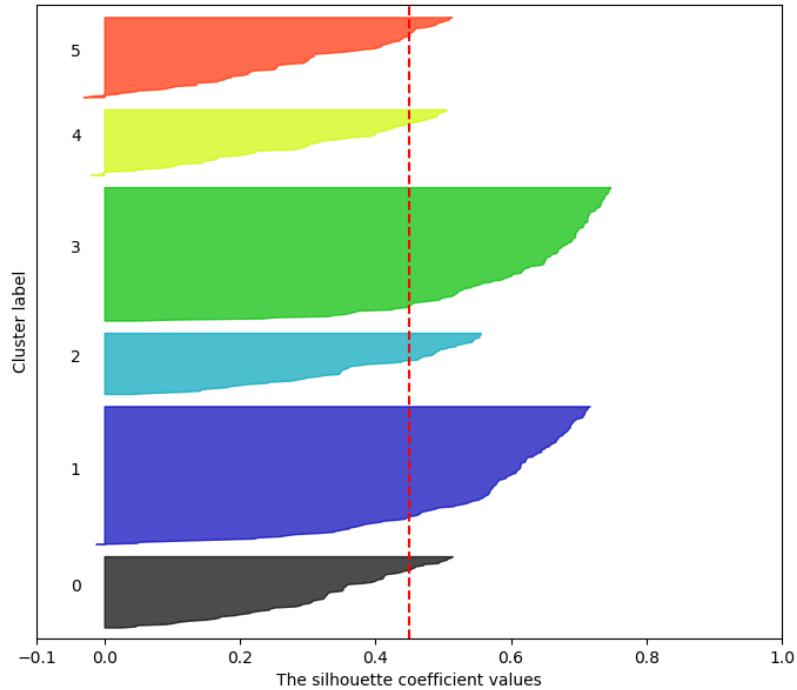
Silhouette Analysis

- The Silhouette Analysis is a method of validating the consistency within clusters of data.
- To calculate the Silhouette coefficient we need to define the mean distance of a point to all other points in its cluster (**a(i)**) and also define the mean distance to all other points in the closest cluster (**b(i)**). So, the Silhouette coefficient is:
- $$s(i) = \frac{(b(i)-a(i))}{\max(b(i),a(i))}$$

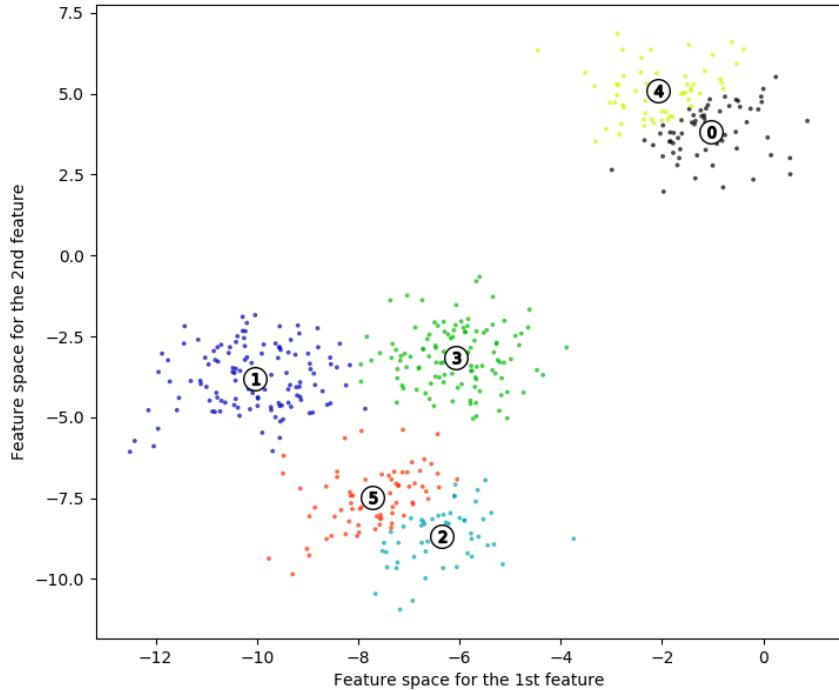


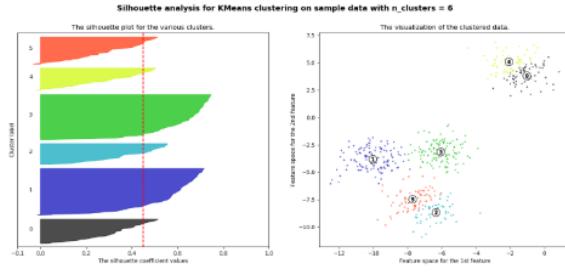
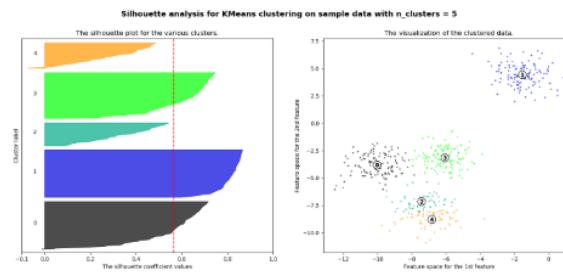
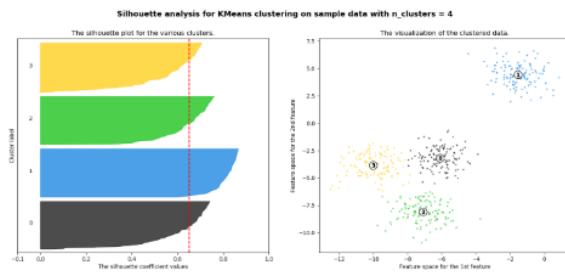
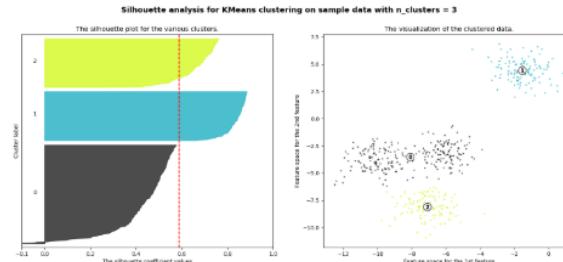
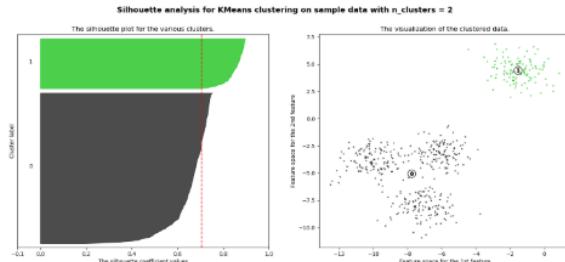
Silhouette analysis for KMeans clustering on sample data with $n_clusters = 6$

The silhouette plot for the various clusters.



The visualization of the clustered data.





n_clusters = 2 The average silhouette_score is : 0.704
n_clusters = 3 The average silhouette_score is : 0.588
n_clusters = 4 The average silhouette_score is : 0.650
n_clusters = 5 The average silhouette_score is : 0.563
n_clusters = 6 The average silhouette score is : 0.450

Agglomerative Hierarchical Clustering

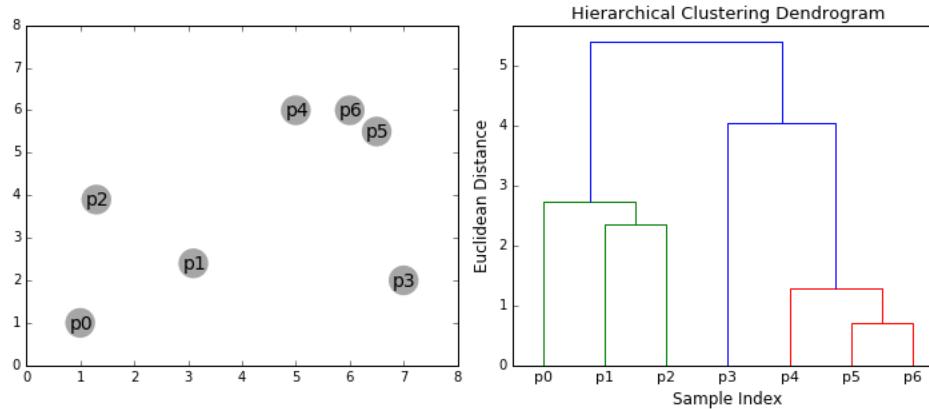


Agglomerative Hierarchical Clustering Introduction

- Hierarchical clustering algorithms is a bottom-up approach to clustering.
- Bottom-up algorithms treat each data point as a single cluster at the beginning and then successively merge or agglomerate pairs of clusters until all clusters have been merged into a single cluster that contains all data points.
- This hierarchy of clusters is represented as a tree or often referred to as a dendrogram.
- The root of the tree is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample.



Agglomerative Hierarchical Clustering Introduction

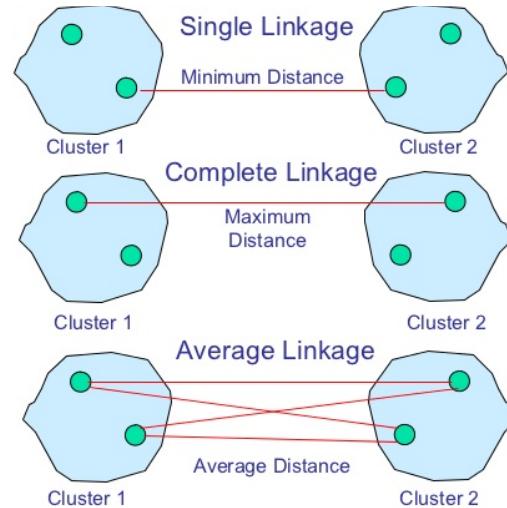


<https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68>



Agglomerative Hierarchical Clustering: Step 1

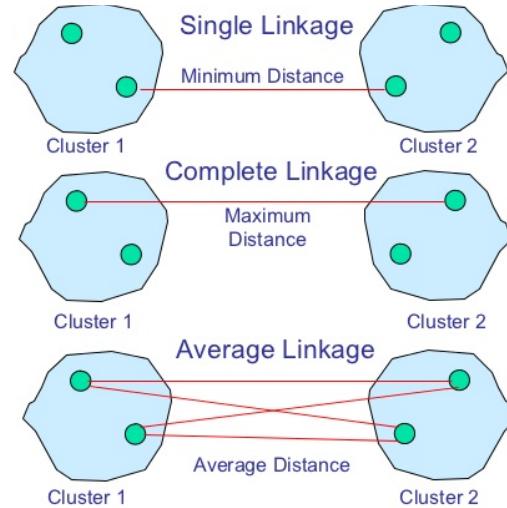
- We first begin by treating each data point as a single cluster
- For X data points we have X clusters.
- We use a distance metric (pre-selected) that measures the **distance between two clusters**.
- A commonly used metric is **average linkage** which defines the distance between two clusters to be the average distance between data points in the first cluster and data points in the second cluster.





Agglomerative Hierarchical Clustering: Step 2

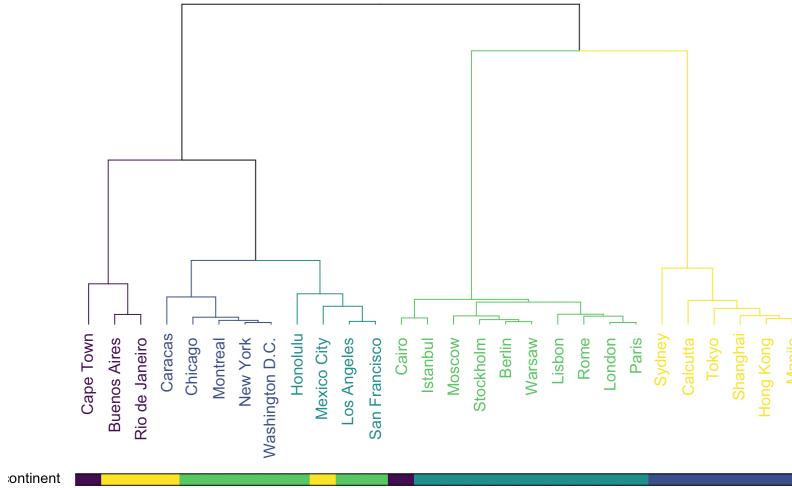
- For each iteration, we combine two clusters into one.
- The two clusters to be combined are selected as those with the smallest average linkage – this means that according to our average linkage distance metric, these two clusters have the smallest distance between each other and therefore are the most similar and should be combined.





Agglomerative Hierarchical Clustering: Step 3

- We repeat Step 2 until we reach the root of the tree, which is the final cluster that encapsulates all data points.





Advantages of Agglomerative Hierarchical Clustering

- We don't need to specify the number of clusters, k
- The algorithm is not too sensitive to the choice of distance metric with all of them working equally well in most cases.
- It works well in uncovering naturally hierarchical data

Disadvantages:

- It is quite slow and doesn't scale well

Mean-Shift Clustering



Mean-Shift Clustering

- Mean shift clustering is a sliding-window-based algorithm that attempts to find dense areas of data points.
- It is a centroid-based algorithm whereby its objective is to locate the center points of each cluster.
- It does this by **updating candidates for center points to be the mean of the points within the sliding-window**.
- These candidate windows are then filtered in a post-processing stage to eliminate near-duplicates, forming the final set of center points and their corresponding groups



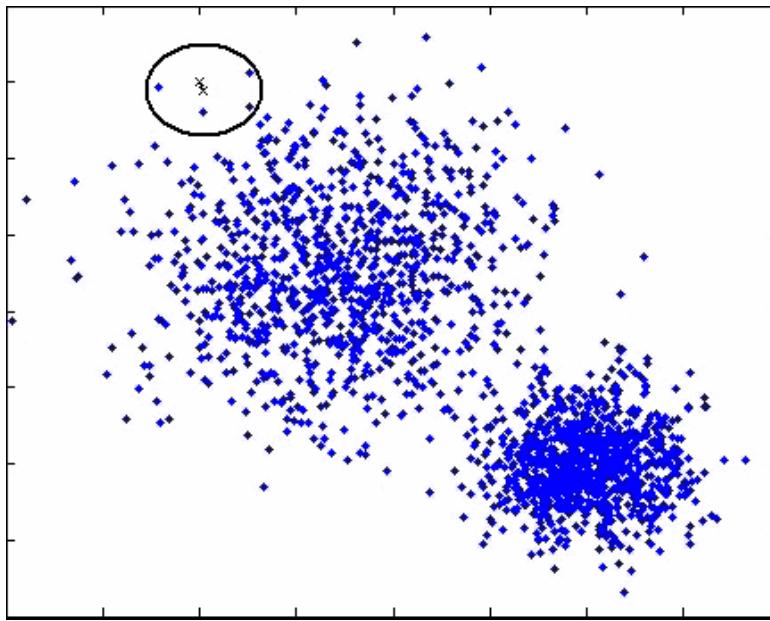
Mean-Shift Clustering Steps

Let's start with thinking about a set of 2-dim x,y points.

1. We first initialize a circular sliding window (radius r) starting at a randomly selected point.
2. Think of Mean Shift as a density finding algorithm, that keeps shifting the window to higher and higher densities of points until it converges.
3. The density within the sliding window is proportional to the number of points inside it.
4. The sliding window is shifted according to the mean until there is no direction at which a shift can accommodate more points inside the kernel.
5. We do this entire process above is done with a few sliding windows until all points lie within a window. When multiple sliding windows overlap the window containing the most points is preserved. The data points are then clustered according to the sliding window in which they reside.



Mean-Shift Clustering Demo



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)



Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

- DBSCAN is similar to Mean Shift, where it too is a density-based clustered algorithm
- However, it has a few notable advantages.



DBSCAN – Step 1 & 2

Step 1

- DBSCAN starts off with an arbitrary starting data point that has not been visited.
- The neighborhood of this point is extracted using a distance epsilon ε (All points which are within the ε distance are neighborhood points)

Step 2:

- If there are a sufficient number of points (according to minPoints) within this neighborhood then the clustering process starts and the current data point becomes the first point in the new cluster.
- If not, the point will be labeled as noise and the point is marked as “visited”.



DBSCAN – Step 3 & 4

Step 3

- For this first point in the new cluster, the points within its ϵ distance neighborhood also become part of the same cluster. This procedure of making all points in the ϵ neighborhood belong to the same cluster is then repeated for all of the new points that have been just added to the cluster group.

Step 4

- This process of steps 2 and 3 is repeated until all points in the cluster are determined, this occurs when all points have been visited and assigned a cluster.



DBSCAN – Step 5

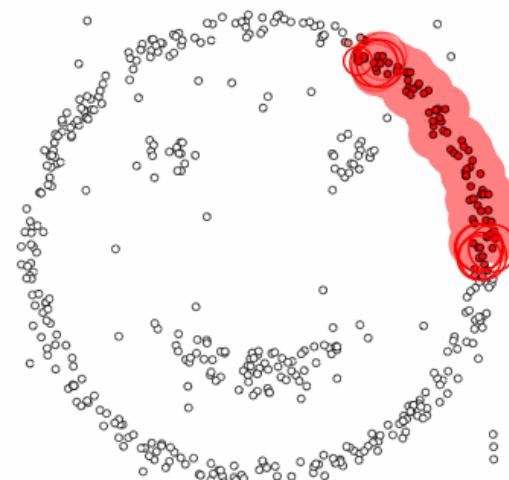
Step 5

- Once we're done with the current cluster, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.
- This process repeats until all points are marked as visited. Since at the end of this all points have been visited, each point will have been marked as either belonging to a cluster or being noise.



DBSCAN Demo

epsilon = 1.00
minPoints = 4



Restart



Pause



DBSCAN Advantages and Disadvantages

Advantages:

- No preset K number of clusters needs to be set
- It can identify outliers
- It can find erratically sized and shaped clusters well

Disadvantages:

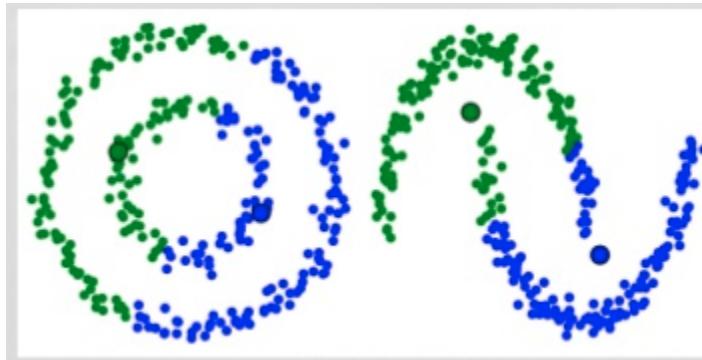
- It doesn't perform well when clusters are of varying densities (due to the setting of ε and minPoints for identifying the neighborhood points, as these will vary from cluster to cluster when they have different densities).
- High-Dimensional data poses problems when choosing ε

Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)



Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)

- EM Clustering solves one the main weaknesses of K-Means, which is it's naïve use of the mean value for the cluster center.
- The image below shows two scenarios where K-means fails due to the mean values of the clusters are tightly packed.





Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)

- Gaussian Mixture Models (GMM) allows more flexibility than K-Means by assuming the points are Gaussian Distributed.
- This way we now have two parameters to describe the cluster shape, the mean and standard deviation.



EM Clustering Using GMM **Steps 1 & 2**

Step 1

- Like K-Means, we begin by selecting the number of clusters and randomly initializing the Gaussian distribution parameters for each cluster.

Step 2

- Given these Gaussian distributions for each cluster, we compute the probability that each data point belongs to a particular cluster. The closer a point is to the Gaussian's center, the more likely it belongs to that cluster.
- This works well for Normally Distributed data as we are assuming that most of the data lies closer to the center of the cluster.



EM Clustering Using GMM Steps

Step 3

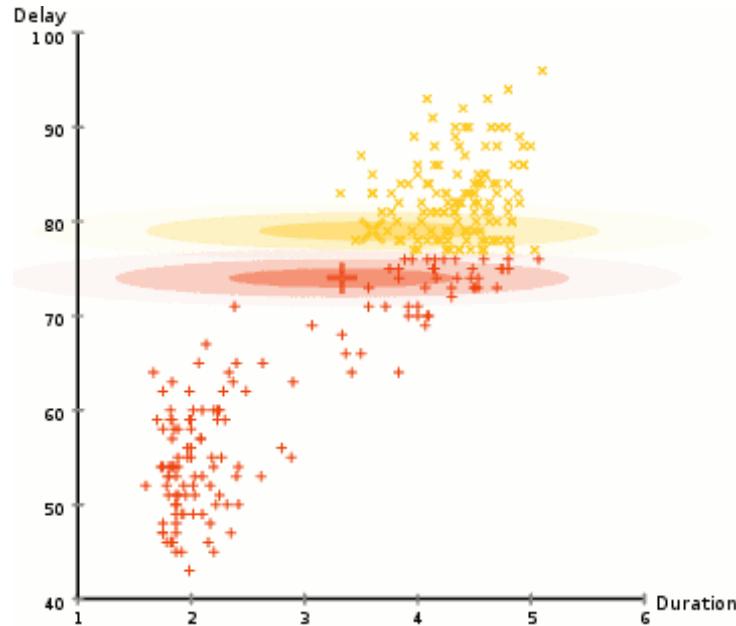
- Based on these probabilities a new set of parameters for the Gaussian distributions is computed such that we maximize the probabilities of data points within the clusters.
- We compute these new parameters using a weighted sum of the data point positions, where the weights are the probabilities of the data point belonging in that particular cluster.

Step 4

- We repeat Step 2 and 3 iteratively until convergence.



EM Clustering Using GMM Demo





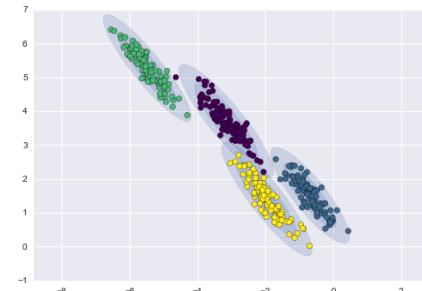
EM Clustering Using GMM Advantages and Disadvantages

Advantages

- GMMs are a lot more flexible in terms of cluster covariance than K-Means
- the clusters can take on any ellipse shape, rather than being restricted to circles
- Due to the use of probabilities data points can have multiple clusters e.g. a point can have 0.65 probability in being in one cluster and 0.35 in another.

Disadvantages

- Choosing K and scaling to higher dimensions

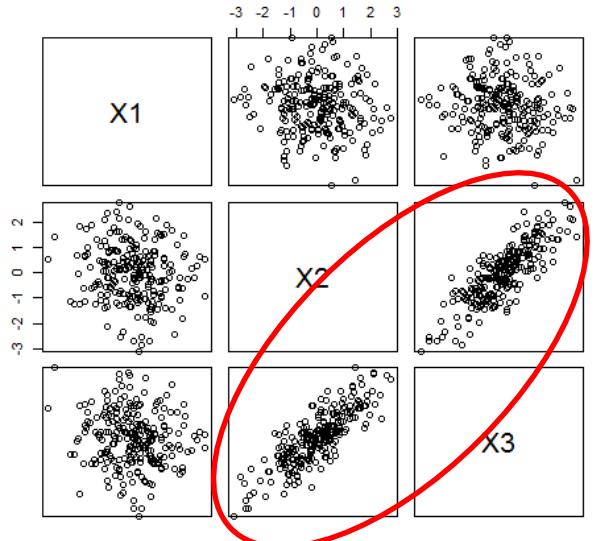


Principal Component Analysis



Why do we need PCA?

- Many times when working with large datasets with **many dimensions** things get too **computationally expensive** and confusing to keep track off
- However, many times we look at the data itself, we see variables that are **strongly correlated** and hence possibly redundant. E.g. a persons height and weight
- What if there was a way to **reduce the dimensionality** of our data while still retaining it's information?
- That is what **Principal Component Analysis** achieves





What is PCA Exactly?

- PCA is an algorithm that compresses your dataset's dimensions from a **higher to lower dimensionality**.
- It does this based on the **eigenvectors** of the variance in your dataset
- PCA is extremely used in **Data Compression** saving loads in processing time, as well as in **Visualizations** of high dimensional data in 2 or 3 dimensions. This is very helpful when doing **cluster analysis**.
- More technically, PCA finds a **new set of dimensions** such that:
 - All the dimensions are orthogonal (and thus linearly independent)
 - Ranked according to the variance of data along them. This means the first principal component contains the most information about the data.



How does PCA Work?

1. Calculate **the Covariance Matrix (X)** of the our data points (i.e. how our variables all relate to one another)
 2. Calculate the **Eigenvectors** and their **Eigenvalues**
 3. Sort our Eigenvectors according to their Eigenvalues in from largest to smallest.
 4. Select a set of **Eigenvectors (k)** to represent our data in k-dimensions
 5. Transform our original n-dimensional dataset into k-dimensions
-
- What exactly are Eigenvectors & Eigenvalues?



Eigenvectors & Eigenvalues

- The **eigenvectors** or principal components of our covariance matrix represent the **vector directions of the new feature space**.
- The **eigenvalues** represent the **magnitudes** of these vectors.
- As we are looking at our covariance matrix the eigenvalues thus quantify the **contributing variance of each vector** to the overall variance in our dataset



Understanding them intuitively

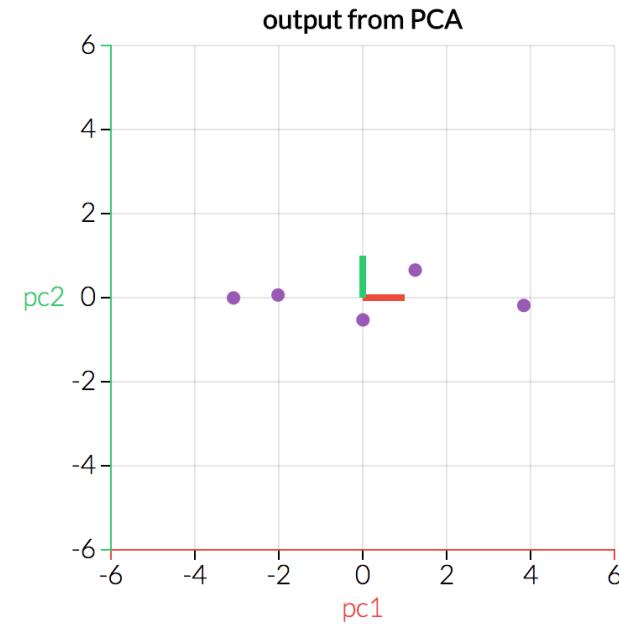
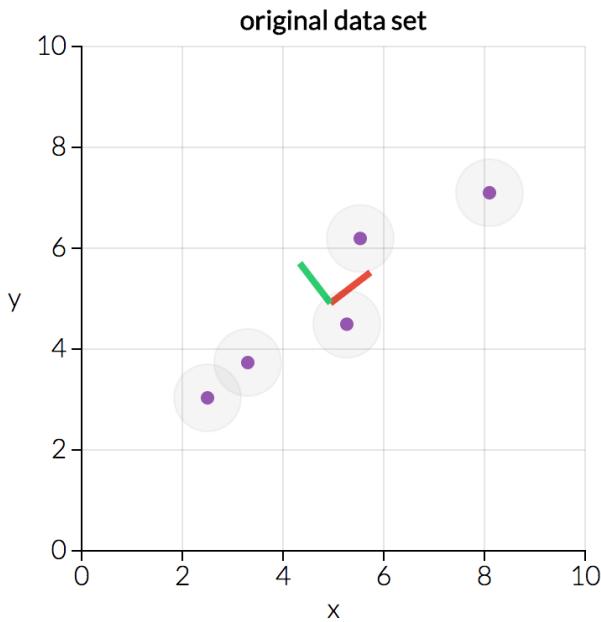
- Imagine you have a bunch of people. Every person has a group of friends they talk to, with some frequency. And every person has a credibility rating (though the same person may have different credibility ratings from different people).
- Distribute an amount X of gossip to each person, and let them talk to each other.
- The largest eigenvalue gives you an idea of the fastest rate at which gossip can grow in this social circle.
- The corresponding eigenvector gives you an idea of how much gossip each person should start with in order to obtain this maximal growth rate. (In particular, if you want a story to spread rapidly, the largest components of the principal eigenvector identify who you should tell the story to)



<https://www.quora.com/What-is-the-physical-meaning-of-the-eigenvalues-and-eigenvectors>

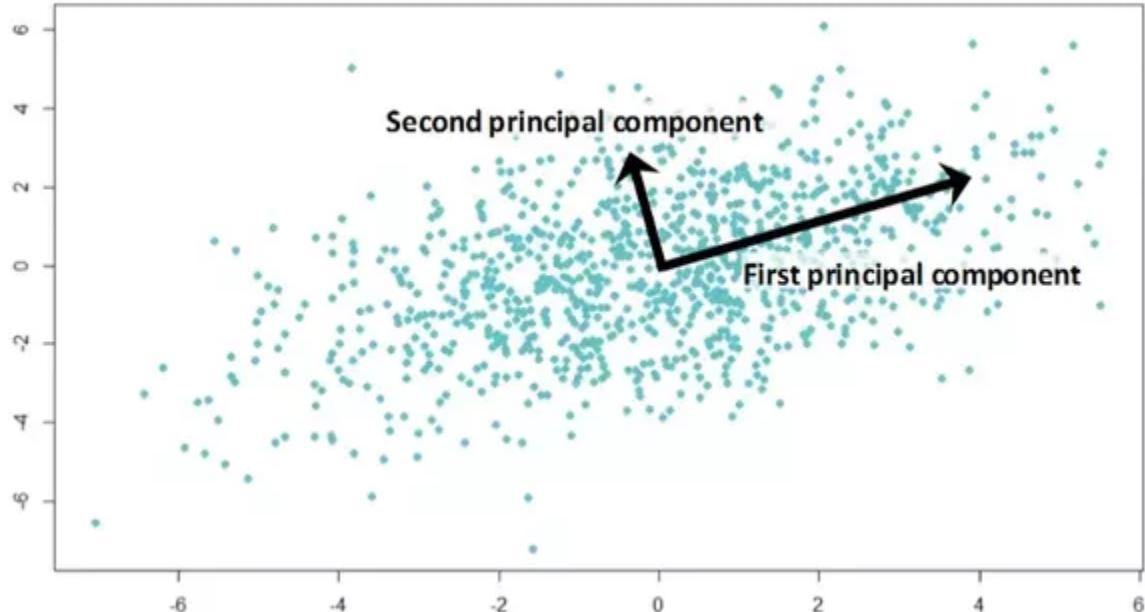


PCA Output





PCA Output



<https://medium.com/@sadatnazrul/the-dos-and-donts-of-principal-component-analysis-7c2e9dc8cc48>



Points to Remember about PCA

- Always normalize your dataset before performing PCA
- Principal components are always linearly independent
- Every principal component will be Orthogonal/Perpendicular to every other principal component
- Using PCA we can use k-dimensions to represent some value (always less than 100%, your goal would be to get it as close to 100% as possible with as small as k as possible)
- Numerically we compute PCA using SVD (Singular Value Decomposition)
- The goal of PCA is to create a new data that represents the original with some loss due to compression/reduced dimensionality
- $New\ Data_{k \times n} = [top\ k\ eigenvector]_{k \times m} [original\ dataset]_{m \times n}$



PCA Disadvantages

- PCA works only if the observed variables or data is linearly correlated. If there is no correlation within our dataset PCA will fail to find the adequate components to represent our data with less dimensions
- PCA by nature is lossy and thus information will be lost when we reduce dimensionality
- It is sensitive to scaling which is why all data should be normalized
- Visualizations are hard to interpret real meaning as they do not relate to the original data features.



PCA In Python!

t-Distributed Stochastic Neighbor Embedding (t-SNE)



t-Distributed Stochastic Neighbor Embedding (t-SNE)

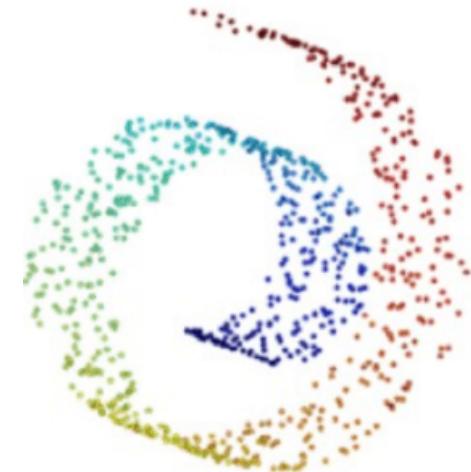
Introduction

- **t-SNE is another Dimensionality Reduction algorithm that is very popular.**
- **Published in 2008, in practice it has proven more effective than PCA.**



Why is t-SNE better than PCA?

- PCA is almost the defacto standard in dimensionality reduction tasks and is a part of possibly every Machine Learning class.
- PCA is good as it creates low-dimensional embeddings that preserve the overall variance of the dataset.
- However, PCA is a Linear Projection which means it's unable to capture non-linear dependencies.
- For example, PCA would be unable to unroll the data structure (right)
- t-SNE is not limited to linear projections which makes it applicable to many more datasets.





How does t-SNE Work?

- t-SNE uses local structure. It attempts to map points of higher dimension onto a lower dimensional space so that the distances between points remain almost the same.
- There are in fact other dimensionality reduction algorithms that attempt this such as Local Linear Embeddings and Kernel PCA. However, t-SNE works best in practice.
- The reason t-SNE works so well is that it solves the crowding problem caused by the curse of dimensionality. When mapping these higher dimensional points onto a lower dimension we end up with all the points squashed, causing crowding.
- t-SNE solves this by making the optimization spread-out.



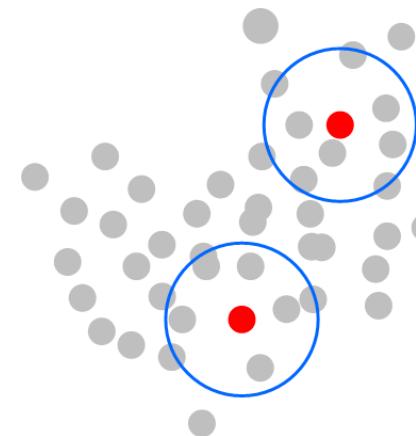
How does t-SNE Work?

- Additionally t-SNE uses **stochastic neighbors**
- This allows us to have no clear line between which points are neighbors of the other points.
- The lack of defined boundaries is a huge advantage as it allows t-SNE to naturally take both the global and local structure into account.
- The local structure is more important than global structure, however the points that are far away aren't ignored completely. This allows for a well-balanced dimensionality reduction.



The t-SNE algorithm – Step 1

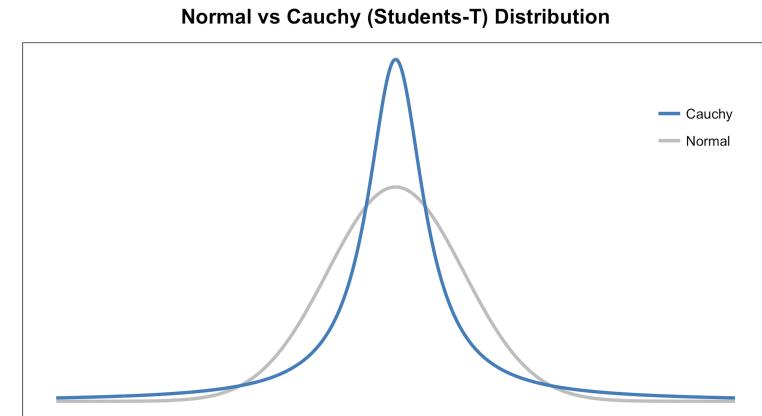
- Firstly, we measure similarities between points in the higher dimensional space.
- Imagine we have a bunch of 2d points scattered (see right). For each point we center a Gaussian distribution over the point.
- We then measure the density of all points under that Gaussian distribution.
- This is effectively converting the high-dimensional Euclidean distances between data points into conditional probabilities that represent similarities.





The t-SNE algorithm – Step 2

- We perform Step 1, however this time we use a Student t-distribution with one degree of freedom (also called a Cauchy distribution)
- The t-distribution shape allows better modeling for distances that are far apart.





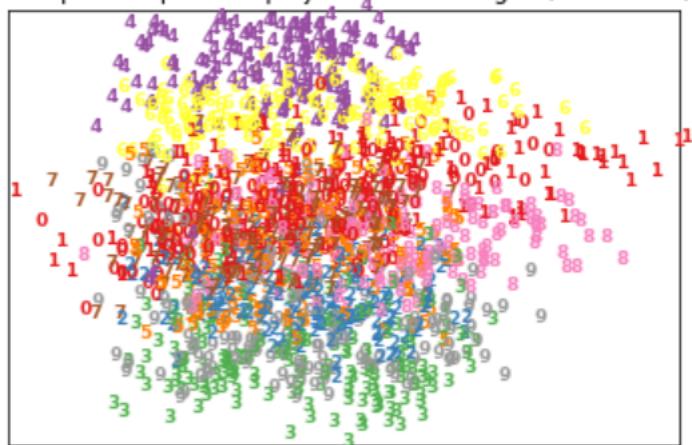
The t-SNE algorithm – Step 3

- In the last step we want these set of probabilities from the low-dimensional space to reflect those of the high dimensional as best as possible (as we want the two map structures to be similar).
- We then measure the difference between the probability distributions of the two-dimensional spaces using Kullback-Liebler divergence (KL), often written as $D(p,q)$.
- KL-divergence is an asymmetrical approach that compares two distributions (“distance” between two distributions).
- We then minimize our KL cost function using gradient descent.

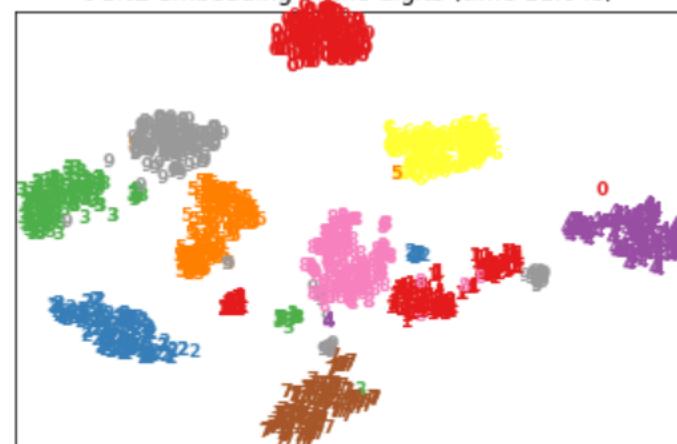


Visual Comparison of t-SNE vs PCA

Principal Components projection of the digits (time 0.04s)



t-SNE embedding of the digits (time 11.04s)



Introduction to Recommendation Engines





Netflix's Recommendations

NETFLIX Home TV Shows Movies Recently Added My List

Because you watched Love, Death & Robots ›

Top Picks for Rajeev

Because you watched Better Call Saul

641



Amazon's Recommendations

Inspired by your shopping trends



Frequently bought together



Total price: £477.42

[Add all three to Basket](#)

i These items are dispatched from and sold by different sellers. [Show details](#)

This item: Canon EF 70-300 mm f/4-5.6 IS II USM Lens - Black £449.00

JJC Reversible Lens Hood Shade for Canon EF 70-300mm f/4-5.6 IS II USM Replaces Canon Lens Hood ET... £14.99

Hoya 67mm UV(C) Digital HMC Screw-in Filter,Y5UVC067 £13.43



YouTube GB

Search

7

Home

Trending

Subscriptions

Library

Recommended



JVNA Live Ep:008 | Alchemy | Melodic Dubstep, Future...
JVNA
53K views • 4 weeks ago



Relaxing Music Mix | BEAUTIFUL PIANO
Epic Music World
38M views • 3 years ago



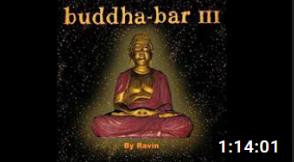
Soca 2019 / Soca 2020- Duty Free 2019 (The Best...)
REDYMIX DMTV
15K views • 1 month ago



Nuclear Fusion Energy: The Race to Create a Star on...
Motherboard
3.9M views • 2 years ago



Paramore: Playing God [OFFICIAL VIDEO]
Fueled By Ramen
68M views • 8 years ago



Buddha-Bar III - CD1
Buddha-Bar Official Channel
186K views • 5 months ago



YEAH yeah yeah yeah
3:46



ピアノ曲 勉強用
1:05:09



MACHINE LEARNING
Living in the Age of AI 41:17

643



How do they know us so well?





Are Big Tech Companies Spying on Us?

- No
- Well technically no.
- Through advances in Recommendation System Machine Learning methods, companies can create very accurate user specific recommendations!



Why are Recommendation Systems so Important?

- **Information Overload** – Think about online retailers like Amazon, eBay or Netflix. What problem can you easily foresee? There is just too much products or movies to choose from. Yes we can search for something we want, but what about things we'd potentially like, but didn't know existed.
- In this new era of Big Data, we need systems with heuristic techniques that **make our process of selection easier!**

“

*“Recommender Systems aim to help
a user or a group of users
to select items from a crowded item
or information space.”*

(MacNee et. al 2006)



Section Overview

In this section we'll learn

- Intuition behind Recommendation Systems – How do we review items?
- Collaborative Filtering and Content-based filtering

**Before recommending, how do we
rate or review Items?**



Let's try to Build an Item Comparison Tool

- Suppose we have website called “**I like to Watch Movies**”, a Netflix competitor
- We allow users can rate movies by giving a thumbs **up** or **down** to rank it.
- And now we want to get a list of our highest rated movies
- **How do we do this?**



I Like to Watch Movies





Approach 1 – Net Score

- We take the Net Score of Positive ratings– Negative ratings
 - $Net\ Score = (Positive\ Ratings) - (Negative\ Ratings)$

Movie	Positive Ratings	Negative Ratings	Net Score	Ave Percent Positive
A	750	500	250	60%
B	5000	4000	1000	56%

- Our algorithm scores Movie B higher, but is this right? **Nope**



Examples of sites making this mistake

Urban Dictionary

2. **normal**

209 up, 50 down 

A word made up by this corrupt society so they could single out and attack those who are different

Normal is nothing but a word made up by society

[conformists](#) [worker bees](#) [in crowd](#) [followers](#) [mindless](#)

by [Bill](#) Oct 6, 2005 [share this](#) [add comment](#)

3. **normal**

118 up, 25 down 



Approach 2 – Average Rating

- $\text{Average Rating} = \frac{\text{Positive Ratings}}{\text{Total Ratings}}$

Movie	Positive Ratings	Negative Ratings	Net Score	Ave Percent Positive
A	750	500	250	60%
B	5000	4000	1000	56%
C	9	1	8	90%

- Our algorithm now scores Movie C the highest but again is this right?
Nope. Movies with very little reviews will dominate the rankings.



Sites making this mistake - Amazon

4



EVGA GeForce GTX 1660 Ti XC Black Gaming, 6GB GDDR6, HDB Fan Graphics Card 06G-P4-1261-KR

★★★★★ ▼ 2

£265⁹⁴ £279.99

✓prime | Same-Day

FREE delivery Today, Oct 30

More buying choices

£247.32 (16 used & new offers)

5



MSI NVIDIA GTX 970 Gaming Twin Frozr HDMI DVI-I DP Graphics Card (4GB, PCI Express, DDR5, 256 Bit)

★★★★★ ▼ 908

£429⁰⁰

£5.95 delivery

Only 1 left in stock.

More buying choices

£116.99 (5 used & new offers)

6



EVGA Geforce GTX 1050 TI GeForce GTX1050TI Graphic Card 4096 MB

★★★★★ ▼ 15

More buying choices

£226.42 (2 new offers)

654



Approach 3 **CORRECT** Score = Lower bound of Wilson score confidence interval for a Bernoulli parameter

- It can be seen that we need to balance the proportion of positive ratings with the uncertainty of a small number of observations.
- Fortunately, the math for this was worked out in 1927 by Edwin B. Wilson.
- What we want to ask is: Given the ratings I have, there is a 95% chance that the “real” fraction of positive ratings is at least what?
- Wilson gives the answer. Considering only positive and negative ratings (i.e. not a 5-star scale), the lower bound on the proportion of positive ratings is given by:

$$\left(\hat{p} + \frac{z_{\alpha/2}^2}{2n} \pm z_{\alpha/2} \sqrt{\frac{[\hat{p}(1 - \hat{p}) + z_{\alpha/2}^2/4n]}{n}} \right) / (1 + z_{\alpha/2}^2/n).$$

User Collaborative Filtering and Item/Content-based Filtering



Recommendation Systems

- As we discussed in the introduction to this section, when we have too much choices, we tend to avoid choosing.
- We need a tool that can 'know' what we'd like. Imagine someone who knows you inside out, could be your best friend, your spouse or parent. But they can easily go into a store and pick things you'd like.
- They, in essence act like a Recommendation System
- But how do they do this? And how would an AI system figure this out?



Types of Recommendation Systems

Let's think about two ways we can do this for let's say, an online retailer like Amazon.

1. Do we have users that buy similar items? Let's say our system has records that we have a subset of users who bought Metallica albums, and most of these customers also bought Megadeath albums too. Therefore, we can infer if someone has purchases Metallica albums, they are a likely candidate to purchase a Megadeath album. This is called **Collaborative Filtering**.
2. Another approach is, what if we know a user has been searching for formal wear suits online. We know intrinsically that formal wear suits need to be paired with appropriate shoes. As such, we can recommend the user purchase our top rated shoes. This is called **Content or Item based filtering**.



Collaborative filtering User to User

- The Collaborative filtering algorithm is used to recommend products based on the history of user behaviors and consequently looks at the **similarities between users**
- A Collaborative filter uses something called the **user-to-item matrix** to find similar users.



User-to-item Matrix

		Image	Book	Video	Game
A	User icon	Green thumbs up	Red thumbs down	Green thumbs up	Green thumbs up
B	User icon		Green thumbs up	Red thumbs down	Red thumbs down
C	User icon	Green thumbs up	Green thumbs up	Red thumbs down	
D	User icon	Red thumbs down		Green thumbs up	
E	User icon	Green thumbs up	Green thumbs up	?	Red thumbs down

User	Item A	Item B	Item C	Item D
1	1	0	1	1
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	1	0	0



User-to-item Matrix Explained

- In our matrix, each row represents an individual customer
- Each column represents items in the inventory
- Therefore, a 1 or 0 indicates that the user in that row has purchased (clicked, watched, rated highly, liked, rated, etc.) the item
- From this matrix we constructed, we can calculate the similarity of users using some method to measure distance between users. A popular and successful method is the **Cosine Similarity**



Using the Cosine Similarity

- $\text{Cosine Similarity}(a, b) = \cos(a, b) = \frac{a \cdot b}{\|a\| \times \|b\|}$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

- A and B are used to represent two users, A and B
- A_i and B_i represent each item User A and B purchased.



Cosine Similarity Explained

Imagine we have two sentences to compare

- Amy likes mangoes more than apples
- Sam likes potatoes, Sam likes mangoes

Create a list of words: “Amy, likes, mangoes, more, than, apples, Sam, potatoes”



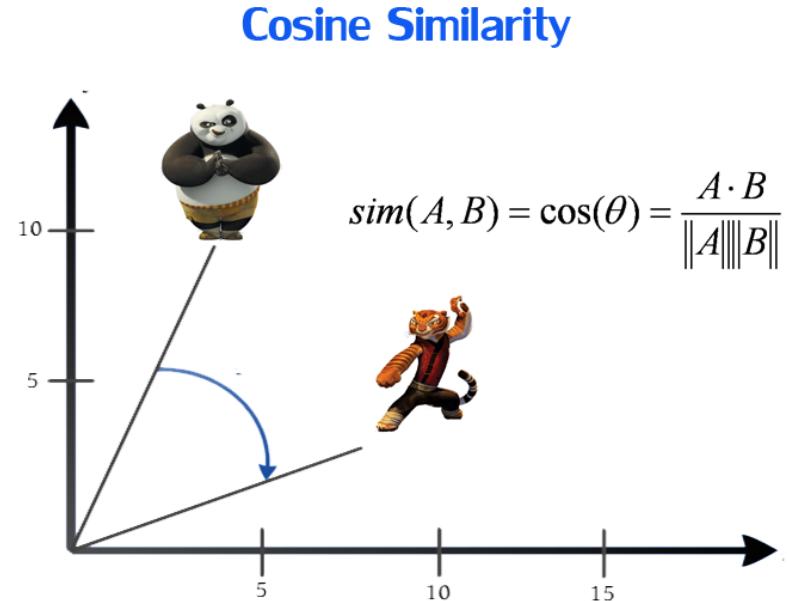
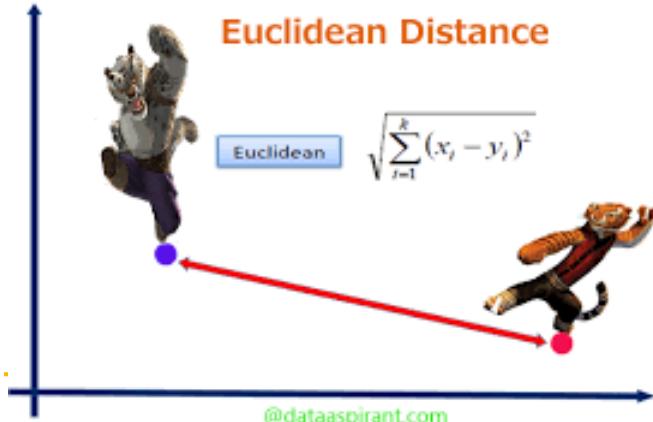
Cosine Similarity Explained continued

Words	Sentence 1	Sentence 2
Amy	1	0
Likes	1	2
Mangoes	1	1
More	1	0
Than	1	0
Apples	1	0
Sam	0	1
potatoes	0	1



Cosine Similarity Explained continues

- Sentence 1 = [1,1,1,1,1,1,0,0]
- Sentence 2 = [0,2,1,0,0,0,1,1]
- Cosine Distance = 0.463





Item-based approach collaborative filtering

- In Collaborative Filtering, we calculated similarities between the two users by building a user-to-item matrix.
- However, for a **item-based approach**, we need to calculate similarities between the two items.
- This means that we need to build and use an item-to-user matrix, (we do this by simply transposing the user-to-item matrix).



User to User Collaborative Filtering Disadvantages

- Systems performed poorly when they had many items but comparatively few ratings
- Computing similarities between all pairs of users was expensive
- User profiles changed quickly and the entire system model had to be recomputed



How item-item based filtering solved this problem

- Item-item models resolve these problems in systems that have more users than items.
- Item-item models use rating distributions per item, not per user. With more users than items, each item tends to have more ratings than each user, so an item's average rating usually doesn't change quickly.
- This leads to more stable rating distributions in the model, so the model doesn't have to be rebuilt as often. When users consume and then rate an item, that item's similar items are picked from the existing system model and added to the user's recommendations.

Case Study – User Collaborative Filtering and Item/Content-based filtering in Python

The Netflix Prize and Matrix Factorization and Deep Learning as Latent-Factor Methods



Collaborative Filtering Recap

- Looks relatively simple to create and underlying theory is relatively simple
- Can be applied to many types of examples
- High accuracy

However, it's not all good, let's take a look at some of the potential issues we face when using Collaborative filtering.



Collaborative Filtering Challenges

- **Sparsity** – the number of purchases made by each user is extremely small compared to the number of items that exist.
- **Cold-start issues** – we don't have anything to recommend to new users as they haven't made any purchases/reviews/likes yet
- How to recommend a **new item?**
- **Scalability** – the more users grow the larger the computational requirements to calculate distances between them.



Latent-Factor Methods were introduced to help with Scalability Issues

- Latent-factor methods create a new and most times a reduced feature space of the original user or item vectors, leading to reduced noise and faster computations in real-time.

There are two latent-factor methods:

1. *Matrix factorization*
2. *Deep learning*



Matrix Factorization

- **Matrix Factorization** was popularly used in the Netflix Prize Competition.
- Recommendations are a vital part of business in our modern era, so much so that Netflix offered \$1M USD in a competition to improve recommendation performance over their in-house algorithm.
- On September 21, 2009, the grand prize was given to the BellKor's Pragmatic Chaos team which bested Netflix's own algorithm for predicting ratings by 10.06%

Netflix Prize

Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard

Display top leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	No Grand Prize candidates yet	--	--	--
Grand Prize - RMSE <= 0.8563				
1	PragmaticTheory	0.8584	9.78	2009-06-16 01:04:47
2	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
3	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
4	Dace	0.8604	9.56	2009-04-22 05:57:03
5	BigChaos	0.8613	9.47	2009-06-15 18:03:55
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
6	BellKor	0.8620	9.40	2009-06-17 13:41:48
7	Gravity	0.8634	9.25	2009-04-22 18:31:32
8	Opera Solutions	0.8640	9.19	2009-06-09 22:24:53
9	xilector	0.8640	9.19	2009-06-17 12:47:27
10	BruceDengDaoCiYiYou	0.8641	9.18	2009-06-02 17:08:31
11	Ces	0.8642	9.17	2009-06-12 23:04:25
12	majia2	0.8642	9.17	2009-06-15 03:35:00
13	xiangliang	0.8642	9.17	2009-06-13 16:35:35
14	Feeds2	0.8647	9.11	2009-06-16 22:21:19
15	Just a guy in a garage	0.8650	9.08	2009-05-24 10:02:54
16	Team ESP	0.8653	9.05	2009-06-16 05:25:11
17	pengpengzhou	0.8654	9.04	2009-05-05 18:18:03
18	NewNetflixTeam	0.8657	9.01	2009-05-31 07:30:22
19	J.Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	Vandelay Industries !	0.8658	9.00	2009-05-11 00:43:14



675



Singular Value Decomposition (SVD)

- Singular Value Decomposition (SVD) decomposes the preference matrix as:

- $P_{m \times n} = U_{m \times m} \Sigma_{m \times n} V_{n \times n}$

- **U and V are unitary matrixes. For 4 users and 5 items, we will have:**

$$P_{4 \times 5} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ u_{41} & u_{42} & u_{43} & u_{44} \end{bmatrix} \bullet \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 \\ 0 & 0 & 0 & \sigma_4 & 0 \end{bmatrix} \bullet \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \\ v_{31} & v_{32} & v_{33} & v_{34} & v_{35} \\ v_{41} & v_{42} & v_{43} & v_{44} & v_{45} \\ v_{51} & v_{52} & v_{53} & v_{54} & v_{55} \end{bmatrix}$$

- Where $\sigma_1 > \sigma_2 > \sigma_3 > \sigma_4$, therefore the preference for the first item is written as:
- $p_{11} = \sigma_1 \mu_{11} v_{11} + \sigma_2 \mu_{12} v_{21} + \sigma_3 \mu_{13} v_{31} + \sigma_4 \mu_{14} v_{41}$
- **Vector Form** - $p_{11} = \underbrace{\sigma}_{\sigma} \circ \underbrace{\rightarrow u_1}_{u_1} \cdot \underbrace{\rightarrow v_1}_{v_1}$
- So now we can select the top two features based on the sigmas:
- $p_{11} \approx \sigma_1 \mu_{11} v_{11} + \sigma_2 \mu_{12} v_{21}$



Simon Funk's SVD

There were still two issues with using SVDs

1. The way missing values are imputed can have an undesirable impact on the outcome
2. The computational complexity for training can be high when all the entries are considered

During the Netflix Prize content, Simon Funk developed a solution. Where only non-missing entries (p_{ij}) are considered.

$$\min_{\vec{u}_i, \vec{v}_j} \sum_{p_{ij}} (p_{ij} - \vec{u}_i \cdot \vec{v}_j)^2$$

Therefore, the estimated score for the item j from user i is:

$$\vec{u}_i \cdot \vec{v}_j$$



Deep Learning Embedding

- Deep learning offered a more flexible method to include various factors into modeling and creating embeddings.
- The workings for many of these methods can be a bit complicated to explain, however in essence, they formulate the problem as a classification problem where each item is one class.
- There have many advances that can deal with millions of users and items.
- Multi-Phase modeling such as used by YouTube divided the modeling process into two steps where the first uses only user-item activity data to select hundreds of candidates out of millions. Then in the second phase it uses more information about the candidate videos to make another selection and ranking.
- All Recommender systems should not overfit historical user-item preference data (exploitation), to avoid getting stuck in a local optimal.

Introduction to Natural Language Processing



Introduction to Natural Language Processing

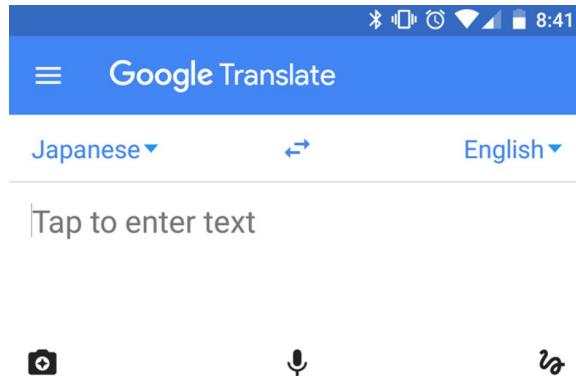
- Natural Language Processing (NLP) is a sub-field of Artificial Intelligence that deals with the processing and understanding of human language.
- It is the foundation of language translation, topic modeling, document indexing, information retrieval and extraction.
- Current hot topics in NLP involve search engines, chat bots, sentiment analysis, summarization of documents, marking essays and grammatical checking and improving writing.





NLP's role in Businesses - Translating

- Imagine translating your website or business communication into different languages:



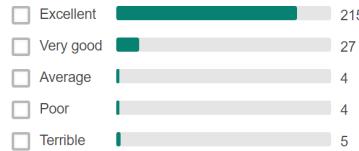


NLP's role in Businesses - Summarizing information

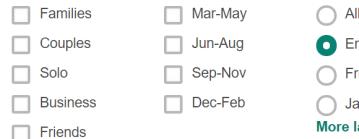
Reviews (265)

[Write a review](#)

Traveller rating



Traveller type



Time of year



Language

[More languages ▾](#)

Show reviews that mention

[All reviews](#)[chicken](#)[papadoms](#)[curry](#)[naan breads](#)[prawn puri](#)[dumplings](#)[king prawns](#)[nepalese food](#)[delicious food](#)[tasted amazing](#)[fantastic restaurant](#)[small restaurant](#)[great menu](#)[recommend this restaurant](#)[saturday night](#)[visited this restaurant](#)[attentive service](#)

1 - 10 of 255 reviews

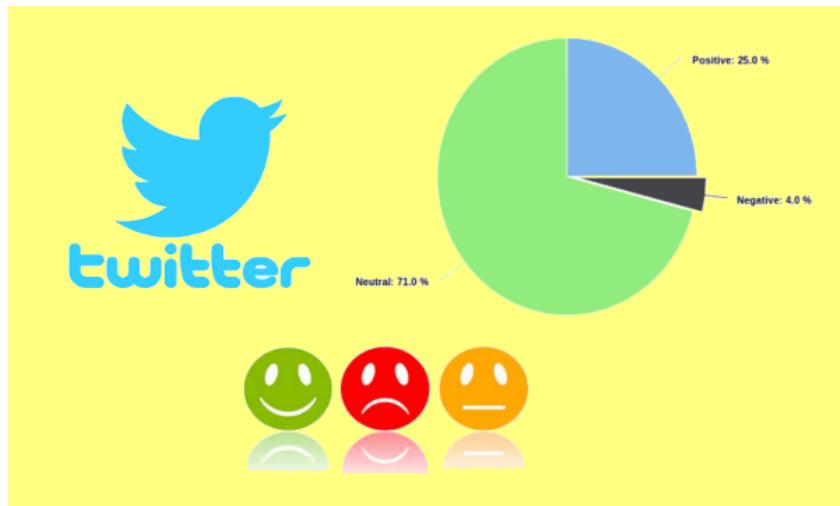
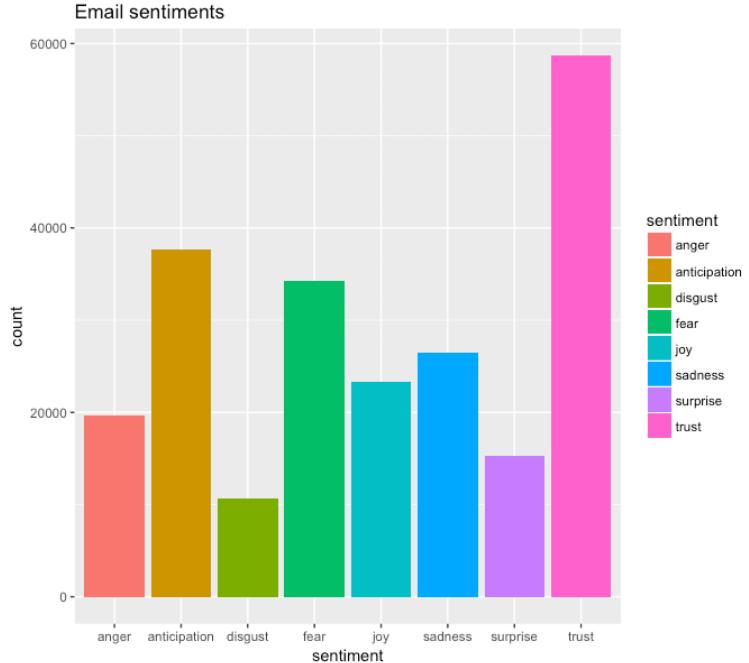
Read reviews that mention

[small amount](#)[easy to apply](#)[dry hair](#)[short hair](#)[matt and messy](#)[works well](#)[hair gel](#)[much better](#)[hair in place](#)[easy to use](#)[fairly short](#)[rough paste](#)[strong hold](#)[makes it easy](#)[easy to wash](#)

198 customer reviews

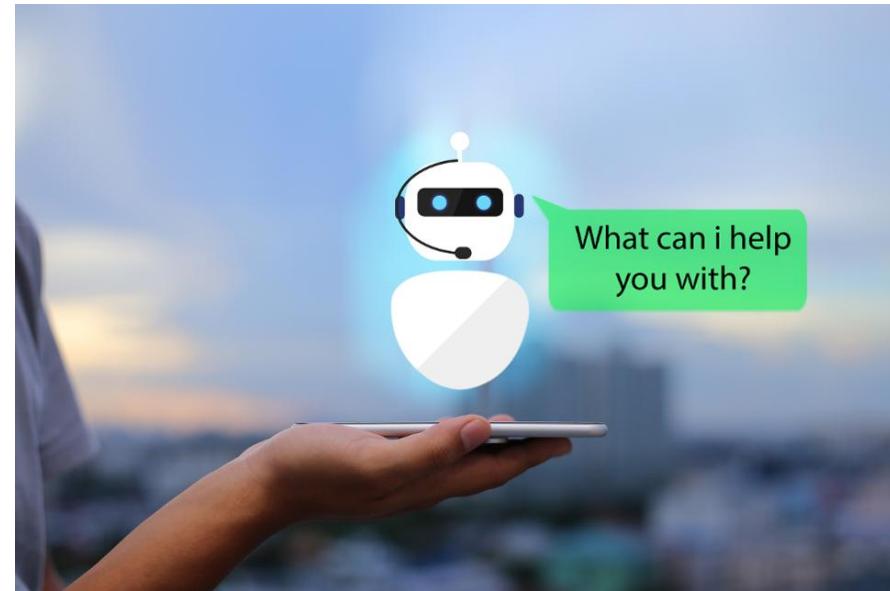


NLP's role in Businesses - Sentiment analysis





NLP's role in Businesses - Chatbots





NLP's role in Businesses – Information Extraction and Search





NLP's role in Businesses – Auto Responders and Auto Complete

The screenshot shows a messaging application interface. A dark gray message bubble at the top contains the text "Taco Tuesday". Below it, a message from Jacqueline Bruzek is shown, consisting of a small profile picture, the name "Jacqueline Bruzek", and a close button. Further down, another message bubble contains the text "Taco Tuesday". Below this, a message from "Hey Jacqueline," reads: "Haven't seen you in a while and I hope you're doing well."



Main Topics in NLP

- Bag of Words Model
- Tokenization
- Predictions
- Sentiment Analysis - Case Study – Sentiment of Airline Tweets
- Summarization - Case Study – Amazon Review Summaries
- Text Classification - Case Study - Spam Filter

Modeling Language – The Bag of Words Model



Bag of Words Modeling

- When training NLP models, we need to understand how we create or format our data.
- In English we've got hundreds of thousands of words, which begs the question, how do we represent our words as data inputs to a ML model?



Typical Machine Learning Inputs

Input #	Output - Gender
Height	
Weight	
Hair Length	0 – Male 1 - Female

Input #	Output – Diabetes Risk
Height	
Body Fat	
Weight	Percentage Output

- Typical Datasets we've seen so far have simple or intuitive inputs.
- How do we represent language or text inputs in this?
- We use Tokenization!



Tokenization

- **Tokenization** is the process where we take an input string of text and split it up (parse) into individual string elements.
- It is a vectorization technique that allows us to represent words as real numbers



Bag of Words and Tokenization Example

1. *We have landed on the moon*
2. *The USA has landed on the moon*

[We, have, landed, on, the, moon]

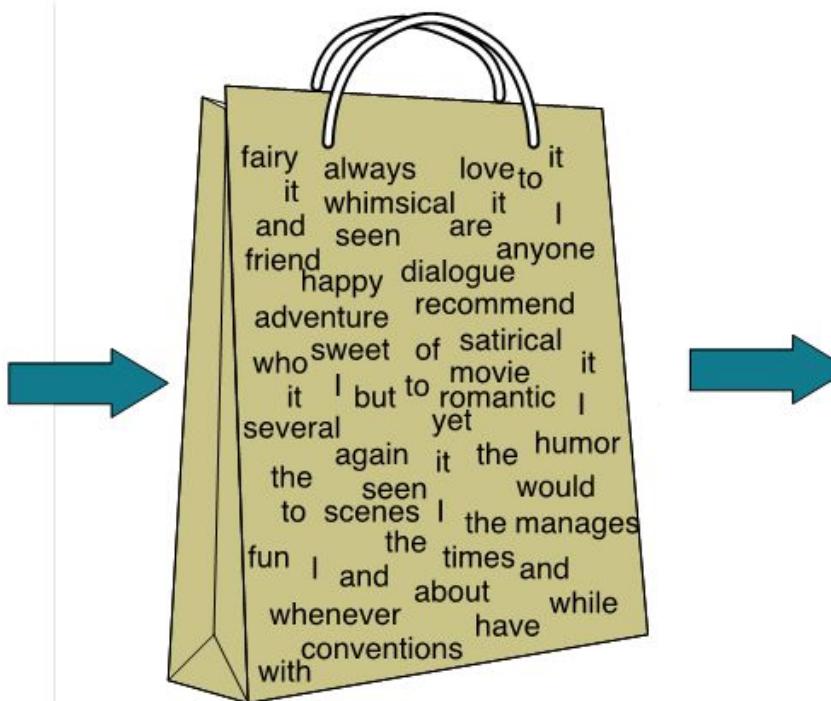
[The, USA, has, landed, on, the, moon]

#	We	the	have	USA	has	landed	on	moon
1	1	1	1	0	0	1	1	1
2	0	1	0	1	1	1	1	1



The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...



Understanding our input data

1. *We have landed on the moon*
2. *The USA has landed on the moon*

#	We	the	have	USA	has	landed	on	moon
1	1	1	1	0	0	1	1	1
2	0	1	0	1	1	1	1	1



Original Document	Document Vector	Class Label
We have landed on the moon	[1,1,1,0,0,1,1,1]	Negative
The USA has landed on the moon	[0,1,0,1,1,1,1,1]	Positive

Normalization, Stop Word Removal, Lemmatizing/Stemming



Words are Messy

"We ain't gonna get it in time"

"This meal was pleasantly exquisite!"

"Vote 4 PSG.LGD at The International 2019"

"OMG lol jus txt me pls, tttyl"

"U wot mate?"



- Words in any language (we use English here) can get extremely messy! Any NLP project requires extensive cleaning, but how do we go about doing this?



Normalization Types

- Case Normalization
- Removing Stop Words
- Removing Punctuation and Special Symbols
- Lemmatising/Stemming



Case Normalization

Case normalization is simply the standardizing of all case for words found in our document. Example, changing sentences like:

- “**Hi John, today we’ll go to the market**”:
- “**hi john, today we’ll go to the market**

Typically, case doesn’t actually change any meaning, however there are cases (pun intended) where it can, example:

- Reading is a city in the UK which is different to the act of reading.
- April and May are both common names and months.



Removing Stop Words

- Stop words are common words that do not act additional information or predictive value due to how common they are in normal text. Examples of common stop words are:
 - I
 - is
 - and
 - a
 - are
 - the



Removing Punctuation and Special Symbols

- Self explanatory, but very important step in our cleaning process. However, there are instances where punctuation adds mean such as it's vs. its. Example of punctuation and special symbol removals are:
- !"#\$%&'()*+,-./';<=>?@[\\]^_`{}|~



Lemmatising/Stemming

- Both lemmatizing and stemming are techniques that seek to reduce inflection forms to normalize words with the same lemma.
- Lemmatising does this by considering the context of the word while stemming does not.
- However, most current lemmatizers or stemmer libraries are not highly accurate.

Example

caresses	→	caress
ponies	→	poni
caress	→	caress
cats	→	cat

Stemming from Stanford NLP

TF-IDF Vectorizer (Term Frequency — Inverse Document Frequency)



Vectorizing Text

- Sometimes our good old Bag of Words model isn't always good enough, it's often a good way to explain how simple NLP classifiers are built, but in reality we need sometimes more complex methods.
- Methods that can account the importance of each term in a document/text.
- One of the simplest and most versatile vectorizers is the **TF-IDF Vectorizer** (Term Frequency — Inverse Document Frequency)



TF-IDF Vectorizer

The TF-IDF statistic for a term i in document j is calculated as shown below:

- $TF - IDF(i, j) = TF \times IDF$

Where

$$TF(i, j) = \frac{\text{number of times term } i \text{ is in document } j}{\text{total number of terms in document}}$$

$$IDF(i) = \ln \left(\frac{\text{total number of documents}}{\text{number of documents containing term } i} \right)$$

Word2Vec - Efficient Estimation of Word Representations in Vector Space



Vectorizing Text

- In our previous slides we explained Bag of Words modeling and the TF-IDF vectorizer. Both are useful, but more advances in vectorizing documents have been developed since.
- One such is the Word2Vec model developed by a Google Research team.



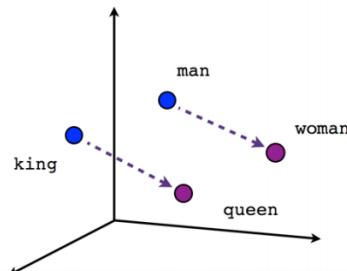
Vectorizing Text

- Word2vec has been pre-trained over large corpora
- It uses an unsupervised learning method to determine semantic and syntactic meaning from word co-occurrence, which is then used to construct vector representations for every word in the vocabulary.

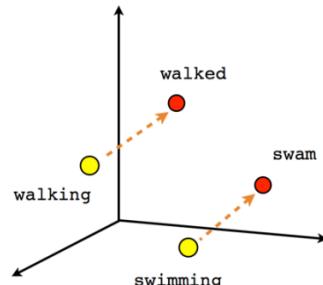


From the Word2Vec Paper - **Read** – “*Efficient Estimation of Word Representations in Vector Space*” <https://arxiv.org/pdf/1301.3781.pdf>

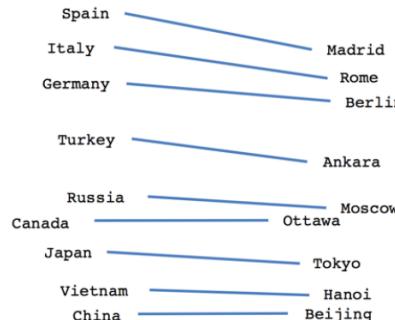
“Using a word offset technique where simple algebraic operations are performed on the word vectors, it was shown for example that $\text{vector}(\text{"King"}) - \text{vector}(\text{"Man"}) + \text{vector}(\text{"Woman"})$ results in a vector that is closest to the vector representation of the word Queen.”



Male-Female



Verb tense



Country-Capital



Training Word2Vec

- Two model architectures were used to train word2vec:
 - Continuous Bag of Words
 - Skip Gram
- They both use a **context window** to determine the contextually similar words, example, using a window with a fixed size n means that all the words within n units from the target word belong to its context.



Context Window

- Let's use a fixed window of size 2 on the following sentence:
 - The **quick brown fox** jumped over the lazy dog
- Let's look at **Fox** as our target word example.
- The words in blue (**quick, brown, jumped, over**) belong to the context of **Fox**
- Word2Vec has the ability that with enough examples of contextual similarity, the network learns the correct associations between words.



Context Window

- This assumption made by the design of the Word2Vec model is that, “*words which are used and occur in the same contexts tend to purport similar meaning*”
- The context window of Word2Vec is dynamic which has a max size.
- The context is samples from the max window size with a probability of $1/d$ where d is the distance between the word to target.
- In our previous example, the target word **fox** using a dynamic context window with maximum window size of 2. (brown, jumped) have a **1/1** probability of being included in the context since they are one word away from fox. (quick, over) have a **1/2** probability of being included in the context since they are two words away from fox.



Continuous Bag of Words & Skip Gram

- Using this method, the Continuous Bag of Words and the Skip Gram models separate the data into observations of target words and their context.
- Continuous Bag of Words** - In our Fox example, the context is 'quick, brown, jumped, over'. These form the features for the Fox class.
- Skip Gram** - Here we structure the data so that the target is used to predict the context, so here we'll use Fox to predict the context 'quick, brown, jumped, over'.



Building the Neural Network Model

- Word2Vec trains a shallow neural network over data structured using either the Continuous Bag of Words Model or Skip Gram.
- Example of a simple Continuous Bag of Words model with a fixed context window of 1 on a simple corpus (right)
- Let's use our context window to include words that follows the target. In this example we can assume the words at the end of the sentence is the first word of the next sentence.
- Here we can see that simple patterns emerge, where "Math" and "Programing" are both context to "like".

I like math.

I like programming.

Today is Friday.

Today is a good day.

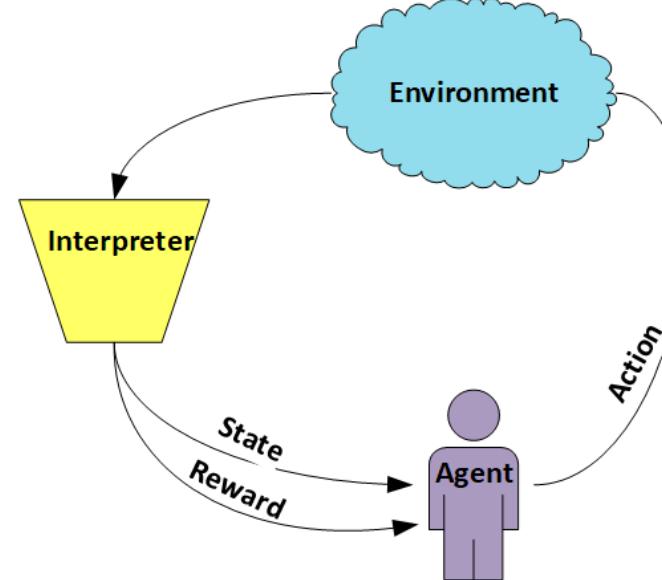
- like* is the context of target *I*
- math* is the context of target *like*
- programming* is also the context of target *like*

Reinforcement Learning



Introduction to Reinforcement Learning

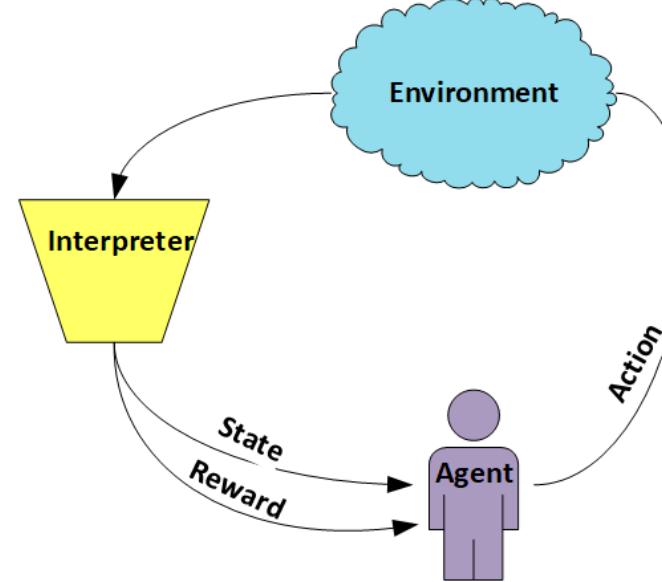
- In Reinforcement learning, we teach our algorithm, or '**agent**' in an **environment** that produces a **state** and **reward**.
- The agent performs **actions/interacts** with this environment which results in various responses.





Learning in Reinforcement Learning

- The agent in this environment examines the **state** and the **reward** information it receives.
- It chooses an action that **maximizes the reward** feedback it receives.
- The **agent learns** by repeated interaction with the environment.

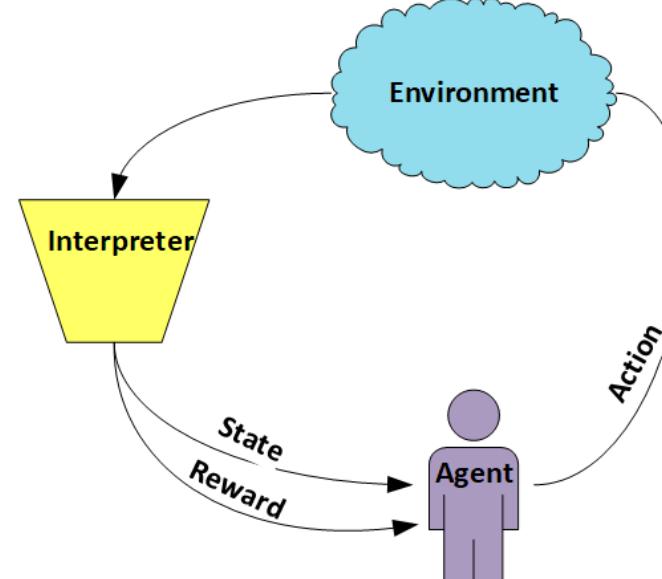




Learning in Reinforcement Learning

A successful agent needs to:

- Learn the interaction between states, actions and their corresponding rewards
- Determine which action(s) are the best to take





Q Learning

Q-learning is a model-free reinforcement learning algorithm.

The goal of Q-learning is to learn a **policy**, which tells an agent what action to take under what circumstances.

It does not require a model of the environment, and it can handle problems with stochastic transitions and rewards, without requiring adaptations.



Q Learning State Reward Tables

	Action 1	Action 2
State 1	0	5
State 2	5	0
State 3	0	5
State 4	5	0

- This is a simple State-Action-Reward table where in this 'game' we can see at each state the best reward the agent needs to take.
- If an agent randomly explored this game, and summed up which actions received the most reward in each of the four states (and stored this in an array), then it would basically learn the functional form of the table above.



State Reward Tables – Deferred Learning

	Action 1	Action 2
State 1	10	0
State 2	5	0
State 3	5	0
State 4	5	40

- After extensive trials an agent will be able to learn that taking Action 2 repeatedly for States 1,2,3 and 4 lead to the greatest reward..



The Q Learning Rule

- We can define the Q-Learning update rule as:
- r – Reward
- γ – Discounts reward impact (0 to 1)
- $\max_{\hat{a}} Q(s, \hat{a})$ - This is the maximum Q value possible in the next state. It represents the maximum future reward thus to encourage the agent to aim for the max reward in as little action steps

Introduction to Big Data



Big Data

- **Big Data** hype!
- Is Big Data AI?
- And what is 'Big'?



ALSO
BIG DATA
EVERY
EXAMPLES
TOOLS
DISK TARGET
APPLIED
SHARED SENSOR
RECONSIDER DEFINITION
CURRENT
MAY MOVING
WITHIN THOUGHT
ZETTABYTES
PRACTITIONERS CAPTURE BUSINESS
INTERNET DISTRIBUTED SETS GENOMICS
DESCRIBING RADIO-FREQUENCY COMPLEXITY
MANAGEMENT TERABYTES CASE
PETABYTES SYSTEMS INCLUDE
ELAPSED FORMS TOLERABLE
INCLUDE IS
ABILITY
SIZE MPP
STORAGE
PARALLEL
SINCE MASSIVELY GROW
QUALITIES
SAN
GARTNER WORKING AMOUNT
CURRENTLY
DIFFICULTY
OPPORTUNITIES
GROW
SEARCH
RECORDS
HUNDREDS
NETWORKS
Databases
CAPACITY
NOW
BIOLOGICAL
PROCESSING
UBQUITOUS
BURIED
WORLD'S
DESKTOP
CONNECTOMICS
ORGANIZATIONS
RELATIONAL
SOCIAL
INDEXING
CITATION
LARGER
SET
CONTINUES
USE
TENS
COMPLEX
ANALYTICS
RECORDS
HUNDREDS
NETWORKS
Databases
SEARCH



Big Data Defined

- Big Data refers to:
 - Huge volume of data
 - Processed by non traditional means
- The data is huge
- Small data can sometimes become big

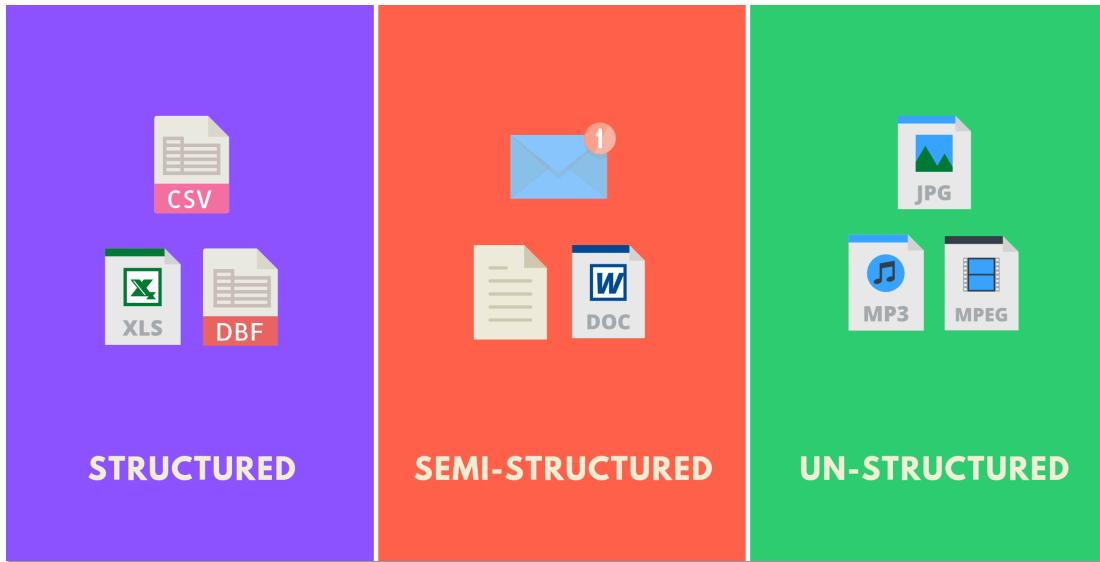


Examples of Big Data

- User information for millions of customers
- User generated information, think of a social media behemoths like Facebook and Twitter
- Behavioral data (e.g tracking user movements on a website)
- Image data
- Text Data



Classifications of Big Data





Structured Data

- This is data that has a proper defined structure to it, examples include:
 - Existing Database data
 - CSV or any other structured or organized data



Semi-Structured Data

- This is data that doesn't have a defined structure, example the text in documents can be messy and very disorganized

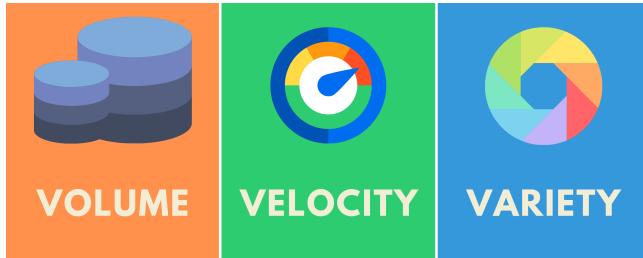


Un-Structured Data

- This is data that has no structure at all, examples include:
 - Video
 - Audio
 - Images



Big Data Characteristics - The 3Vs



- **Volume** – The amount of data generated, think of how much thousands of point of sale terminals can generate sales data
- **Velocity** – The speed at which this data is generated. Think about how fast thousands of IoT sensors can create data
- **Variety** – the various types of data being generated, eg. Think of an organizations cloud storage that holds everyone's documents, emails, audio, video etc.

Challenges in Big Data



Typical Data Storage Challenges





Challenges of Dealing with Big Data

- Storage Requirements
- Keep forever?
- Can we keep adding new data easily?
- How long do reports take to generate?



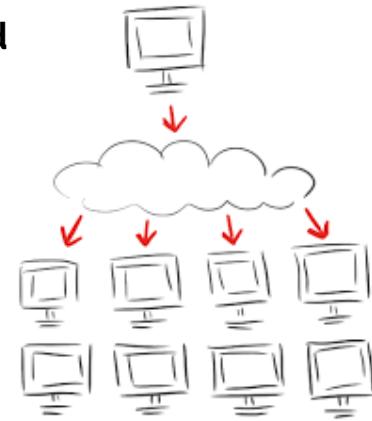
Big Data Solutions

- New and advanced databases and data structure
- Using **Distributed** workload



Distributed Data/Computing

- Big Data storage and analysis requires new tools (software and hardware)
- Instead of a supercomputer can we just use multiple computers?
- Introducing MapReduce

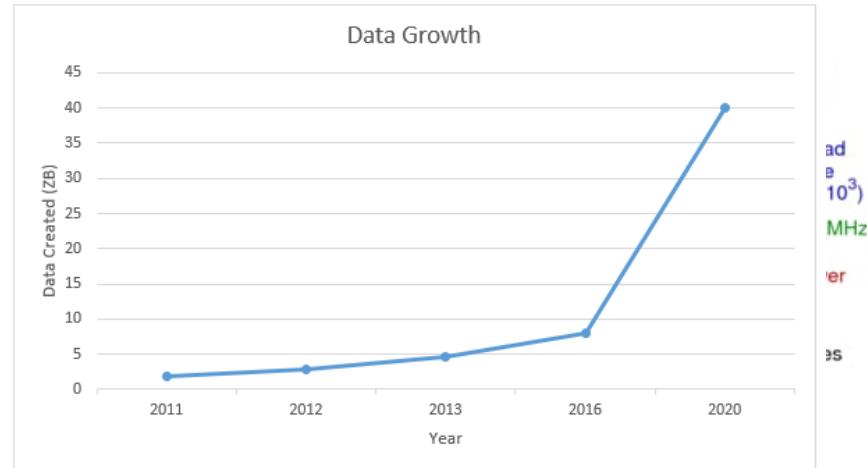


Hadoop, MapReduce and Spark



Big Data History

- While computer speed increased linearly in the last few decades
- Data growth did Not





MapReduce



- MapReduce is a programming framework for Big Data processing on **distributed platforms** created by Google in 2004. It can run calculations over thousands of computers in parallel
- It evolved further with advances by Hadoop and Spark, but one of the key paradigms set out by MapReduce was the Map and Reduce phase, still used by Hadoop and Spark today.



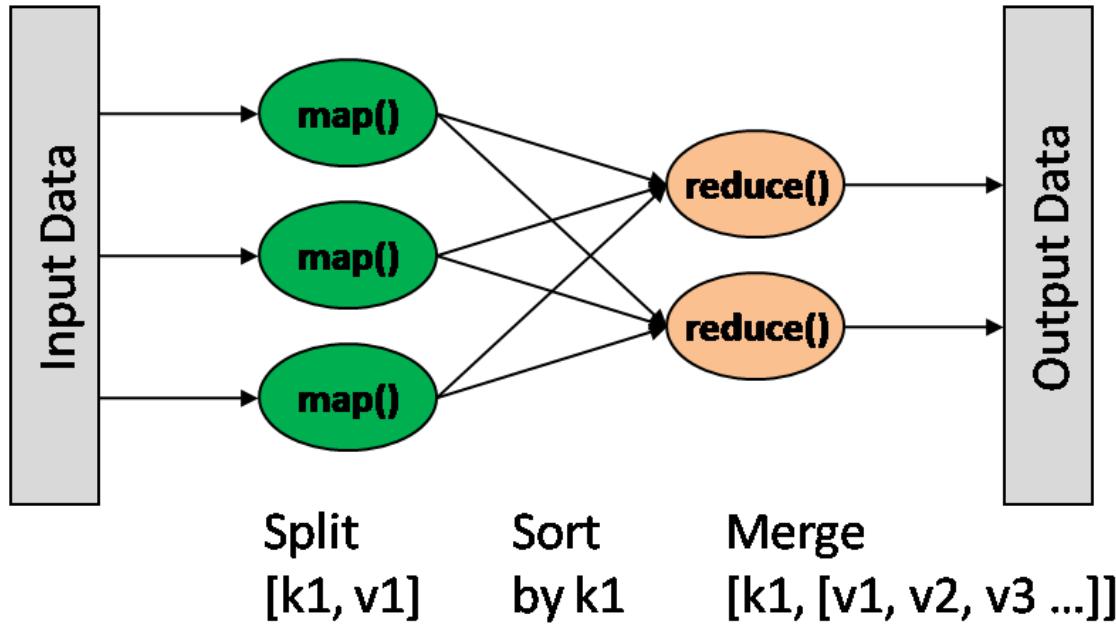
MapReduce



- **Map phase:** The user specifies a map function that is applied to each key-value pair, producing other key-value pairs, referred to as intermediate key-value pairs.
- **Reduce phase:** In this phase the intermediate key-value pairs are grouped by key and for each group the user applies a reduce function, producing other key-value pairs, which is the output of the round.
- A program written in MapReduce is automatically parallelized without the programmer having to care about the underlying details of partitioning the input data, scheduling the computations or handling failures.



MapReduce



Hadoop

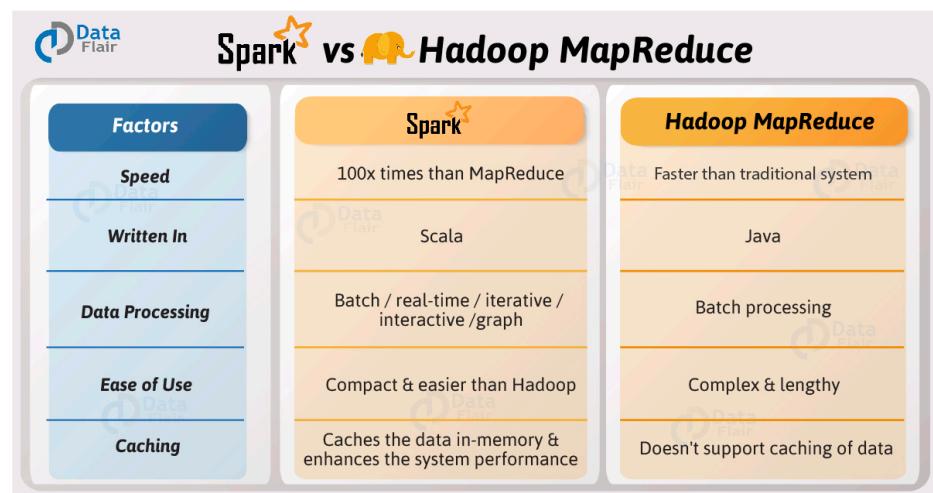


- Hadoop is a JAVA based open source system developed by Apache in 2006 and provides a software framework for distributed storage and processing of big data using the MapReduce programming model.
- Hadoop is a **Processing Engine/Framework** and introduced the HDFS (Hadoop Distributed File System), a batch processing engine (MapReduce) & a Resource Management Layer (YARN).
- Hadoop provided the ability to analyze large data sets. However, it relied heavily on **disk storage as opposed to memory** for computation. Hadoop was therefore very slow for calculations that require multiple passes over the same data.
- This allowed the hardware requirements for Hadoop operations to be cheap (hard disk space is far cheaper than ram) it made accessing and processing data much slower.
- However, Hadoop had poor support for SQL and machine learning implementations.

Spark

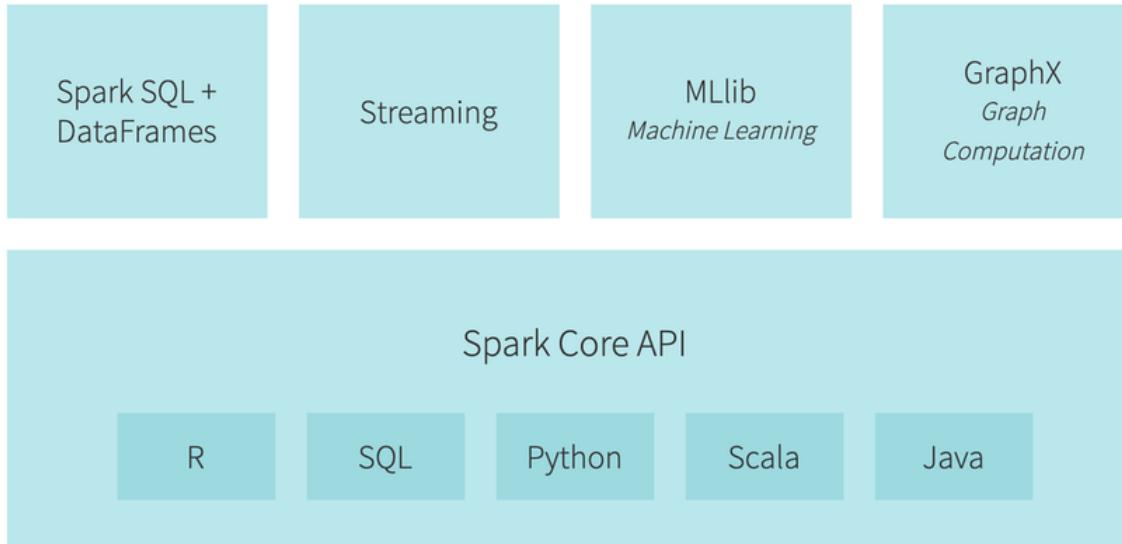


- The UC Berkeley AMP Lab and Databricks spearheaded the development of Spark that aimed to solve many of deficiencies, performance issues and complexities of Hadoop
- It was initially released in 2014 and donated to the Apache Software Foundation
- Spark powers a stack of libraries including SQL and DataFrames, MLlib for machine learning, GraphX, and Spark Streaming. You can combine these libraries seamlessly in the same application.
- As such, Spark was a big hit with Data Scientists!





Spark Components



Source: <https://databricks.com/spark/about>



RDDs – Resilient Distributed Data Set

- **RDD** is Spark's representation of a dataset which is distributed across the RAM of a cluster of machines. It is the primary data abstraction in Apache Spark and is often referred to as the Spark Core.

The features of RDDs are:

- **Resilient**, i.e. fault-tolerant with the help of RDD lineage graph and so able to re-compute missing or damaged partitions due to node failures.
- **Distributed** with data residing on multiple nodes in a cluster.
- **Dataset** is a collection of partitioned data with primitive values or values of values, e.g. tuples or other objects (that represent records of the data you work with).

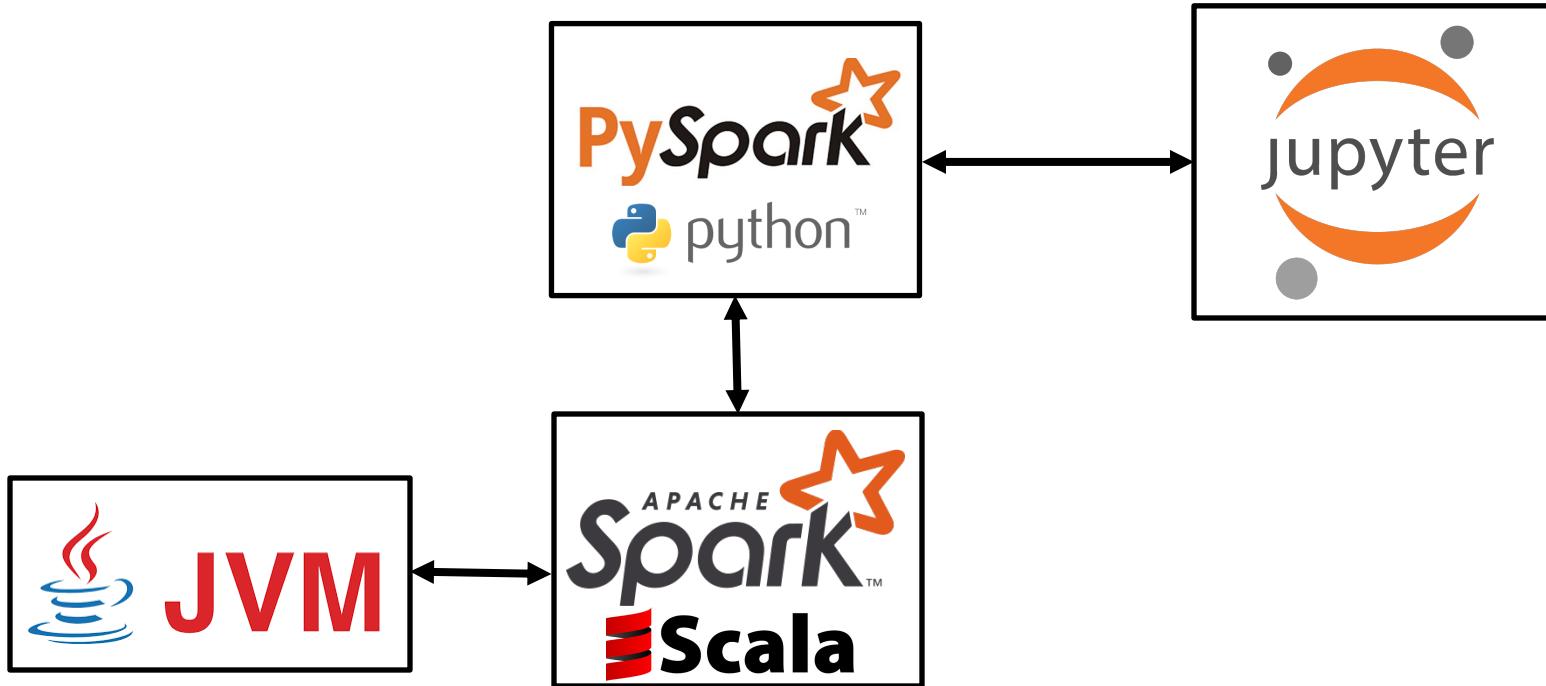
Introduction to PySpark

PySpark



- Even though the Spark toolkit was written in **Scala**, a language that compiles to byte code for the **Java Virtual Machine (JVM)**, as such numerous implementations or wrappers have been developed for R, Java, SQL and of course Python!
- It was made possible by **Py4j** which enables Python programs running in a Python interpreter to dynamically access Java objects in a JVM. As such, PySpark is API that allows us to interface with RDDs in Python
- As such, Python users can now work with **RDDs** in Python programming language also

PySpark Overview



PySpark in Industry

- **Netflix** has used PySpark internally to power many of their backend machine learning tasks (apparently almost one **trillion** per day!)
- The **healthcare** industry has used PySpark to perform analytics including Genome Sequencing
- The **financial** sector has made full use of PySpark for in-house trading, banking, credit risk and insurance use cases.
- **Retail and E-commerce** – Literally begs the use of PySpark given that these businesses have millions and millions of sales and retail data in their data warehouses. Both **eBay** and **Alibaba** are known to use PySpark.



RDDs, Transformations, Actions, Lineage Graphs & Jobs



RDDs (Resilient Distributed Data) in Python

- Loading data with PySpark creates an **RDD object**
- It is **immutable** (means once it's made it can't be altered)
- **RDDs do not have a schema.**
- An RDD object can then run any of the methods accessible to that object.



Lazy Evaluation and Pipelining

- Spark's RDD implementation allows us to evaluate code "lazily".
- A regular compiler (like Python) sequentially evaluates each expression it comes across. A lazy compiler doesn't continually evaluate expressions.,
- Pipelining is how we chain together calculations in Spark and understanding it is critical in working with Spark.



Types of Spark Methods

- **Transformations:**
 - `map()`
 - `reduceByKey()`
- **Actions:**
 - `take()`
 - `reduce()`
 - `saveAsTextFile()`
 - `collect()`



Transformations

- Transformations are one of the methods you can perform on an RDD in Spark.
- They **are lazy operations** that create one or more new RDDs (because RDDs are **immutable**, they can't be altered in any way once they've been created)
- Transformations take an RDD as an input and apply some function on them and outputs one or more RDDs.
- Let's talk about lazy evaluation - as the Scala compiler comes across each Transformation, it doesn't actually build any new RDDs yet. Instead, it **constructs a chain (or pipeline)** of hypothetical RDDs that would result from those Transformations which will only be evaluated once an **Action** is called.
- This chain of hypothetical, or "child", RDDs are all connected logically back to the original "parent" RDD, this concept is called the **lineage graph**.



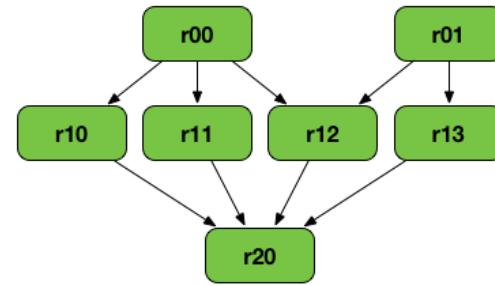
Actions

- Actions are any operations on RDDs that do not produce an RDD output
- Typically these include operations such as getting a count, max, min etc.



Lineage Graphs

- Remember the **chain or pipeline** constructed due to our Lazy Evaluation? These were **Transformation** operations that are only evaluated when an **Action** is called. This chain of hypothetical, or “child”, RDDs are all connected logically back to the original “parent” RDD, this concept is called the **lineage graph**.
- A **lineage graph** outlines a “logical execution plan”. The compiler begins with the earliest RDDs that aren’t dependent on any other RDDs, and follows a logical chain of Transformations until it ends with the RDD that an Action is called on
- Lineage Graphs are the drivers of Spark’s fault tolerance. If a Node fails, the information about what that node was supposed to do is held in the lineage graph and thus can be done elsewhere.

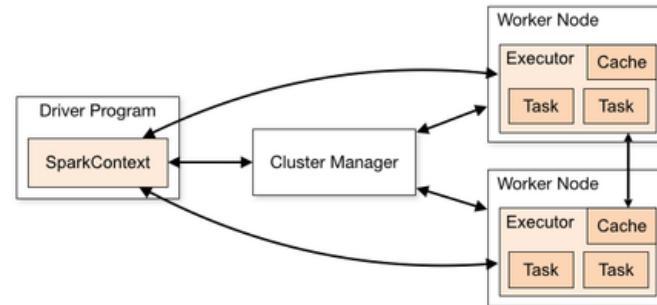


Visualization of example lineage graph;
r00, r01 are parent RDDs, r20 is final
RDD (source Jacek Laskowski -
<https://jaceklaskowski.gitbooks.io/maste-ring-apache-spark/spark-rdd-lineage.html#logical-execution-plan>)



Spark Applications and Jobs

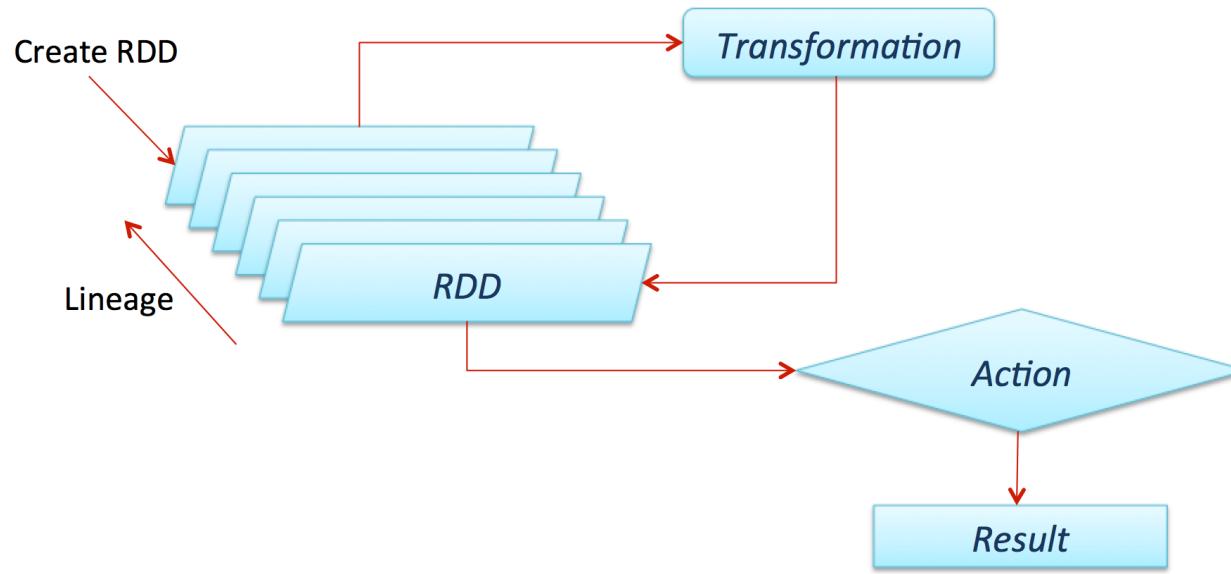
- In Spark, whenever processing needs to be done, there is a **Driver** process that is in charge of taking the user's code and converting it into a set of multiple tasks.
- There are also **executor** processes, each operating on a separate node in the cluster, that are in charge of running the tasks, as delegated by the driver.
- Each **driver** process has a set of **executors** that it has access to in order to run tasks.
- A **Spark application** is a user built program that consists of a driver and that driver's associated executors.
- A **Spark job** is task or set of tasks to be executed with executor processes, as directed by the driver.



Visualization of Spark Architecture (from Spark API
- <https://spark.apache.org/docs/latest/cluster-overview.html>)



Spark Overview



Simple Data Cleaning in PySpark

Machine Learning in PySpark

Customer Lifetime Value (CLV)



Customer Lifetime Value

- Why is CLV Important?
- Suppose our Marketing Department has told us the **Customer Acquisition Cost** (Marketing Spending divided by the number of new customers) is \$100 per customer.
- How do you know if that's worth it?
- That's where CLV comes in. we need to determine what the expected lifetime value a customer has to our business.



Customer Lifetime Value Defined

Definition - The present value of the expected sum of discounted cash flows of an individual customer.

This effectively means, it's the total purchases made (cash flow) over the lifetime of that customer.



Customer Lifetime Value Application

- For CLV to be useful we need to **accurately predict how much the customer will spend in future**
- CLV in our situation applies to non-contractual customers who have a **continuous opportunity to purchase**
- **Contractual customers** are subscription based customers
- **Discrete** (opposite of continuous in our case) has limited windows of purchase opportunity e.g. concert tickets or seasonal businesses



Benefits of knowing your CLV

- Determine the traits of your most valuable customers and find similar customers
- Know how much you should be spending to acquire a particular type of customer
- Push the marketing channels that bring you your most valuable customers
- Use your best customers for market research and product feedback



What CLV is Not!

- There is a lot of misconception about CLV, with many marketing texts and tutorials defining it's calculation as:

"CLV is calculated by multiplying the Average Order Value by number of Expected Purchases and Time of Engagement."

Why is this wrong?

- CLV isn't a calculation per se, it's a prediction and this definition is leads one to think it's simply the spend of a customer over time (based only on past purchases).
- We're interested in the Expected Value of a customer's returns



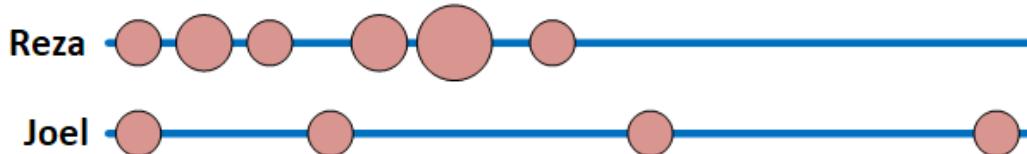
Another CLV Pitfall

- Another common mistake used when determining CLV is that it is calculated over all customers.
- This neglects the fact that we have different types of customers.



Another CLV Pitfall Example

- Imagine we have two customers, Reza and Joel.
- Reza discovered our business and liked it a lot, as such made many frequent purchases early on, but then lost interest and stopped buying
- Joel, buys less frequently but is still a regular customer.



- If we based on our CLV on purchase frequency and average order value, we'd be misled into thinking Reza is a more valuable customer.
- This type of analysis failed to consider that Reza churned.

Buy-til-you-die (BTYD) models



The buy-til-you-die model

- In 1987 by researchers at Wharton and Colombia developed a model called the Pareto/NBD (Negative Binomial Distribution) that could be used for estimating the number of future purchases a customer will make.
- This was the foundation of buy-til-you-die (BTYD) models.



BTYD Method

- BTYD models predict purchasing activity of a customer using two stochastic probabilistic models.
 1. The probability of customer making a repeat purchase
 2. The probability of a customer churns or 'dies'
- To get the above we need two pieces of information:
 1. Recency – the time since a customers last purchase
 2. Frequency - the number of repeat purchases placed by that customer in the given time period



Using BTYD to Estimate Expected No. of Future Purchases

- Our goal with the BTYD model is to estimate the expected number of future purchases each customer will place over a time period given their recency and frequency.
- This can expressed as:
- $E[X(t)] = \text{expected number of transactions in time of length } t$
(given a customers's recency and frequency)



Obtaining a Customers Residual Lifetime Value

- Once we have the **Expected number of future purchases** for each customer, we can multiply it by that customers average order value to get their **Residual Lifetime Value (RLV)**.

RLV = expected future purchases + expected average order value

- Residual lifetime value is the amount of additional value we can expect to collect from a customer over a given time period.**
- To get our CLV just add the sum of each customer's past purchases to their RLV!**



The Beta-Geometric/Negative Binomial Distribution

- In 2003, Peter Fader and Bruce Hardie published their seminal paper on a simplified version of the Pareto/NB.
- It was called the Beta-Geometric/NBD (BG/NBD) and it was much easier to implement (thanks to great packages created around it)



The `lifetimes` Module in Python

- Implementing the BG/NBD model in Python is relatively simple using the `lifetimes` module created by Cameron Davidson-Pilon, the former head of Data Science at Shopify.
- All the `lifetimes` value model needs is a simple transaction log, with a customer ID, date of order, and order amount.
- Let's go into Python and experiment with it!



Deploying Machine Learning Models in Production using Flask & Heroku

NLP dataset for
model creation

Machine Learning
CI/CD
Flask & Heroku



Deploying your Machine Learning Model

- Models don't just sit in your Jupyter notebooks
- They need to be deployed
- But how exactly do we make our models accessible to the world?

```
❶ 1 docs_new = ['Anti-aliasing was turned on using OpenGL', 'The earth was made in 7 days', 'Biden can become the President of U.S']
2
3 X_new_counts = count_vect.transform(docs_new)
4 X_new_tfidf = tfidf_transformer.transform(X_new_counts)
5 predicted = clf.predict(X_new_tfidf)
6 predicted

❷ array([1, 3, 4])

[ ] 1 dict = {"0": "atheism", "1": "computer graphics", "2": "medical science", "3": "christianity", "4": "politics"}
2
3 for p in predicted:
4     print(dict[str(p)])
```

computer graphics
christianity
politics



Cloud Deployments

- There are many ways to deploy models and servers to various cloud services
- For Example, using AWS's EC2 (a web service that provides secure, resizable compute capacity in the cloud)
- Or Azure by using a Web App from their App Service
- Or Google Cloud via their App Engine
- We're going to use Heroku

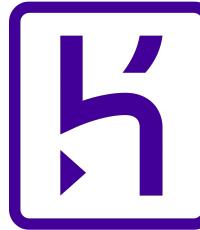


Google Cloud



HEROKU

Why Heroku?



HEROKU



- AWS and the others requires too much manual configuration and also a credit card to sign up
- Heroku allows users free access to test their platform without a credit card
- It's an integrated CI/CD platform (Continuous Integration, Continuous Deployment)
- Connects directly to your GitHub account and automatically deploys



A bit about Continuous Integration/Continuous Deployment

- CI/CD allows development teams to deliver code changes more frequently and reliably
- CI/CD tools help store the environment-specific parameters that must be packaged with each delivery. CI/CD automation then performs any necessary service calls to web servers, databases, and other services that may need to be restarted or follow other procedures when applications are deployed.
- Tests are performed and need to be passed before deployed live



Steps to Deploy our Machine Learning Model

- Create a model and serialize it using pickle
- Create a Flask App that creates a local server to serve your ML Model
- Push Flask App to GitHub
- Link GitHub Repository to Heroku
- Deploy from Heroku



Deep Learning Recommendation Engines



Why Deep Learning for Recommendation Engines?

Neural Network Embeddings

- Embeddings are a way to represent discrete — categorical — variables as continuous vectors. They place similar entities closer to one another in the embedding space.
- To create embeddings, we need a neural network embedding model and a supervised machine learning task. The end outcome of our network will be a representation of each book as a vector of 50 continuous numbers.
- While the embeddings themselves are not that interesting — they are just vectors — they can be used for three primary purposes:
- Finding nearest neighbors in the embedding space
- As input to a machine learning model
- Visualization in low dimensions



US 2020 Presidential Election Simulation

FiveThirtyEight
dataset from
196 pollsters

EDA
Statistics



Covid-19 Data Analysis

Covid-19 Case
World Wide
Dataset

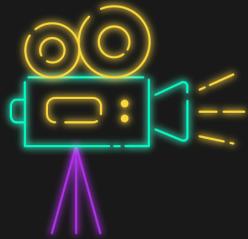
EDA
Statistics
Visualizations



Deep Learning Recommendation Systems

Wikipedia Movie
Scraped Dataset

Deep Learning
Recommendation
Systems



Streaming Movies Data Analysis

OTT Movie Dataset -
Netflix, Hulu, Amazon
Prime & Disney+

Data Analysis
Visualizations



Market Basket Analysis Apriori Algorithm

Grocery Dataset

Statistics
Visualizations



2009 vs 2014: Indian Election Analysis

Indian 2014 and 2019
Election Results
Datasets

Data Analysis
Visualizations



Supply-Chain for Shipping Data Analytics

Shipping Cargo
Dataset

Data Analysis
Visualizations



Olympics Analysis - The Greatest Olympians



Olympics and Winter
Olympics datasets

Data Analysis
Visualizations



Is Home Advantage Real? Soccer & Basketball Analysis

NBA Dataset & Scrapped data
from footballdatabase.com
league games

Statistics
Correlations
Visualizations

DATA SCIENCE, ANALYTICS & AI FOR BUSINESS & THE REAL WORLD™



Africa Economic, Banking & Systematic Crisis Data

Reinhart et. al's Global
Financial Stability
dataset

Statistics
Visualizations

DATA SCIENCE, ANALYTICS & AI FOR BUSINESS & THE REAL WORLD™



IPL Cricket Data Analysis

Indian Premier League
(Cricket)
Ball-By-Ball Cricket Data

Statistics
Visualizations



Pizza Restaurant Analytics

Pizza Restaurants and the
Pizza They Sell (US) -
Datafiniti's Business Database

Data Analysis
Visualizations



World Cup (Soccer) Prediction using Machine Learning

FIFA Rankings,
International Matches and
World Cup Schedule

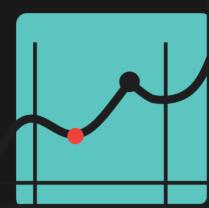
Data Analysis
Visualizations
Machine Learning



Brewery and Pub Data Analysis

Breweries and Brewpubs
in the USA - Datafiniti's
Business Database

Data Analysis
Visualizations



Stock Market Trading Bot with Reinforcement Learning

Quandl Stock Data for
Apple and Microsoft

Reinforcement
Learning



Predicting Diabetes using Healthcare Data

National Health and
Nutrition Examination
Survey (NHANES) data

Statistics
Machine Learning



Sales Demand Forecasting

Rossmann Store - sales
using store, promotion, and
competitor data

Timeseries Forecasting
Statistics
Visualizations



Forecasting Brent Oil Prices with Facebook's Prophet

Brent Oil price dataset

Timeseries Forecasting
Visualizations



Credit Card Fraud Detection

Credit cards transactions
for September 2013 by
european cardholder

Machine Learning
Data Analysis
Visualizations

DATA SCIENCE, ANALYTICS & AI FOR BUSINESS & THE REAL WORLD™



Airline Tweets Sentiment Analysis

US Airlines Tweets Dataset

Natural Language
Processing (NLP)
Visualizations



Amazon Review Summarizer

American Airline Tweets
Dataset

Natural Language
Processing (NLP)
Visualizations



Spam Detector

Spam Ham dataset of text
messages

Natural Language
Processing (NLP)
Machine Learning