

# Introduction to Data Science

Unit 1

# Course Contents

## **Introduction to Data Science** [10 Hrs.]

Introduction to data science, Applications of data science; Limitations of data science Commonly used tools in data science, their strengths and common use-cases: R/RStudio, Python/Pandas/Jupyter Notebooks, Excel/Tableau/PowerBI;

Data Science life-cycle/Common methodologies for data science: CRISP-DM, OSEMN Framework, TDSP lifecycle;

Review of statistics and probability: Probability distributions, compound events and independence. Statistics: Centrality measures, variability measures, interpreting variance. Correlation analysis: Correlation coefficients, autocorrelation

# Before we begin.... Let's understand the data first

- Data is generated in various sources



Mobile Apps



Computer Applications



Point of Sale



Stock Market

Today	THU	FRI	SAT	SUN	MON	TUE
 Sunny	 Sunny	 More sun than clouds	 Passing clouds	 More sun than clouds	 Scattered clouds	 Scattered clouds
66° 43°	69° 39°	72° 44°	78° 47°	78° 53°	77° 52°	75° 55°

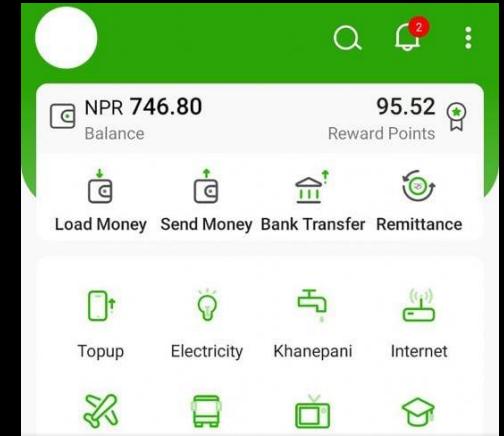
Weather



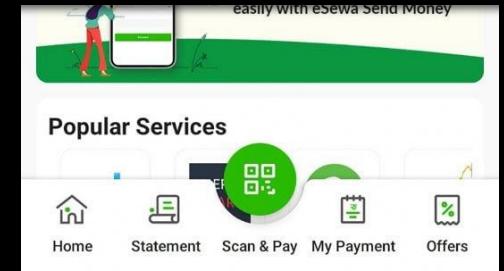
Security Surveillance

# Before we begin.... Let's understand the data first (contd.)

- Also,



& many more



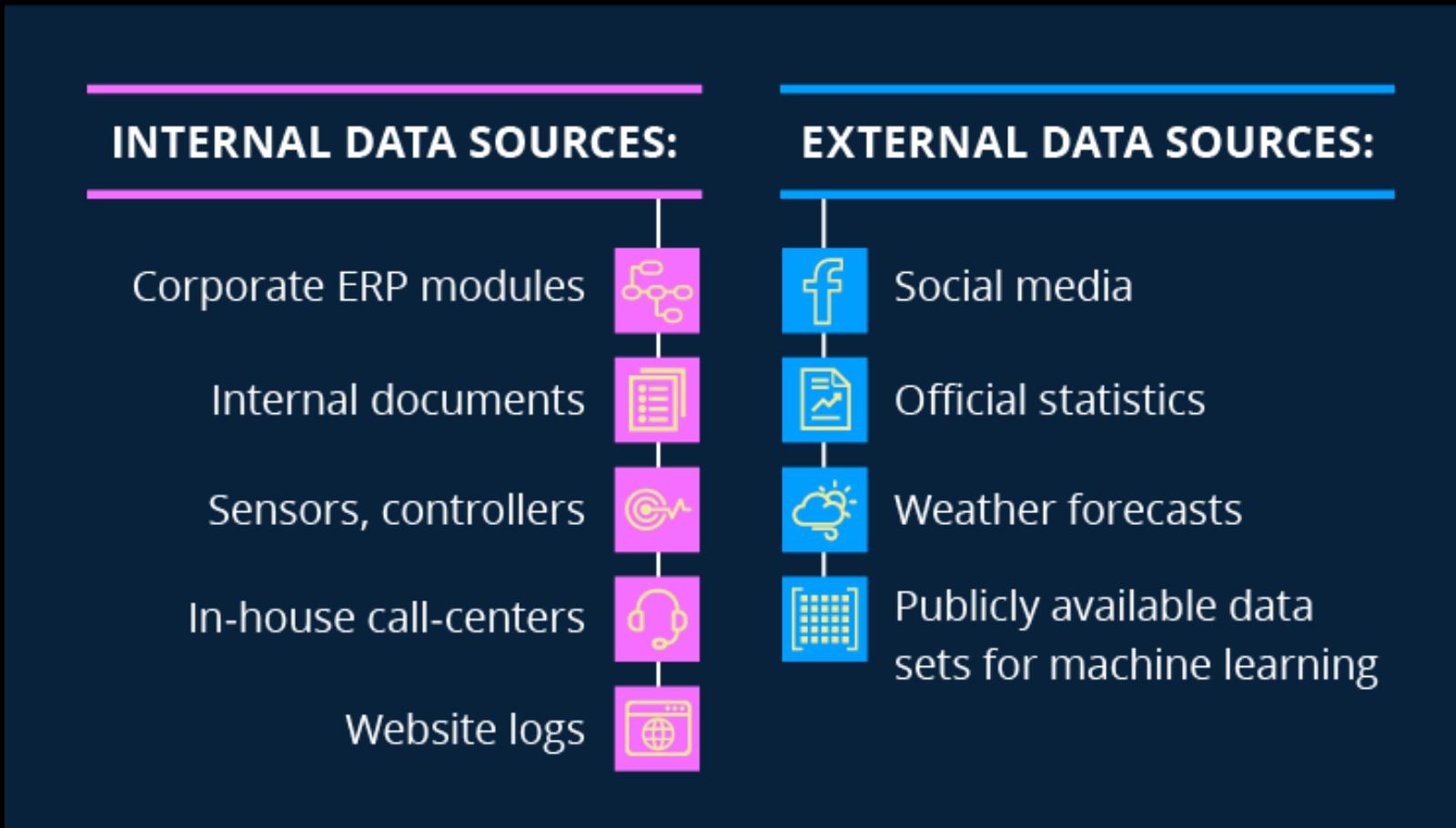
Astronomy

Medical Diagnosis

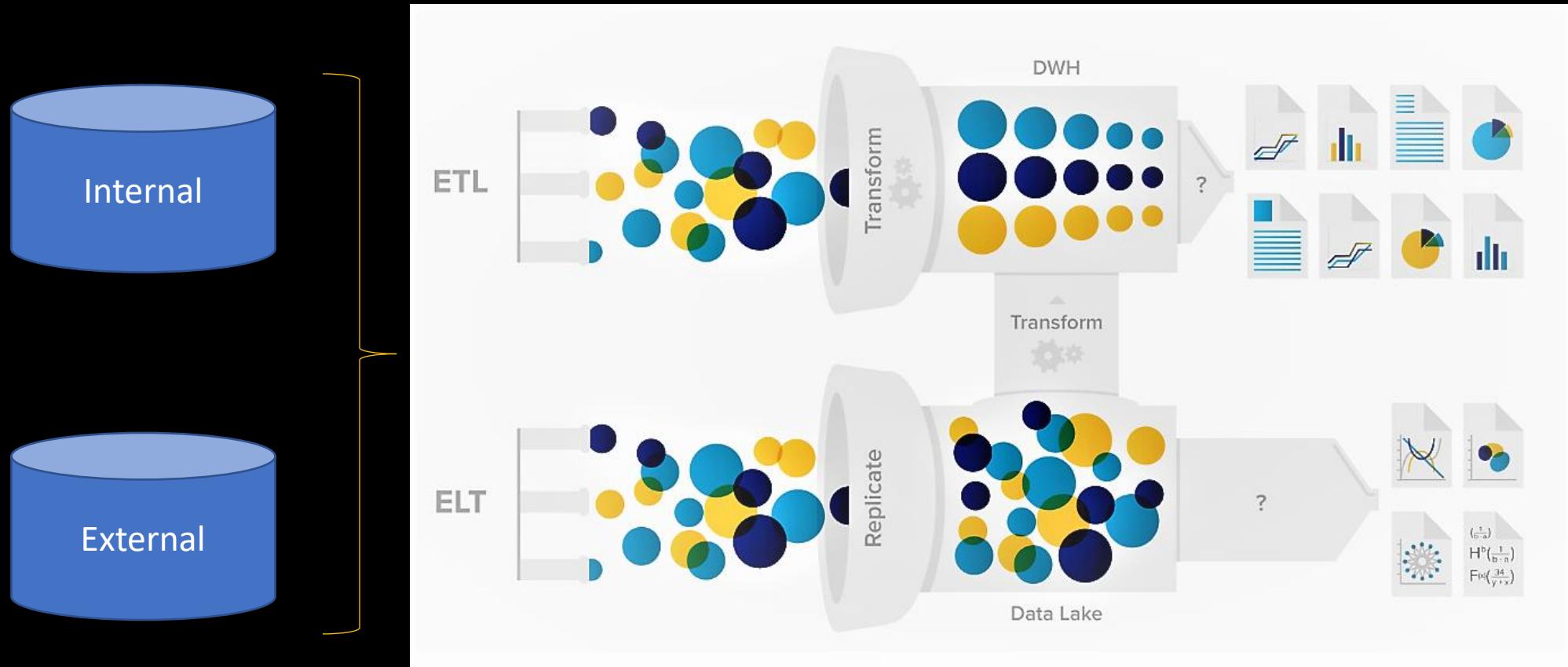
Scientific Study

Financial Transactions

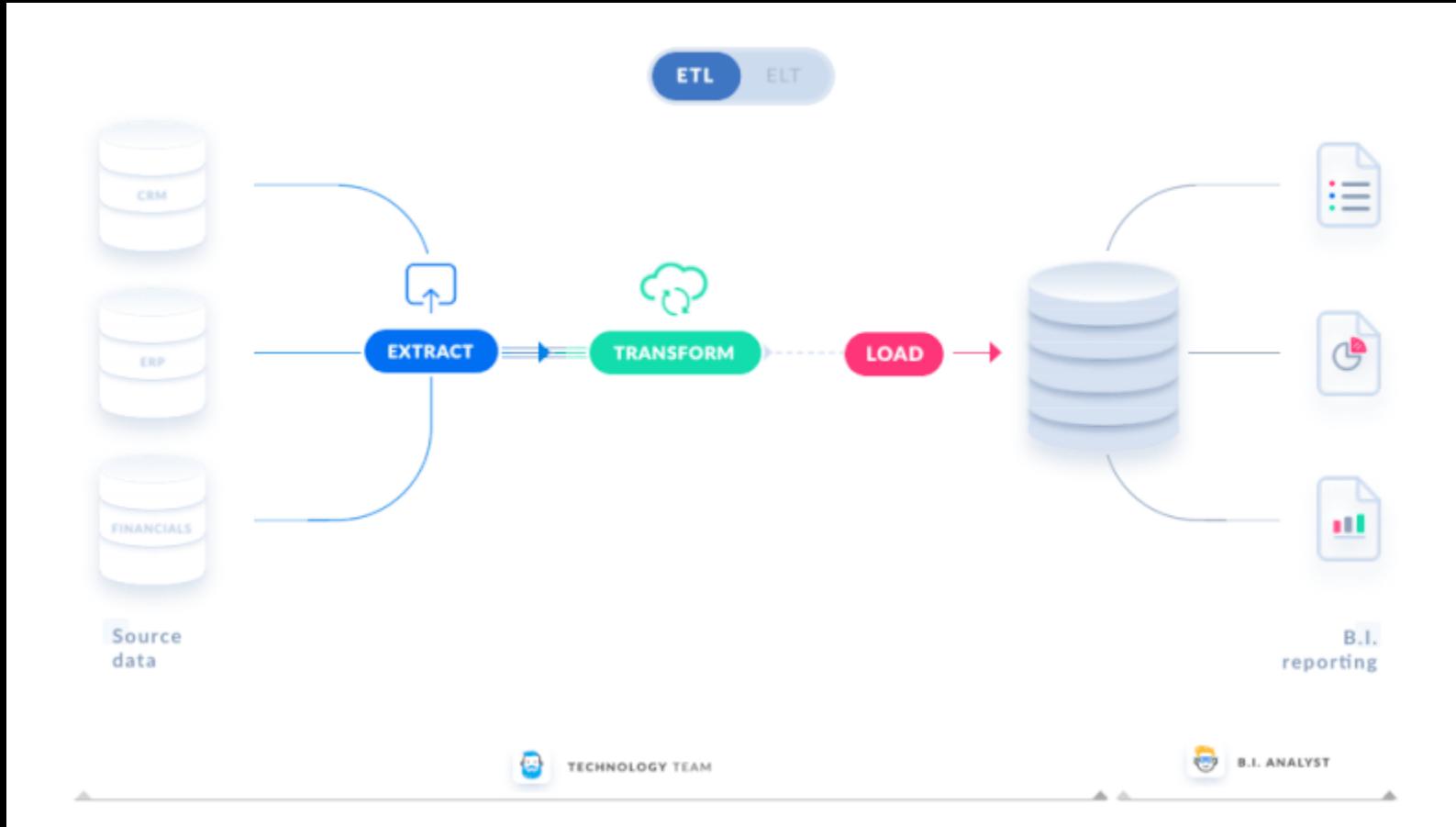
# And, we categorize all these data sources into:



# Data migrates from various sources to Data Warehouse (Data Lake)



# ETL vs ELT tools



# Data Warehouse

- A data warehouse is a huge collection of business data used to help an organization make decisions.
- The large amount of data in data warehouses comes from different sources such as internal applications such as marketing, sales, and finance; customer-facing apps; and external partner systems, among others.
- On a technical level, a data warehouse periodically pulls data from those apps and systems; then, the data goes through formatting and import processes to match the data already in the warehouse.

# Data Warehouse (contd.)

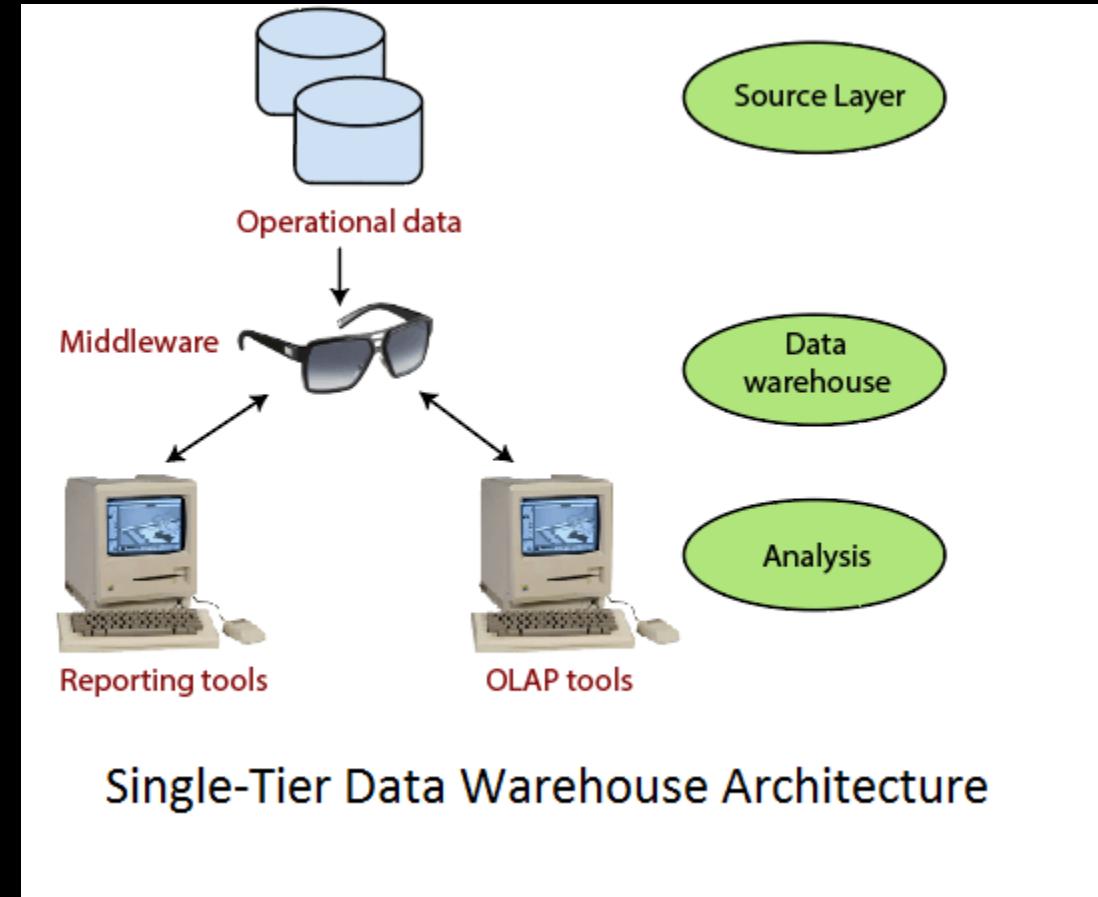
- The data warehouse stores this processed data so it's ready for decision makers to access.
- How frequently data pulls occur, or how data is formatted, etc., will vary depending on the needs of the organization.

# Characteristics of Datawarehouse

1. **Subject Oriented**: focused on specific subject area
2. **Integrated**: Integrates data from multiple sources
3. **Time Variant**: Stores historical data
4. **Non Volatile**: Permanent Storage

# Data Warehouse Architecture

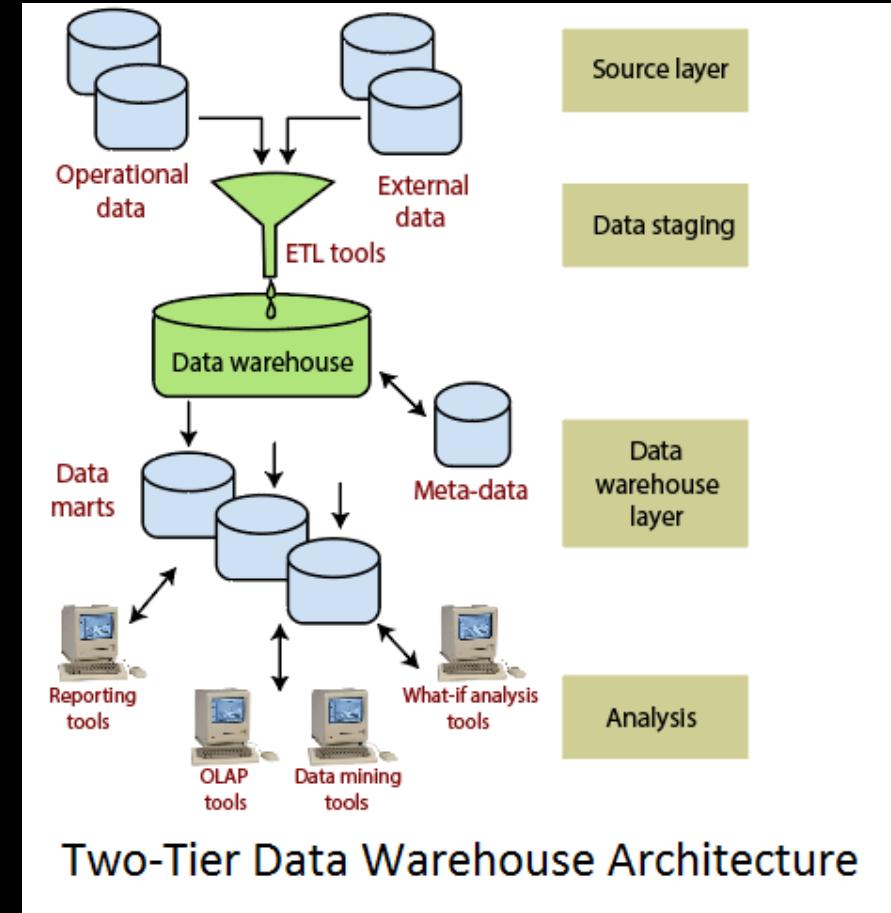
Single-Tier architecture is not periodically used in practice. Its purpose is to minimize the amount of data stored to reach this goal; it removes data redundancies.



# Data Warehouse Architecture (contd.)

Although it is typically called two-layer architecture to highlight a separation between physically available sources and data warehouses, in fact, consists of four subsequent data flow stages:

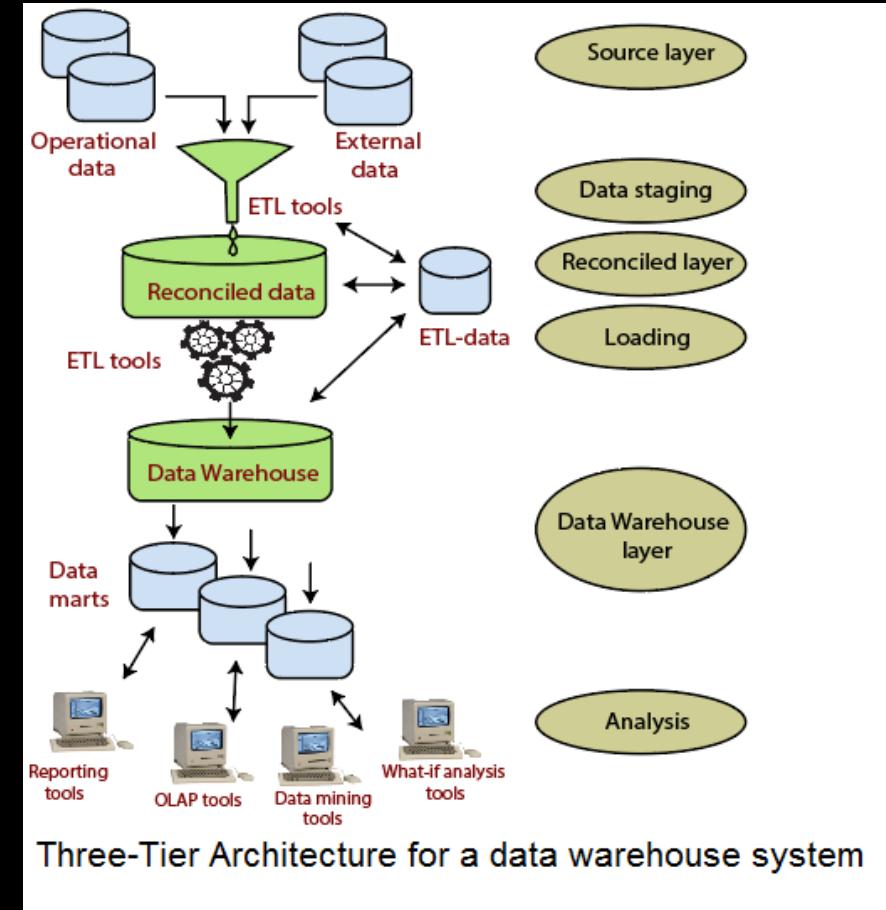
- a) Source layer
- b) Data Staging
- c) Data Warehouse layer
- d) Analysis



# Data Warehouse Architecture (contd.)

The three-tier architecture consists of the source layer (containing multiple source system), the reconciled layer and the data warehouse layer (containing both data warehouses and data marts). The reconciled layer sits between the source data and data warehouse.

The main advantage of the reconciled layer is that it creates a standard reference data model for a whole enterprise. At the same time, it separates the problems of source data extraction and integration from those of data warehouse population. In some cases, the reconciled layer is also directly used to accomplish better some operational tasks, such as producing daily reports that cannot be satisfactorily prepared using the corporate applications or generating data flows to feed external processes periodically to benefit from cleaning and integration.



# Data Lake

- A data lake is a central storage repository that holds big data from many resources in a raw, granular format.
- It can store structured, semi-structured or unstructured data, which means data can be kept in a more flexible format for future use.
- When storing data, a data lake associates it with identifiers and metadata tags for faster retrieval.
- The term “data lake” refers to the ad hoc nature of data in a data lake, as opposed to the clean and processed data stored in traditional data warehouse system.

# Data Lake (contd.)

- A data lake works on a principle called **schema-on-read**.
- This means that there is no predefined schema into which data needs to be fitted before storage.
- Only when the data is read during processing is it parsed and adapted into a schema as needed.
- This feature saves a lot of time that's usually spent on defining a schema. This also enables data to be stored as is, in any format.

Data scientists can access, prepare, and analyze data faster and with more accuracy using data lakes. For analytics experts, this vast pool of data — available in various non-traditional formats — provides the opportunity to access the data for a variety of use cases like sentiment analysis or fraud detection.

# Data lake vs data warehouse - Similarities

- A data lake and a data warehouse are similar in their basic purpose and objective, which make them easily confused:
  - Both are storage repositories that consolidate the various data stores in an organization.
  - The objective of both is to create a one-stop data store that will feed into various applications.

# Data lake vs data warehouse - Differences

- **Schema-on-read vs schema-on-write**

- The schema of a data warehouse is defined and structured before storage (schema is applied while writing data). A data lake, in contrast, has no predefined schema, which allows it to store data in its native format.
- In a data warehouse most of the data preparation usually happens before processing. In a data lake, it happens later, when the data is actually being used.

- **Complex vs simple user accessibility**

- As data is not organized in a simplified form before storage, a data lake often needs an expert with a thorough understanding of the various kinds of data and their relationships, to read through it. A data warehouse, in contrast, is easily accessible to both tech and non-tech users due its well-defined and documented schema. Even a new member on the team can begin to use a warehouse quickly.

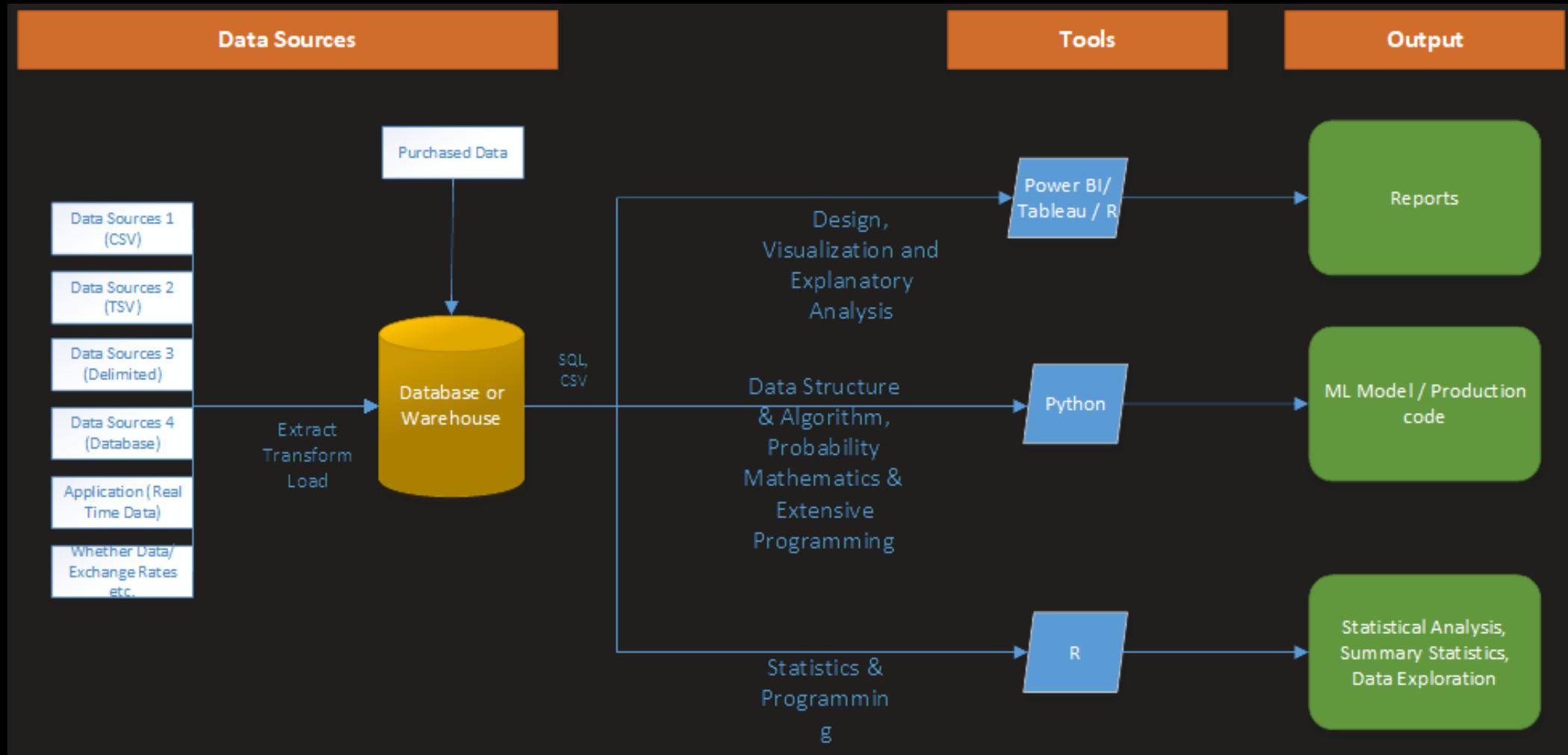
- **Flexibility vs rigidity**

- With a data warehouse, not only does it take time to define the schema at first, it also takes considerable resources to modify it when requirements change in the future. However, data lakes can adapt to changes easily. Also, as the need for storage capacity increases, it is easier to scale the servers on a data lake cluster.

# Data lake vs data warehouse - Differences

Characteristics	Data Warehouse	Data Lake
Data	Relational data from transactional systems, operational databases, and line of business applications	All data, including structured, semi-structured, and unstructured
Schema	Often designed prior to the data warehouse implementation but also can be written at the time of analysis  (schema-on-write or schema-on-read)	Written at the time of analysis (schema-on-read)
Price/Performance	Fastest query results using local storage	Query results getting faster using low-cost storage and decoupling of compute and storage
Data quality	Highly curated data that serves as the central version of the truth	Any data that may or may not be curated (i.e. raw data)
Users	Business analysts, data scientists, and data developers	Business analysts (using curated data), data scientists, data developers, data engineers, and data architects
Analytics	Batch reporting, BI, and visualizations	Machine learning, exploratory analytics, data discovery, streaming, operational analytics, big data, and profiling

# Thus,



# Introduction to Data Science

- Over the past few years, there's been a lot of hype in the internet about “data science” and “Big Data.”
- But, what actually “Data Science” is? And what does “Big Data” means?
- How “Data Science” and “Big Data” are related?
- Is data science the science of Big Data?
- Is data science only the stuff going on in companies like Google and Facebook and tech companies?
- Why do many people refer to Big Data as crossing disciplines (astronomy, finance, tech, etc.) and to data science as only taking place in tech? Just how big is big? Or is it just a relative term?

# Introduction to Data Science (contd.)

## Is it new Age Thing?

- Statisticians already feel that they are studying and working on the “Science of Data.” That’s their bread and butter.
- Many of the algorithms ( *e.g. linear regression, logistic regression, Bayesian Statistics, and even neural network* ) we used today were discovered long back in the past.
- However, many of the methods and techniques we’re using—and the challenges we’re facing now—are part of the evolution of everything that’s come before.

# Introduction to Data Science (contd.)

## Why now?

- We have massive amounts of data about many aspects of our lives, and, simultaneously, an abundance of inexpensive computing power.
- *Shopping, communicating, reading news, listening to music, searching for information, expressing our opinions*—all this is being tracked online, as most people know.
- What people might not know is that the “**datafication**” of our **offline behavior** has started as well, mirroring the online data collection revolution.
- Put the two together, and there’s a lot to learn about our behavior and, by extension, who we are as a species.

# Introduction to Data Science (contd.)

## Why now?

- It's not just Internet data, though—it's finance, the medical industry, pharmaceuticals, bioinformatics, social welfare, government, education, retail, and the list goes on.
- But it's not only the massiveness that makes all this new data interesting (or poses challenges).
- It's that the data itself, often in real time, becomes the building blocks of data products.

# Introduction to Data Science (contd.)

## Why now?

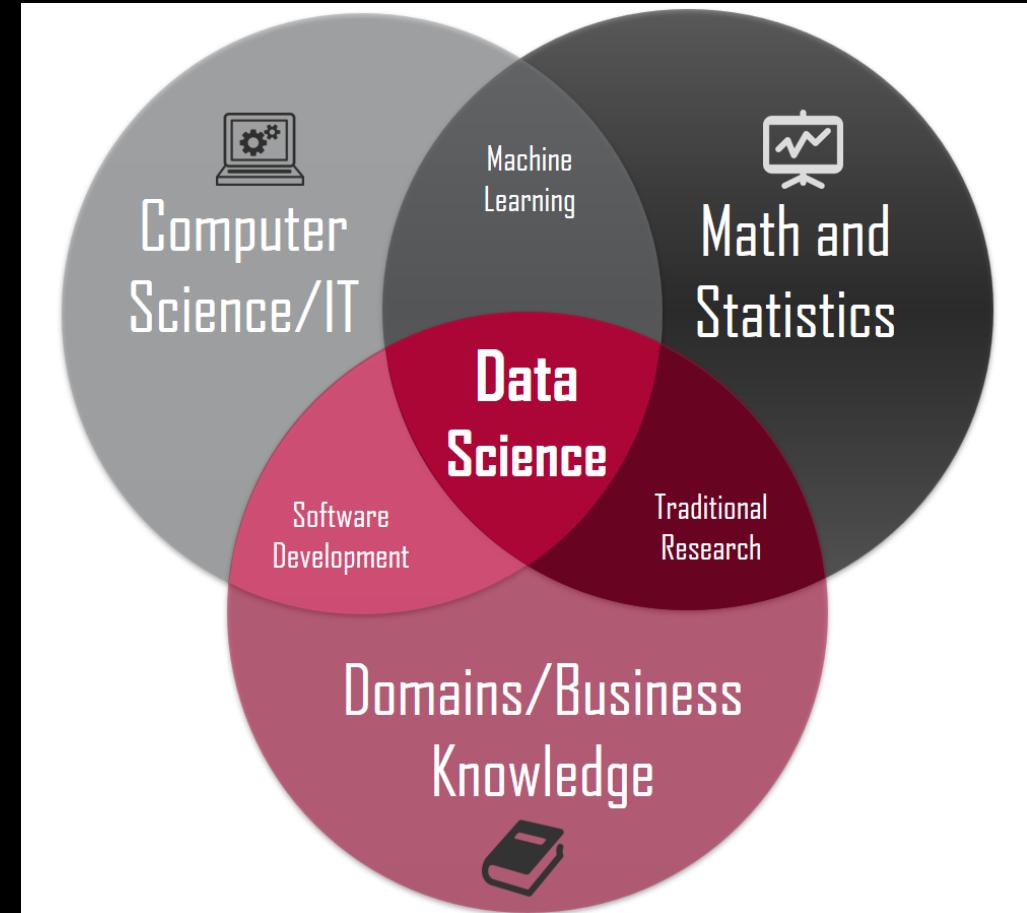
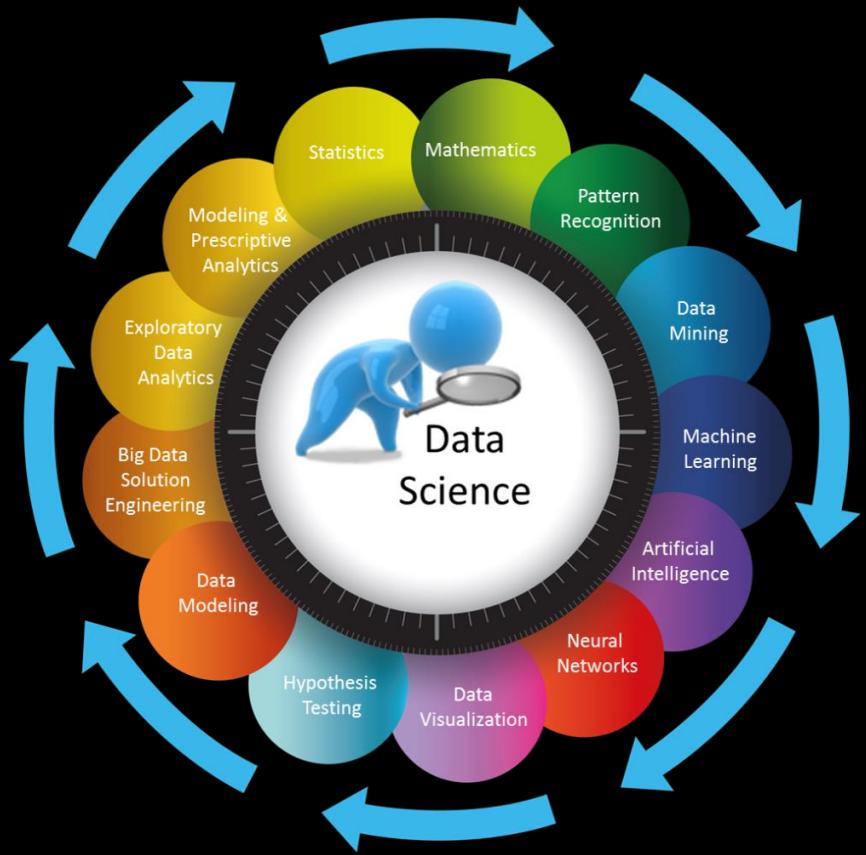
- We're witnessing the beginning of a massive, culturally saturated feedback loop where our behavior changes the product and the product changes our behavior.
- Technology makes this possible: infrastructure for large-scale data processing, increased memory, and bandwidth, as well as a cultural acceptance of technology in the fabric of our lives.

# Introduction to Data Science (contd.)

## So, what is data science?

- Is it new, or is it just statistics or analytics rebranded? Is it real, or is it pure hype? And if it's new and if it's real, what does that mean?
- This is an ongoing discussion, and many people have their own definition, their own explanations.

# Introduction to Data Science (contd.)



*Drew Conway's Venn diagram of data science*

# Introduction to Data Science (contd.)

- Metamarket CEO Mike Driscoll defines data science as:
  - Data science, as it's practiced, is a blend of Red-Bull-fueled hacking and espresso-inspired statistics.
  - But data science is not merely hacking—because when hackers finish debugging their Bash one-liners and Pig scripts, few of them care about non-Euclidean distance metrics.
  - And data science is not merely statistics, because when statisticians finish theorizing the perfect model, few could read a tab-delimited file into R if their job depended on it.
  - Data science is the civil engineering of data. Its acolytes possess a practical knowledge of tools and materials, coupled with a theoretical understanding of what's possible.

# Introduction to Data Science (contd.)

- Who is data scientist?

A data scientist is someone who knows how to extract meaning from and interpret data, which requires both tools and methods from statistics and machine learning, as well as being human. She spends a lot of time in the process of collecting, cleaning, and munging data, because data is never clean. This process requires persistence, statistics, and software engineering skills—skills that are also necessary for understanding biases in the data, and for debugging logging output from code.

# Introduction to Data Science (contd.)

- Who is data scientist?

A **chief data scientist** should be **setting the data strategy of the company**, which involves a variety of things: setting everything up from the engineering and infrastructure for collecting data and logging, to privacy concerns, to deciding what data will be user-facing, how data is going to be used to make decisions, and how it's going to be built back into the product. She should manage a team of engineers, scientists, and analysts and should communicate with leadership across the company, including the CEO, CTO, and product leadership. She'll also be concerned with patenting innovative solutions and setting research goals.

# Introduction to Data Science (contd.)

- Who is data scientist?

Once she gets the data into shape, a crucial part is exploratory data analysis, which combines visualization and data sense. She'll find patterns, build models, and algorithms—some with the intention of understanding product usage and the overall health of the product, and others to serve as prototypes that ultimately get baked back into the product. She may design experiments, and she is a critical part of data driven decision making. She'll communicate with team members, engineers, and leadership in clear language and with data visualizations so that even if her colleagues are not immersed in the data themselves, they will understand the implications.

# Applications of Data Science

## 1. Finance

- Risk Management
- Financial Analysis
- Stock price Prediction

## 2. Retailing

- Inventory Replenishment
- Customer Segmentation
- Promotion campaigning
- Sales Campaigning
- Demand and Supply projection

## 3. Banking

- Resource Utilization
- Fraud Detection
- Loan Management
- Predictive Analysis
- Customer Segmentation

## 4. Manufacturing

- Resource Utilization
- Monitoring of energy costs and optimize production hour.
- Identify potential problems through analysis of continuous stream of data

# Applications of Data Science (contd.)

## 5. Health

- Medical Image Analysis
- Genetics and Genomics
- Drug Discovery
- Predictive Modeling for Diagnosis
- Health bots or virtual assistants

## • Other Applications

- Fraud and Risk Detection
- Personalized Advertising
- Content Recommendation
- Advanced Image Recognition
- Airline Route Planning
- Gaming etc.

# Limitations of Data Science

- It's a Blurry Team
- Mastering Data Science is near to impossible.
- Large amount of Domain knowledge is required
- Arbitrary data may yield unexpected result.
- Data alone cannot guide decisions because they are a means of orienting you toward
- Difficulty in devising a conclusion if dataset is smaller.
- Garbage in Garbage Out

# Commonly used tools

- Python
- R
- Tableau
- Power BI
- Excel / Google Sheets
- Jupyter
- Weka
- Apache Spark
- Apache Hadoop
- MATLAB
- Julia
- Scala
- Azure/Google Cloud/AWS
- MLOps
- AutoML

# Commonly used library

## Python

- NumPy
- SciPy
- Pandas
- Matplotlib
- Scikit-Learn
- Tensorflow
- Pytorch
- BeautifulSoup
- NLTK
- OpenCV
- ScraPY
- XGBoost
- Seaborn

## R

- Dplyr
- Tidyr
- Readr
- Stringr
- Ggplot2
- Lubridate
- Jsonlite
- BioConductor
- Shiny
- Knitr
- Caret
- Rmarkdown

# Basic Steps for Data Science



# Let's make hands dirty

Task 01 – Build Classification Model to classify Survival in Titanic

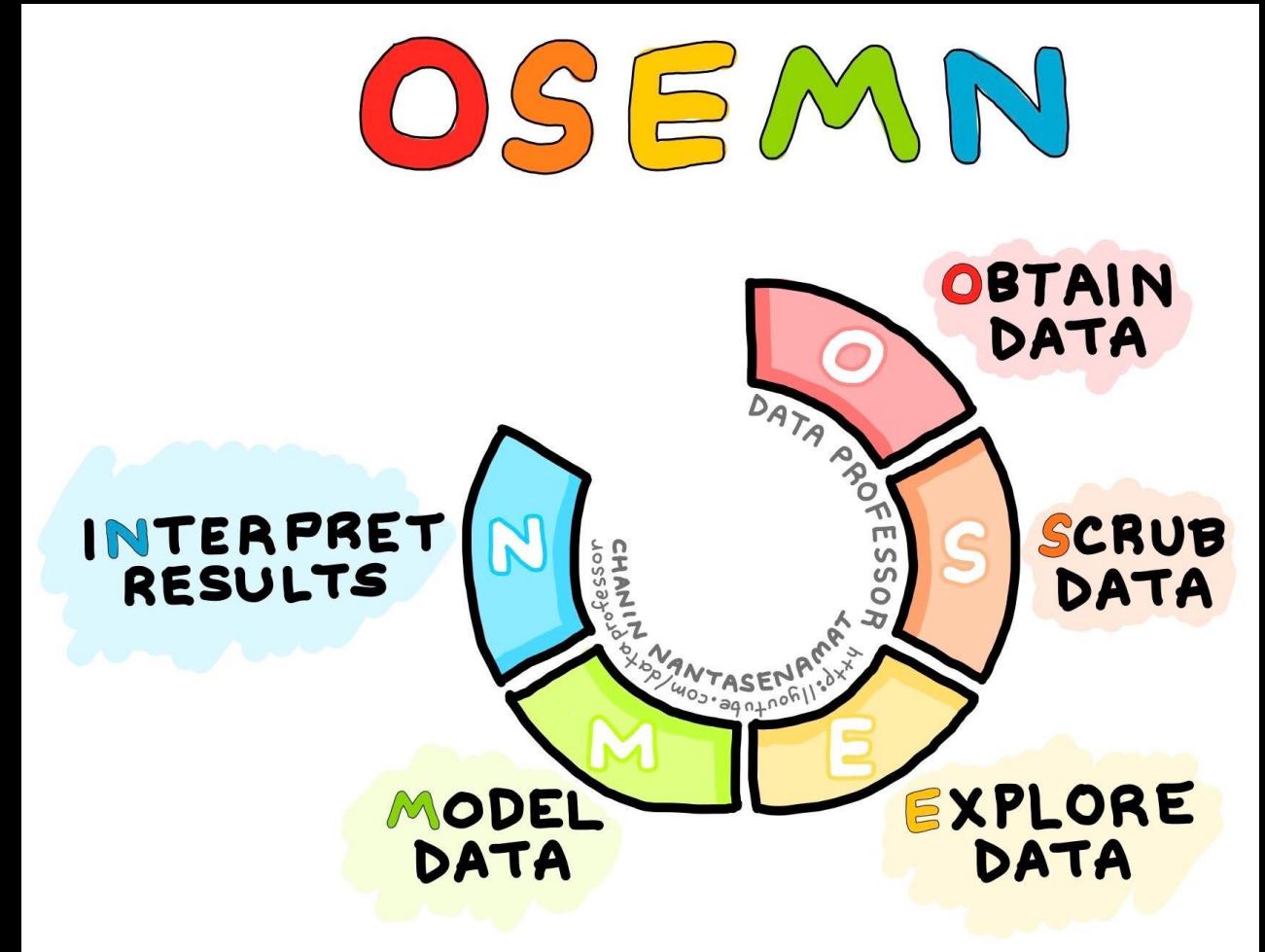
URL - [shorturl.at/grxBJ](http://shorturl.at/grxBJ)

# Data Science Life Cycle

# OSEMN Framework

# OSEMN Framework

- Obtain
- Scrub
- Explore
- Model
- iNterpret



# OSEMN Framework (Contd.)

- Benefits:
  - Simple – It distills the complex process of a data science project into five clear steps. This is especially noteworthy given that this general process and the modern concept of data science were still new when Mason and Wiggins created OSEMN in 2010.
  - Catchy – OSEMН is Awesome!
  - Makes sense – The steps presented have a logical flow representative of the general data science life cycle.
  - Provides a shared understanding – OSEMН creates a taxonomy to help define how a data science project progresses.

# OSEMN Framework (Contd.)

- Shortcomings:
  - **Misses business understanding** – The framework starts with Obtain which ignores the key base questions that should come first, namely: “Should I invest time on this project?” and “What outcome am I trying to drive?”
  - **Doesn’t consider deployment** – OSEMN implicitly assumes that you are delivering a one-time output. In reality, you often need to deploy a model in a production system so that it continues to provide value over time.
  - **Ignores teamwork** – Data science is increasingly a team sport. Yet, OSEMN ignores the broader team aspect of modern projects.
  - **It’s linear** – OSEMN proceeds in a waterfall-like manner with each phase following the other. In reality, you often switch back and forth between phases as needed. Moreover, you will want frequent decision points where you re-assess and adjust your plan based on recent learnings.

# Team Data Science Process (TDSP)

# TDSP

- The Team Data Science Process (TDSP) is an agile, iterative data science methodology to deliver predictive analytics solutions and intelligent applications efficiently.
- TDSP helps improve team collaboration and learning by suggesting how team roles work best together.
- TDSP includes best practices and structures from Microsoft and other industry leaders to help toward successful implementation of data science initiatives.
- The goal is to help companies fully realize the benefits of their analytics program.

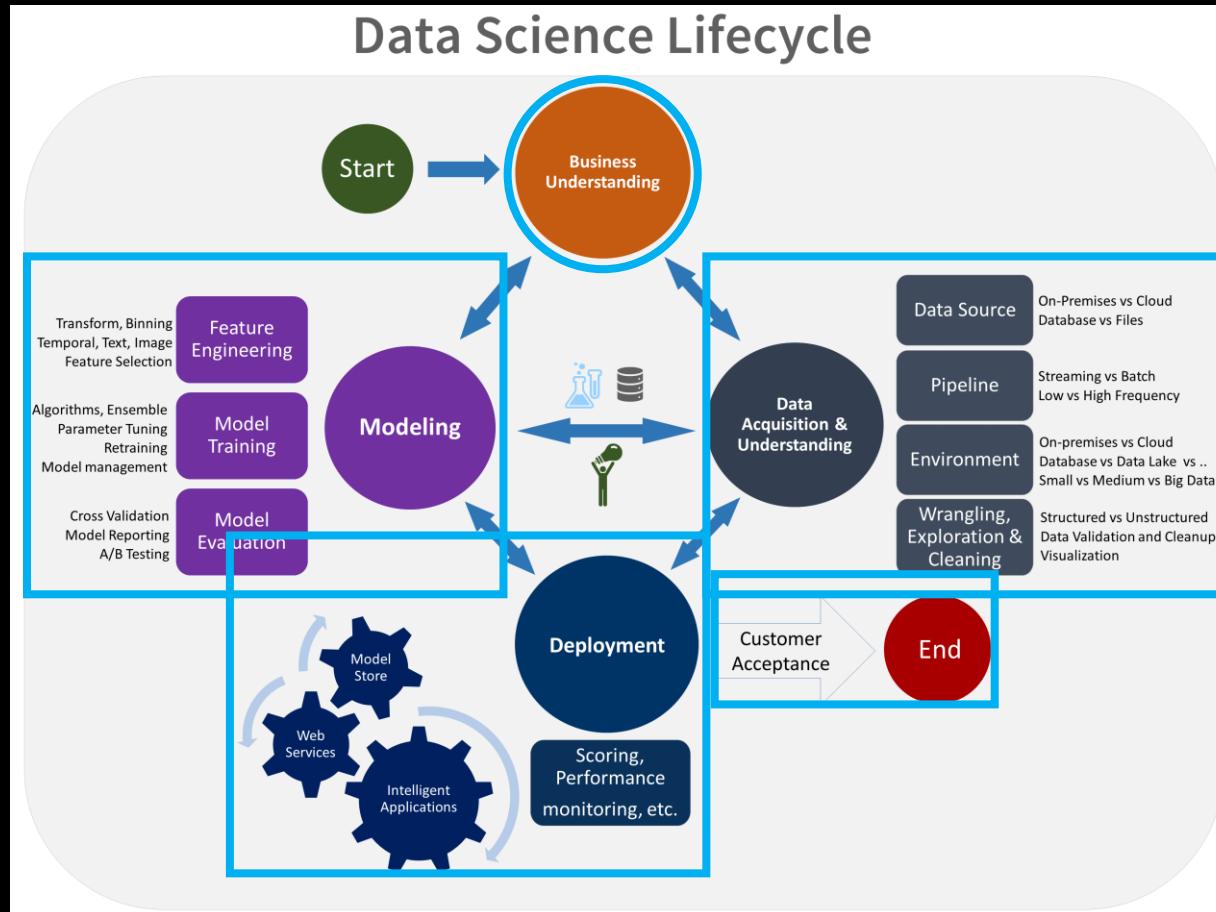
# TDSP (contd.)

- Key Components of TDSP
  - A data science lifecycle
  - A standardized project structure
  - Infrastructure and resources
  - Tools and utilities

# TDSP (contd.)

- Data Science Lifecycle
  - The Team Data Science Process (TDSP) provides a lifecycle to structure the development of your data science projects.
  - The lifecycle outlines the full steps that successful projects follow.
  - The lifecycle outlines the major stages that projects typically execute, often iteratively:
    - Business Understanding
    - Data Acquisition and Understanding
    - Modeling
    - Deployment

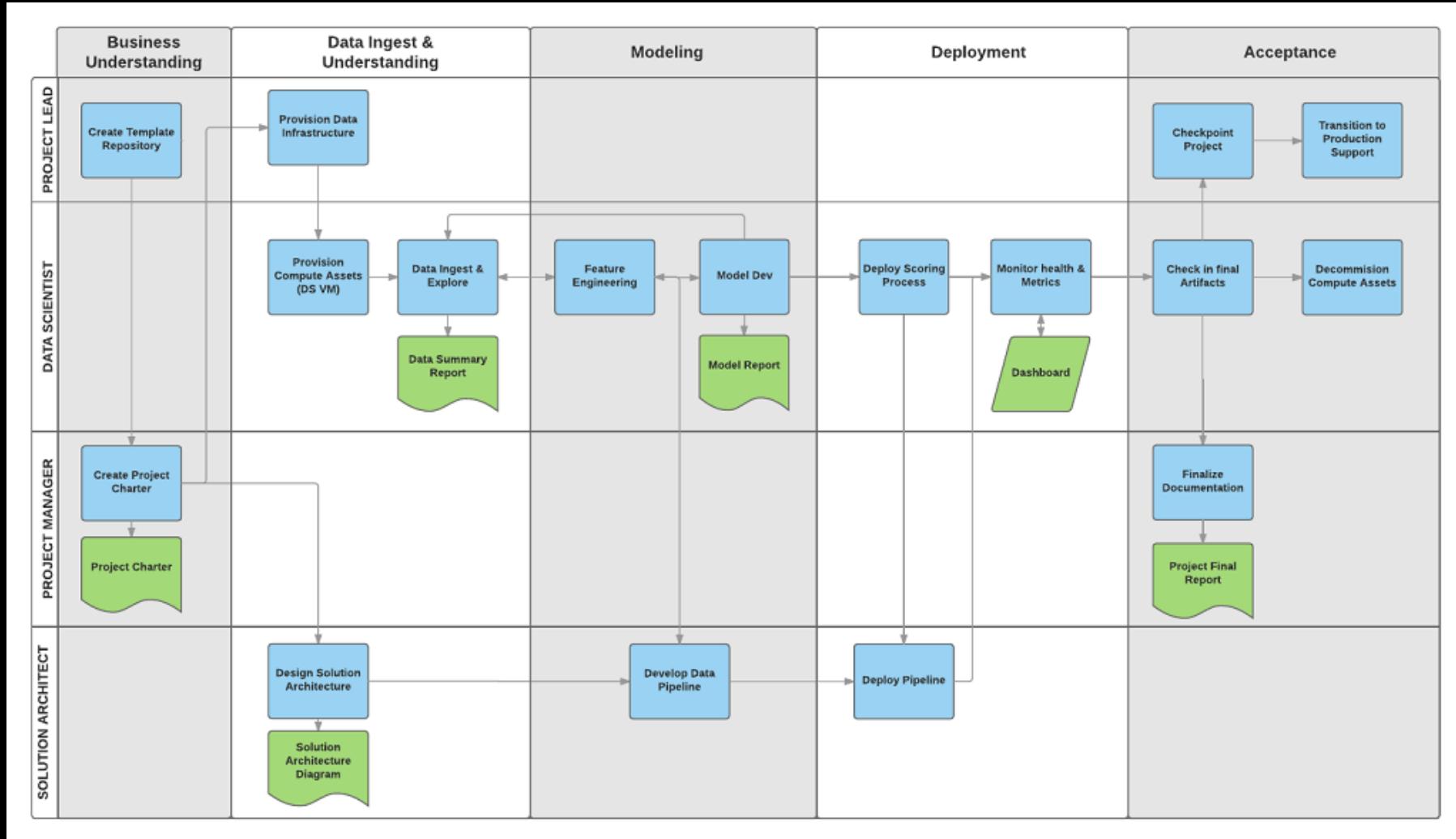
# TDSP (contd.)



# TDSP (contd.)

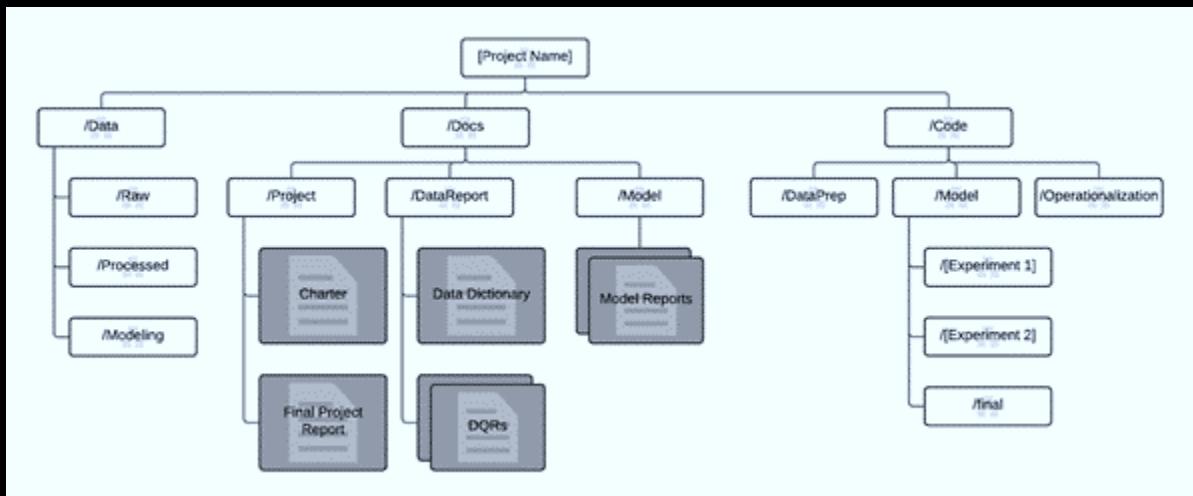
- The goals, tasks, and documentation artifacts for each stage of the lifecycle in TDSP are described in the Team Data Science Process lifecycle topic.
- These tasks and artifacts are associated with project roles:
  - Solution architect
  - Project manager
  - Data engineer
  - Data scientist
  - Application developer
  - Project lead

# TDSP (contd.)



# TDSP (contd.)

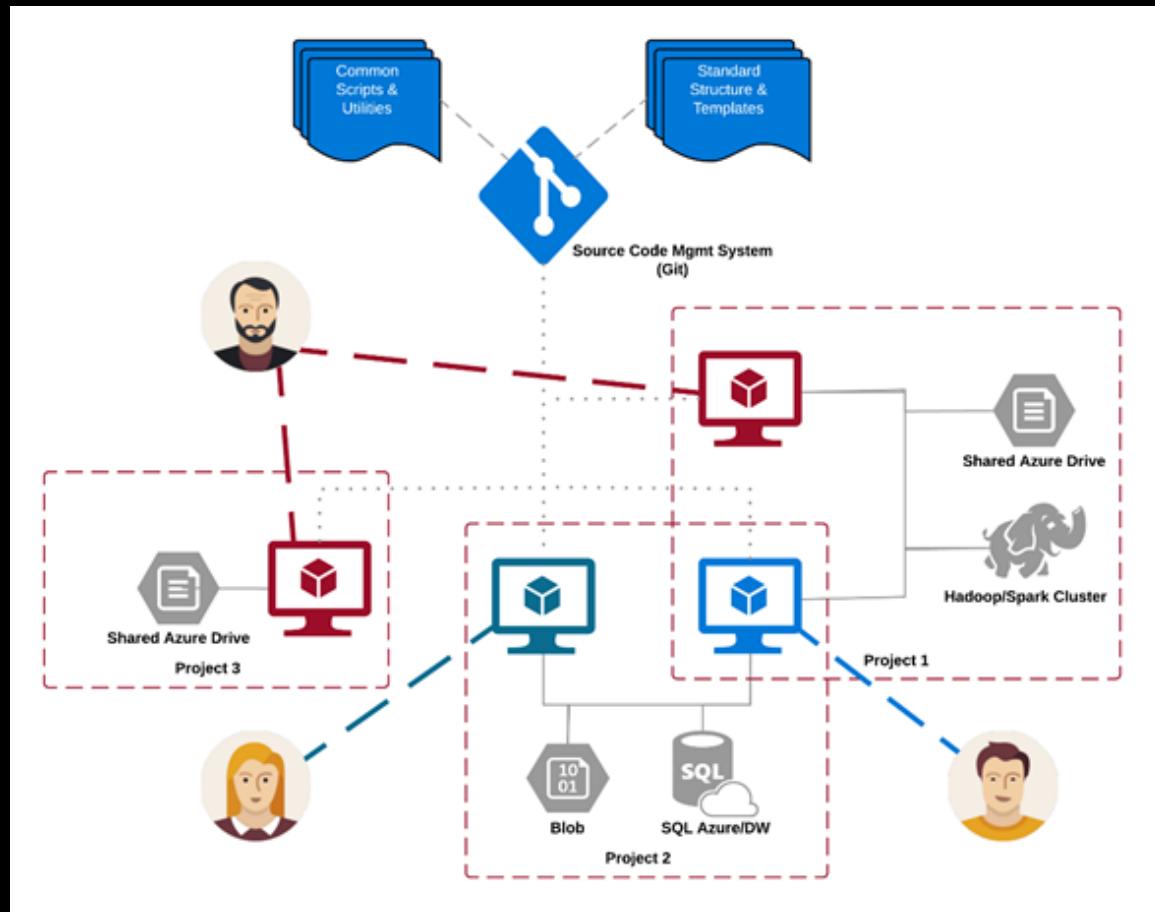
- **Standardized project structure**
  - Having all projects share a directory structure and use templates for project documents makes it easy for the team members to find information about their projects.
  - All code and documents are stored in a version control system (VCS) like Git, TFS, or Subversion to enable team collaboration.
  - Tracking tasks and features in an agile project tracking system like Jira, Rally, and Azure DevOps allows closer tracking of the code for individual features. Such tracking also enables teams to obtain better cost estimates.
  - TDSP recommends creating a separate repository for each project on the VCS for versioning, information security, and collaboration. The standardized structure for all projects helps build institutional knowledge across the organization.



# TDSP (contd.)

- Infrastructure and resources for data science projects
  - TDSP provides recommendations for managing shared analytics and storage infrastructure such as:
    - cloud file systems for storing datasets
    - databases
    - big data (SQL or Spark) clusters
    - machine learning service
  - The analytics and storage infrastructure, where raw and processed datasets are stored, may be in the cloud or on-premises.
  - This infrastructure enables reproducible analysis.
  - It also avoids duplication, which may lead to inconsistencies and unnecessary infrastructure costs. Tools are provided to provision the shared resources, track them, and allow each team member to connect to those resources securely.
  - It is also a good practice to have project members create a consistent compute environment. Different team members can then replicate and validate experiments.

# TDSP (contd.)



Here is an example of a team working on multiple projects and sharing various cloud analytics infrastructure components.

# TDSP (contd.)

- Tools and utilities for project execution
  - Introducing processes in most organizations is challenging.
  - Tools provided to implement the data science process and lifecycle help lower the barriers to and increase the consistency of their adoption.
  - TDSP provides an initial set of tools and scripts to jump-start adoption of TDSP within a team.
  - It also helps automate some of the common tasks in the data science lifecycle such as data exploration and baseline modeling.
  - There is a well-defined structure provided for individuals to contribute shared tools and utilities into their team's shared code repository.
  - These resources can then be leveraged by other projects within the team or the organization.
  - Microsoft provides extensive tooling inside Azure Machine Learning supporting both open-source (Python, R, ONNX, and common deep-learning frameworks) and also Microsoft's own tooling (AutoML).

# TDSP (contd.)

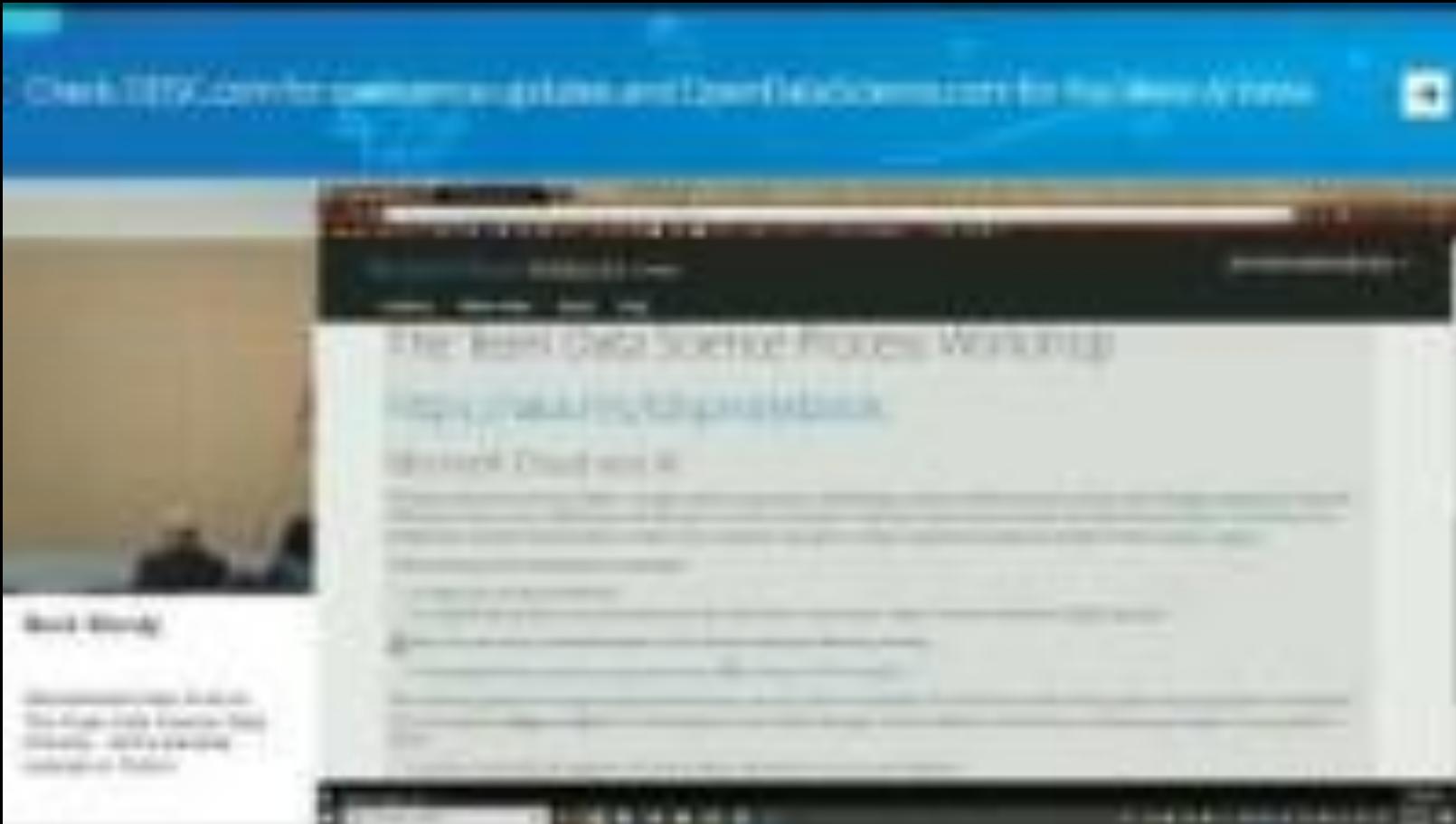
## Benefits

- Elaborated Documentation
- Agile
- Familiar
- Data Science Native
- Flexible
- Detailed
- Free Templates

## Short comings

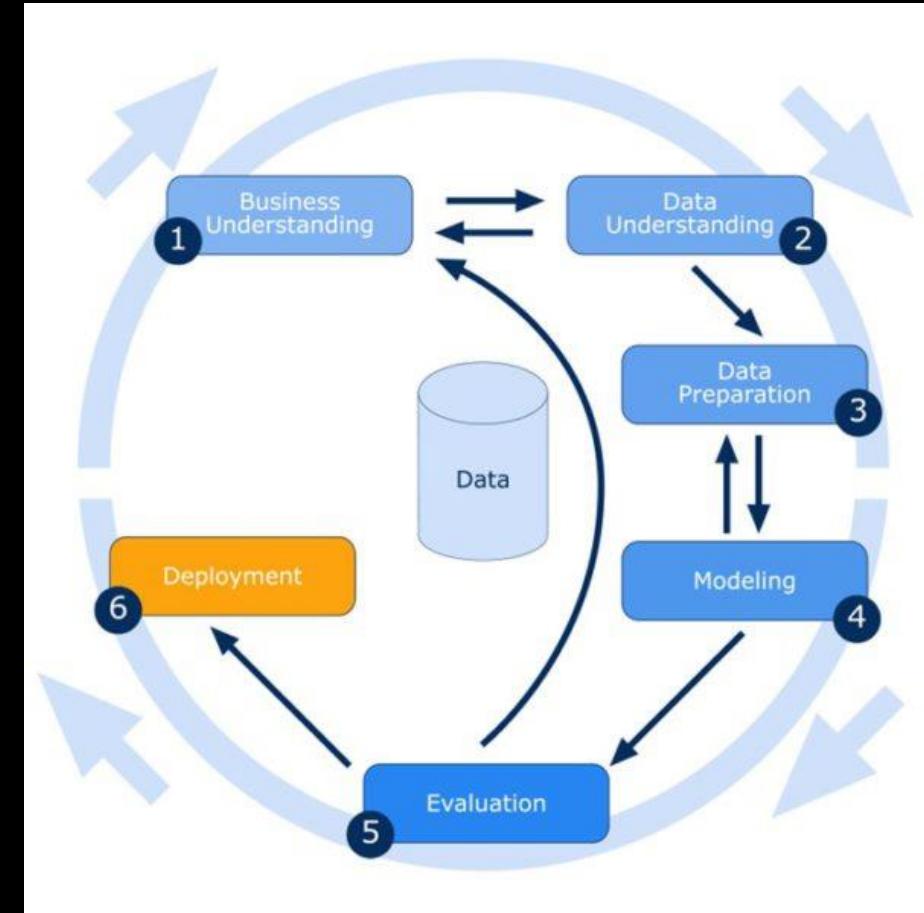
- Fixed Sprints: TDSP leverages fixed-length planning sprints which many data scientists struggle with.
- Some Inconsistencies: Not all of Microsoft's documentation is consistent.

# TDSP (contd.)



# CRISP-DM

- The CRoss Industry Standard Process for Data Mining (CRISP-DM) is a process model that serves as the base for a data science process. It has six sequential phases:
  - Business understanding
  - Data understanding
  - Data preparation
  - Modeling
  - Evaluation
  - Deployment



<https://www.datascience-pm.com/crisp-dm-2/>

<https://web.archive.org/web/20220401041957/https://www.the-modeling-agency.com/crisp-dm.pdf>

# CRISP-DM (contd.)

## I. Business Understanding

- Any good project starts with a deep understanding of the customer's needs. Data mining projects are no exception and CRISP-DM recognizes this.
- The Business Understanding phase focuses on understanding the objectives and requirements of the project. Aside from the third task, the three other tasks in this phase are foundational project management activities that are universal to most projects:
  1. **Determine business objectives:** You should first “thoroughly understand, from a business perspective, what the customer really wants to accomplish.” (CRISP-DM Guide) and then define business success criteria.
  2. **Assess situation:** Determine resources availability, project requirements, assess risks and contingencies, and conduct a cost-benefit analysis.
  3. **Determine data mining goals:** In addition to defining the business objectives, you should also define what success looks like from a technical data mining perspective.
  4. **Produce project plan:** Select technologies and tools and define detailed plans for each project phase.

# CRISP-DM (contd.)

## II. Data Understanding

- Next is the Data Understanding phase. Adding to the foundation of Business Understanding, it drives the focus to identify, collect, and analyze the data sets that can help you accomplish the project goals. This phase also has four tasks:
  1. **Collect initial data:** Acquire the necessary data and (if necessary) load it into your analysis tool.
  2. **Describe data:** Examine the data and document its surface properties like data format, number of records, or field identities.
  3. **Explore data:** Dig deeper into the data. Query it, visualize it, and identify relationships among the data.
  4. **Verify data quality:** How clean/dirty is the data? Document any quality issues.

# CRISP-DM (contd.)

## III. Data Preparation

- A common rule of thumb is that 80% of the project is data preparation.
- This phase, which is often referred to as “data munging”, prepares the final data set(s) for modeling. It has five tasks:
  1. **Select data:** Determine which data sets will be used and document reasons for inclusion/exclusion.
  2. **Clean data:** Often this is the lengthiest task. Without it, you’ll likely fall victim to garbage-in, garbage-out. A common practice during this task is to correct, impute, or remove erroneous values.
  3. **Construct data:** Derive new attributes that will be helpful. For example, derive someone’s body mass index from height and weight fields.
  4. **Integrate data:** Create new data sets by combining data from multiple sources.
  5. **Format data:** Re-format data as necessary. For example, you might convert string values that store numbers to numeric values so that you can perform mathematical operations.

# CRISP-DM (contd.)

## IV. Modeling

- What is widely regarded as data science's most exciting work is also often the shortest phase of the project.
- Here you'll likely build and assess various models based on several different modeling techniques. This phase has four tasks:
  1. **Select modeling techniques:** Determine which algorithms to try (e.g. regression, neural net).
  2. **Generate test design:** Pending your modeling approach, you might need to split the data into training, test, and validation sets.
  3. **Build model:** As glamorous as this might sound, this might just be executing a few lines of code like “`reg = LinearRegression().fit(X, y)`”.
  4. **Assess model:** Generally, multiple models are competing against each other, and the data scientist needs to interpret the model results based on domain knowledge, the pre-defined success criteria, and the test design.

# CRISP-DM (contd.)

## V. Evaluation

- Whereas the Assess Model task of the Modeling phase focuses on technical model assessment, the Evaluation phase looks more broadly at which model best meets the business and what to do next. This phase has three tasks:
  1. **Evaluate results:** Do the models meet the business success criteria? Which one(s) should we approve for the business?
  2. **Review process:** Review the work accomplished. Was anything overlooked? Were all steps properly executed? Summarize findings and correct anything if needed.
  3. **Determine next steps:** Based on the previous three tasks, determine whether to proceed to deployment, iterate further, or initiate new projects.

# CRISP-DM (contd.)

## VI. Deployment

*“Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process across the enterprise.”*

– CRISP-DM Guide

- A model is not particularly useful unless the customer can access its results. The complexity of this phase varies widely. This final phase has four tasks:
  - **Plan deployment:** Develop and document a plan for deploying the model.
  - **Plan monitoring and maintenance:** Develop a thorough monitoring and maintenance plan to avoid issues during the operational phase (or post-project phase) of a model.
  - **Produce final report:** The project team documents a summary of the project which might include a final presentation of data mining results.
  - **Review project:** Conduct a project retrospective about what went well, what could have been better, and how to improve in the future.

# Is CRISP-DM Agile or Waterfall?

- Some argue that it is flexible and agile and while others see CRISP-DM as rigid. What really matters is how you implement it.
- **Waterfall**
  - On one hand, many view CRISP-DM as a rigid waterfall process – in part because of its reporting requirements are excessive for most projects. Moreover, the guide states in the business understanding phase that “the project plan contains detailed plans for each phase” – a hallmark aspect of traditional waterfall approaches that require detailed, upfront planning.
  - Indeed, if you follow CRISP-DM precisely (defining detailed plans for each phase at the project start and include every report) and choose not to iterate frequently, then you’re operating more of a waterfall process.
- **Agile**
  - On the other hand, CRISP-DM indirectly advocates agile principles and practices by stating: “The sequence of the phases is not rigid. Moving back and forth between different phases is always required. The outcome of each phase determines which phase, or particular task of a phase, has to be performed next.”
  - Thus if you follow CRISP-DM in a more flexible way, iterate quickly, and layer in other agile processes, you’ll wind up with an agile approach.

Example: *To illustrate how CRISP-DM could be implemented in either an Agile or waterfall manner, imagine a churn project with three deliverables: a voluntary churn model, a non-pay disconnect churn model, and a propensity to accept a retention-focused offer.*

# CRISP-DM (contd.)

## Waterfall: Horizontal Slicing



## Agile: Vertical Slicing



# CRISP-DM (contd.)

## Which is better?

- When possible, take an agile approach and slice vertically so that:
  - Stakeholders get value sooner
  - Stakeholders can provide meaningful feedback
  - The data scientists can assess model performance earlier
  - The project team can adjust the plan based on stakeholder feedback

# CRISP-DM (contd.)

## Benefits

- Generalizable
- Common Sense
- Adoptable
- Right Start
- Strong Finish
- Flexible

## Shortcomings

- Rigid
- Documentation Heavy
- Not Modern
- Not a Project Management Approach

# Review of statistics and probability

# Probability

- Probability is a measure of the likelihood of an event to occur. Many events cannot be predicted with total certainty.
- We can predict only the chance of an event to occur i.e., how likely they are going to happen, using it.
- Probability can range from 0 to 1, where 0 means the event to be an impossible one and 1 indicates a certain event.
- Probability for Class 10 is an important topic for the students which explains all the basic concepts of this topic.
- The probability of all the events in a sample space adds up to 1.

# Probability

- For example, when we toss a coin, either we get Head OR Tail, only two possible outcomes are possible (H, T).
- But when two coins are tossed then there will be four possible outcomes, i.e  $\{(H, H), (H, T), (T, H), (T, T)\}$ .

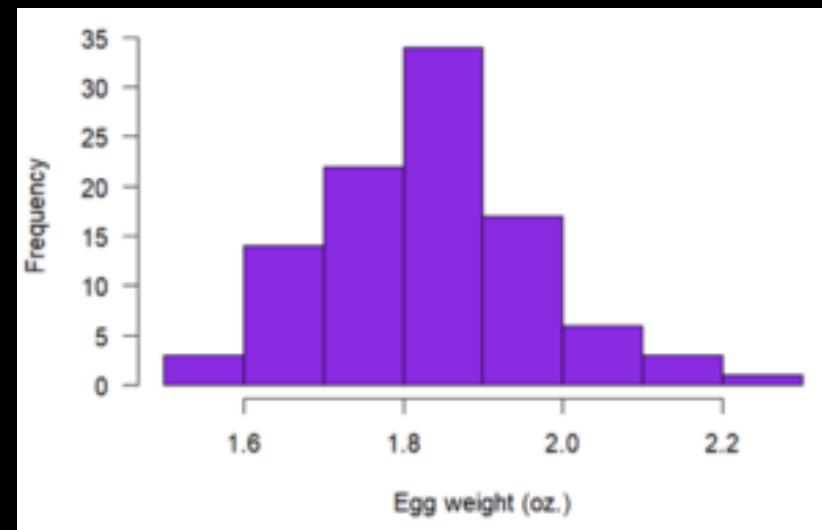
$$\text{Probability of event to happen } P(E) = \frac{\text{Number of favourable outcomes}}{\text{Total Number of Outcomes}}$$

# Probability Distribution

- A probability distribution is an idealized frequency distribution.
- A frequency distribution describes a specific sample or dataset. It's the number of times each possible value of a variable occurs in the dataset.
- The number of times a value occurs in a sample is determined by its probability of occurrence.
- The higher the probability of a value, the higher its frequency in a sample.
- More specifically, the probability of a value is its relative frequency in an infinitely large sample.
- Infinitely large samples are impossible in real life, so probability distributions are theoretical. They're idealized versions of frequency distributions that aim to describe the population the sample was drawn from.
- Probability distributions are used to describe the populations of real-life variables, like coin tosses or the weight of chicken eggs.

# Probability Distribution (contd.)

- Imagine that an egg farmer wants to know the probability of an egg from her farm being a certain size.
- The farmer weighs 100 random eggs and describes their frequency distribution using a histogram:



# Probability Distribution (contd.)

- Thus, a probability distribution is a mathematical function that describes the probability of different possible values of a variable.
- It is a mathematical description of a random phenomenon in terms of its sample space and the probabilities of events (subsets of the sample space).
- Probability distributions are often depicted using graphs or probability tables.
- There are many types of probability distributions, including the normal distribution, binomial distribution, Poisson distribution, and exponential distribution

# Compound events

- A compound event is an event that includes two or more simple events.
- Simple events are events that can have only one outcome, while compound events can have multiple different outcomes.
- Compound events can be made up of a number of **independent events** (*events in which the outcome of one event has no effect on the probability of the other*) or **dependent events** (*events in which the outcome of one event affects the probability of another*).

# Compound Probability

- A compound probability is the probability of a compound event.
- Generally, it is the ratio of favorable outcomes to the total number of outcomes within the sample space of the compound event and can be calculated using one of two rules: **the addition rule** *and* **the multiplication rule**.

# Compound Probability

- **Addition Rule:**

For mutually exclusive

$$P(A \text{ or } B) = P(A) + P(B)$$

For non-mutually exclusive events

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B)$$

- **Product Rule:**

For independent events

$$P(A \text{ and } B) = P(A).P(B)$$

For dependent events

$$P(A \text{ and } B) = P(A).P(B|A) \text{ or } P(B).P(A|B)$$

# Centrality Measures

- Measures of central tendency help you find the middle, or the average, of a dataset.
- The 3 most common measures of central tendency are the mode, median, and mean.
  - Mode: the most frequent value.
  - Median: the middle number in an ordered dataset.
  - Mean: the sum of all values divided by the total number of values.

# Centrality Measures (contd.)

- Mean
  - The arithmetic mean of a dataset (which is different from the geometric mean) is the sum of all values divided by the total number of values.
  - It's the most commonly used measure of central tendency because all values are used in the calculation.
  - Outliers can significantly increase or decrease the mean when they are included in the calculation.
  - Since all values are used to calculate the mean, it can be affected by extreme outliers.
  - An outlier is a value that differs significantly from the others in a dataset.

# Centrality Measures (contd.)

Example: Finding the mean

Participant	1	2	3	4	5
Reaction time (milliseconds)	287	345	365	298	380

First you add up the sum of all values:

$$\sum x = 287 + 345 + 365 + 298 + 380 = 1675$$

Then you calculate the mean using the formula

$$\frac{\sum x}{n}$$

There are 5 values in the dataset, so  $n = 5$ .

$$\bar{x} = \frac{1675}{5} = 335$$

Mean ( $\bar{x}$ ): 335 milliseconds

# Centrality Measures (contd.)

## Example: Mean with an outlier

In this dataset, we swap out one value with an extreme outlier.

Participant	1	2	3	4	5
Reaction time (milliseconds)	832	345	365	298	380

$$\sum x = 832 + 345 + 365 + 298 + 380 = 2\,220$$

$$\bar{x} = \frac{\sum x}{n} = \frac{2\,220}{5} = 444$$

Due to the outlier, the mean ( $\bar{x}$ ) becomes much higher, even though all the other numbers in the dataset stay the same.

**Mean: 444 milliseconds**

# Centrality Measures (contd.)

- Median

- The median of a dataset is the value that's exactly in the middle when it is ordered from low to high.

**Example: Finding the median**

You measure the reaction times of 7 participants on a computer task and categorize them into 3 groups: slow, medium or fast.

Participant	1	2	3	4	5	6	7
Speed	Medium	Slow	Fast	Fast	Medium	Fast	Slow

To find the median, you first order all values from low to high. Then, you find the value in the middle of the ordered dataset—in this case, the value in the 4th position.

Ordered dataset	Slow	Slow	Medium	Medium	Fast	Fast	Fast
-----------------	------	------	--------	--------	------	------	------

**Median: Medium**

# Centrality Measures (contd.)

## Median of an odd-numbered dataset

For an odd-numbered dataset, find the value that lies at the  $\frac{(n + 1)}{2}$  position, where  $n$  is the number of values in the dataset.

### Example

You measure the reaction times in milliseconds of 5 participants and order the dataset.

Reaction time (milliseconds)	287	298	345	365	380
------------------------------	-----	-----	-----	-----	-----

The middle position is calculated using  $\frac{(n + 1)}{2}$ , where  $n = 5$ .

$$\frac{(5 + 1)}{2} = 3$$

That means the median is the 3rd value in your ordered dataset.

**Median: 345 milliseconds**

# Centrality Measures (contd.)

- Mode
  - The mode is the most frequently occurring value in the dataset. It's possible to have no mode, one mode, or more than one mode.
  - To find the mode, sort your dataset numerically or categorically and select the response that occurs most frequently.
  - The mode is most applicable to data from a nominal level of measurement. Nominal data is classified into mutually exclusive categories, so the mode tells you the most popular category.
  - For continuous variables or ratio levels of measurement, the mode may not be a helpful measure of central tendency.

# Centrality Measures (contd.)

Example: Ratio data with no mode

You collect data on reaction times in a computer task, and your dataset contains values that are all different from each other.

Participant	1	2	3	4	5	6	7	8	9
Reaction time (milliseconds)	267	345	421	324	401	312	382	298	303

In this dataset, there is no mode, because each value occurs only once.

# Variability Measures

- Variability describes how far apart data points lie from each other and from the center of a distribution.
- Along with measures of central tendency, measures of variability give you descriptive statistics that summarize your data.
- Variability is also referred to as spread, scatter or dispersion. It is most commonly measured with the following:
  - Range: the difference between the highest and lowest values
  - Interquartile range: the range of the middle half of a distribution
  - Standard deviation: average distance from the mean
  - Variance: average of squared distances from the mean

# Variability Measures (contd.)

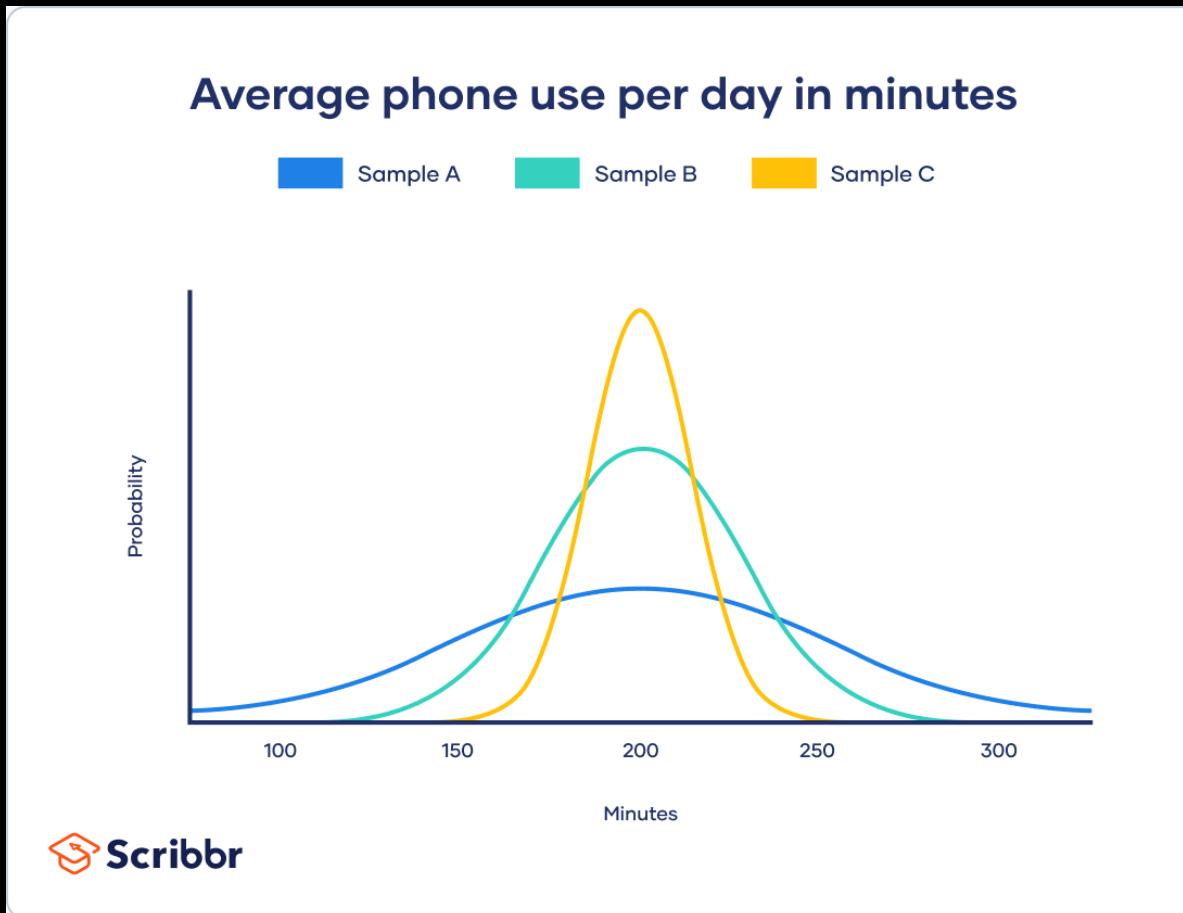
- Why does it matter?

While the central tendency, or average, tells you where most of your points lie, variability summarizes how far apart they are. This is important because the amount of variability determines how well you can generalize results from the sample to your population.

Low variability is ideal because it means that you can better predict information about the population based on sample data. High variability means that the values are less consistent, so it's harder to make predictions.

Data sets can have the same central tendency but different levels of variability or vice versa. If you know only the central tendency or the variability, you can't say anything about the other aspect. Both of them together give you a complete picture of your data.

# Variability Measures (contd.)



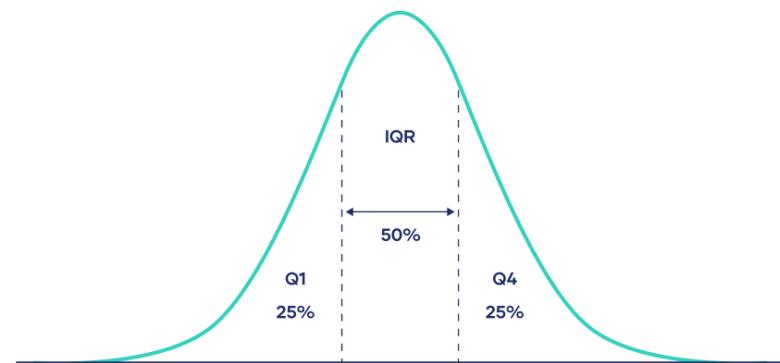
Although the data follows a normal distribution, each sample has different spreads. Sample A has the largest variability while Sample C has the smallest variability.

# Variability Measures (contd.)

- **Inter Quartile Range**

- The interquartile range gives you the spread of the middle of your distribution.
- For any distribution that's ordered from low to high, the interquartile range contains half of the values.
- While the first quartile (Q1) contains the first 25% of values, the fourth quartile (Q4) contains the last 25% of values.
- The interquartile range is the third quartile (Q3) minus the first quartile (Q1). This gives us the range of the middle half of a data set.

Interquartile range on a normal distribution



# Variability Measures (contd.)



## Five-number summary

Every distribution can be organized using a five-number summary:

- Lowest value
- Q1: 25th percentile
- Q2: the median
- Q3: 75th percentile
- Highest value (Q4)

These five-number summaries can be easily visualized using box and whisker plots.

# Variability Measures (contd.)

- Standard deviation
  - The standard deviation is the average amount of variability in your dataset.
  - It tells you, on average, how far each score lies from the mean. The larger the standard deviation, the more variable the data set is.
  - There are six steps for finding the standard deviation by hand:
    - List each score and find their mean.
    - Subtract the mean from each score to get the deviation from the mean.
    - Square each of these deviations.
    - Add up all of the squared deviations.
    - Divide the sum of the squared deviations by  $n - 1$  (for a sample) or  $N$  (for a population).
    - Find the square root of the number you found.

# Variability Measures (contd.)

## Standard deviation example

Step 1: Data (minutes)	Step 2: Deviation from mean	Steps 3 + 4: Squared deviation
72	$72 - 207.5 = -135.5$	18360.25
110	$110 - 207.5 = -97.5$	9506.25
134	$134 - 207.5 = -73.5$	5402.25
190	$190 - 207.5 = -17.5$	306.25
238	$238 - 207.5 = 30.5$	930.25
287	$287 - 207.5 = 79.5$	6320.25
305	$305 - 207.5 = 97.5$	9506.25
324	$324 - 207.5 = 116.5$	13572.25
Mean = 207.5	Sum = 0	Sum of squares = 63904

Step 5

## Standard deviation example

Because you're dealing with a sample, you use  $n - 1$ .

$$n - 1 = 7$$

$$63904 / 7 = 9129.14$$

Step 6

## Standard deviation example

$$s = \sqrt{9129.14} = 95.54$$

The standard deviation of your data is 95.54. This means that on average, each score deviates from the mean by 95.54 points.

# Variability Measures (contd.)

## Standard deviation formula for populations

If you have data from the entire population, use the population standard deviation formula:

Formula	Explanation
$\sigma = \sqrt{\frac{\sum (X - \mu)^2}{N}}$	<ul style="list-style-type: none"><li>• <math>\sigma</math> = population standard deviation</li><li>• <math>\sum</math> = sum of...</li><li>• <math>X</math> = each value</li><li>• <math>\mu</math> = population mean</li><li>• <math>N</math> = number of values in the population</li></ul>

# Variability Measures (contd.)

- Variance
  - The variance is the average of squared deviations from the mean. A deviation from the mean is how far a score lies from the mean.
  - Variance is the square of the standard deviation. This means that the units of variance are much larger than those of a typical value of a data set.
  - While it's harder to interpret the variance number intuitively, it's important to calculate variance for comparing different data sets in statistical tests like ANOVAs.
  - Variance reflects the degree of spread in the data set. The more spread the data, the larger the variance is in relation to the mean.

# Variability Measures (contd.)

## Variance example

To get variance, square the standard deviation.

$$s = 95.5$$

$$s^2 = 95.5 \times 95.5 = 9129.14$$

The variance of your data is 9129.14.

## Variance formula for populations

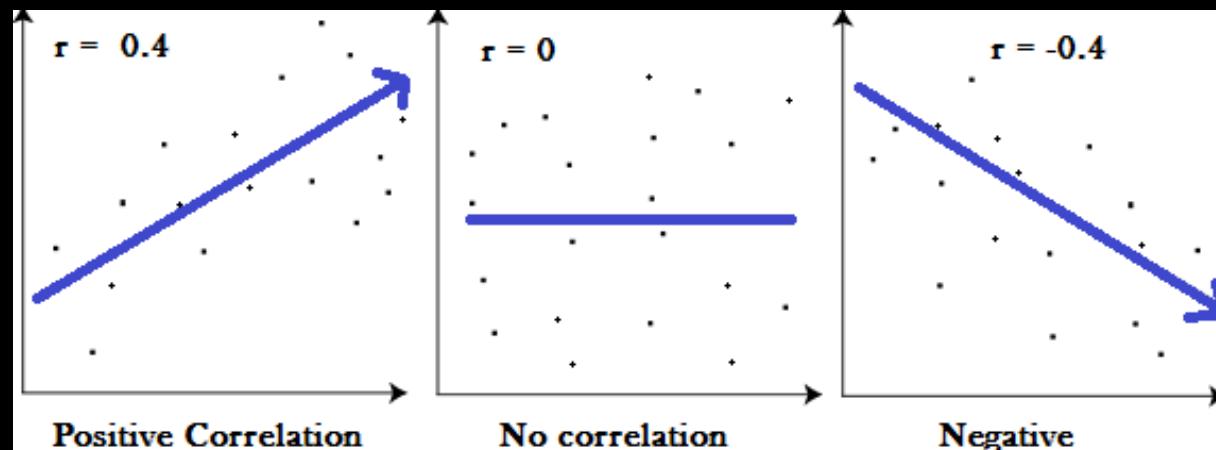
Formula	Explanation
$\sigma^2 = \frac{\sum(X - \mu)^2}{N}$	<ul style="list-style-type: none"><li>• <math>\sigma^2</math> = population variance</li><li>• <math>\sum</math> = sum of...</li><li>• <math>=</math> each value</li><li>• <math>\mu</math> = population mean</li><li>• <math>N</math> = number of values in the population</li></ul>

# Correlation Analysis

- Correlation is used to test relationships between quantitative variables or categorical variables.
- In other words, it's a measure of how things are related. The study of how variables are correlated is called correlation analysis.
- Correlations are useful because if you can find out what relationship variables have, you can make predictions about future behavior.
- Knowing what the future holds is very important in the social sciences like government and healthcare.

# Correlation Analysis

- A **correlation coefficient** is a way to put a value to the relationship. Correlation coefficients have a value of between -1 and 1.
- A “0” means there is no relationship between the variables at all, while -1 or 1 means that there is a perfect negative or positive correlation (*negative or positive correlation here refers to the type of graph the relationship will produce*).

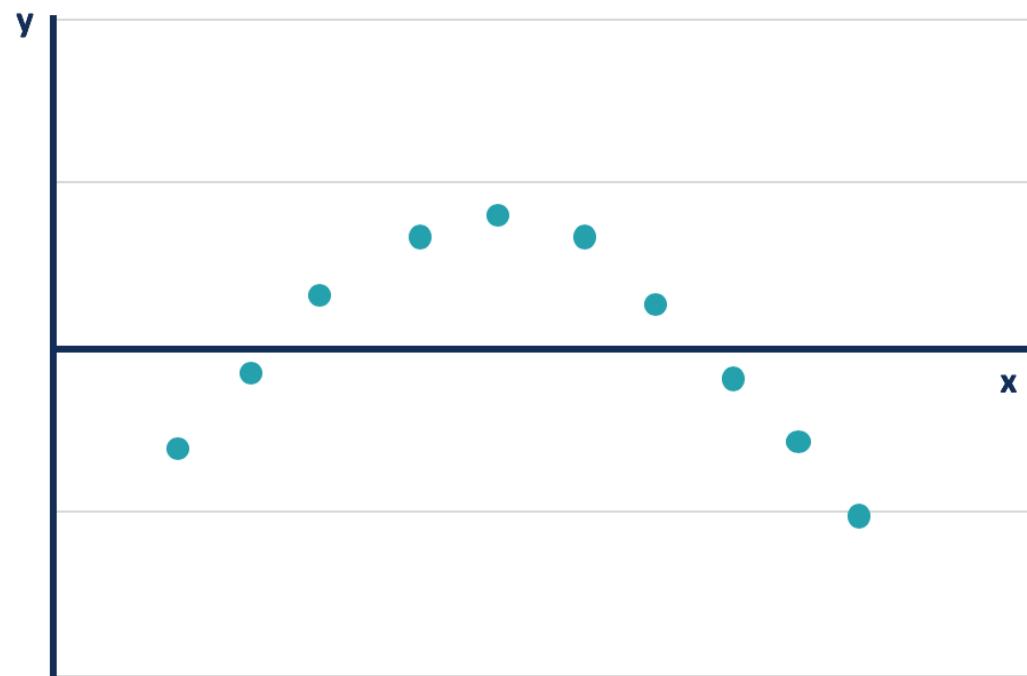


# Autocorrelation

- Autocorrelation is a mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals.
- It's conceptually similar to the correlation between two different time series, but autocorrelation uses the same time series twice: once in its original form and once lagged one or more time periods.
- For example,
  - if it's rainy today, the data suggests that it's more likely to rain tomorrow than if it's clear today.
  - When it comes to investing, a stock might have a strong positive autocorrelation of returns, suggesting that if it's "up" today, it's more likely to be up tomorrow, too.
- Naturally, autocorrelation can be a useful tool for data scientist to utilize; particularly for technical analysts.

# Autocorrelation (contd.)

**Positive Autocorrelation**



**Negative Autocorrelation**



# Autocorrelation (contd.)

- Autocorrelation can be applied to different numbers of time gaps, which is known as lag.
- A lag 1 autocorrelation measures the correlation between the observations that are a one-time gap apart.
- For example, to learn the correlation between the temperatures of one day and the corresponding day in the next month, a lag 30 autocorrelation should be used (assuming 30 days in that month).

# Autocorrelation (contd.)

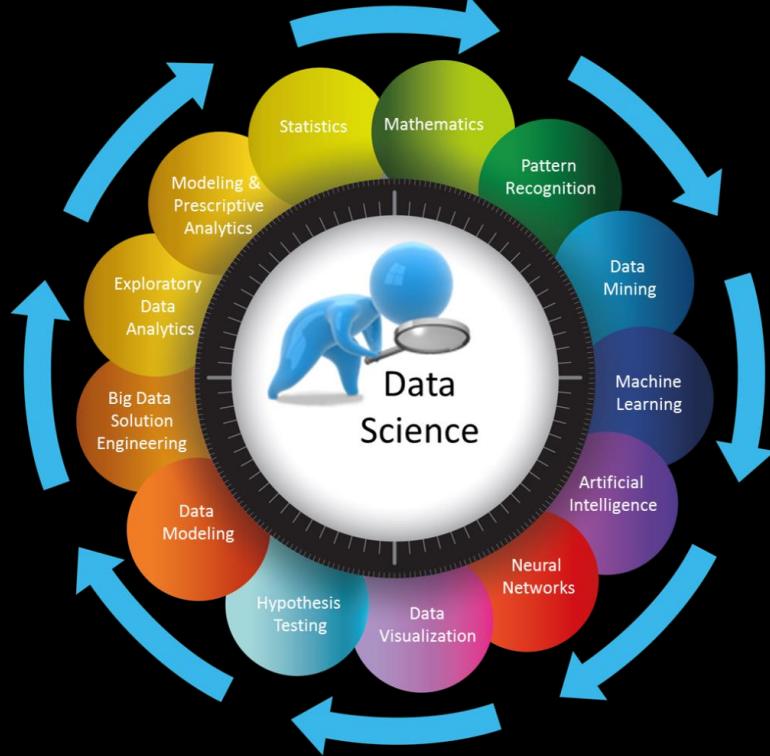
- The Durbin-Watson statistic is commonly used to test for autocorrelation.
- It can be applied to a data set by statistical software. The outcome of the Durbin-Watson test ranges from 0 to 4.
- An outcome closely around 2 means a very low level of autocorrelation.
- An outcome closer to 0 suggests a stronger positive autocorrelation, and an outcome closer to 4 suggests a stronger negative autocorrelation.

# Autocorrelation (contd.)

- It is necessary to test for autocorrelation when analyzing a set of historical data.
- For example, in the equity market, the stock prices on one day can be highly correlated to the prices on another day.
- However, it provides little information for statistical data analysis and does not tell the actual performance of the stock.
- Therefore, it is necessary to test for the autocorrelation of the historical prices to identify to what extent the price change is merely a pattern or caused by other factors.
- In finance, an ordinary way to eliminate the impact of autocorrelation is to use percentage changes in asset prices instead of historical prices themselves.

# End of the Chapter

Next: Data Munging (Chapter 2)



# Data Munging

Unit 2

# What is Data Munging?

- a.k.a **Data Wrangling**, process of processes designed to transform raw data into more readily used formats.
- The exact methods differ from project to project depending on the data you're leveraging and the goal you're trying to achieve.
- Any analyses a business performs will ultimately be constrained by the data that informs them.
- If data is incomplete, unreliable, or faulty, then analyses will be too—diminishing the value of any insights gleaned.
- Data Munging seeks to remove that risk by ensuring data is in a reliable state before it's analyzed and leveraged.

# What is Data Munging? (contd.)

- It's important to note that data wrangling can be time-consuming and taxing on resources, particularly when done manually.
- This is why many organizations institute policies and best practices that help employees streamline the data cleanup process—for example, requiring that data include certain information or be in a specific format before it's uploaded to a database.

# Understanding Data Munging through Analogy



Harvest / Buy Vegetables  
( Import )



Wash vegetables  
( Cleaning )



Chop vegetables  
( Transformation )

# Data Quality

- **Data Quality** refers to the state of data that determine how well suited the data is to meet the purpose.
- It is a measure of the condition of data in terms of
  - Data accuracy
  - Completeness
  - Consistency
  - Timeliness
  - Uniqueness
  - Integrity

# Data Quality (contd.)

- **Data accuracy:** Not a false Information - How accurate the data is?

E.g. In resume, developer wrote 10 years of experience on Mobile App development using Flutter.

- **Completeness:** *Comprehensiveness or wholeness of data* – Do we have complete information?

All data fields contains data required for analysis.

# Data Quality (contd.)

## Ways to ensure data completeness

1. Determine which information is critical
2. Make certain fields required
3. Use data profiling techniques
4. Assemble a data quality team
5. Use automation and AI technologies
6. Use the right data source

# Data Quality (contd.)

- **Consistency:** *Data change across all the related source; different version of data exists across different sources – Is data valid and accurate?*
- **Timeliness:** *Data belongs to the same time period as required*

E.g. You are assigned to predict future sales from today. But data for last 2 years is not so consistent due to newer system replacement.

# Data Quality (contd.)

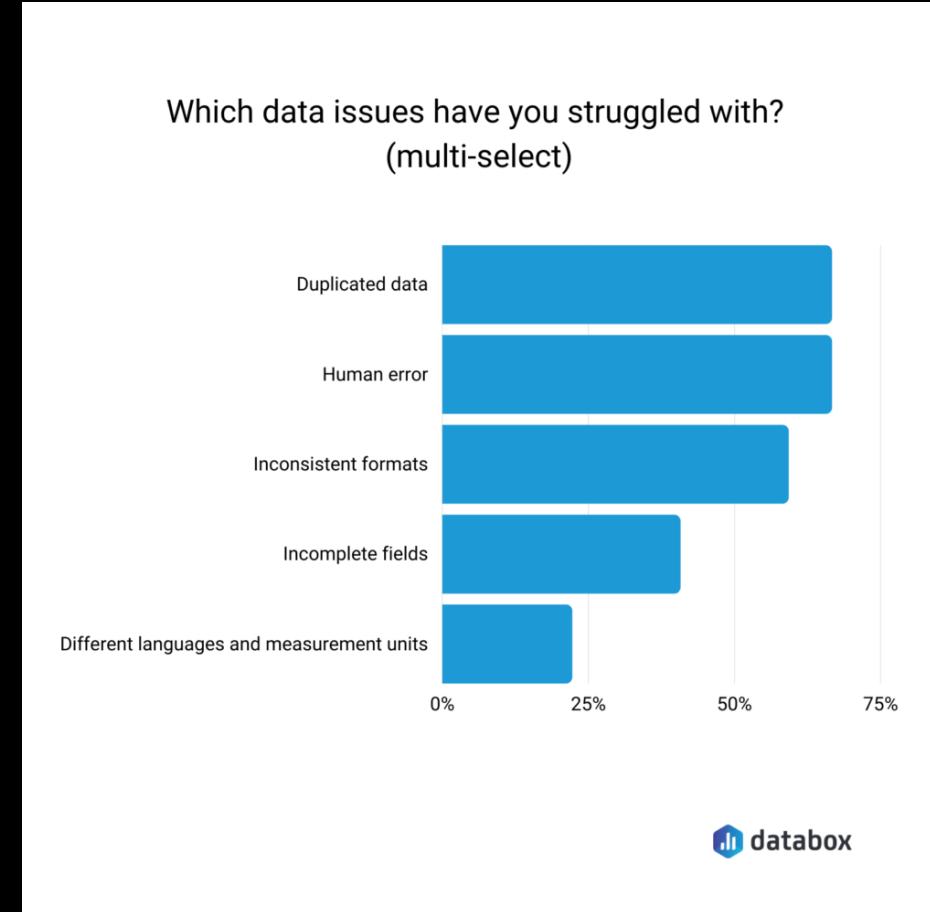
- **Uniqueness:** *No duplicate record*
- **Conformity:** *Data follows standard format – Is data in format we require?*

Redundant/Inconsistent Data Example					
ID	Organizer	Date	Test Location	Meeting	Meeting With
1	John Smith	20-01-2021	Bermingham	Evening	Rosina
2	John Doe	20-01-2021	Mexico	Evening	Rosina
3	John Doe	20-01-2021	Mexico	Evening	Rosina
4	John Doe	20-01-2021	Mexico	Evening	Rosina
5	John Doe	20-01-2021	Mexico	Evening	Rosina
6	John Smith	20-02-2021	Bermingham	Evening	Rosina

*E.g. Sales data should follow 2023-03-24T17:03:41+01:00 format i.e we also require time data with date.*

# Common Issues with Real World Data

1. Duplicates
2. Missing Data
3. Non-Standard Data
4. Unit Mismatch
- Others,
5. Data Ambiguity
6. Hidden data
7. Human Error
8. Inconsistent Data
9. Data Overload



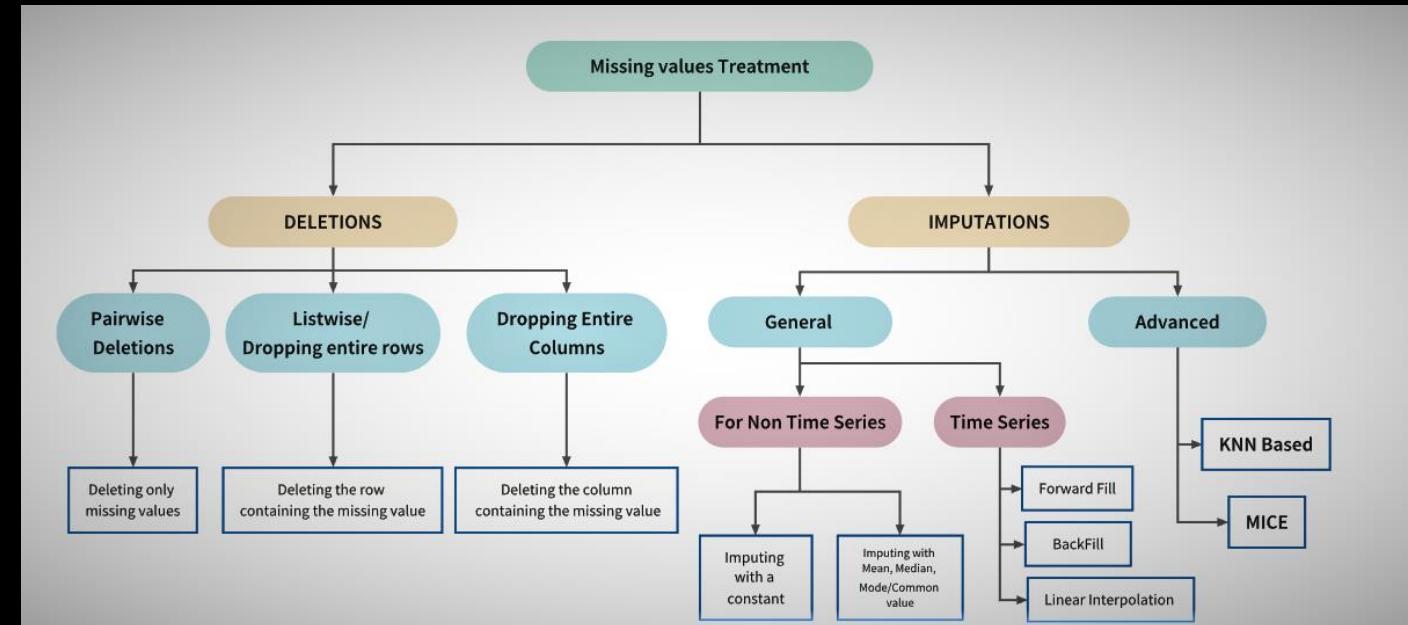
# Ways to Clean Data

1. Dealing with Missing Values
2. Data Enrichment
3. Data Standardization/ Normalization

# Ways to Clean Data (contd.)

## 1. Dealing with Missing Data

- Remove missing record
- Imputation
  - Fill with mean, mode or median
  - constant value
- Apply Algorithms or ML Techniques



<https://ml-concepts.com/2021/10/07/handling-missing-values-in-a-dataset/>

# Ways to Clean Data (contd.)

## 2. Data Enrichment

- Data Enrichment is the process used to enhance, refine or improve the data quality.
- Data enrichment is also a process that takes raw data points and merges them with similar data points in a larger database.
- We enhance the existing information by supplementing missing or incomplete data with relevant context obtained from additional sources.
- This is achieved by merging third-party data from an external authoritative source with an existing database of first-party customer data.

# Ways to Clean Data (contd.)

## 2. Data Enrichment (contd.)

- Businesses carry out Data Enrichment to improve the information they currently have so they can make better-informed decisions. Apart from that, it helps businesses perform the following operations:
  - Define and manage hierarchies.
  - Create new business rules for labeling and sorting data on the fly.
  - Investigate and process data that is multilingual and multi-structured.
  - Process text and Semi-structured Data more efficiently.
  - Reduce costs and optimize sales.
  - Perform Predictive Analysis.

# Ways to Clean Data (contd.)

## 2. Data Enrichment (contd.)

- Why do we need to Enrich Data?

*(In the context of business )*

- #1 – Enriched Data Enhances Your Personalization Strategy
- #2 – Enriched Data Produces Greater Customer Insights
- #3 – Enriched Data Empowers You to Hone in on VIPs
- #4 – Enriched Data Enhances Campaign Performance
- #5 – Enriched Data Leads to Better Decision Making

# Ways to Clean Data (contd.)

## 2. Data Enrichment (contd.)

### Techniques for Data Enrichment

- Web Scrapping
- Manual Research
- Data Append
- Data Segmentation
- Data Imputation
- Entity Extraction
- Data Categorization
- Extrapolation
- Data Correction
- Data Augmentation

# Ways to Clean Data (contd.)

## 3. Data Standardization / Normalization

Source 1					
Name	Email Address	Phone Number	DOB	Gender	Residential Address
John Oneel	john.neal@gmail.com	5164659494	14/2/1987	M	11400 W Olimpic BL # 200

Source 2						
First Name	Last Name	Email Address	Phone Number	DOB	Gender	Residential Address
Johnathan	O'Neal	john.neal_gmail.co m	+1 516-465-9494	2/14/1987	Male	11400 W Olympic 200

# Ways to Clean Data (contd.)

## 3. Data Standardization / Normalization (contd.)

In the example above, you can see the following types of inconsistencies:

- **Structural:** The first source covers Customer Name as a single field, while the second one stores it as two fields – First and Last Name.
- **Pattern:** The first source has a valid email pattern enforced on the email address field, while the second one is visibly missing the @ symbol.
- **Data type:** The first source only allows digits in the Phone Number field, while the second one has a string type field that contains symbols and spaces as well.
- **Format:** The first source has date of birth in the format MM/DD/YYYY, while the second one has it in the format DD/MM/YYYY.
- **Domain value:** The first source allows Gender value to be stored as M or F, while the second source stores the complete form – Male or Female.

# Ways to Clean Data (contd.)

## 3. Data Standardization / Normalization (contd.)

- Why do we need data standard definition?

A data standard definition helps identify:

- The data assets crucial to your business process,
- The necessary data fields of those assets,
- The data type, format, and pattern their values must conform to,
- The range of acceptable values for these fields, and so on.

# Ways to Clean Data (contd.)

## 3. Data Standardization / Normalization

- Data normalization consists of remodeling numeric columns to a standard scale.
- Data normalization is generally considered the development of clean data. :
- Goal of the Data Normalization
  - Data normalization is the organization of data to appear similar across all records and fields.
  - It increases the cohesion of entry types, leading to cleansing, lead generation, segmentation, and higher quality data.

# Ways to Clean Data (contd.)

- ~~Data Standardization / Normalization~~

## Techniques

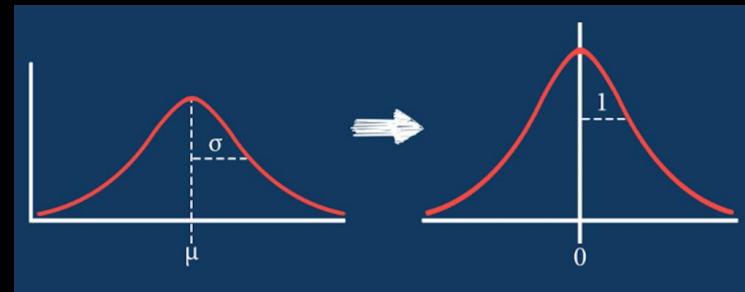
### a) Min-Max Normalization

$$v' = \frac{v - \text{min}_F}{\text{max}_F - \text{min}_F} (\text{new\_max}_F - \text{new\_min}_F) + \text{new\_min}_F ,$$

### a) Z-score Normalization

$$Z = \frac{x - \mu}{\sigma}$$

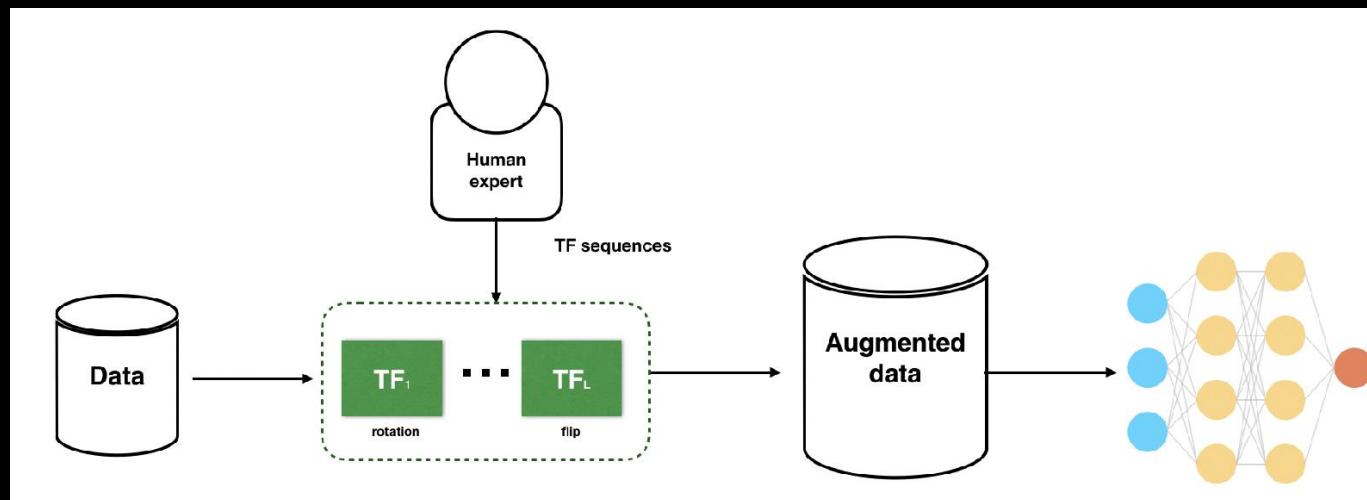
Score      Mean  
                SD



# Ways to Clean Data (contd.)

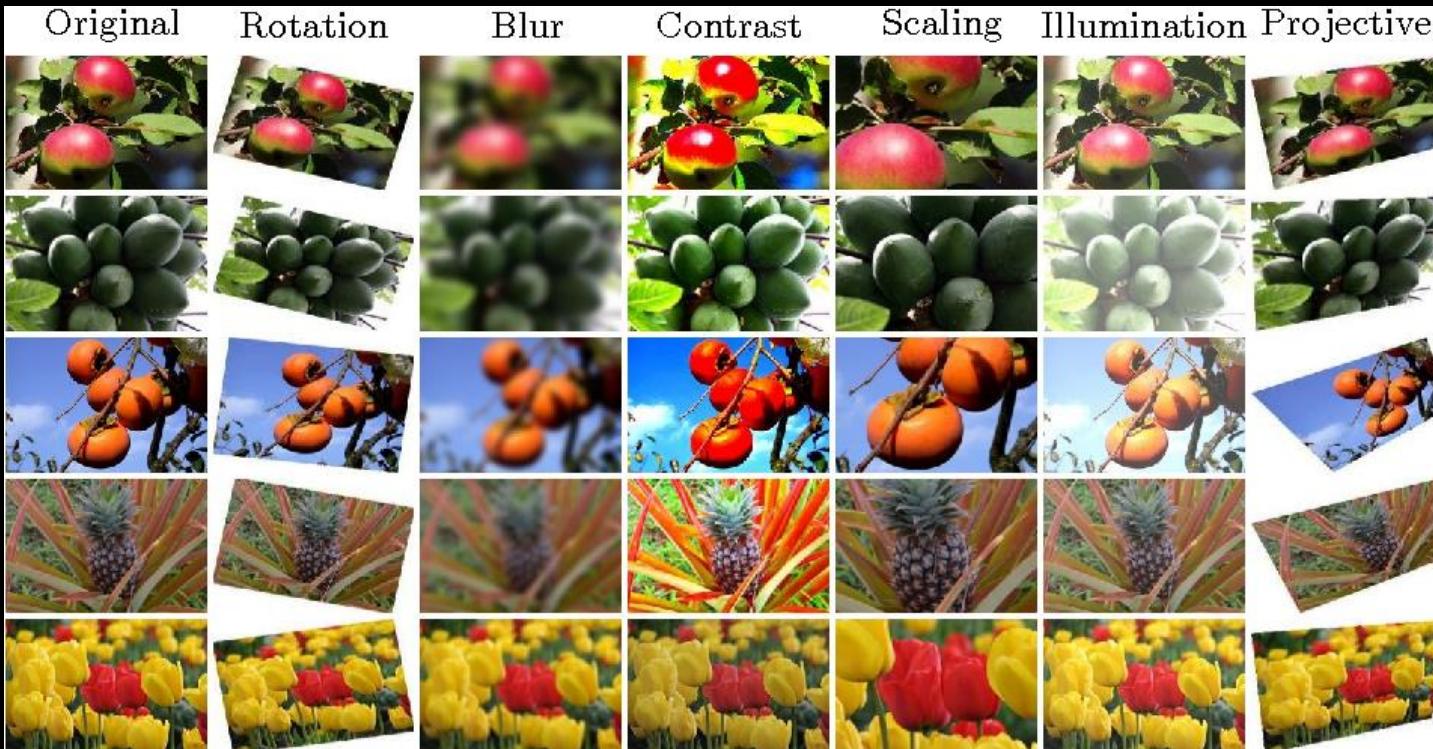
- **Data Augmentation**

- Data augmentation is a set of techniques to artificially increase the amount of data by generating new data points from existing data.
- This includes making small changes to data or using deep learning models to generate new data points.



# Ways to Clean Data (contd.)

- **Data Augmentation**



# Ways to Clean Data (contd.)

- Data Augmentation
  - Advanced models for data augmentation are
    - **Adversarial training/Adversarial machine learning:** It generates adversarial examples which disrupt a machine learning model and injects them into a dataset to train.
    - **Generative adversarial networks (GANs):** GAN algorithms can learn patterns from input datasets and automatically create new examples which resemble training data.
    - **Neural style transfer:** Neural style transfer models can blend content image and style image and separate style from content.
    - **Reinforcement learning:** Reinforcement learning models train software agents to attain their goals and make decisions in a virtual environment.

# Preprocessing for Text

- Lowering the Text
- Remove Numbers
- Remove URLs
- Remove Symbols
- Remove Punctuations
- Remove Whitespace
- Remove Stopwords
- Stemming and Lemmatization
- Vectorization and Semantic Representation

# Preprocessing for Image

- **Rescaling**
- **Grayscale**
- **Samplewise Centering**
- **Featurewise centering**
- **Augmentation Transformations**

<https://www.intel.com/content/www/us/en/developer/articles/technical/hands-on-ai-part-14-image-data-preprocessing-and-augmentation.html>

# Data Formats

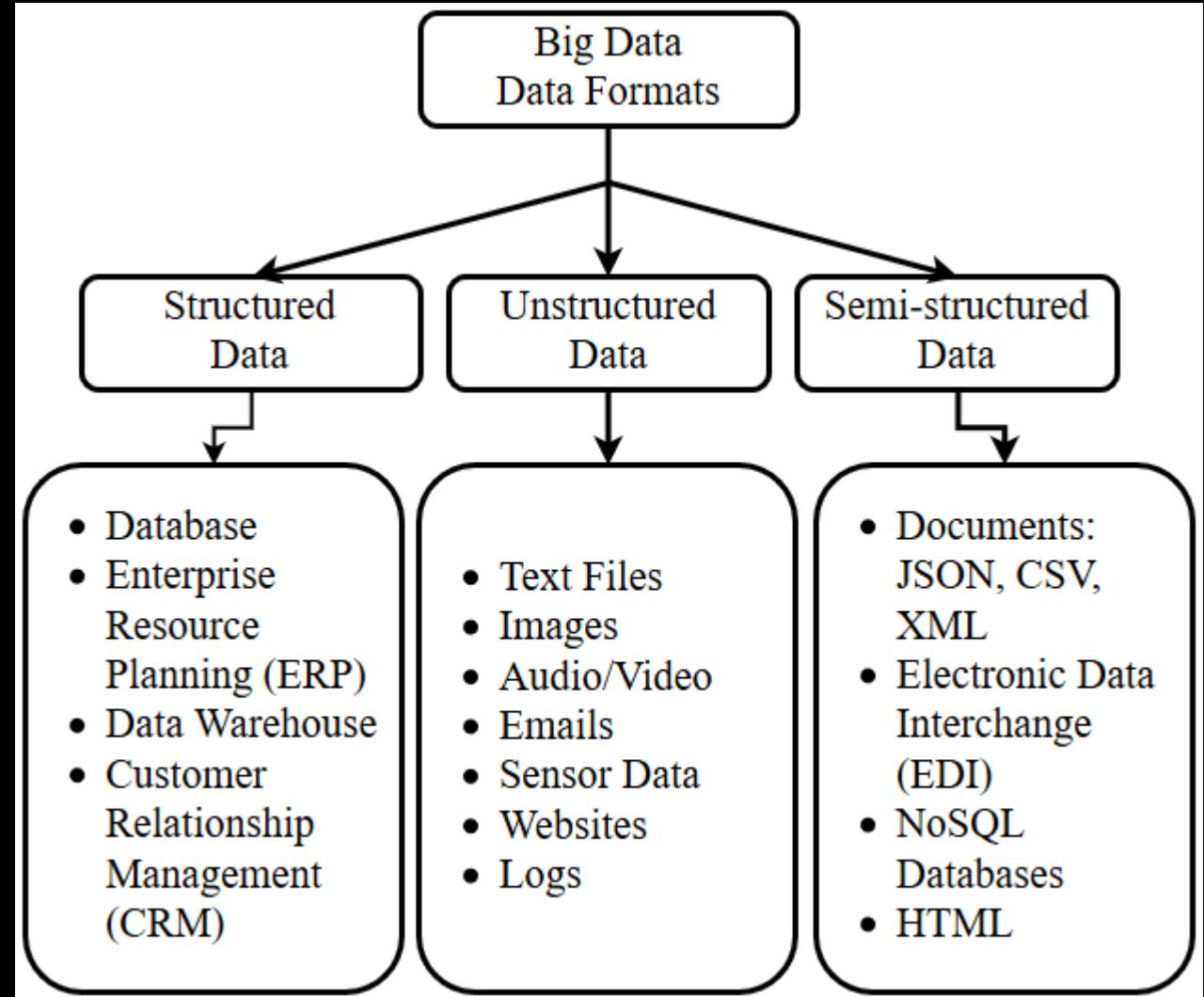
- Data format is the definition of the structure of data within a database or file system that gives the information its meaning.
- How data is organized and presented.
- Arrangement of data fields in a specific way.
- Some information can be organized in different ways. (xml, text, word, pdf).

# Why Data Format is important?

- Source data can come in many different data formats.
- To run analytics effectively, a data scientist must first convert that source data to a common format for each model to process.
- With many different data sources and different analytic routines, that data wrangling can take 80 to 90 percent of the time spent on developing a new model.
- Having a model-driven architecture that simplifies the conversion of the source data to a standard, easy-to-use format ready for analytics reduces the overall time required and allows the data scientist to focus on machine learning model development and the training life cycle.

# Data Formats

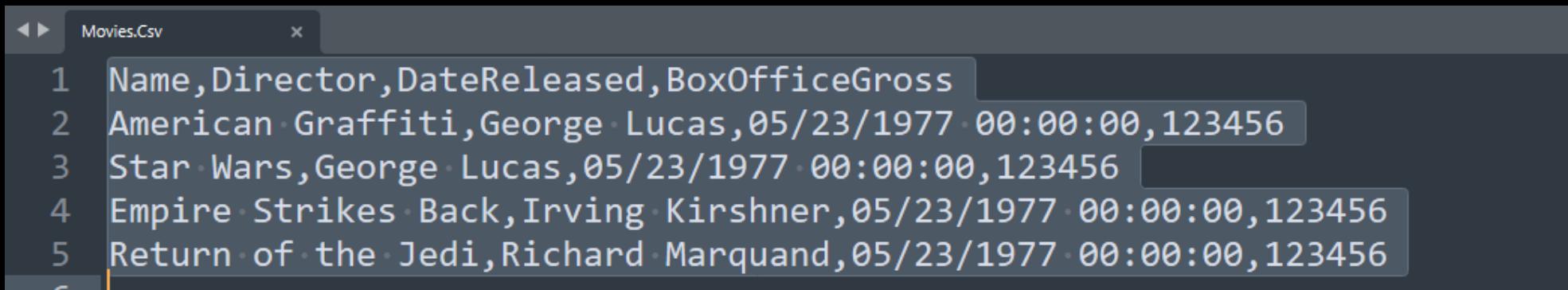
- CSV
- Delimited
- Spreadsheets
- JSON
- XML
- Tabular
- Relational
- Key-Value Pair
- Parquet



# Data Formats - CSV

- **CSV (Comma Separated Value)**

- Tabular data but presented in plain text format where each field is separated by comma (,)
- CSV files are often used to transfer data between sources.
- Common format to be manipulated using text editor or commonly Excel.



The screenshot shows a text editor window titled "Movies.Csv". The content of the file is a CSV (Comma Separated Value) file with five rows of movie data. The columns are labeled: Name, Director, DateReleased, and BoxOfficeGross. The data is as follows:

	Name	Director	DateReleased	BoxOfficeGross
1	American Graffiti	George Lucas	05/23/1977	00:00:00,123456
2	Star Wars	George Lucas	05/23/1977	00:00:00,123456
3	Empire Strikes Back	Irving Kirshner	05/23/1977	00:00:00,123456
4	Return of the Jedi	Richard Marquand	05/23/1977	00:00:00,123456

# Data Formats - DSV

- **DSV (Delimited Separated Value)**

- Tabular data but presented in plain text format where each field is separated by different delimiter such as semicolon ( ; ), pipe ( | ), slash ( / ), tab , colon ( : ) etc.
- CSV is also a common and special type of DSV.
- Typically a delimited file format is indicated by a specification.
- Some specifications provide conventions for avoiding delimiter collision, others do not.
- **Delimiter collision** is a problem that occurs when a character that is intended as part of the data gets interpreted as a delimiter instead.

Name	RollNumber	Grade
Aman	21	C
Rahul	22	A
Vishal	12	B
Jyoti	55	A
Swati	27	B
Kishan	23	D

# Data Formats - DSV

- **DSV (Delimited Separated Value)**
  - Following are the problems that often occurs with different delimiters
    1. Comma and space separated format often suffer from delimiter Collision.
    2. One problem with tab-delimited text files is that tabs are difficult to distinguish from spaces; therefore, there are sometimes problems with the files being corrupted when people try to edit them by hand.
    3. A delimiter cannot be binary zero, a line-feed character, a carriage-return, or a blank space

# Data Formats - Spreadsheets

- **Spreadsheets**
  - Spreadsheets are very common computer software for data preparation, presentation, manipulation and analysis.
  - We can also import data from various format and convert to wide range of formats.
  - Commonly used extensions are .odt, .csv, .xls, .xlsx

# Data Formats – JSON

- **JSON (JavaScript Object Notation)**

- JSON is a language-independent open data format that uses human-readable text to express data objects consisting of attribute-value pairs.
- Although originally derived from the JavaScript scripting language, JSON data can be generated and parsed with a wide variety of programming languages.
- JSON is also a plain text data but with a light weighted data interchange format.
- JSON, today, is commonly used with API to exchange data between computers.

```
{"employees": [  
    { "firstName": "John", "lastName": "Doe" },  
    { "firstName": "Anna", "lastName": "Smith" },  
    { "firstName": "Peter", "lastName": "Jones" }  
]
```

# Data Formats - XML

- **XML (Extensible Markup Language)**
  - A simple text-based format for representing structured information: documents, data, configuration, books, transactions, invoices, and much more.
  - XML is one of the most widely-used formats for sharing structured information today: between programs, between people, between computers and people, both locally and across networks.
  - XML was designed to store and transport data

```
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>
```

# Data Formats - XML

- **XML (Extensible Markup Language)**
  - If you are already familiar with HTML, you can see that XML is very similar.
  - However, the syntax rules of XML are strict: XML tools will not process files that contain errors, but instead will give you error messages so that you fix them.
  - This means that almost all XML documents can be processed reliably by computer software.
  - XML Does Not Use Predefined Tags.

# Data Formats - Tabular

- **Tabular**

- "Tabular" is simply information presented in the form of a table with rows and columns.
- It represents the structured data format.
- A data table is a neat and convenient way to present a large body of information that includes repeating data elements.
- For example, each entry in a list of company clients contains the client's name, title, address, phone number and other identifying information.
- This information can be listed in tabular format -- that is, in rows and columns -- by using separate columns for each data element.
- Columns are usually identified with headers such as "Client Name," "Street Address" and "Email Address," and each row contains all the information for a single client.

# Data Formats - Relational

- **Relational**
  - A relational model organizes data into one or more tables (or "relations") of columns and rows, with a unique key identifying each row.
  - Rows are also called records or tuples.
  - Columns are also called attributes. Generally, each table/relation represents one "entity type" (such as customer or product).
  - A system used to maintain relational databases is a **relational database management system (RDBMS)**.
  - A relationship is maintained between tables.
  - For example, each row of a class table corresponds to a class, and a class corresponds to multiple students, so the relationship between the class table and the student table is "one to many".

# Data Formats – Key Value

- **Key-Value format**

- A key-value database, a.k.a key-value store, associates a value (which can be anything from a number or simple string to a complex object) with a key, which is used to keep track of the object.
- In its simplest form, a key-value store is like a dictionary/array/map object as it exists in most programming paradigms, but which is stored in a persistent way and managed by a Database Management System (DBMS).

```
{  
    name: "John",  
    age : 35,  
    dob : ISODate("01-05-1990"),  
    profile_pic : "https://example.com/john.jpg",  
    social : {  
        twitter : "@mongojohn",  
        linkedin : "https://linkedin.com/abcd_mongojohn"  
    }  
}
```

# Data Formats – Key Value

- **Key-Value format**
  - Key-value databases use compact, efficient index structures to be able to quickly and reliably locate a value by its key, making them ideal for systems that need to be able to find and retrieve data in constant time.
  - Redis, for instance, is a key-value database that is optimized for tracking relatively simple data structures (primitive types, lists, heaps, and maps) in a persistent database.

# Data Formats – Parquet

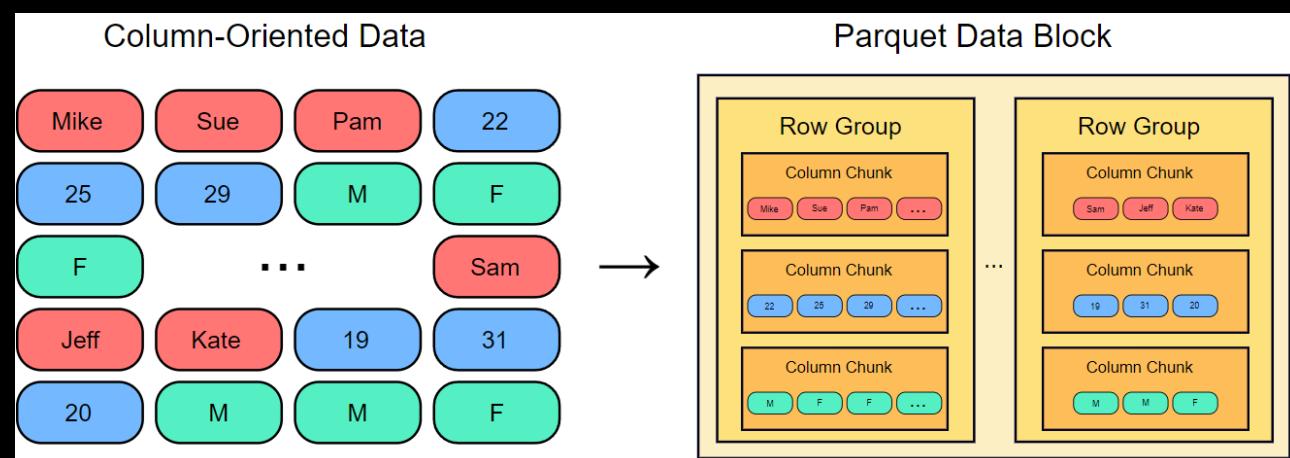
- **Parquet**

- Apache Parquet is an open source, column-oriented data file format designed for efficient data storage and retrieval.
- It provides efficient data compression and encoding schemes with enhanced performance to handle complex data in bulk.
- Apache Parquet is designed to be a common interchange format for both batch and interactive workloads.
- It is similar to other columnar-storage file formats available in Hadoop, namely RCFile and ORC.

# Data Formats – Parquet

- **Parquet**

- Characteristics of Parquet
- Free and open source file format.
- Language agnostic.
- Column-based format
- Used for analytics (OLAP)
- Highly efficient data compression and decompression.
- Supports complex data types and advanced nested data structures.



# Data Formats based on Structure

	Structured	Semi-structured	Unstructured
Examples	Relational databases	Csv, JSON, XML	Word documents, pdfs, web pages.
Flexibility	Low	Medium	High
Robustness	High	Medium	Low
Ease of analysis/query	High	Medium	Low
Human Readability	Low	Medium	High

# Row Based vs Column Based



## ROW-STORES

- "traditional"
- typically OLTP
- partitioned horizontally
- stores all row's values together as a tuple
- write data quickly
- indexing can drastically improve query response time



## COLUMNAR-STORES

- typically OLAP
- partitioned vertically
- stores multiple row's column values together
- reads are efficient due to only reading relevant columns
- column compression works well, speeding up reads

# Wide table format

- Wide, or unstacked data is presented with each different data variable in a separate column.

Person	Age	Weight	Height
Bob	32	168	180
Alice	24	150	175
Steve	64	144	165

# Narrow table format

- Narrow, stacked, or long data is presented with one column containing all the values and another column listing the context of the value.

Person	Variable	Value
Bob	Age	32
Bob	Weight	168
Bob	Height	180
Alice	Age	24
Alice	Weight	150
Alice	Height	175
Steve	Age	64
Steve	Weight	144
Steve	Height	165

*This is often easier to implement; addition of a new field does not require any changes to the structure of the table, however it can be harder for people to understand.*

# Why do we need to perform format conversion?

- **System Integration/Compatibility**: to make it easier for systems to talk to each other.
- **Readability**: easier to read and understand.
- **Better Representation**: Some data is better represented in certain formats.
- **Efficiency**: Some data formats are more efficient for analysis (faster/lesser cost).

# Data Validation

- Data validation is the practice of checking the integrity, accuracy and structure of data before it is used for a business operation.
- Data validation operation results can provide data used for data analytics, business intelligence or training a machine learning model.
- Data can be examined as part of a validation process in a variety of ways, including data type, constraint, structured, consistency and code validation.
- For example, if data is not in the right format to be consumed by a system, then the data can't be used easily, if at all.

# Data Validation

- Each type of data validation is designed to make sure the data meets the requirements to be useful.

# Techniques for data validation

## 1. Data Type Check

- A data type check confirms that the data entered has the correct data type.
- For example, a field might only accept numeric data. If this is the case, then any data containing other characters such as letters or special symbols should be rejected by the system.

<https://www.analyticsvidhya.com/blog/2021/05/data-validation-in-machine-learning-is-imperative-not-optional/>

<https://corporatefinanceinstitute.com/resources/data-science/data-validation/>

# Techniques for data validation (contd.)

## 2. Range Check

- A range check will verify whether input data falls within a predefined range.
- For example, latitude and longitude are commonly used in geographic data. A latitude value should be between -90 and 90, while a longitude value must be between -180 and 180. Any values out of this range are invalid.

# Techniques for data validation (contd.)

## 3. Format Check

- Many data types follow a certain predefined format.
- A common use case is date columns that are stored in a fixed format like “YYYY-MM-DD” or “DD-MM-YYYY.”
- A data validation procedure that ensures dates are in the proper format helps maintain consistency across data and through time.

# Techniques for data validation (contd.)

## 4. Consistency Check

- A consistency check is a type of logical check that confirms the data's been entered in a logically consistent way.
- An example is checking if the delivery date is after the shipping date for a parcel.

# END OF THE CHAPTER

Next Chapter: Chapter 4 – Machine Learning

# Data Analysis Technique

Unit - 3

# Feature Generation

- Feature Generation (also known as **feature construction** or **feature engineering**) is the process of transforming features into new features that better relate to the target.
- This can involve mapping a feature into a new feature using a function like log, or creating a new feature from one or multiple features using multiplication or addition.
- Feature Generation can improve model performance when there is a feature interaction.
- Two or more features interact if the combined effect is (greater or less) than the sum of their individual effects.

<https://www.turintech.ai/feature-generation-what-it-is-and-how-to-do-it/>

# Feature Generation (contd.)

- It is possible to make interactions with three or more features, but this tends to result in diminishing returns.
- Feature Generation is often overlooked as it is assumed that the model will learn any relevant relationships between features to predict the target variable.
- However, the generation of new flexible features is important as it allows us to use less complex models that are faster to run and easier to understand and maintain.

# Feature Selection

- Feature selection, one of the main components of feature engineering, is the process of selecting the most important features to input in machine learning algorithms.
- Feature selection chooses optimal set of features from all available.

# Why do we need feature selection?

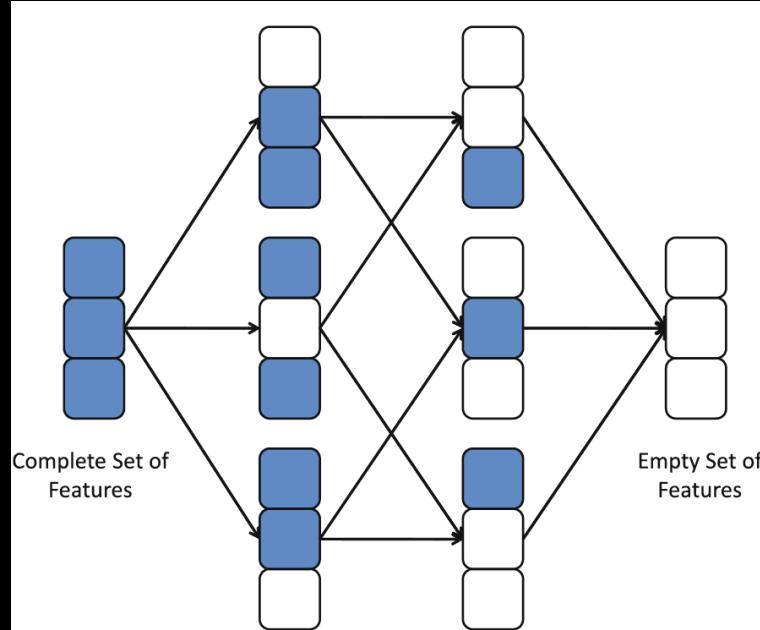
- Remove irrelevant data/noise
- Increase predictive accuracy of learning model
- Reduce Dimensionality
- Increase learning efficiency / reduce training time
- Reduce the complexity of resulting model by reducing the dimensionality of data as well as model..

# Perspectives of Feature Selection

- Search for the best subset of features
- Criteria for evaluating different subsets

# Perspectives of Feature Selection

Search the best subset of features



- Feature Selection can be considered as a search problem, where each state of the search space corresponds to a concrete subset of features selected.
- The selection can be represented as a binary array, with each element corresponding to the value 1, if the feature is currently selected by the algorithm and 0, if it does not occur.
- There should be a total of  $2^M$  subsets where  $M$  is the number of features of a data set.

# Perspectives of Feature Selection

Search the best subset of features

## Search Directions

- **Sequential Forward Generation (SFG)**: It starts with an empty set of features  $S$ . As the search starts, features are added into  $S$  according to some criterion that distinguish the best feature from the others.  $S$  grows until it reaches a full set of original features. The stopping criteria can be a threshold for the number of relevant features  $m$  or simply the generation of all possible subsets in brute force mode.
- **Sequential Backward Generation (SBG)**: It starts with a full set of features and, iteratively, they are removed one at a time. Here, the criterion must point out the worst or least important feature. By the end, the subset is only composed of a unique feature, which is considered to be the most informative of the whole set. As in the previous case, different stopping criteria can be used.

# Perspectives of Feature Selection

Search the best subset of features

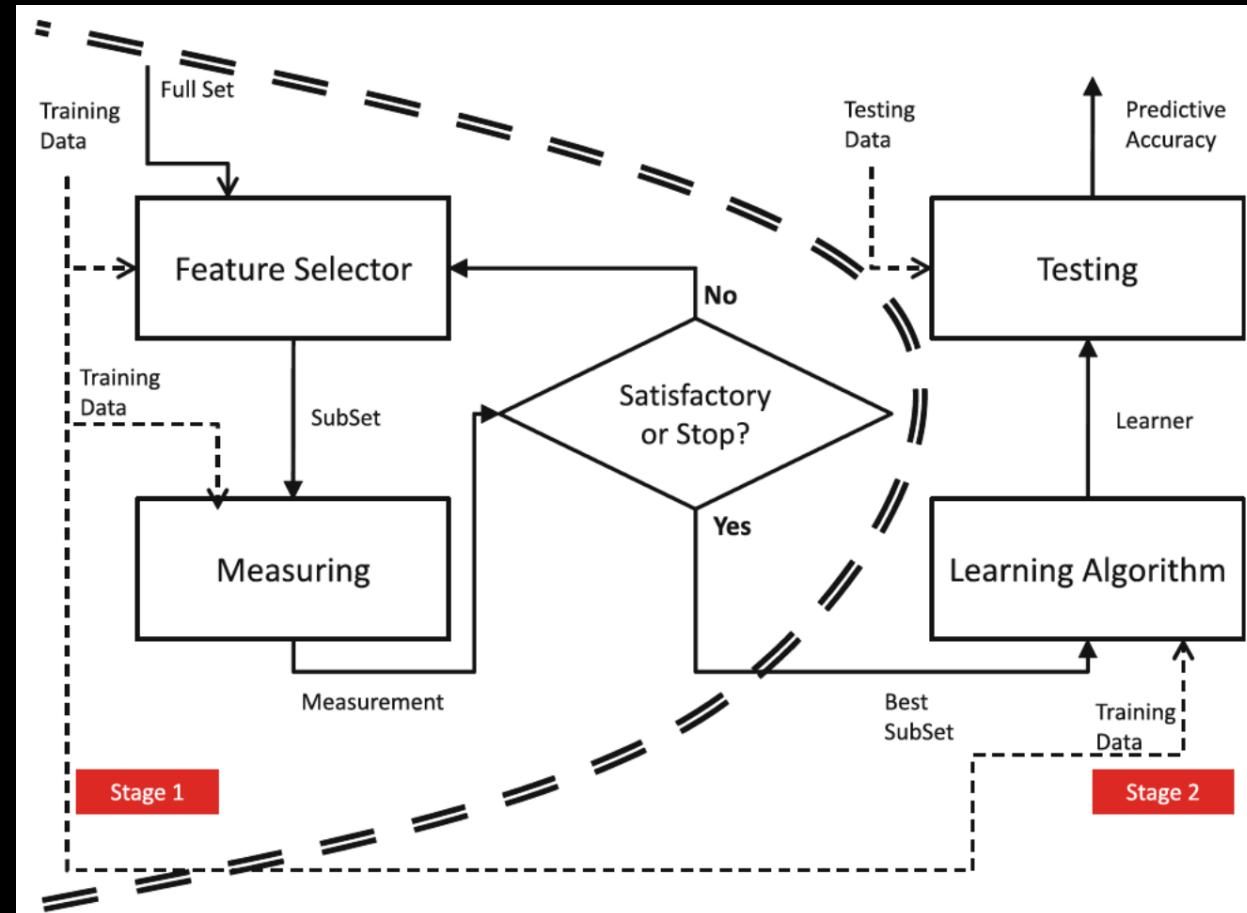
## Search Directions

- **Bidirectional Generation (BG)**: Begins the search in both directions, performing SFG and SBG concurrently. They stop in two cases:
  1. when one search finds the best subset comprised of  $m$  features before it reaches the exact middle, **or**
  2. both searches achieve the middle of the search space. It takes advantage of both SFG and SBG.
- **Random Generation (RG)**: It starts the search in a random direction. The choice of adding or removing a feature is a random decision. RG tries to avoid the stagnation into a local optima by not following a fixed way for subset generation. Unlike SFG or SBG, the size of the subset of features cannot be stipulated..

# Perspectives of Feature Selection - Filters

- measuring uncertainty, distances, dependence or consistency is usually cheaper than measuring the accuracy of a learning process. Thus, filter methods are usually faster.
- it does not rely on a particular learning bias, in such a way that the selected features can be used to learn different models from different DM techniques.
- it can handle larger sized data, due to the simplicity and low time complexity of the evaluation measures.

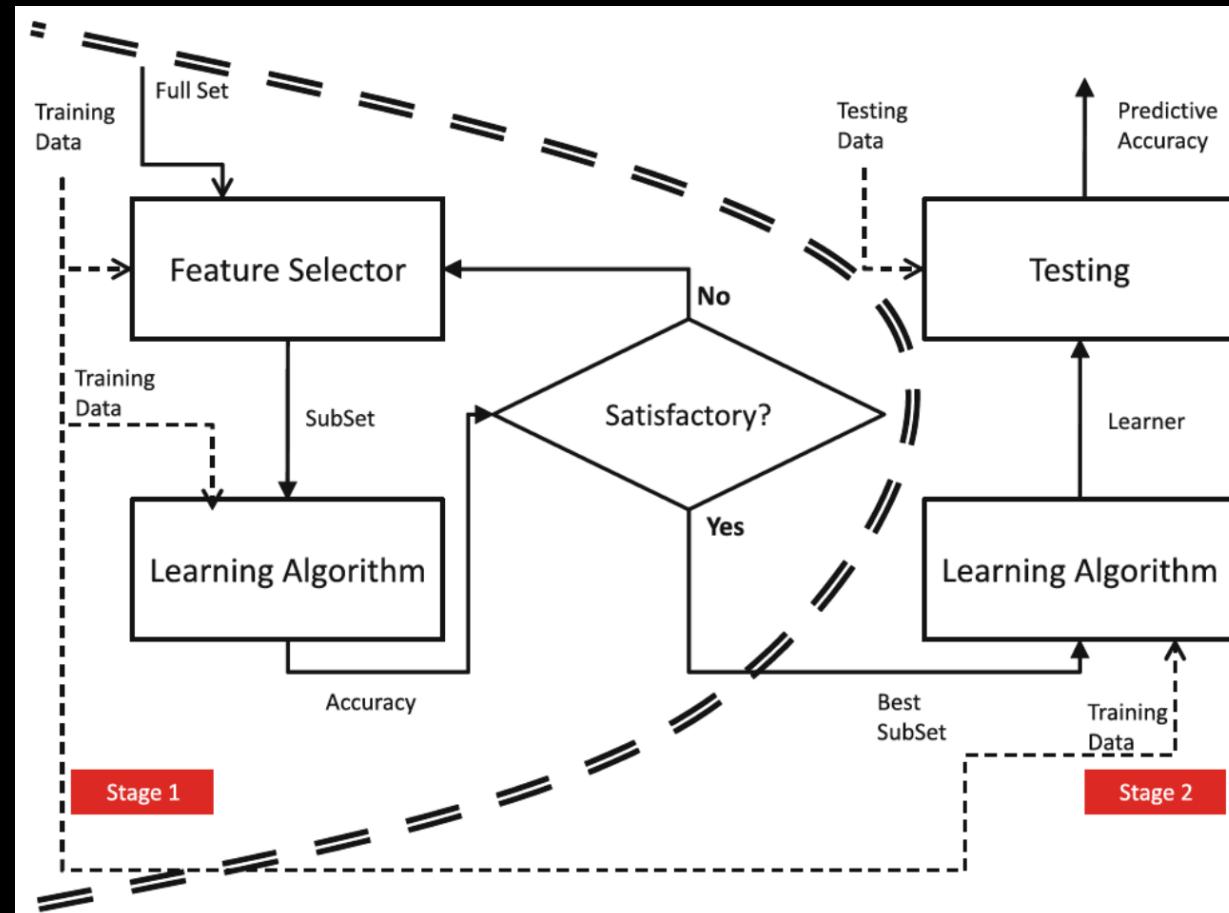
# Perspectives of Feature Selection – Filters



# Perspectives of Feature Selection - Wrappers

- can achieve the purpose of improving the particular learner's predictive performance.
- usage of internal statistical validation to control the overfitting, ensembles of learners and hybridizations with heuristic learning like Bayesian classifiers or Decision Tree induction.
- filter models cannot allow a learning algorithm to fully exploit its bias, whereas wrapper methods do.

# Perspectives of Feature Selection – Wrappers



# Predictive Analytics

- Predictive analytics is the practice of using data, statistical algorithms, and machine learning techniques to predict future outcomes or behavior.
- Predictive analytics is a type of data analytics that uses historical data to make predictions about future outcomes.
- It enables organizations to gain valuable insights, make informed decisions, and anticipate future trends.
- Predictive analytics can be used to identify risks and opportunities, make better decisions, and improve efficiency.
- Predictive analytics is often used in business, but it can also be used in other areas, such as healthcare, education, and government.

# How does predictive analytics work?

- Predictive analytics uses a variety of techniques to analyze data, including:
  - Statistical modeling
  - Machine learning
  - Data mining
- These techniques are used to identify patterns in the data and to build models that can be used to make predictions.

# Key Steps in Predictive Analytics

## a) Problem Definition:

- Clearly define the problem or objective you want to solve using predictive analytics. Identify the specific outcomes or behaviors you want to predict.

## b) Data Collection and Preparation:

- Gather relevant data from various sources, such as databases, APIs, or external datasets.
- Clean, preprocess, and transform the data to ensure its quality and suitability for analysis.

## c) Feature Selection and Engineering:

- Select the most relevant features (variables) that have a significant impact on the outcome.
- Create new features or transform existing ones to improve predictive power.

# Key Steps in Predictive Analytics

## d) Model Selection and Training:

- Choose an appropriate predictive model based on the nature of the problem and data.
- Split the data into training and testing sets.
- Train the model using the training data, adjusting model parameters as needed.

## e) Model Evaluation and Validation:

- Evaluate the model's performance using appropriate metrics (e.g., accuracy, precision, recall, F1-score).
- Validate the model by testing its performance on the unseen testing data.

## f. Deployment and Monitoring:

- Deploy the predictive model into production or real-world applications.
- Continuously monitor the model's performance and update it if necessary.

# Benefits of Predictive Analytics

- Predictive analytics can provide a number of benefits, including:
  - Improved decision-making
  - Increased efficiency
  - Reduced risk
  - Increased profits
  - Enhanced Customer Experience
- Predictive analytics can help businesses to:
  - Target marketing campaigns more effectively
  - Identify potential customers
  - Reduce fraud
  - Improve customer service

# Applications of Predictive Analytics

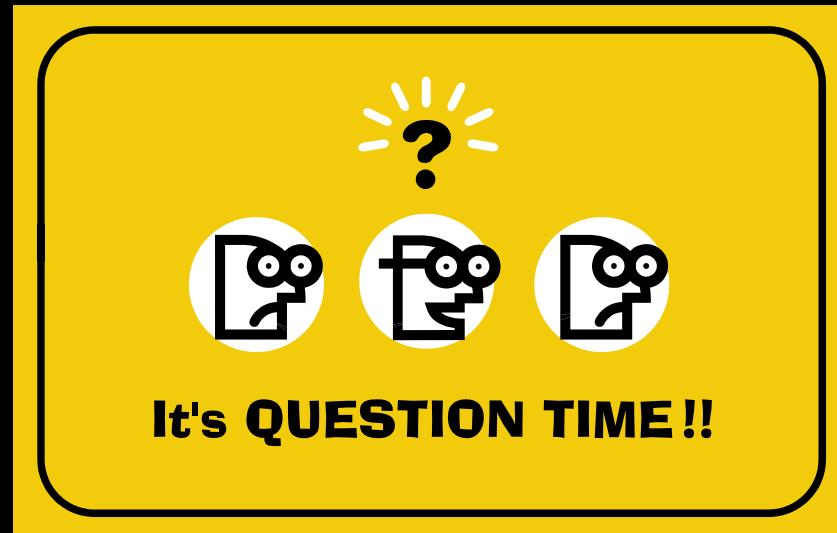
- **Customer Churn Prediction:** Forecasting customer attrition to take proactive retention measures.
- **Sales Forecasting:** Predicting future sales based on historical data and market trends.
- **Fraud Detection:** Identifying fraudulent activities or transactions in real-time.
- **Demand Forecasting:** Estimating future demand for products or services.
- **Predictive Maintenance:** Anticipating equipment failures and optimizing maintenance schedules.
- **Risk Assessment:** Assessing risks and making predictions in finance, insurance, and healthcare.

# Time Series Analysis

# Time series and Trend analysis

A time series consists of a set of observations measured at specified, usually equal, time interval.

Time series analysis attempts to identify those factors that exert influence on the values in the series.



Time series analysis is a basic tool for forecasting.  
Industry and government must forecast future activity  
to make decisions and plans to meet projected  
changes.

An analysis of the trend of the observations is needed  
to acquire an understanding of the progress of events  
leading to prevailing conditions.

The **trend** is defined as the long term underlying growth movement in a time series.

Accurate trend spotting can only be determined if the data are available for a sufficient length of time.

Forecasting does not produce definitive results. Forecasters can and do get things wrong from election results and football scores to the weather.



# Time series examples

- Sales data
- Gross national product
- Share prices
- Exchange rate
- Unemployment rates
- Population
- Foreign debt
- Interest rates

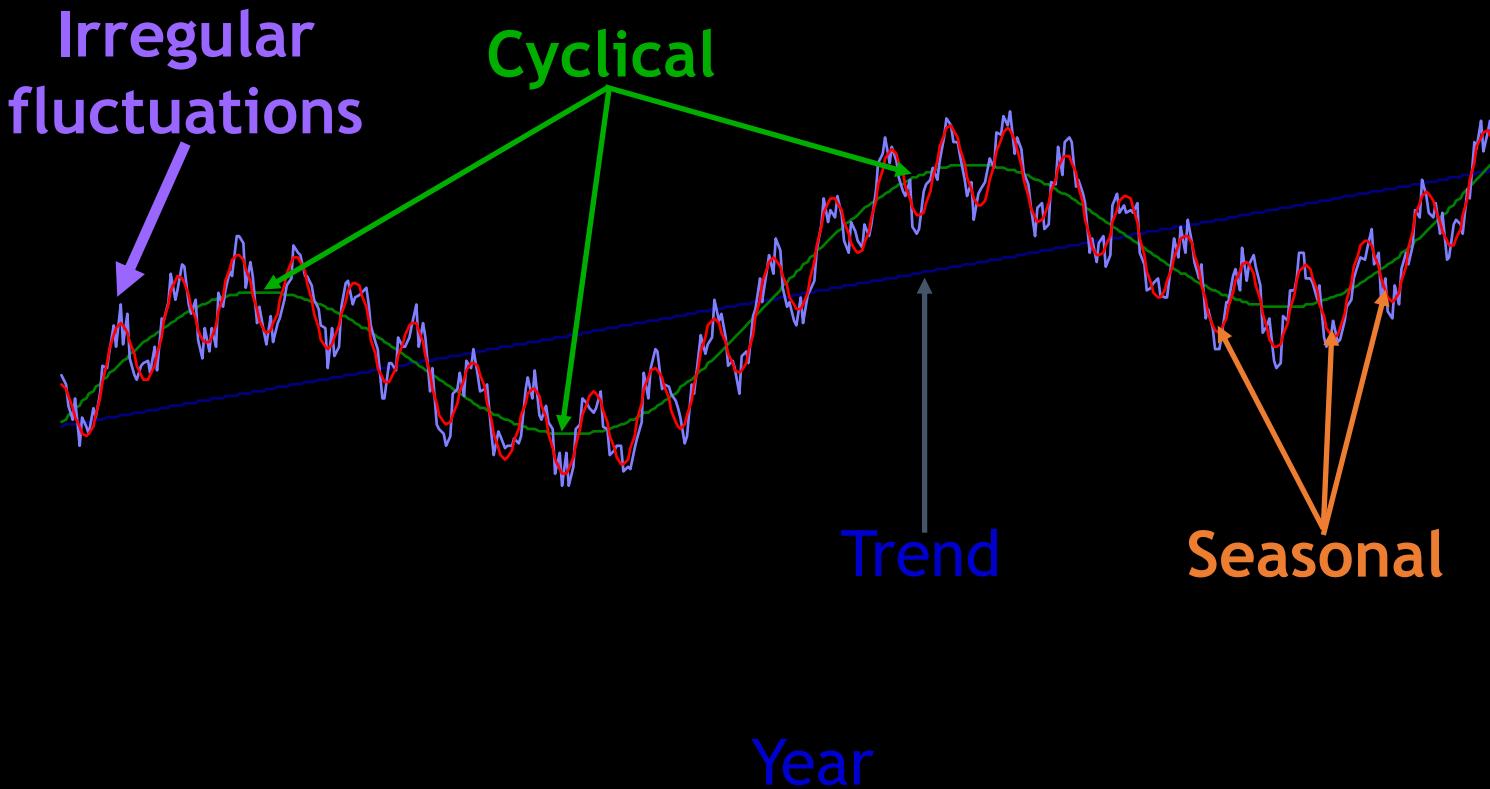


# Time series components

Time series data can be broken into these four components:

1. Secular trend
2. Seasonal variation
3. Cyclical variation
4. Irregular variation

# Components of Time-Series Data



Predicting long term trends without smoothing?

What could go wrong?

Where do you commence your prediction from the bottom of a variation going up or the peak of a variation going down.....

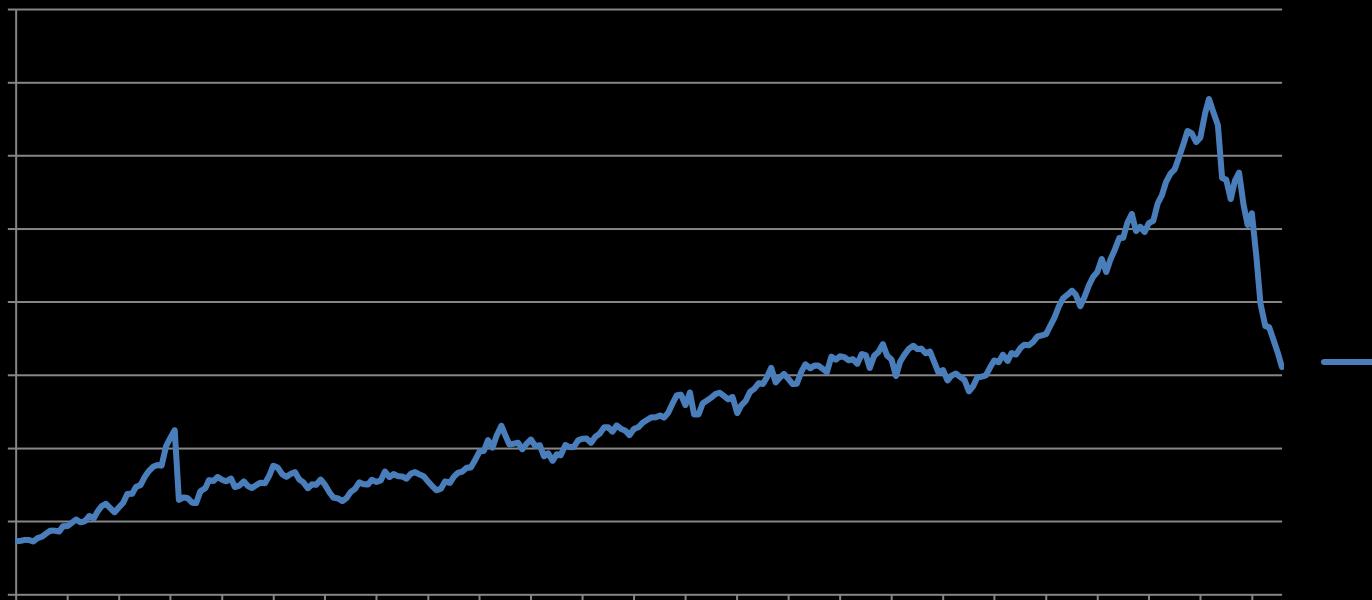
## 1. Secular Trend

This is the long term growth or decline of the series.

- In economic terms, long term may mean >10 years
- Describes the history of the time series
- Uses past trends to make prediction about the future
- Where the analyst can isolate the effect of a secular trend, changes due to other causes become clearer

## Secular Trend

A secular trend identifies the underlying trend (direction) of the data: – increasing, decreasing or remaining constant. It is the long term direction of the data, usually described by the “line of best fit”. And is deduced over a large number of periods. The following chart is a long term graph of the ASX200.



## Look out

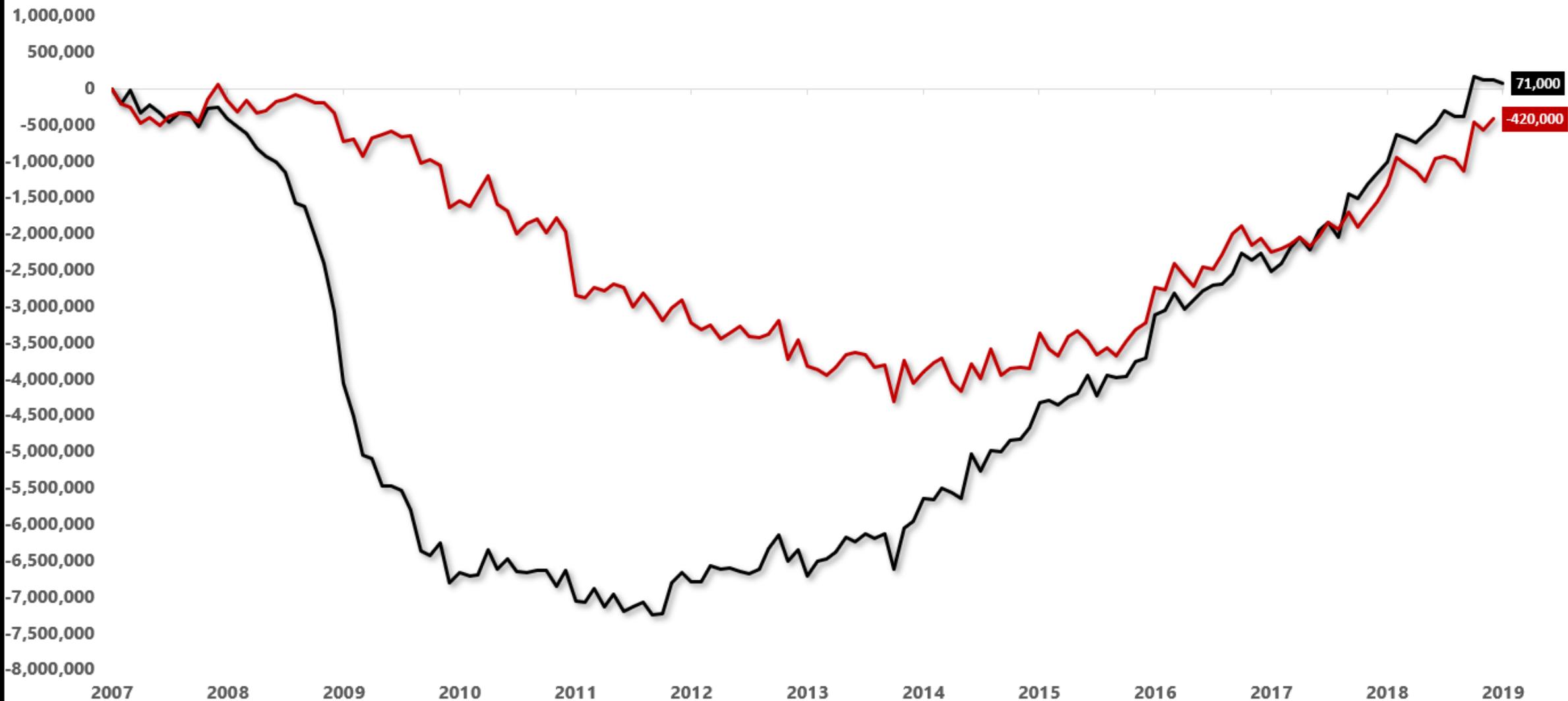
While trend estimates are often reliable, in some instances the usefulness of estimates is reduced by:

- by a high degree of irregularity in original or seasonally adjusted series or
- by abrupt change in the time series characteristics of the original data



## Employment Level Vs. Population Cumulative Change Since 2007

— Employment Level: 25 to 54 Years Cumulative Change    — Active Population: 25 to 54 Cumulative Change



## 2. Seasonal Variation

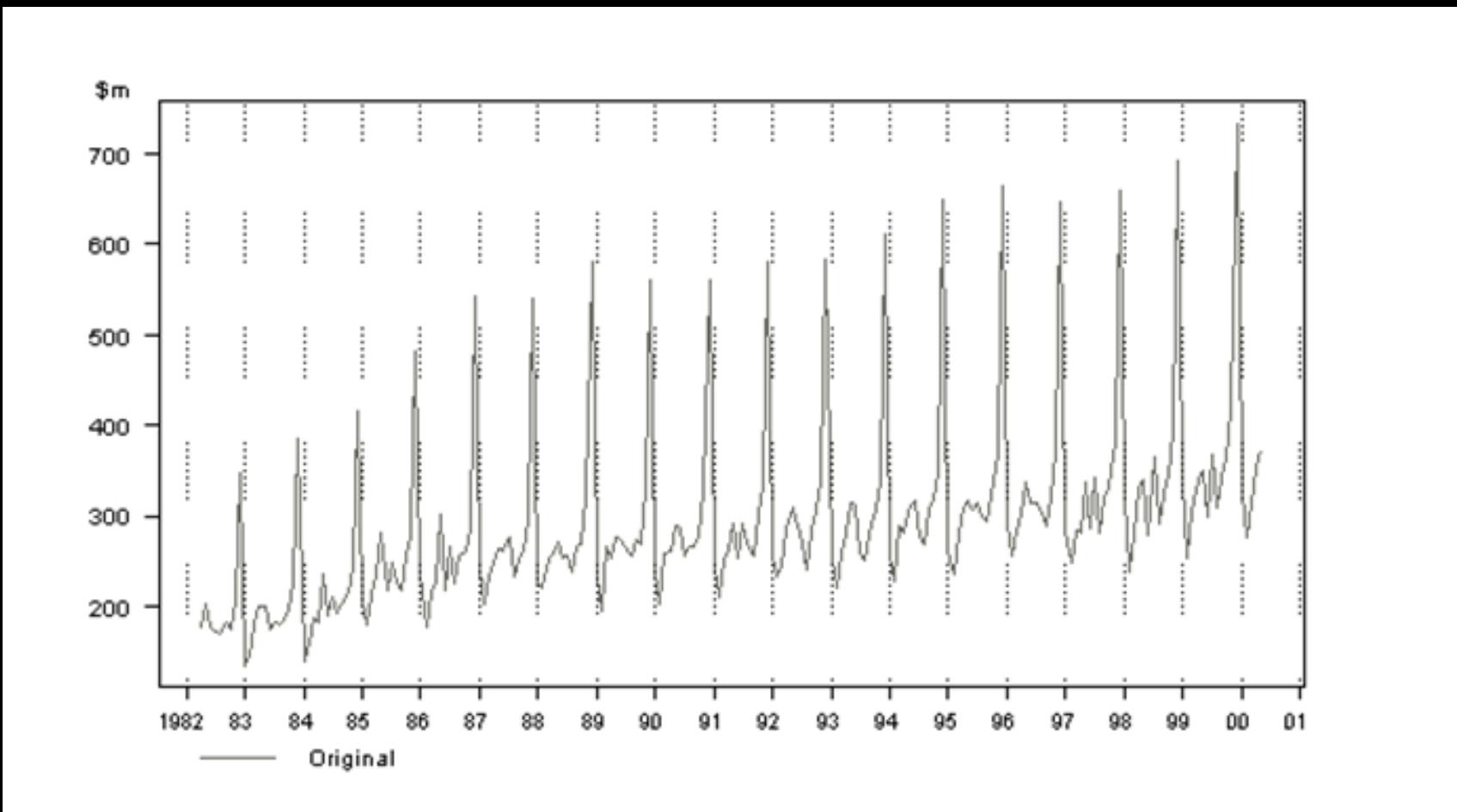
The seasonal variation of a time series is a pattern of change that **recurs** regularly over time.

Seasonal variations are usually due to the differences between seasons and to festive occasions such as Easter and Christmas.

Examples include:

- Air conditioner sales in Summer
- Heater sales in Winter
- Flu cases in Winter
- Airline tickets for flights during school vacations

# Monthly Retail Sales in NSW Retail Department Stores



## **Seasonal Movement**

Seasonal movement refers to regular periodic fluctuations that occur in each time period – yearly, monthly, daily. Some examples are speciality cards for Valentine's Day, monthly travel passes and off-peak heating.

Seasonal variations greatly impact on the outcomes of recorded data and often belie the underlying trend. Businesses need to identify the seasonal impact:

- So that a measurement (index) can be used to adjust the expected outcome.
- In order to recognise the direction of the underlying trend.

### 3. Cyclical variation

Cyclical variations also have recurring patterns but with a longer and more **erratic time scale** compared to Seasonal variations.

The name is quite misleading because these cycles can be far from regular and it is usually impossible to predict just how long periods of expansion or contraction will be.

There is no guarantee of a regularly returning pattern.

# Cyclical variation

Example include:

- Floods
- Wars
- Changes in interest rates
- Economic depressions or recessions
- Changes in consumer spending

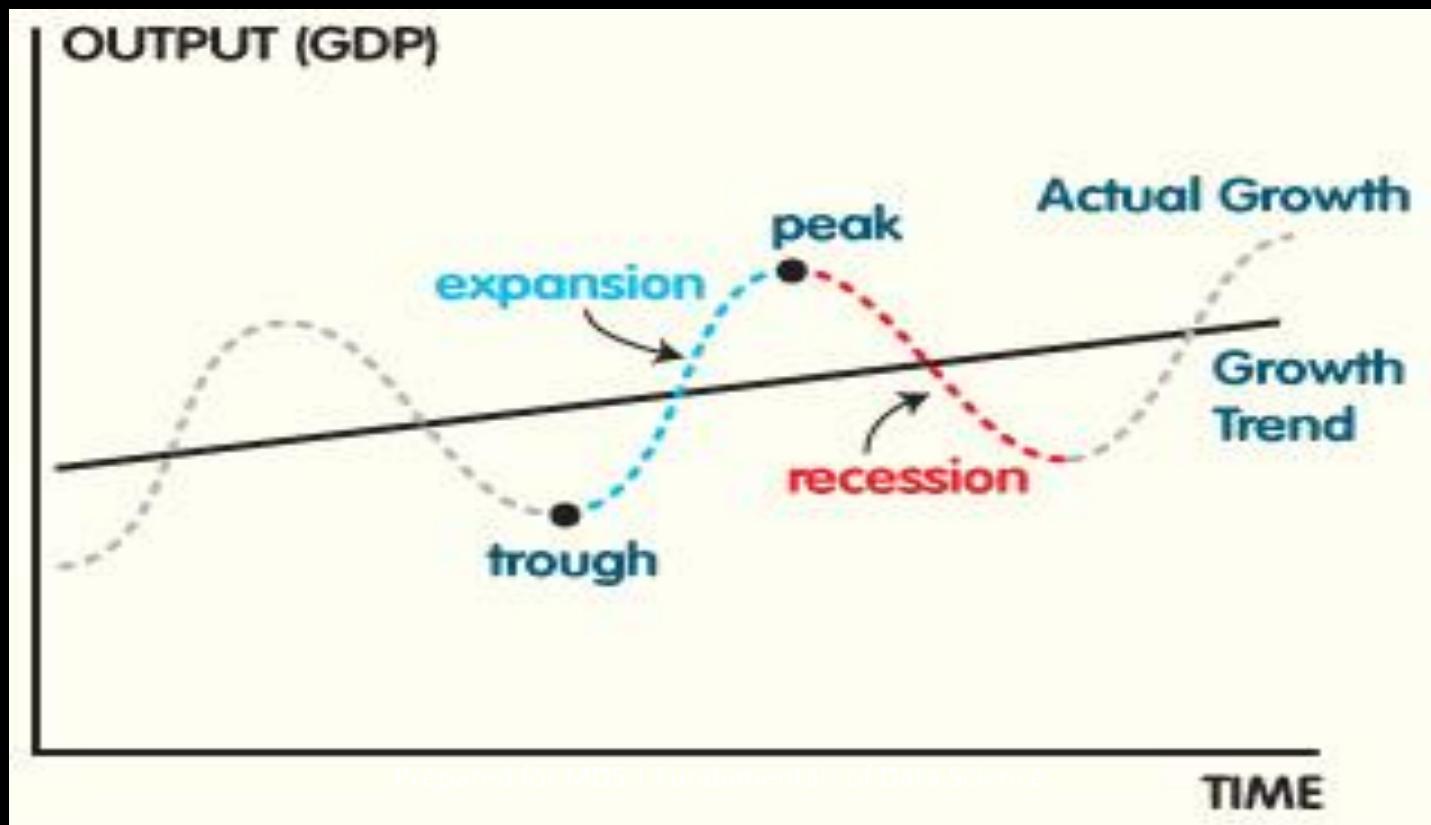


## **Cyclical Movement**

This reflects the level of business activity and economic movement over time by fluctuating patterns, known as the economic cycle. These variations measure periods of expansion and contraction in industry and the economy. Their regularity and intensity are not predictable, however certain economic indicators contribute to their existence – level of investment, confidence in the economy, GDP, trade indexes and government policy.

## Cyclical variation

This chart represents an economic cycle, but we know it doesn't always go like this. The timing and length of each phase is not predictable.



## 4. Irregular variation

An irregular (or random) variation in a time series occurs over varying (usually short) periods.

It follows no pattern and is by nature unpredictable.

It usually occurs randomly and may be linked to events that also occur randomly.

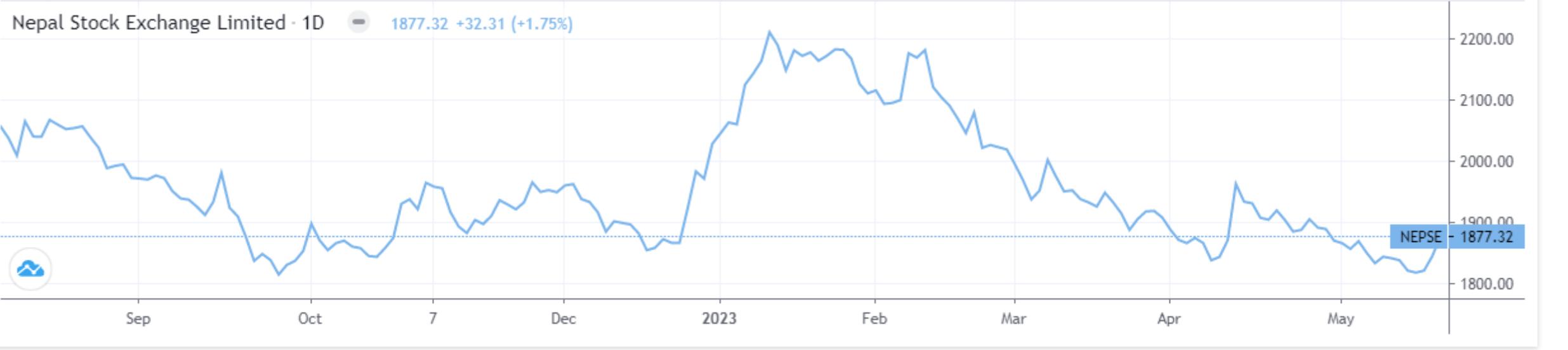
Irregular variation cannot be explained mathematically.

## Irregular variation

If the variation cannot be accounted for by secular trend, season or cyclical variation, then it is usually attributed to irregular variation.

Example include:

- Sudden changes in interest rates
- Collapse of companies
- Natural disasters
- Sudden shifts in government policy
- Dramatic changes to the stock market
- Effect of Middle East unrest on petrol prices



## **Irregular Movements**

These patterns refer to random variations that impact greatly on the level of business activity, often called natural variation. Some examples are extreme weather patterns (flood, fire, cyclone), extreme business variation (stock market crash, drop in \$A), political climate (sudden elections, wars, death of a leader), and industry changes (pilot strikes, waterside strikes.). The resulting patterns will exert a great pressure on the predicted underlying trends and for this reason must be accounted for when planning for the future. However, the irregular movements are unpredictable.

# Time Series Model - Examples

- Autoregressive (AR) model
- Moving average (MA) model
- Autoregressive moving average (ARMA) model
- Autoregressive integrated moving average (ARIMA) model
- Seasonal autoregressive integrated moving average (SARIMA) model
- Vector autoregressive (VAR) model
- Vector error correction (VECM) model

# Autoregressive (AR) model

- Autoregressive (AR) models are defined as **regression models** in which the dependent or response variable is a linear function of past values of the dependent/response variable.
- The order of an autoregressive model is denoted as '**p**', which represents the number of lags used to predict the current value.
- For example, if  $p=0$ , then it means that we are predicting the current time-step ( $t$ ) based on the previous time-step ( $t-0$ ).
- If  $p=n$ , then we are predicting time-step ( $t$ ) based on  $n$  past time-steps.
- The general form of an autoregressive model can be represented as:

$$Y_t = c + \phi_p Y_{t-p} + \varepsilon_t$$

# Moving average (MA) model

- A moving average (MA) is a type of model used for time-series forecasting.
- The moving average models are primarily used for stationary data, the data where we don't see significant trends or seasonality.
- There are two different kinds of moving average model. They are **Simple Moving Average (SMA)** and **Weighted Moving Average** model.

# Moving average (MA) model

- **Simple Moving Average (SMA)**

- a type of moving Average model that uses a fixed number of data points for the averaging calculation.
- Easy to calculate and can be implemented in wide in almost any language

$$\hat{y}_{t+1} = \frac{y_t + y_{t-1} + \dots + y_{t-k+1}}{k}$$

- **Weighted Moving Average (SMA)**

- a type of moving Average model that uses a weighting scheme to give more importance to more recent data points.
- This type of MA can be used for time-series forecasting, and can help to reduce the impact of older data points on the average.

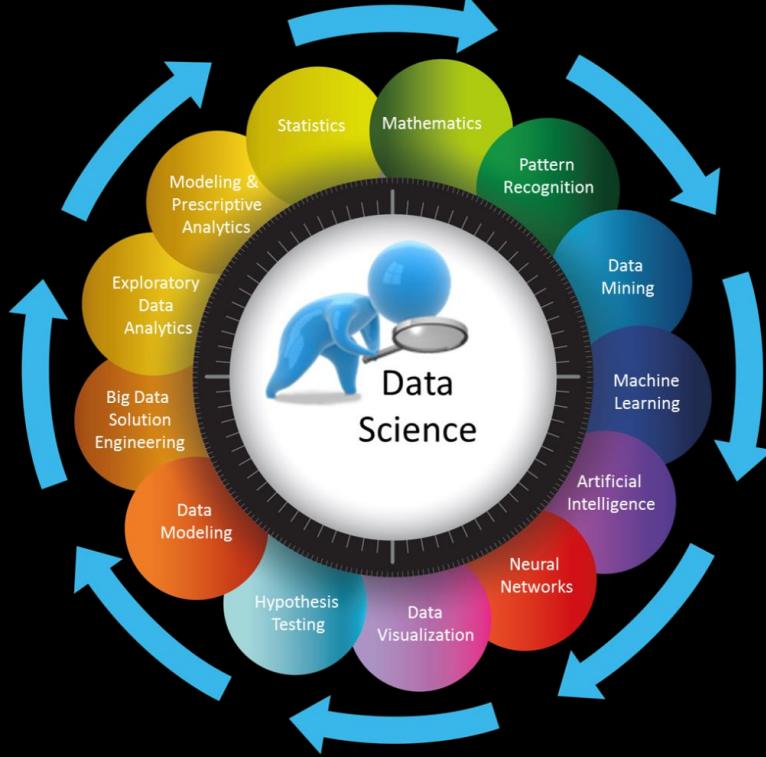
# Time Series Analysis - Applications

- Time series analysis is used for non-stationary data—things that are constantly fluctuating over time or are affected by time.
- Industries like finance, retail, and economics frequently use time series analysis because currency and sales are always changing.
- Stock market analysis is an excellent example of time series analysis in action, especially with automated trading algorithms.
- Likewise, time series analysis is ideal for forecasting weather changes, helping meteorologists predict everything from tomorrow's weather report to future years of climate change.

# Time Series Analysis - Applications

- Weather data
- Rainfall measurements
- Temperature readings
- Heart rate monitoring (EKG)
- Brain monitoring (EEG)
- Quarterly sales
- Stock prices
- Automated stock trading
- Industry forecasts
- Interest rates

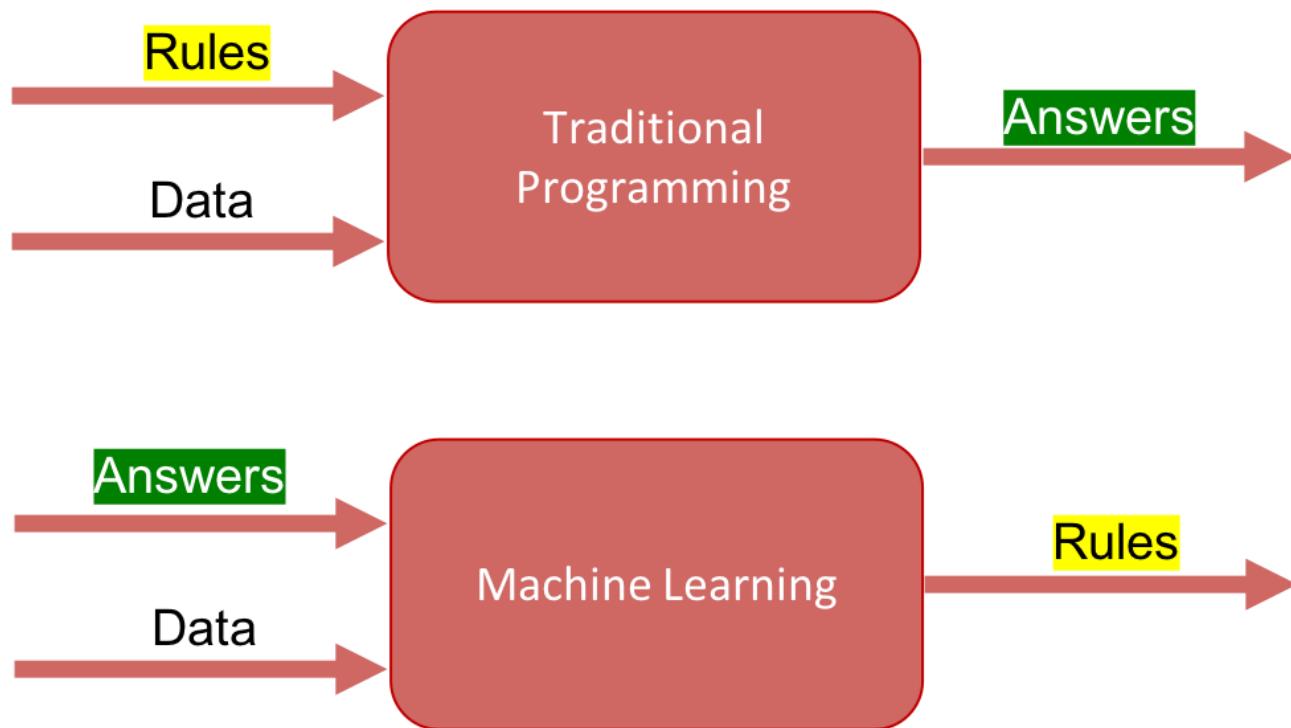
# End of the Chapter



# Machine Learning

Unit 4

# What is Machine Learning?



Machine learning is an application of AI that enables systems to learn and improve from experience without being explicitly programmed.

# What is Machine Learning?

- Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

- IBM

- In 1959, Arthur Samuel (an AI pioneer at IBM), defined the term “Machine Learning” as

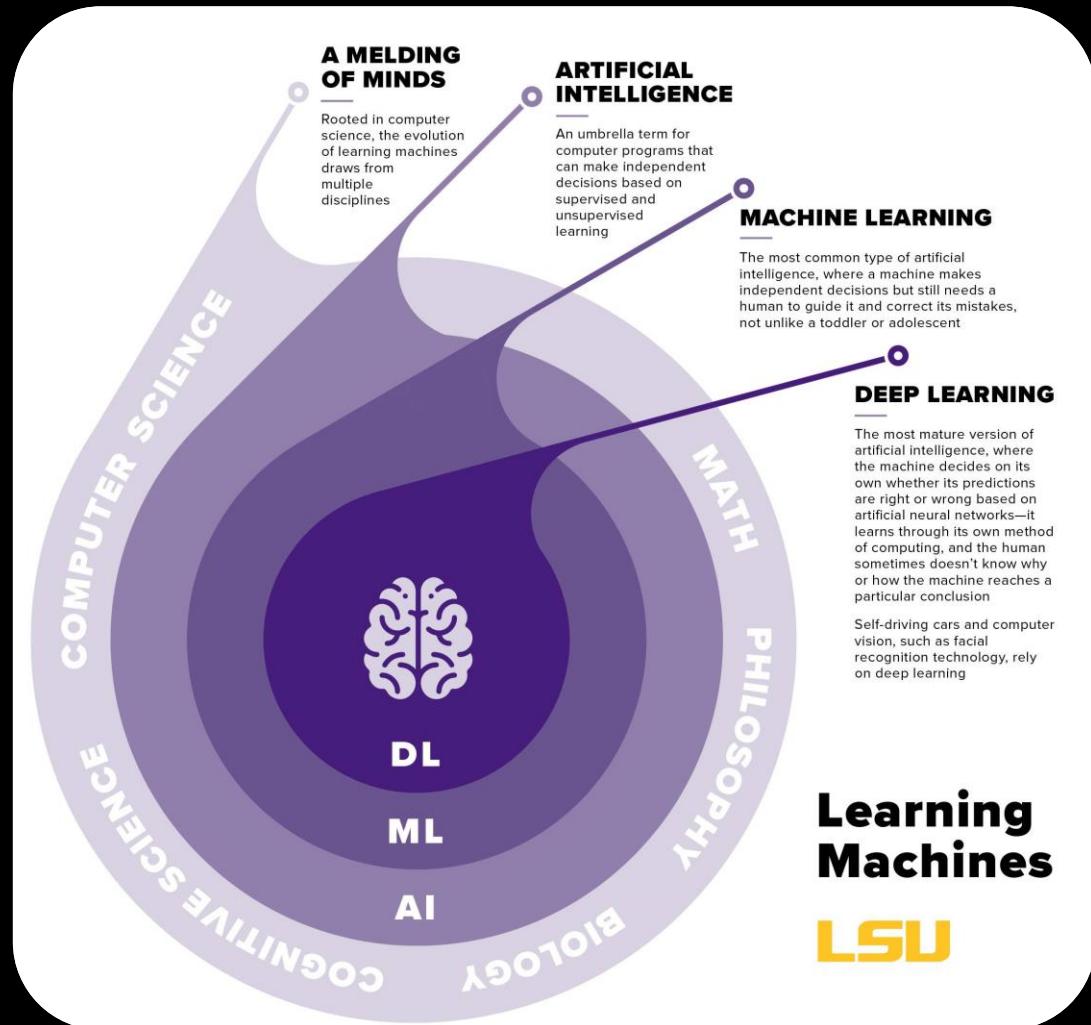
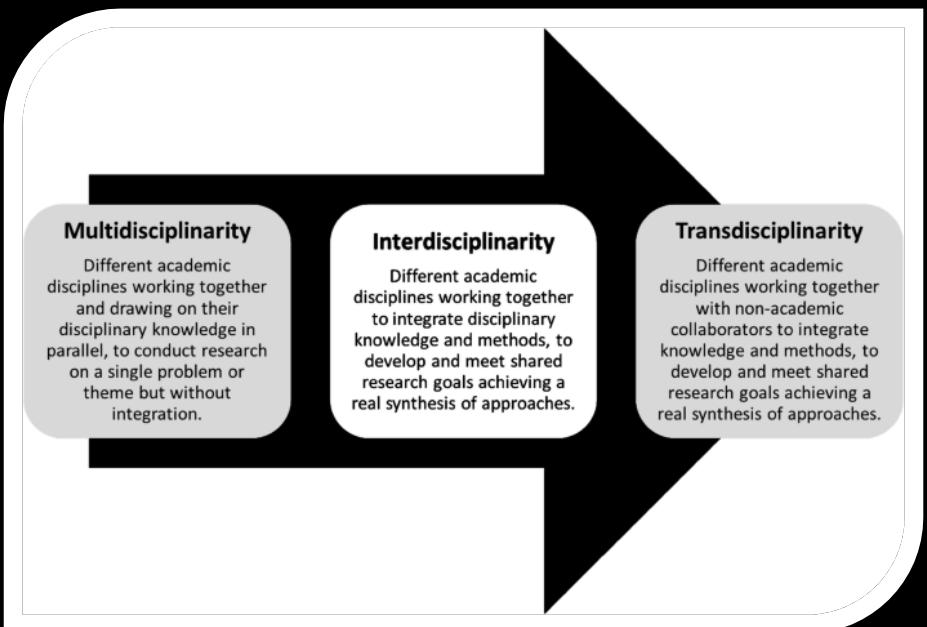
*“Field of study that gives computers the ability to learn without being explicitly programmed.”*

# What is Machine Learning? (contd.)

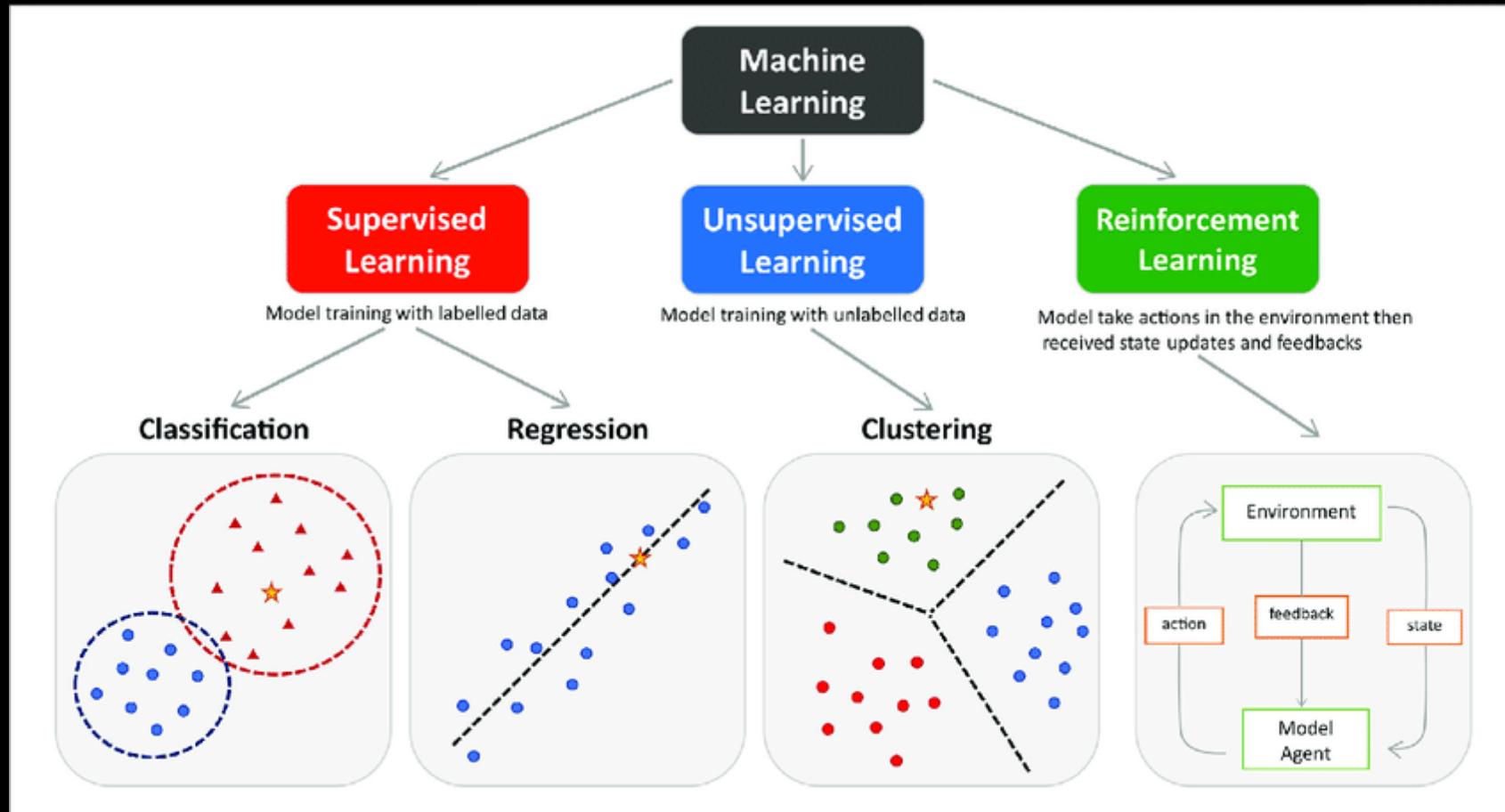
- After that, Tom M. Mitchell in 1997 gave the definition of Machine Learning which is widely quoted all over the globe.
- This is the more formal definition of the algorithms that are studied in the Machine Learning field,

*“A computer program is said to learn from experience E with respect to some class of tasks T and a performance measure P if its performance in tasks T, as measured by P, improves with experience E.”*

# Machine Learning is the subfield of AI and is multidisciplinary



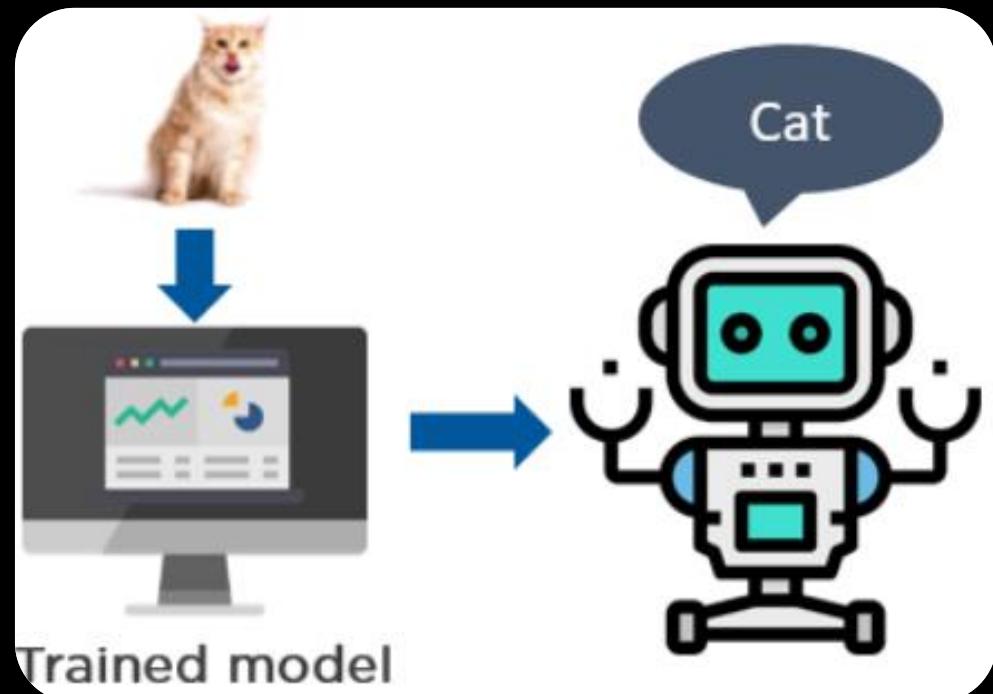
# Different Types of Machine Learning



Also, there is something called **Semi Supervised Learning** and **Self Supervised Learning**.

# 1. Supervised Learning

- A type of learning that uses labelled data (or input with outputs) to train machine learning algorithms.
- The input are provided with corresponding outputs while training ML models.
- Since these algorithm needs external supervision on mapping input and expected output, they are called **supervised**.



# 1. Supervised Learning (contd.)

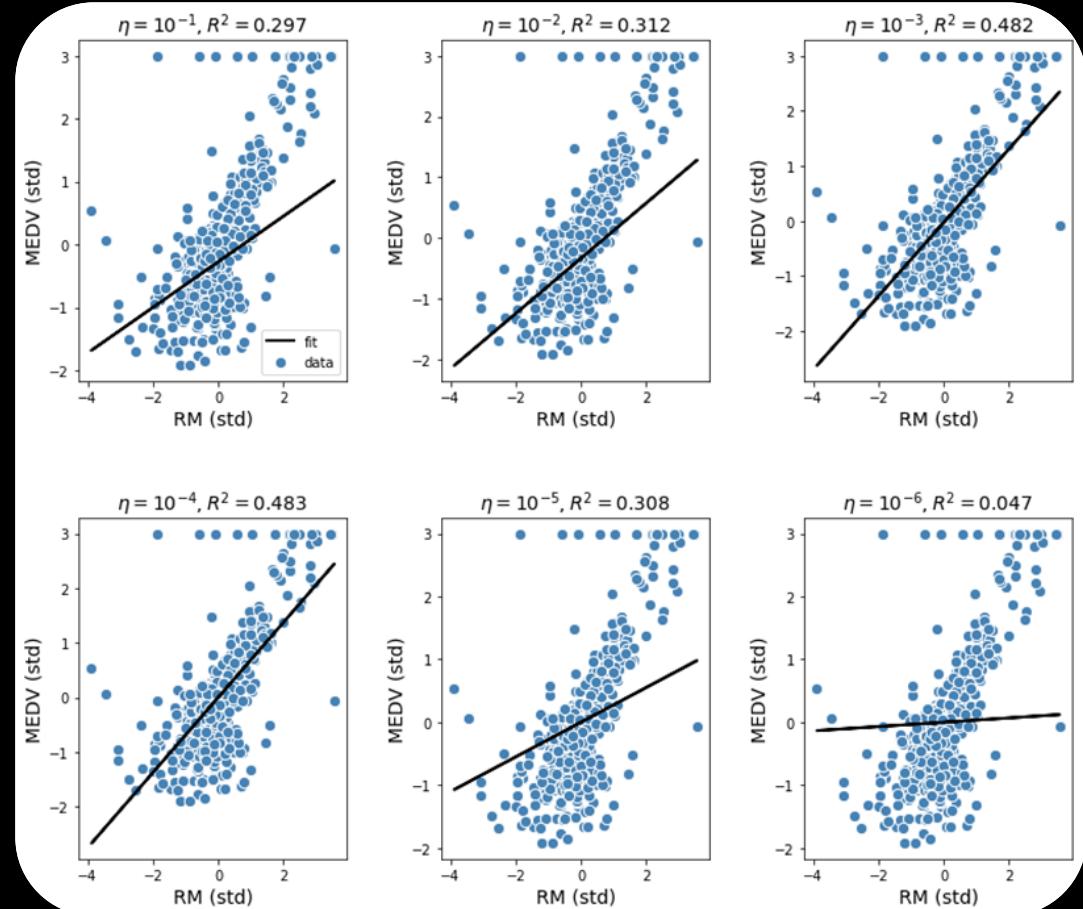
- Even though the data needs to be labeled accurately for this method to work, usually done by experts, supervised learning is extremely powerful when used in the right circumstances.

# 1. Supervised Learning (contd.)

## Working Mechanism

- In supervised learning, the ML algorithm is given a training dataset to work with.
- This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with.
- The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labeled parameters required for the problem.

# 1. Supervised Learning (contd.)



# 1. Supervised Learning (contd.)

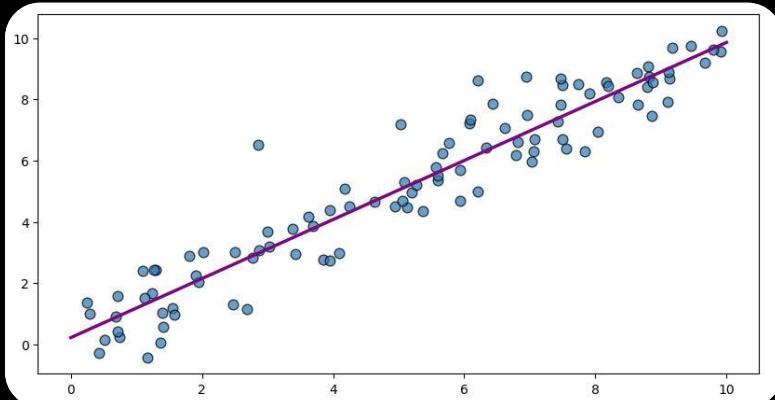
- **Working Mechanism**

- The algorithm then finds relationships between the parameters given, essentially establishing a cause and effect relationship between the variables in the dataset.
- At the end of the training, the algorithm has an idea of how the data works and the relationship between the input and the output.
- This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset.
- This means that supervised machine learning algorithms will continue to improve even after being deployed, discovering new patterns and relationships as it trains itself on new data.

# Types of Supervised Learning

## Regression

- Learning for prediction of value
  - Has continuous output
- Needs corresponding output with input



## Classification

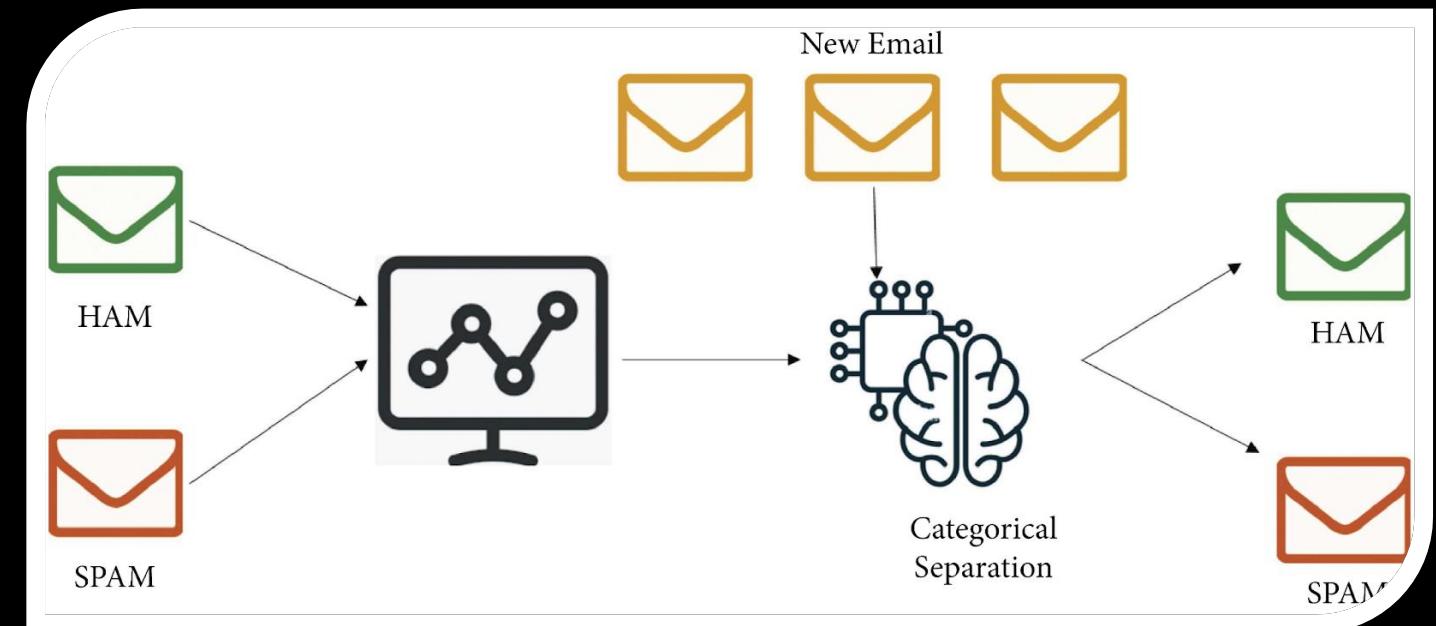
- Learning for classification of objects
- Has discrete output
- Needs Labelled data with input



# 1. Supervised Learning (contd.)

- Applications

- Price Prediction
- Image classification and segmentation
- Disease identification and medical diagnosis
- Fraud detection
- Spam detection
- Speech recognition
- Sentiment Analysis
- Image Captioning
- Image Generation
- Text Generation etc.



# 1. Supervised Learning (contd.)

- **Algorithms**

- Linear regression
- Logistic regression
- Naive Bayes
- Linear discriminant analysis
- Decision trees
- K-nearest neighbor algorithm
- Neural networks (Multilayer perceptron)
- Random forest algorithm
- Support Vector Machine etc.

## 2. Unsupervised Algorithm

- A class of algorithm which has input data but no corresponding output or label.
- The goal for unsupervised learning is to model/ understand/ manipulate the underlying structure or distribution of the data in order to learn more about the data.
- There is no supervision from expert through label or output, thus, called unsupervised algorithm.
- Unsupervised machine learning purports to uncover previously unknown patterns in data, but most of the time these patterns are poor approximations of what supervised machine learning can achieve.

# Types of Unsupervised Algorithms

- **Clustering** allows you to automatically split the dataset into groups according to similarity. Often, however, cluster analysis overestimates the similarity between groups and doesn't treat data points as individuals. For this reason, cluster analysis is a poor choice for applications like customer segmentation and targeting.
- **Anomaly detection** can automatically discover unusual data points in your dataset. This is useful in pinpointing fraudulent transactions, discovering faulty pieces of hardware, or identifying an outlier caused by a human error during data entry.
- **Association mining** identifies sets of items that frequently occur together in your dataset. Retailers often use it for basket analysis, because it allows analysts to discover goods often purchased at the same time and develop more effective marketing and merchandising strategies.
- **Latent variable models** are commonly used for data preprocessing, such as reducing the number of features in a dataset (dimensionality reduction) or decomposing the dataset into multiple components.

## 2. Unsupervised Algorithms (contd.)

- Applications

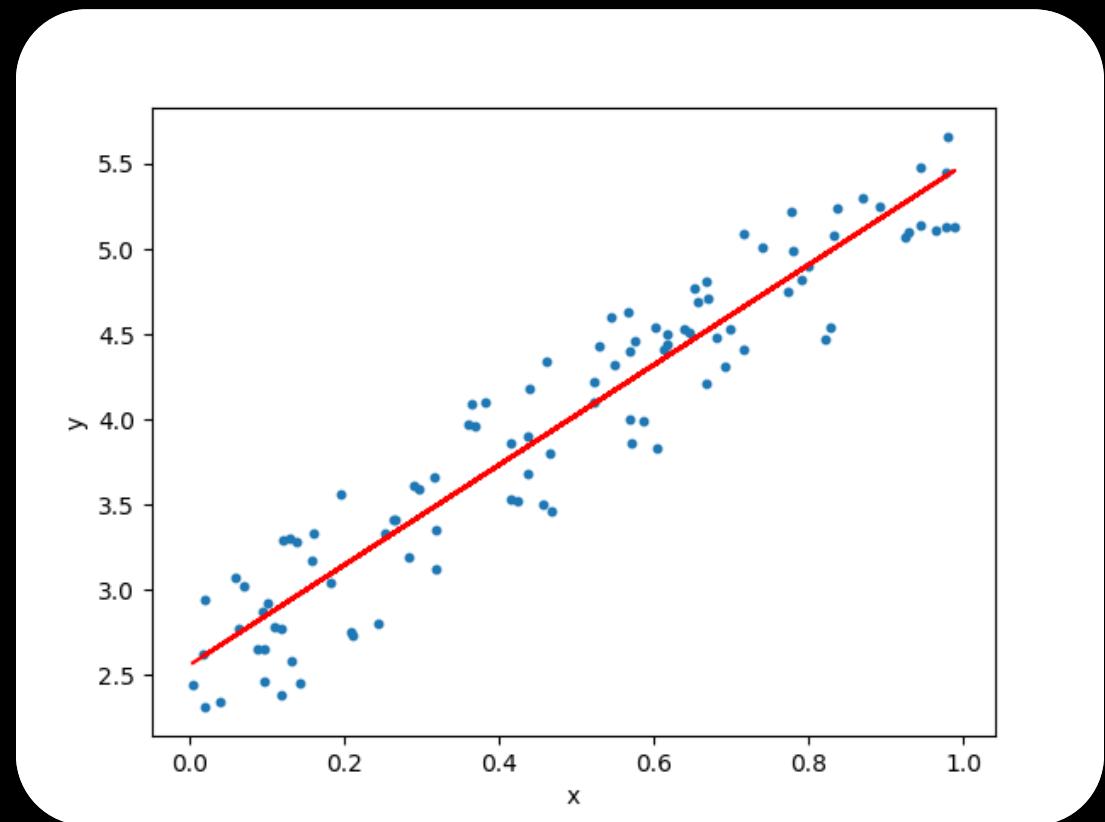
- Market Basket Analysis
- Semantic Clustering
- Delivery Store Optimization
- Identifying Accident Prone Areas
- Customer Segmentation
- Dimensionality Reduction
- Image Segmentation
- Audience segmentation
- Market research
- Recommendation System

## 2. Unsupervised Algorithms (contd.)

- Algorithms
  - K-means clustering
  - Hierarchical clustering
  - Gaussian Mixture Models
  - Apriori algorithms
  - FP Growth
  - Principal Component Analysis
  - Singular Value Decomposition
  - Autoencoders
  - Local Outlier Factor
  - Expected-Maximization

# Linear Regression

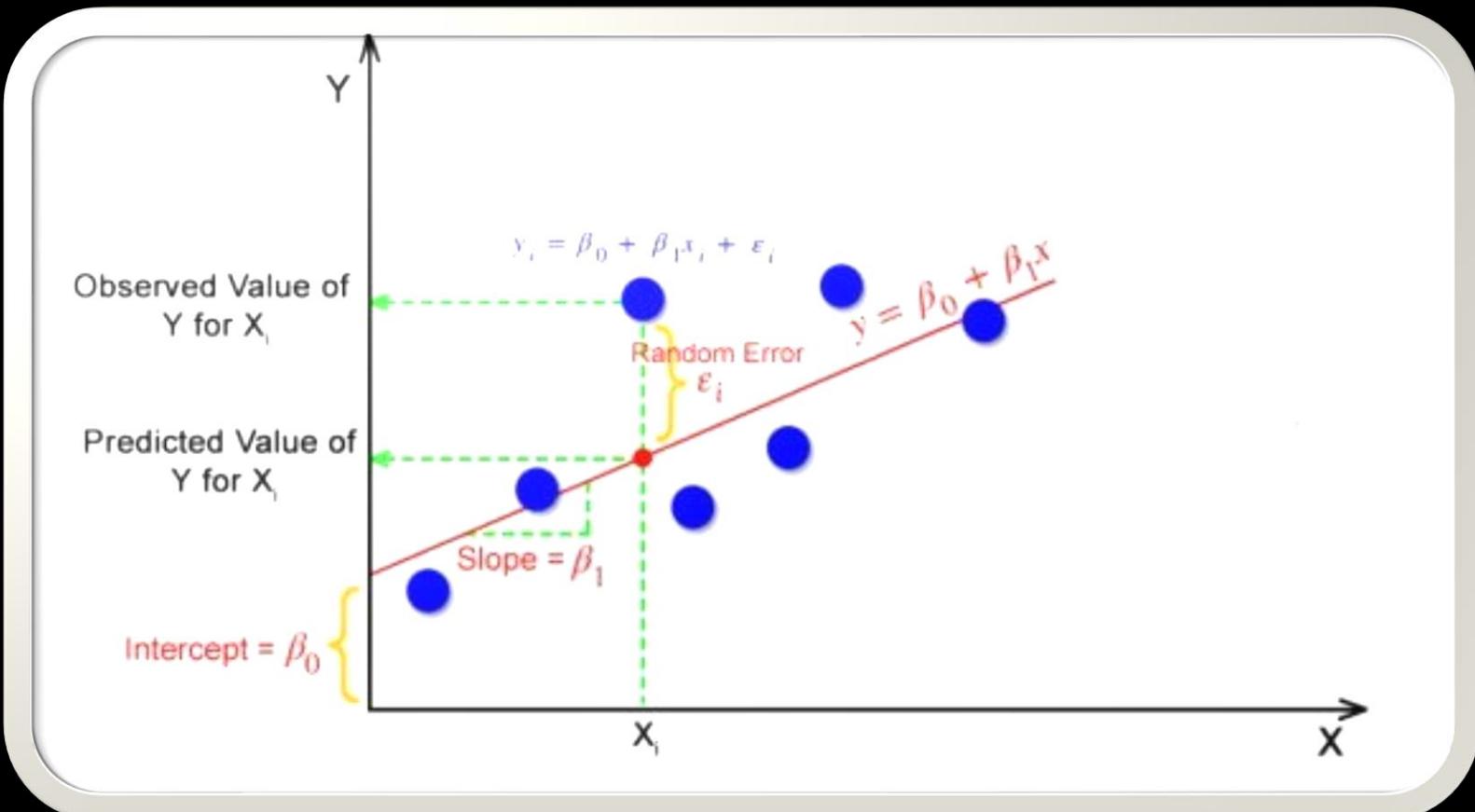
- Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data.
- One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable.
- For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model.



# Linear Regression (contd.)

- Linear Regression is a simple yet powerful and mostly used algorithm in data science.
- Linear regression shows the linear relationship between the independent(predictor) variable i.e. X-axis and the dependent(output) variable i.e. Y-axis, called linear regression.
- If there is a single input variable X(independent variable), such linear regression is called simple linear regression.
- The above graph presents the linear relationship between the output(y) variable and predictor(X) variables.
- The red line is referred to as the best fit straight line.

# Linear Regression (contd.)



# Linear Regression (contd.)

- The goal of the linear regression algorithm is to get the best values for  $\beta_0$  and  $\beta_1$  to find the best fit line.
- In simple terms, the best fit line is a line that fits the given scatter plot in the best way.
- The best fit line is a line that has the least error which means the error between predicted values and actual values should be minimum.

# Linear Regression (contd.)

## Random Error(Residuals)

- In regression, the difference between the observed value of the dependent variable( $y_i$ ) and the predicted value is called the **residuals**.

$$\epsilon = y_{predicted} - y_{actual}$$

- Then we can define the RSS (Residual Sum of Squares) as

$$RSS = e_1^2 + e_2^2 + \dots + e_m^2$$

# Linear Regression (contd.)

- In Linear Regression, generally **Mean Squared Error (MSE)** cost function is used, which is the average of squared error that occurred between the  $y_{predicted}$  and  $y_i$ .

$$MSE = \frac{1}{m} \sum_{i=1}^m (y_i - (\beta_0 + B_1 x))^2$$

- We use different optimization techniques to minimize the error or RSS. Some of them are ordinary least square method, gradient descent etc.

## X Ordinary Least Square Method

- This is a statistical Method for performing linear regression.
- Ordinary least square error is given by:

$$OLSE = \frac{1}{2} \sum_{i=1}^m (h_{\beta}(x^i) - y(i))^2$$

- Using Least Square method, we can write

$$\widehat{\beta}_1 = \frac{\sum_{i=1}^m (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^m (x_i - \bar{x})^2}$$

$$\widehat{\beta}_0 = \bar{y} - \widehat{\beta}_1 \bar{x}$$

# X Ordinary Least Square Method

## Derivation of OLS Estimator

In class we set up the minimization problem that is the starting point for deriving the formulas for the OLS intercept and slope coefficient. That problem was,

$$\min_{\beta_0, \beta_1} \sum_{i=1}^N (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2. \quad (1)$$

As we learned in calculus, a univariate optimization involves taking the derivative and setting equal to 0. Similarly, this minimization problem above is solved by setting the partial derivatives equal to 0. That is, take the derivative of (1) with respect to  $\hat{\beta}_0$  and set it equal to 0. We then do the same thing for  $\hat{\beta}_1$ . This gives us,

$$\frac{\partial W}{\partial \hat{\beta}_0} = \sum_{i=1}^N -2(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \quad (2)$$

and,

$$\frac{\partial W}{\partial \hat{\beta}_1} = \sum_{i=1}^N -2x_i(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \quad (3)$$

Note that I have used  $W$  to denote  $\sum_{i=1}^N (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2$ . Now our task is to solve (2) and (3) using some algebra tricks and some properties of summations. Lets start with the first order condition for  $\hat{\beta}_0$  (this is Equation (2)). We can immediately get rid of the  $-2$  and write  $\sum_{i=1}^N y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i = 0$ . Now lets rearrange this expression and make use of the algebraic fact that  $\sum_{i=1}^N y_i = N\bar{y}$ . This leaves us with,

$$N\hat{\beta}_0 = N\bar{y} - N\hat{\beta}_1\bar{x}. \quad (4)$$

We simply divide everything by  $N$  and amazing, we have the formula that Professor Sadoulet gave in lecture! That is,

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}. \quad (5)$$

Now lets consider solving for  $\hat{\beta}_1$ . This one is a bit more tricky. We can first get rid of the  $-2$  and rearrange Equation (3) to get  $\sum_{i=1}^N x_i y_i - \hat{\beta}_0 x_i - \hat{\beta}_1 x_i^2 = 0$ . Now lets substitute our result for  $\hat{\beta}_0$  into this expression and this gives us,

$$\sum_{i=1}^N x_i y_i - (\bar{y} - \hat{\beta}_1\bar{x})x_i - \hat{\beta}_1 x_i^2 = 0 \quad (6)$$

Note that the summation is applying to everything in the above equation. We can distribute the sum to each term to get,

$$\sum_{i=1}^N x_i y_i - \bar{y} \sum_{i=1}^N x_i + \hat{\beta}_1 \bar{x} \sum_{i=1}^N x_i - \hat{\beta}_1 \sum_{i=1}^N x_i^2 = 0. \quad (7)$$

We have of course used the property that you can always pull a constant term out in front of a summation. Lets again use the property that  $\sum_{i=1}^N y_i = N\bar{y}$  (and of course this also means that  $\sum_{i=1}^N x_i = N\bar{x}$ ). We apply these facts to Equation (7) and solve for  $\hat{\beta}_1$ . This gives,

$$\hat{\beta}_1 = \frac{\sum_{i=1}^N x_i y_i - N\bar{y}\bar{x}}{\sum_{i=1}^N x_i^2 - N\bar{x}^2}. \quad (8)$$

Doesn't quite look like the formula from class, right? Well let us just use a couple more tricks. You can either look up or derive for yourself that  $\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y}) = \sum_{i=1}^N x_i y_i - N\bar{y}\bar{x}$ . You can also easily derive that  $\sum_{i=1}^N (x_i - \bar{x})^2 = \sum_{i=1}^N x_i^2 - N\bar{x}^2$ . These two can be derived very easily using algebra. Now we substitute these two properties into (8) and we have something that looks very, very familiar:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^N (x_i - \bar{x})^2}. \quad (9)$$

\* \* \* \* \*

$$\bar{y}^F = \frac{\sum_{i=1}^N (x_i^F - \bar{x})y_i}{\sum_{i=1}^N (x_i^F - \bar{x})^2} \quad (8)$$

important note: this is not the formula for OLS regression. It is the formula for the weighted least squares regression. The reason is that the OLS regression formula is  $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$ . If we plug in  $x_i^F$  for  $x$  and  $y_i$  for  $y$ , we get  $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_i^F$ . This is not the same as  $\bar{y}^F = \sum_{i=1}^N (x_i^F - \bar{x})y_i / \sum_{i=1}^N (x_i^F - \bar{x})^2$ .

# X Gradient Descent

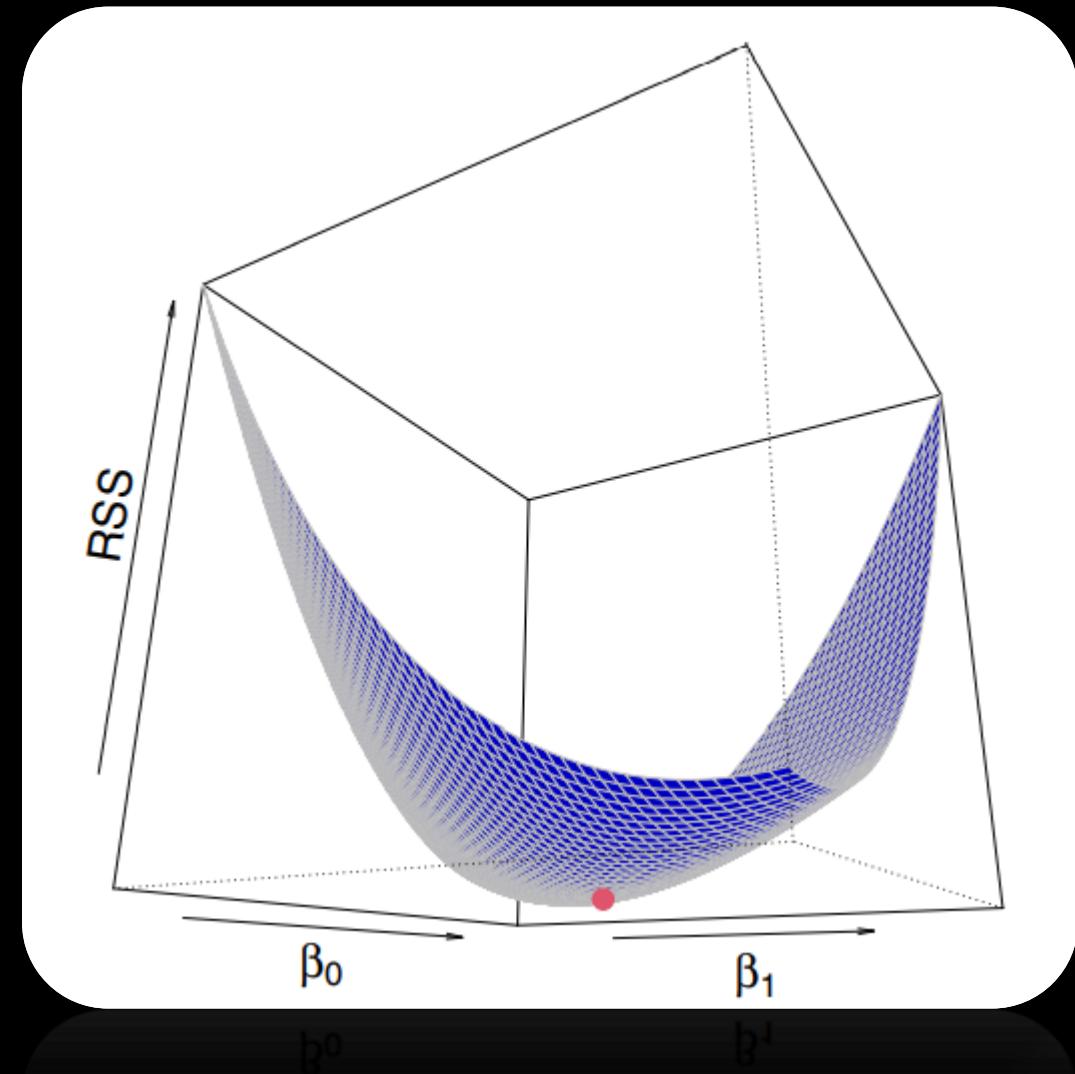
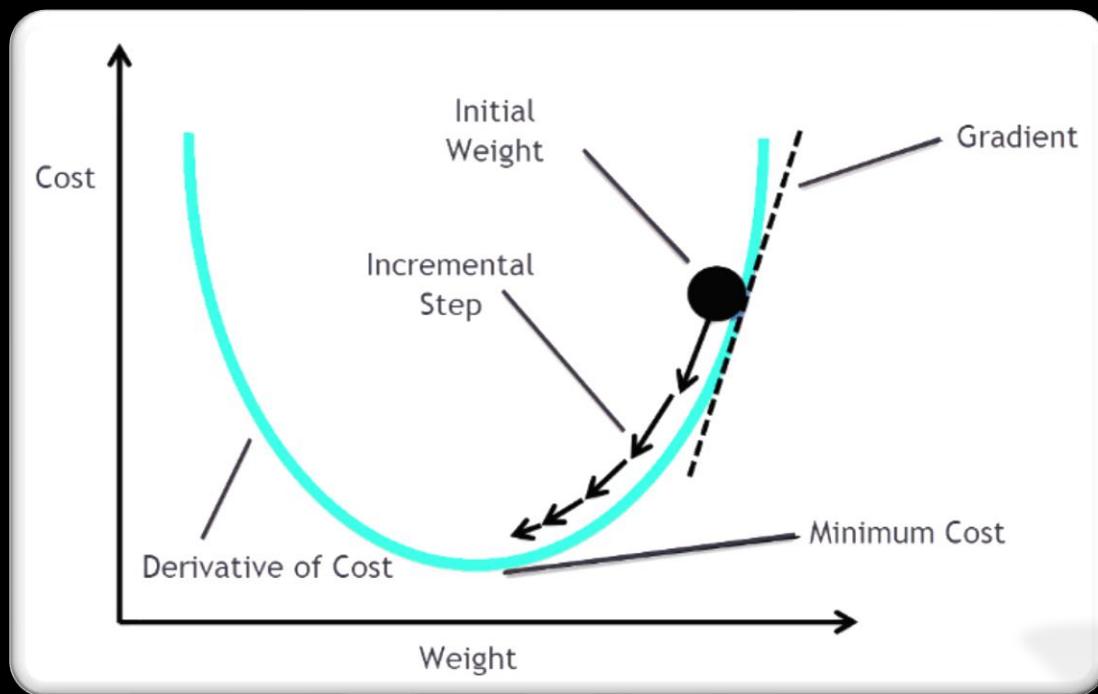
- We use linear regression to predict the dependent continuous variable Y on the basis of independent X. It assumes the relationship between independent and dependent variables to be linear as such:

$$y_i = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$$

- Gradient Descent is one of the optimization algorithms that optimize the cost function(objective function) to reach the optimal minimal solution.
- To find the optimum solution we need to reduce the cost function(MSE) for all data points.
- This is done by updating the values of  $\beta_0$  and  $\beta_1$  iteratively until we get an optimal solution.

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^m (h_\beta(x^{(i)}) - y^{(i)})^2$$

# X Gradient Descent



## X Gradient Descent (contd.)

- We used the update rule as:

$$\beta_j = \beta_j - \alpha \frac{\partial}{\partial \beta_j} J(\beta)$$

And we get the derivate as:

$$\begin{aligned}\frac{\partial}{\partial \beta_j} J(\beta) &= \frac{\partial}{\partial \beta_j} (h_{\beta}(x) - y)^2 \\ &= \frac{1}{2} \frac{\partial}{\partial \beta_j} (h_{\beta}(x) - y)^2 \\ &= \frac{1}{2} \frac{\partial (h_{\beta}(x) - y)^2}{\partial (h_{\beta}(x) - y)} \cdot \frac{\partial}{\partial \beta_j} (h_{\beta}(x) - y) \\ &= \frac{1}{2} 2 (h_{\beta}(x) - y) \cdot \frac{\partial}{\partial \beta_j} \left( \sum_{i=0}^n \beta_i x_i - y \right) \\ &= (h_{\beta}(x) - y) \cdot x_j\end{aligned}$$

## X Gradient Descent (contd.)

- Now, we can rewrite the update rule as:

$$\beta_j = \beta_j - \alpha(h_\beta(x) - y) \cdot x_j$$

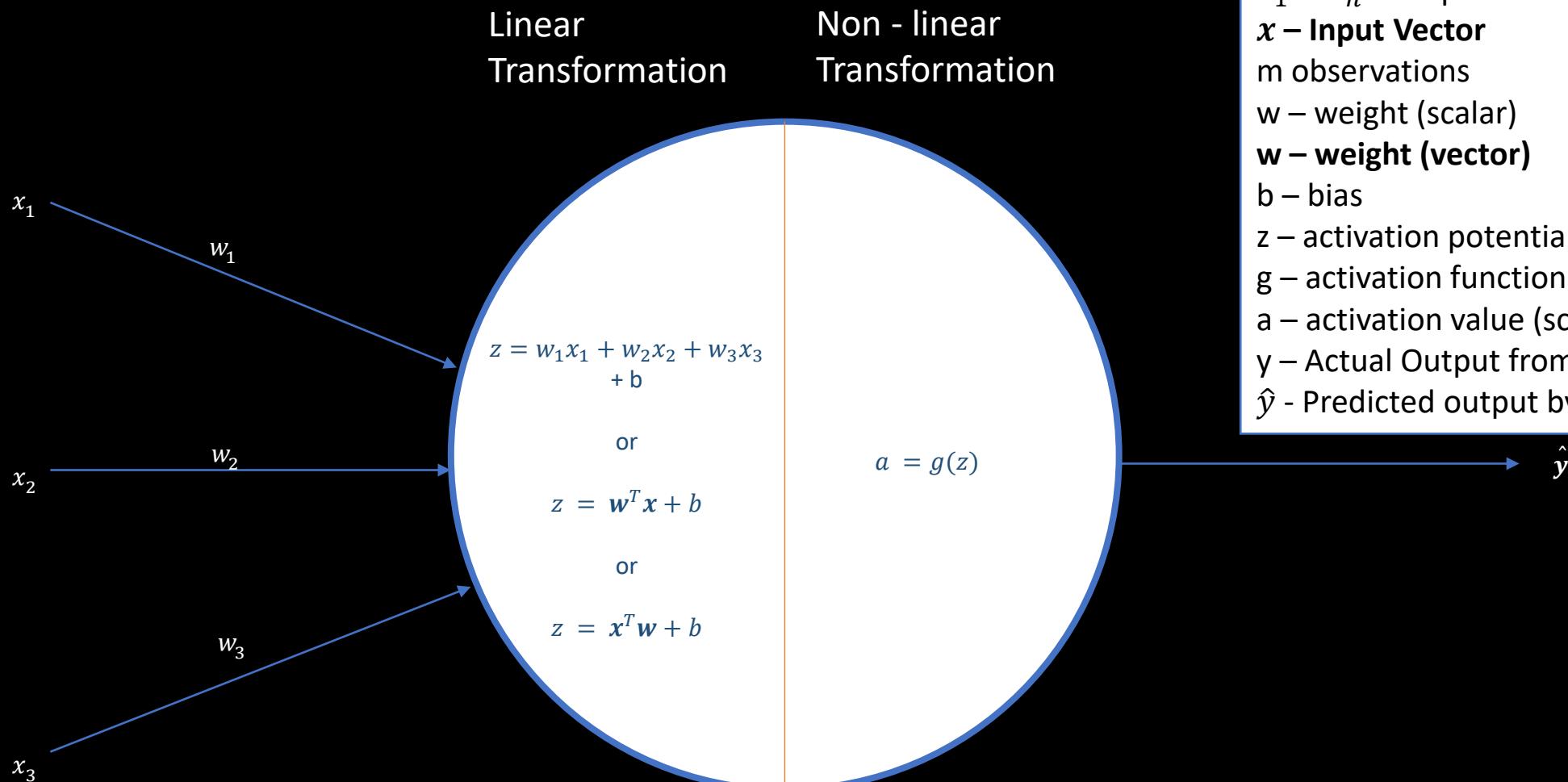
$$\beta_j = \beta_j + \alpha(y - h_\beta(x)) \cdot x_j$$

Which is the required form.

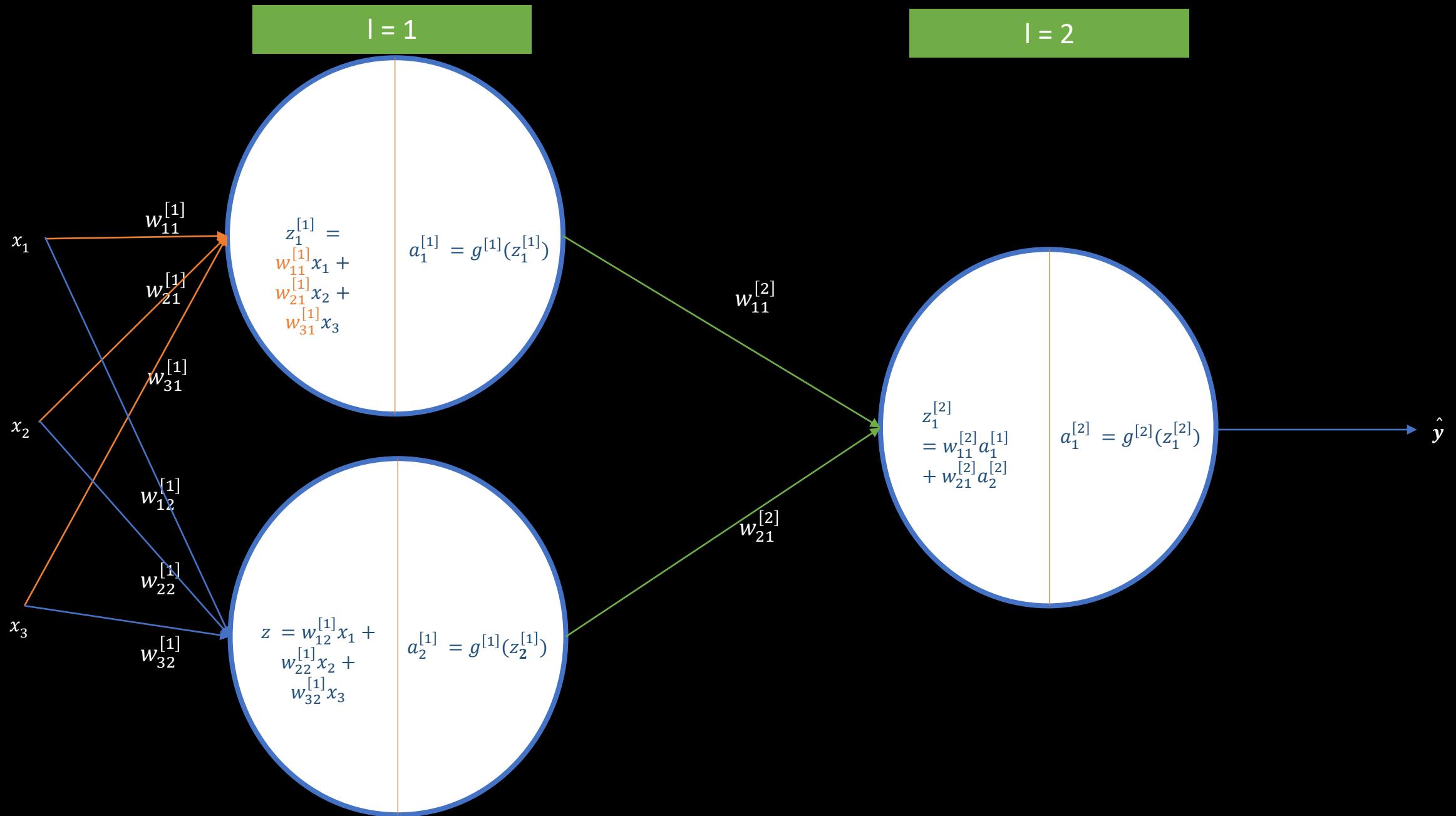
# Neural Network

## Notations

$x_1 \dots x_n$  - n input features  
 **$x$  – Input Vector**  
m observations  
 $w$  – weight (scalar)  
 **$w$  – weight (vector)**  
 $b$  – bias  
 $z$  – activation potential  
 $g$  – activation function  
 $a$  – activation value (scalar)  
 $y$  – Actual Output from input data  
 $\hat{y}$  - Predicted output by network

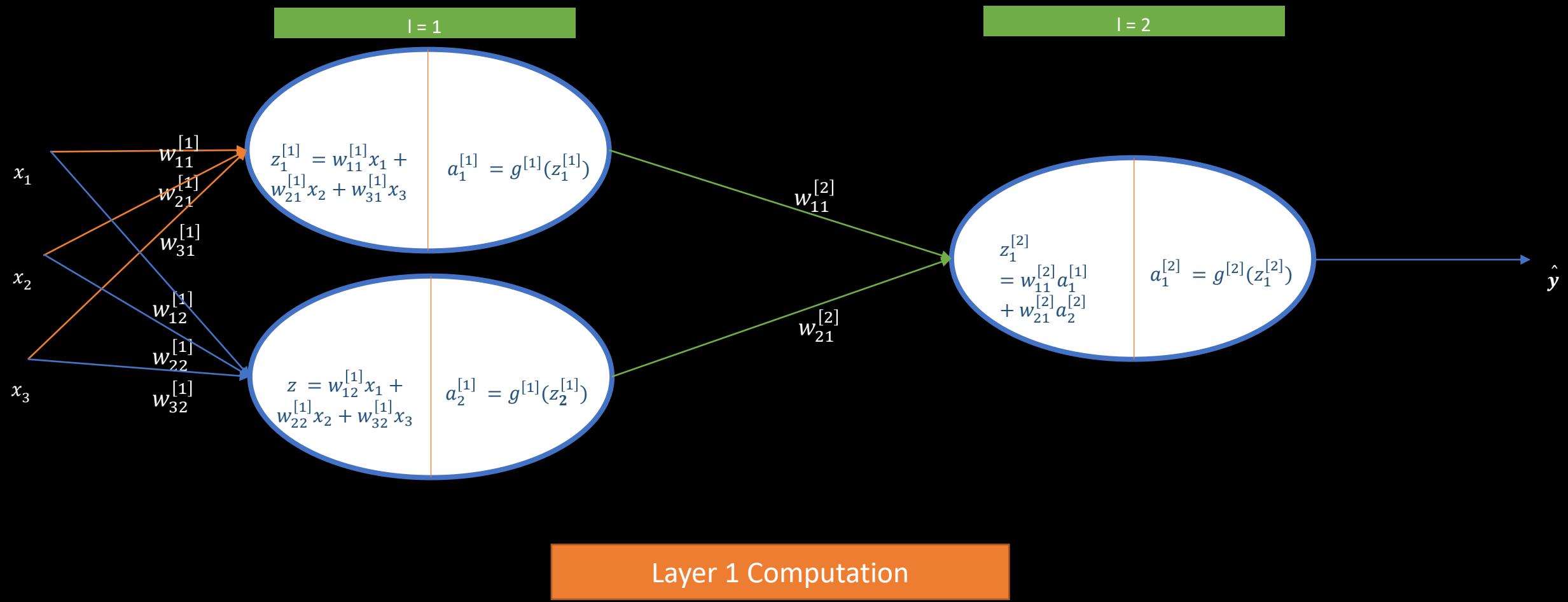


# Forward Propagation



# Forward Propagation

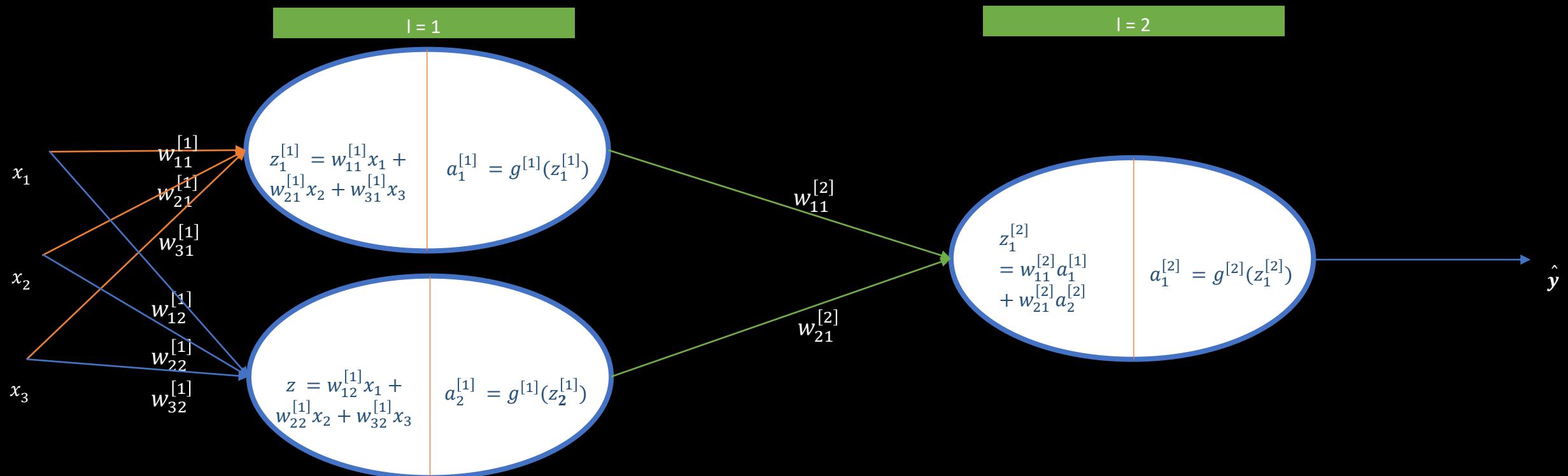
Vectorization Technique I



**Layer 1 Computation**

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \end{bmatrix} \quad W^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{12}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} \\ w_{31}^{[1]} & w_{32}^{[1]} \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \end{bmatrix} \quad Z^{[1]} = XW^{[1]} + b^{[1]}$$

$$A^{[1]} = g^{[1]}(Z^{[1]})$$



Layer 2 Computation

$$\mathbf{A}^{[1]} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \\ a_{41} & a_{42} \end{bmatrix}$$

$$\mathbf{W}^{[2]} = \begin{bmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \end{bmatrix}$$

$$\mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix}$$

$$\begin{aligned} \mathbf{Z}^{[2]} &= \mathbf{A}^{[1]}\mathbf{W}^{[2]} + \mathbf{b}^{[2]} \\ \mathbf{A}^{[2]} &= g^{[2]}(\mathbf{Z}^{[2]}) \end{aligned}$$

### Layer 1 Computation

$$\mathbf{A}_0 = \mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \end{bmatrix} \quad \mathbf{W}^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{21}^{[1]} \\ w_{12}^{[1]} & w_{22}^{[1]} \\ w_{13}^{[1]} & w_{23}^{[1]} \end{bmatrix} \quad \mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \end{bmatrix} \quad \mathbf{Z}^{[1]} = \mathbf{X}\mathbf{W}^{[1]} + \mathbf{b}^{[1]} \\ \mathbf{A}^{[1]} = g^{[1]}(\mathbf{Z}^{[1]})$$

### Layer 2 Computation

$$\mathbf{A}^{[1]} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \\ a_{41} & a_{42} \end{bmatrix} \quad \mathbf{W}^{[2]} = \begin{bmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \end{bmatrix} \quad \mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix} \quad \mathbf{Z}^{[2]} = \mathbf{A}^{[1]}\mathbf{W}^{[2]} + \mathbf{b}^{[2]} \\ \mathbf{A}^{[2]} = g^{[2]}(\mathbf{Z}^{[2]})$$

### Layer lth Computation

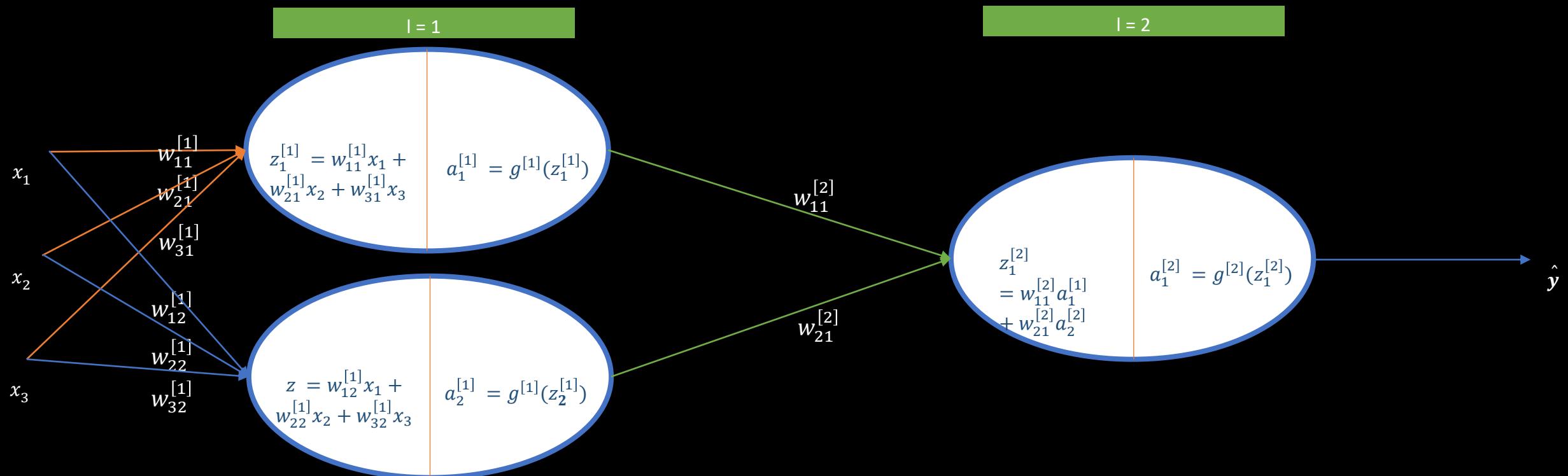
$$\mathbf{Z}^{[l]} = \mathbf{A}^{[l-1]}\mathbf{W}^{[l]} + \mathbf{b}^{[l]} \\ \mathbf{A}^{[l]} = g^{[l]}(\mathbf{Z}^{[l]})$$

```

for l = 1 ... L
{
    z[l] = numpy.matmul(A[l-1], W[l]) + b[l]
    A[l] = sigmoid(z[l])
}
```

# Forward Propagation

Vectorization Technique II

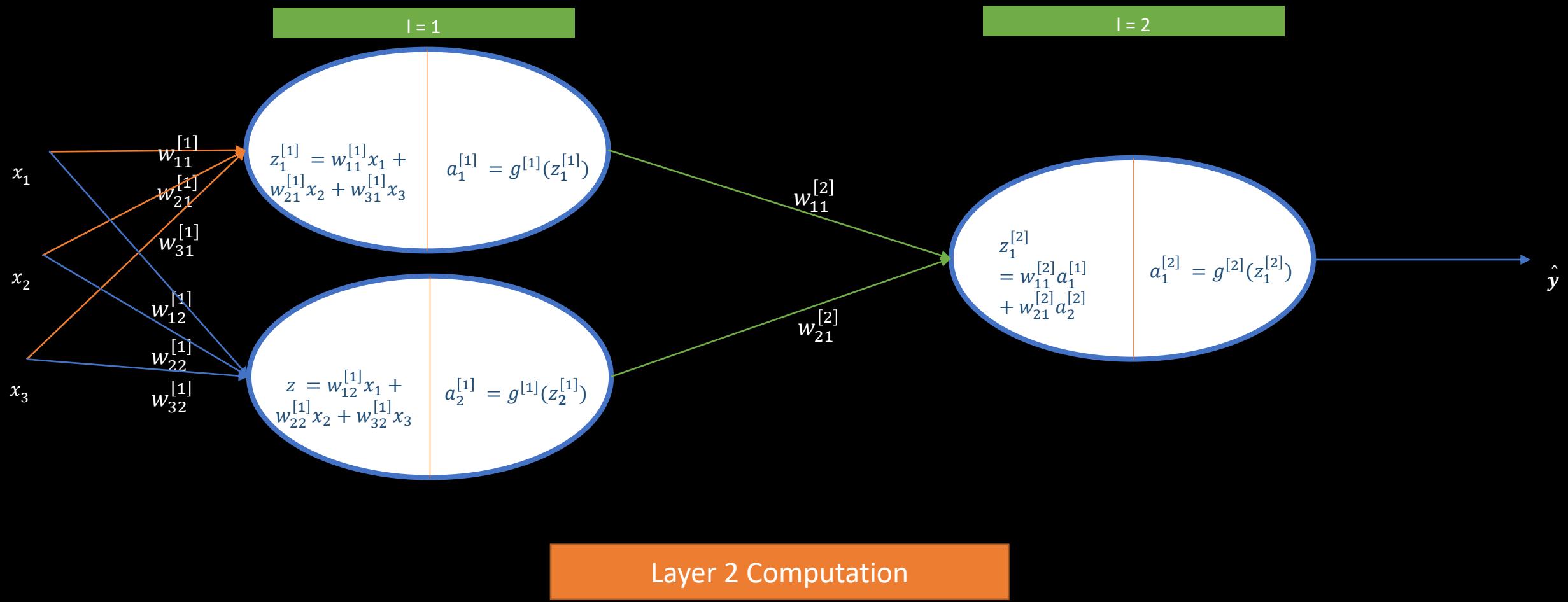


### Layer 1 Computation

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad W^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{21}^{[1]} \\ w_{12}^{[1]} & w_{22}^{[1]} \\ w_{13}^{[1]} & w_{23}^{[1]} \end{bmatrix}$$

$$b^{[1]} = [b_1^{[1]} \quad b_2^{[1]}]$$

$$\begin{aligned} Z^{[1]} &= (W^{[1]})^T X + b^{[1]} \\ A^{[1]} &= g^{[1]}(Z^{[1]}) \end{aligned}$$



$$\mathbf{A}^{[1]} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \\ a_{41} & a_{42} \end{bmatrix} \quad \mathbf{W}^{[2]} = \begin{bmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \end{bmatrix}$$

$$\mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix}$$

$$\begin{aligned} \mathbf{Z}^{[2]} &= (\mathbf{W}^{[2]})^T \mathbf{A}^{[1]} + \mathbf{b}^{[2]} \\ \mathbf{A}^{[2]} &= g^{[2]}(\mathbf{Z}^{[2]}) \end{aligned}$$

## Layer 1 Computation

$$X = \begin{bmatrix} x_{11} & x_{21} & x_{31} & x_{41} \\ x_{12} & x_{22} & x_{32} & x_{42} \\ x_{13} & x_{23} & x_{33} & x_{43} \end{bmatrix} \quad W^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{21}^{[1]} \\ w_{12}^{[1]} & w_{22}^{[1]} \\ w_{13}^{[1]} & w_{23}^{[1]} \end{bmatrix} \quad b^{[1]} = [b_1^{[1]} \quad b_2^{[1]}] \quad Z^{[1]} = (W^{[1]})^T X + b^{[1]}$$
$$A^{[1]} = g^{[1]}(Z^{[1]})$$

## Layer 2 Computation

$$A^{[1]} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{41} \\ a_{12} & a_{22} & a_{32} & a_{42} \end{bmatrix} \quad W^{[2]} = \begin{bmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \end{bmatrix} \quad b^{[2]} = [b_1^{[2]}] \quad Z^{[2]} = (W^{[2]})^T A^{[1]} + b^{[2]}$$
$$A^{[2]} = g^{[2]}(Z^{[2]})$$

## Layer 1 Computation

$$\mathbf{A}^{[0]} = \mathbf{X} = \begin{bmatrix} x_{11} & x_{21} & x_{31} & x_{41} \\ x_{12} & x_{22} & x_{32} & x_{42} \\ x_{13} & x_{23} & x_{33} & x_{43} \end{bmatrix}$$

$$\mathbf{W}^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{21}^{[1]} \\ w_{12}^{[1]} & w_{22}^{[1]} \\ w_{13}^{[1]} & w_{23}^{[1]} \end{bmatrix} \quad \mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} \end{bmatrix}$$

$$\begin{aligned} \mathbf{Z}^{[1]} &= (\mathbf{W}^{[1]})^T \mathbf{X} + \mathbf{b}^{[1]} \\ \mathbf{A}^{[1]} &= g^{[1]}(\mathbf{Z}^{[1]}) \end{aligned}$$

## Layer 2 Computation

$$\mathbf{A}^{[1]} = \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{41} \\ a_{12} & a_{22} & a_{32} & a_{42} \end{bmatrix}$$

$$\mathbf{W}^{[2]} = \begin{bmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \end{bmatrix} \quad \mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix}$$

$$\begin{aligned} \mathbf{Z}^{[2]} &= (\mathbf{W}^{[2]})^T \mathbf{A}^{[1]} + \mathbf{b}^{[2]} \\ \mathbf{A}^{[2]} &= g^{[2]}(\mathbf{Z}^{[2]}) \end{aligned}$$

## Layer lth Computation

$$\begin{aligned} \mathbf{Z}^{[l]} &= (\mathbf{W}^{[l]})^T \mathbf{A}^{[l-1]} + \mathbf{b}^{[l]} \\ \mathbf{A}^{[l]} &= g^{[l]}(\mathbf{Z}^{[l]}) \end{aligned}$$

```

for l = 1 ... L
{
    Z[l] = numpy.matmul(W[l].T, A[l-1]) + b[l]
    A[l] = sigmoid(Z[l])
}

```

# Backward Propagation

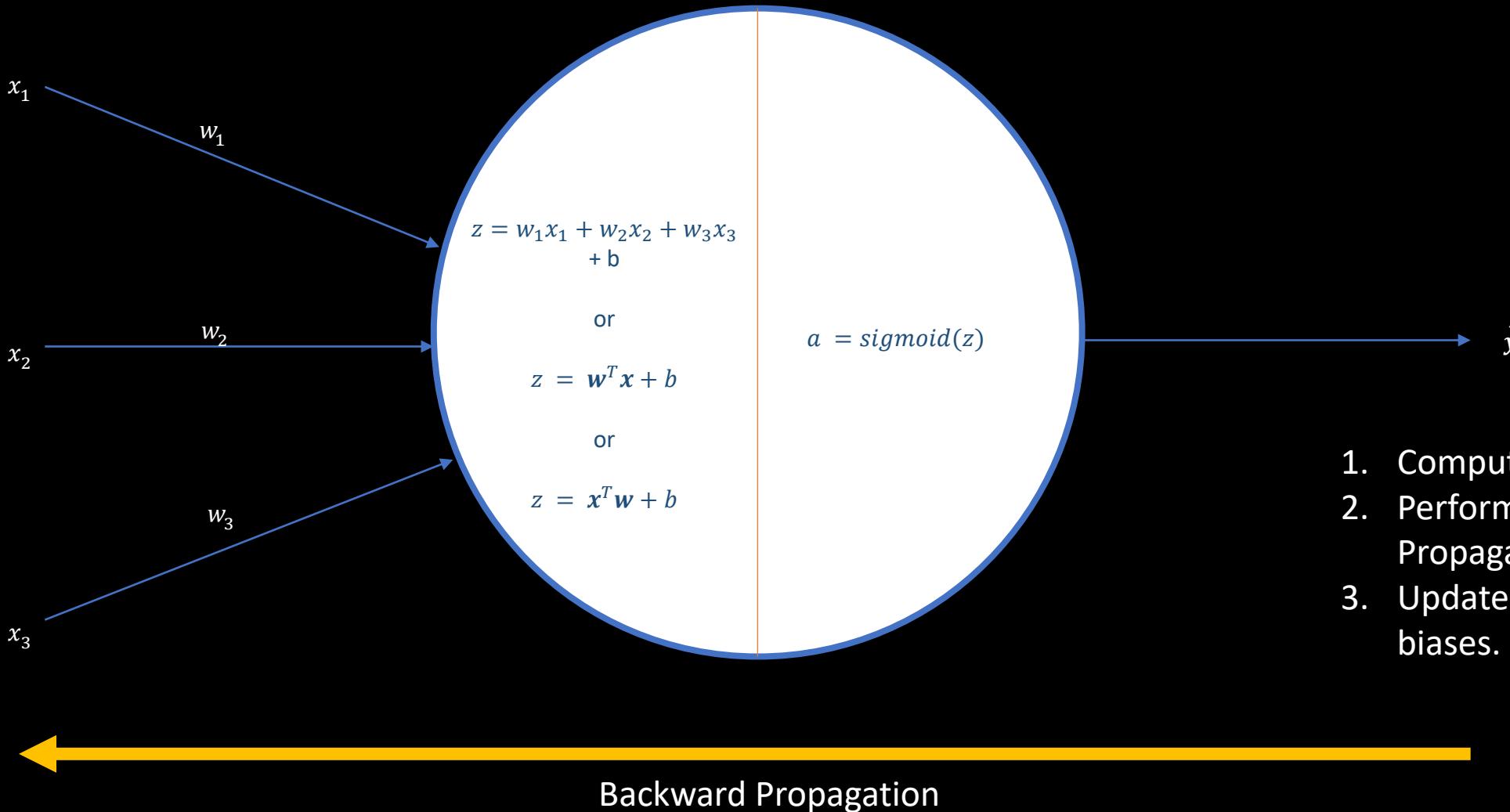
# Backward Propagation

- Backpropagation, short for *backward propagation of errors*.
- It is a widely used method for calculating derivatives inside deep feedforward neural networks.
- Backpropagation forms an important part of a number of supervised learning algorithms for training feedforward neural networks, such as stochastic gradient descent.
- When training a neural network by gradient descent, a loss function is calculated, which represents how far the network's predictions are from the true labels.

# Backward Propagation

- Backpropagation allows us to calculate the gradient of the loss function with respect to each of the weights of the network.
- This enables every weight to be updated individually to gradually reduce the loss function over many training iterations.
- Backpropagation involves the calculation of the gradient proceeding backwards through the feedforward network from the last layer through to the first.
- To calculate the gradient at a particular layer, the gradients of all following layers are combined via the chain rule of calculus.

## Forward Propagation



# Backward Propagation

## 1. Compute *Loss*

$$Loss = y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

This is called cross entropy loss commonly used in classification. We may use other loss function e.g. MSE as well.

## 2. Perform Backward Propagation

a. Compute  $\frac{\partial Loss}{\partial \hat{y}}$

b. Compute  $\frac{\partial Loss}{\partial a} = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a}$

c. Compute  $\frac{\partial Loss}{\partial z} = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial z}$

d. Compute  $\frac{\partial Loss}{\partial w} = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w}$

e. Compute  $\frac{\partial Loss}{\partial b} = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial b}$

## 3. Update Weights and Biases

$$w = w - \alpha \frac{\partial Loss}{\partial w}$$

$$b = b - \alpha \frac{\partial Loss}{\partial b}$$

# KNN (K – Nearest Neighbor) Algorithm

- K-Nearest Neighbor (KNN) is a non-parametric machine learning algorithm used for classification and regression tasks.
- It is a supervised learning technique.
- It is based on the idea of finding the K nearest neighbors to a new observation in the training set and assigning the observation to the class that has the majority of neighbors.
- K-NN algorithm assumes the similarity between the new data points and available data points and put the new point into the class that is most similar to one of the available classes.

# KNN (contd.)

- It is also called a **lazy learner** algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset into memory and when it gets new data, then it classifies that data into a class that is much similar to the new data.

# How KNN works?

- The KNN algorithm works by finding the distance between the new observation and all the other observations in the training set.
- The distance is usually calculated using Euclidean distance.
- The algorithm then selects the K nearest neighbors to the new observation based on their distances.
- Finally, the new observation is assigned to the class that has the majority of neighbors among the K nearest ones.

# Working of KNN Algorithm

- Given training data is:

RIGHTNESS	SATURATION	CLASS
40	20	Red
50	50	Blue
60	90	Blue
10	25	Red
70	70	Blue
60	10	Red
25	80	Blue

The new data point to classify is:

BRIGHTNESS	SATURATION	CLASS
20	35	?

# Working of KNN Algorithm (contd.)

- Next, we compute a distance between new point and the existing points using Euclidean distance.

$$dist = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

BRIGHTNESS	SATURATION	CLASS	DISTANCE
40	20	Red	25
50	50	Blue	33.54
60	90	Blue	68.01
10	25	Red	10
70	70	Blue	61.03
60	10	Red	47.17
25	80	Blue	45

# Working of KNN Algorithm (contd.)

- Now, we rearrange the distances in ascending order:

BRIGHTNESS	SATURATION	CLASS	DISTANCE
10	25	Red	10
40	20	Red	25
50	50	Blue	33.54
25	80	Blue	45
60	10	Red	47.17
70	70	Blue	61.03
60	90	Blue	68.01

Since we chose 5 as the value of **K**, we'll only consider the first five rows.

As you can see above, the majority class within the 5 nearest neighbors to the new entry is **Red**.

# Pros and Cons of KNN

## Pros

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

## Cons

- Always needs to determine the value of K which may be complex some time.
- The curse of dimensionality can affect the performance of the algorithm in high-dimensional feature spaces.
- The computation cost is high because of calculating the distance between the data points for all the training samples.
- Doesn't scale

# How to choose the value of K in KNN?

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as  $K=1$  or  $K=2$ , can be noisy and lead to inaccurate predictions due to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties. However, always use an odd number as the value of K.
- Use Hierarchical clustering.

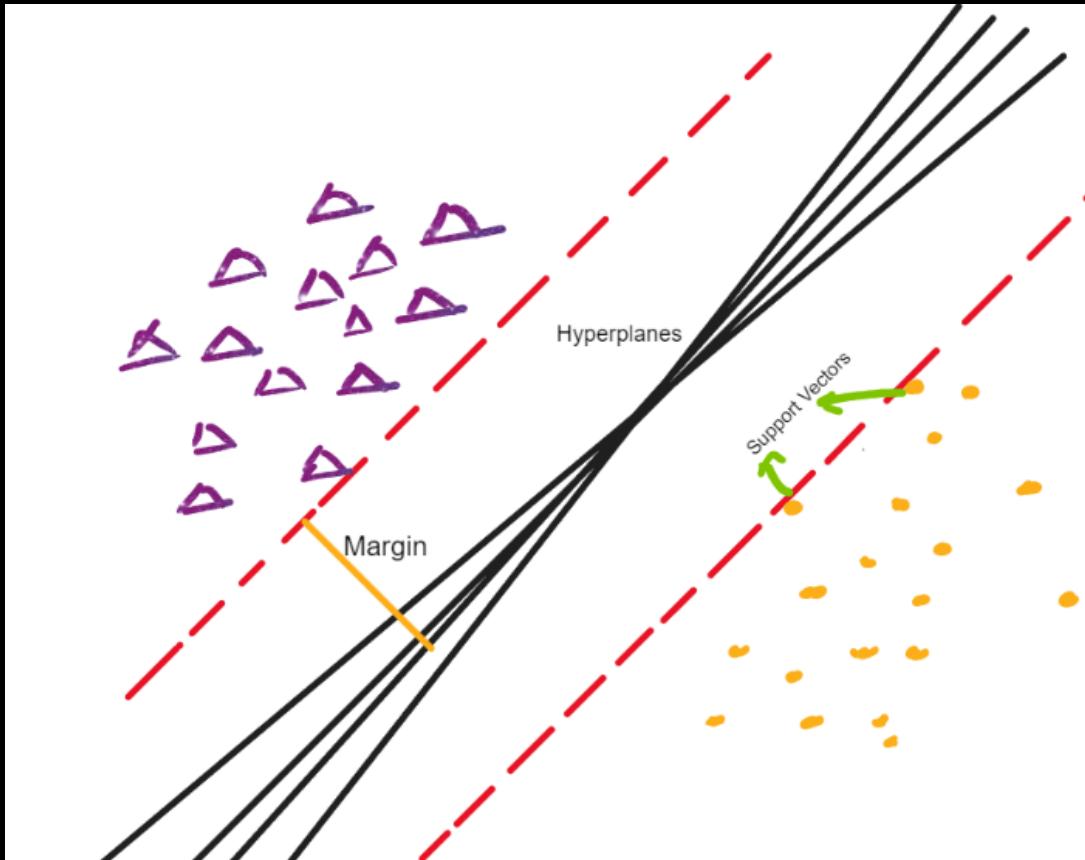
# Applications of K-Nearest Neighbor Algorithm

- KNN can be used in a variety of applications, such as image recognition, spam filtering, and recommendation systems.
- It is particularly useful when the underlying data distribution is unknown or when the dataset is small.

# Support Vector Machines (SVM)

- Support Vector Machines (SVM) is a powerful supervised machine learning algorithm used for classification and regression analysis.
- Support Vector Machine (SVM) is non parametric supervised machine learning algorithm that fits best for classification problem. But it also works well with regression problem.
- In N-dimensional input space, our goal using SVM is to learn a **hyperplane** that best separates the input space into classes.
  - Hyperplane is a line in 2D space and just a point in 1D space.
- Multiple hyperplane can exits in given input space that separates classes. And, the optimal hyperplane is the one among all plausible hyperplanes that separates classes with largest margin.

# SVM (contd.)



- Here, the distance between hyperplane and the closest data point is called **margin**.
- The margin is computed only with the closest points called '**support vectors**' as the perpendicular distance from these support vectors with the hyperplane.

# SVM

- The distance of any line,  $ax + by + c = 0$  from a given point say,  $(x_0, y_0)$  is given by  $d$ .

$$d = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

- Similarly, the distance of a hyperplane equation  $w^T X + b = 0$  from a given point  $x_0$  can be written as:

$$d = \frac{|w^T x_0 + b|}{\|w\|_2}$$

## SVM (contd.)

- They are called `support vectors` because they support the construction of hyperplane determining their orientation.
- Thus in SVM, the optimal hyperplane is obtained from the learning process using training data through the maximization of margin.
- SVM is not just limited to linear classification, SVM's can efficiently perform a non-linear classification, implicitly mapping their inputs into high dimensional feature space.

# Hard Margin vs. Soft Margin

- We have already defined margin as the perpendicular distance between support vectors and the hyperplanes. Also, we choose the hyperplane with maximal margin as the optimal one.
- But, in SVM, we have notion of two different margin viz. Hard Margin and Soft Margin.
- In SVM, the hyperplane is expected to separate the classes clearly such that the points belonging to one class is on one side of the hyperplane only and the points belonging to the another class is on other side of the hyperplane only. This is called **Hard Margin** which is default state of SVM.
- On the other hand, if we allow some data points to cross the boundary set by hyperplane i.e. we allow some misclassification of points by relaxing the hard constraints set by hyperplane. This is called **Soft Margin**.

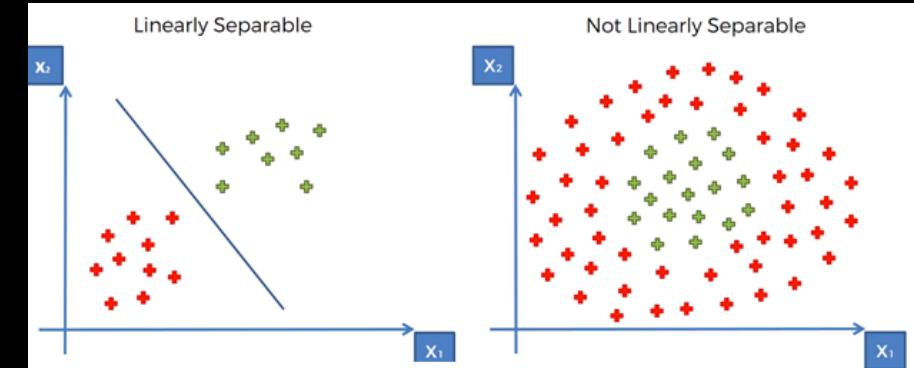
# Hard Margin vs. Soft Margin (contd.)

- We incorporate soft margin by introducing slack variables  $C$ . This is a regularization parameter that defines how much misclassification are we allowing for i.e.  $C$  determines the no. of observations allowed to cross the boundary set by hyperplane.
- The smaller the value of  $C$  the more violations of boundary or misclassification is allowed and is more sensitive to the training data. This is the case of higher variance and lower bias.
- Conversely, the larger the value of the algorithm is more sensitive towards training data causing lower variance and higher bias.
- Conclusion is, hard margin implementation has larger value of  $C$  and Soft margin implementation has lesser value of  $C$ .

# Types of SVM

## Linear SVM

- Linear SVM is used for linearly separable data.
- It finds the optimal hyperplane that separates the two classes using a linear function.
- The hyperplane with the maximum margin is selected as the best classifier.



## Non Linear SVM

- Nonlinear SVM is used for non-linearly separable data.
- It uses a kernel function to transform the data into a higher-dimensional feature space, where a linear boundary can be used to separate the classes.
- The kernel function can be linear, polynomial, Gaussian, or sigmoid.

# Kernel in SVM

- We have already mentioned above that the SVM not only works with linear data but also with non linear data. It does so by transforming lower dimension data into higher dimension such that classes are linearly separable. And this is achieved through the use of so called `Kernel`.
- In SVM, a kernel is a function that performs mapping of the input data from the original feature space to a higher-dimensional space.
- The kernel trick enables SVM to find a nonlinear decision boundary in the feature space by transforming the data into a higher-dimensional space where a linear boundary can be used.
- The learning of hyperplane explained above is the result of linear kernel.

# Types of Kernel

## 1. Linear Kernel

The linear kernel is the dot product between new data point and the support vectors.

For each data point  $x_j$  in  $X$ , compute

$$K(x_i, x_j) = x_i \cdot x_j$$

where,  $x_i$  is the support vector.

# Types of Kernel (contd.)

## 2. Polynomial Kernel

The polynomial kernel is given as:

foreach data point  $x_j$  in  $X$ , compute

$$K(x_i, x_j) = (x_i \cdot x_j + 1)^d$$

where  $d$  is the degree of polynomial and is a hyperparameter which can't be learned through training.

# Types of Kernel (contd.)

## 3. Gaussian Kernel

It is general purpose Kernel and used when there is no prior knowledge about data. It is given as:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

# Types of Kernel (contd.)

## 4. Gaussian Radial Basis Function (RBF) Kernel

It is also general purpose Kernel and used when there is no prior knowledge about data. It is given as:

$$K(x_i, x_j) = \exp(\gamma * \sum(x_j - x_i^2))$$

Where, gamma is the hyperparameter which defines how far the single training example influence is. Thus, higher value of gamma consider points closer to the plausible hyperparameter and conversely, lower value of gamma consider farther points.

# Types of Kernel (contd.)

## 5. Laplace RBF Kernel

It is also general purpose Kernel and used when there is no prior knowledge about data. It is given as:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|}{\sigma}\right)$$

# Pros and Cons of SVM

## Pros

- SVM works well when margin is clearly distinguishable.
- SVM is more effective in high dimensional spaces.
- SVM works in case where the number of dimensions is larger than the number of samples.
- SVM is also memory efficient algorithm since this algorithm uses only the subject of observations called support vectors to determine hyperplane and is suitable for both classification and regression.

# Pros and Cons of SVM (contd.)

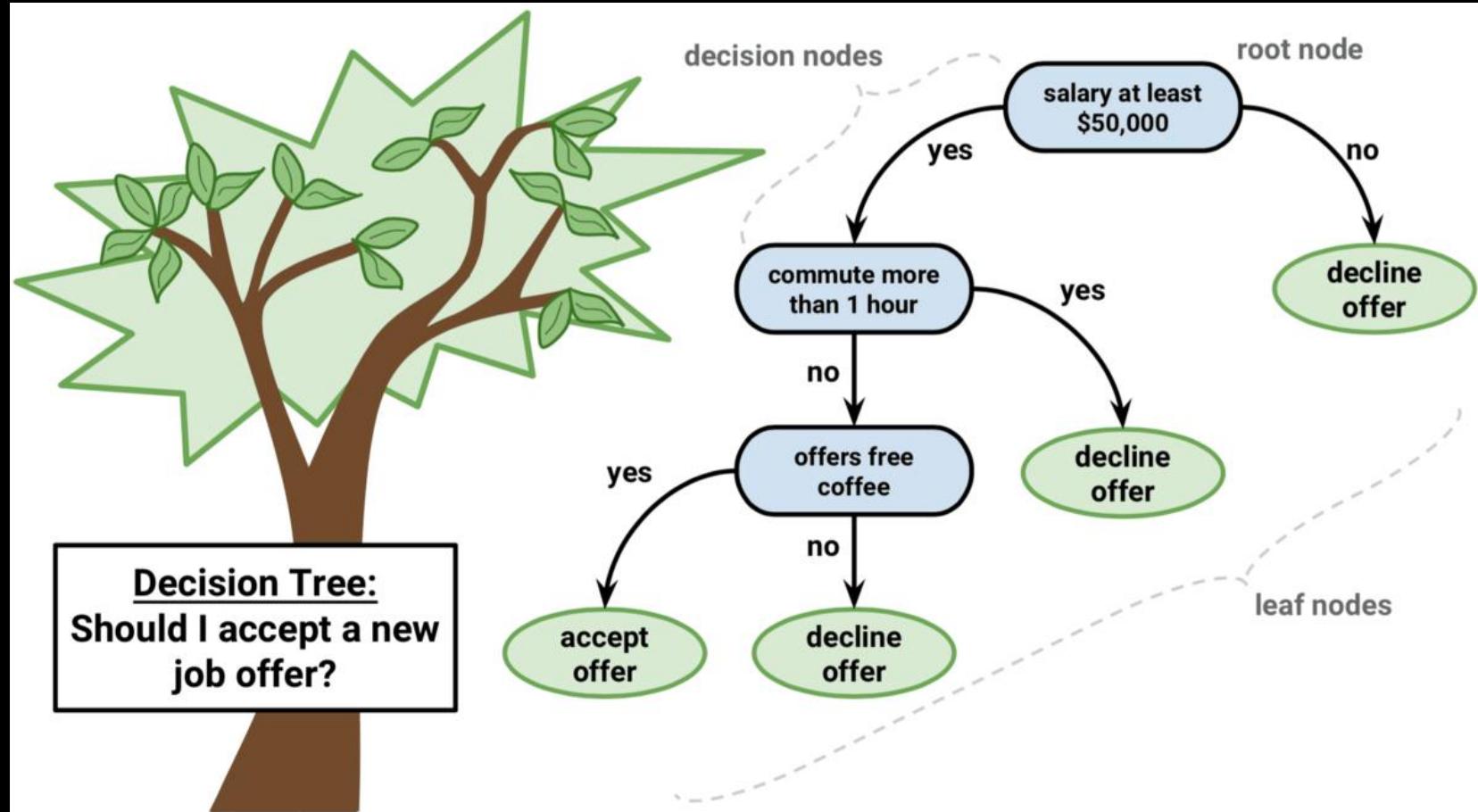
## Cons

- SVM is not preferred with very large data sets.
- SVM does not work well with overlapping classes.
- SVM doesn't work well when number of observations are too much lesser than the number of features.

# Decision Tree

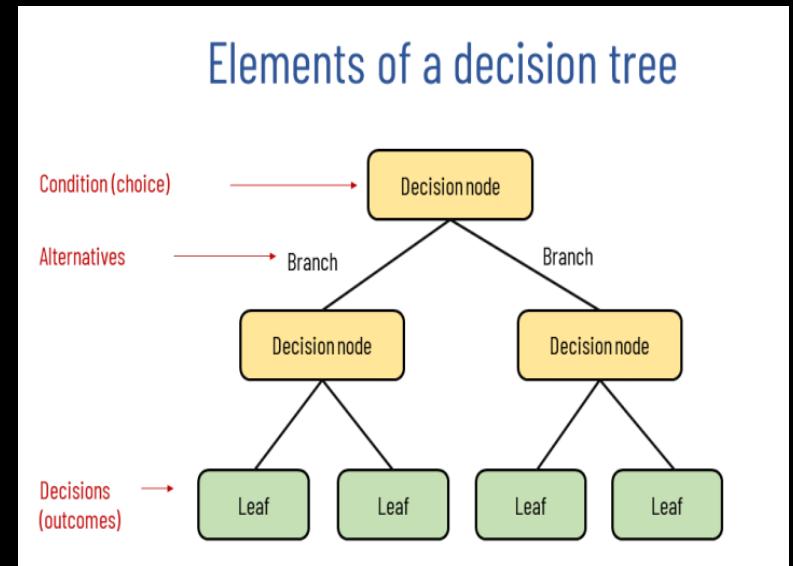
- Decision Trees are also a non-parametric model used for both regression and classification.
- Depiction of inverted tree like structure gave it a name tree and since it is used to make decisions, thus it is called **decision tree**.
- Each internal node denotes an attribute with some branching condition, if it has, extending to child nodes.
- Decision tree is a graphical representation of all possible solutions to a decision.
- In ML, It is constructed by recursively splitting the training data into subsets based on the values of the attributes until a stopping criterion is met, such as the maximum depth of the tree or the minimum number of samples required to split a node.

# Decision Tree



# Decision Tree

- During training, the Decision Tree algorithm selects the best attribute to split the data based on a metric such as entropy or Gini impurity, which measures the level of impurity or randomness in the subsets.
- The goal is to find the attribute that maximizes the information gain or the reduction in impurity after the split.
- Decision trees also provide the foundation for more advanced ensemble methods such as **bagging**, **random forests** and **gradient boosting**.



# Construction of Decision Tree

- There are various techniques to construct decision tree:
  - CART (Classification and Regression Trees)
  - ID3 (Iterative Dichotomizer 3)
  - C4.5
  - CHAID (Chi-Squared Automatic Interaction Detection)

# Classification and Regression Tree (CART)

- CART is simple to understand, interpret, visualize and requires little or no effort for data preparation.
- Moreover, it performs feature selection.
- **Regression trees** are mainly used when the target variable is numerical.
- Here value obtained by a terminal node is always the mean or average of the responses falling in that region. As a result, if any unseen data or observation will predict with the mean value.
- **Classification** is used when the target variable is categorical.
- Here value obtained by a terminal node is the mode of response falling in that region and any unseen data or observation in this region will make a prediction based on the mode value.

## CART (contd.)

- The decision to make a strategic split heavily affects the accuracy of the tree and the decision criteria for regression and classification trees will be different.
- Entropy/Information gain or Gini Index can be used for choosing the best split.
- For a given dataset with different features, to decide which feature to be considered as the root node and which feature should be the next decision node and so on, information gain of each feature should be known.
- The feature which has maximum information gain will be considered as the root node.

# CART (contd.)

- Entropy: Entropy is a measure of disorder or impurity in the given dataset.
- In the decision tree, messy data are split based on values of the feature vector associated with each data point.
- With each split, the data becomes more homogenous which will decrease the entropy. However, some data in some nodes will not be homogenous, where the entropy value will not be small.
- The higher the entropy, the harder it is to draw any conclusion. When the tree finally reaches the terminal or leaf node maximum purity is added.

# CART (contd.)

Information gain indicates how much information a particular variable or feature gives us about the final outcome. It can be found out by subtracting the entropy of a particular attribute inside the data set from the entropy of the whole data set.

- For a dataset that has  $C$  classes and the probability of randomly choosing data from class  $i$  is  $p_i$ . Then entropy  $E(S)$  can be mathematically represented as

$$E(S) = \sum_{i=1}^C -p_i \log_2 p_i$$

Then information gain is given as:

$$Gain(A, S) = E(S) - E(A, S)$$

Where,

$$E(A, S) = \sum_{j=1}^v \frac{|S_j|}{|S|} \cdot E(S_j)$$

$H(S)$  : entropy of whole data set  $S$

$|S_j|$  : number of instances with  $j$  value of an attribute  $A$

$|S|$  : total number of instances in the dataset

$v$  - set of distinct values of an attribute  $A$

$E(S_j)$  - entropy of subset of instances for attribute  $A$

$E(A, S)$  - entropy of an attribute  $A$

# CART (contd.)

- Let's take an example:

Day	Outlook	Humidity	Wind	Play
1	Sunny	High	Weak	No
2	Sunny	High	Strong	No
3	Overcast	High	Weak	Yes
4	Rain	High	Weak	Yes
5	Rain	Normal	Weak	Yes
6	Rain	Normal	Strong	No
7	Overcast	Normal	Strong	Yes
8	Sunny	High	Weak	No
9	Sunny	Normal	Weak	Yes
10	Rain	Normal	Weak	Yes
11	Sunny	Normal	Strong	Yes
12	Overcast	High	Strong	Yes
13	Overcast	Normal	Weak	Yes
14	Rain	High	Strong	No

## CART (contd.)

First in order to find the root node, we calculate the entropy of our dataset as

$$E(S) = - \left( \frac{9}{14} \log_2 \left( \frac{9}{14} \right) + \frac{5}{14} \log_2 \left( \frac{5}{14} \right) \right) = 0.94$$

How?

Check on the decision node, there are two events as outcomes. One is **yes** and the other is **no**. Out of 14 records, 9 are **yes** and 5 are **no**. Thus,

$$P(\text{yes}) = 9/14$$

$$P(\text{no}) = 5/14$$

## CART (contd.)

Next, we compute the information gain for each feature. For that compute,

$$\begin{aligned} E(\text{Outlook}, S) \\ = & -\frac{5}{14} \left( -\left( \frac{2}{5} \log_2 \frac{2}{5} + \frac{3}{5} \log_2 \frac{3}{5} \right) \right) + \frac{4}{14} \left( -\left( \frac{4}{4} \log_2 \frac{4}{4} \right) \right) \\ + & -\frac{5}{14} \left( -\left( \frac{2}{5} \log_2 \frac{2}{5} + \frac{3}{5} \log_2 \frac{3}{5} \right) \right) = 0.693 \end{aligned}$$

## CART (contd.)

Now, Information Gain

$$IG(Outlook) = E(S) - E(Outlook, S) = 0.94 - 0.693 = 0.247$$

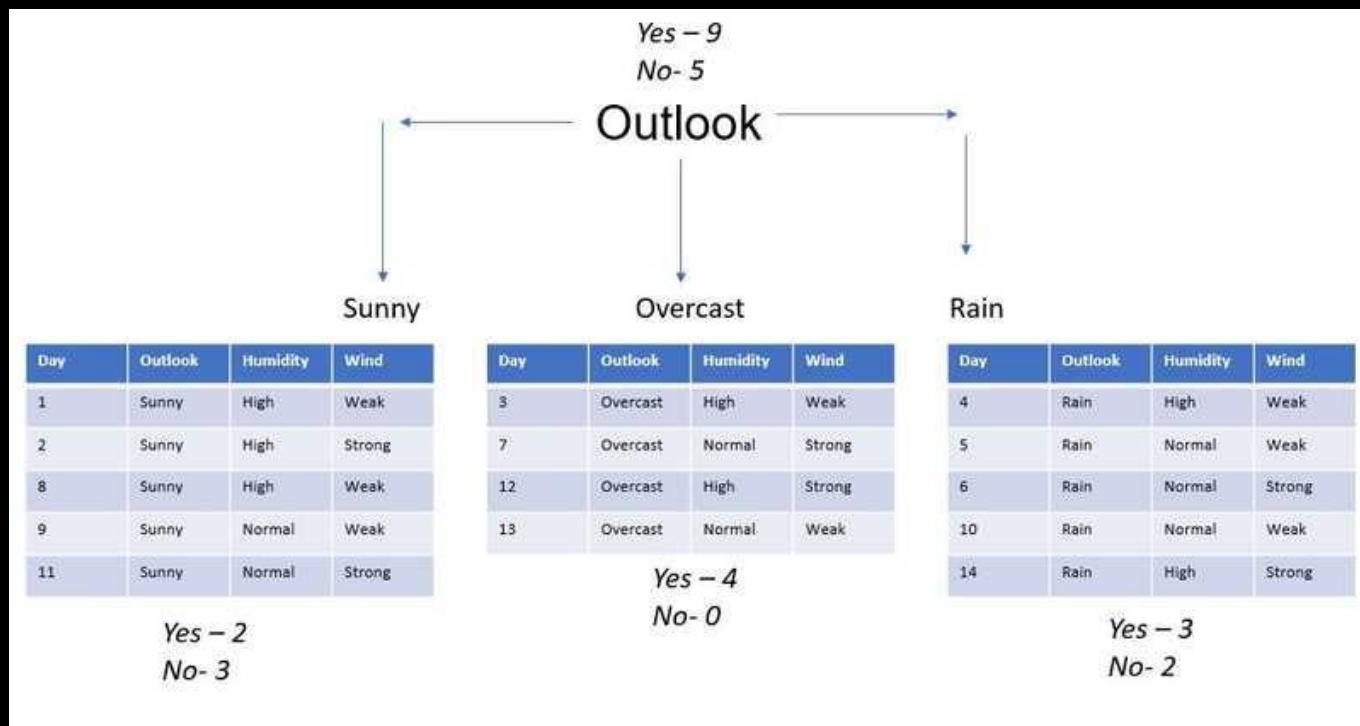
Repeat same for other attributes as well. So,

$$IG(Wind) = 0.94 - \frac{8}{14} \left( -\left(\frac{6}{8} \log_2 \frac{6}{8} + \frac{2}{8} \log_2 \frac{2}{8}\right) \right) + \frac{6}{14} \left( -\left(\frac{3}{6} \log_2 \frac{3}{6} + \frac{3}{6} \log_2 \frac{3}{6}\right) \right) = 0.048$$

$$IG(Humidity) = 0.94 - \frac{7}{14} \left( -\left(\frac{3}{7} \log_2 \frac{3}{7} + \frac{4}{7} \log_2 \frac{4}{7}\right) \right) + \frac{7}{14} \left( -\left(\frac{6}{7} \log_2 \frac{6}{7} + \frac{1}{7} \log_2 \frac{1}{7}\right) \right) = 0.151$$

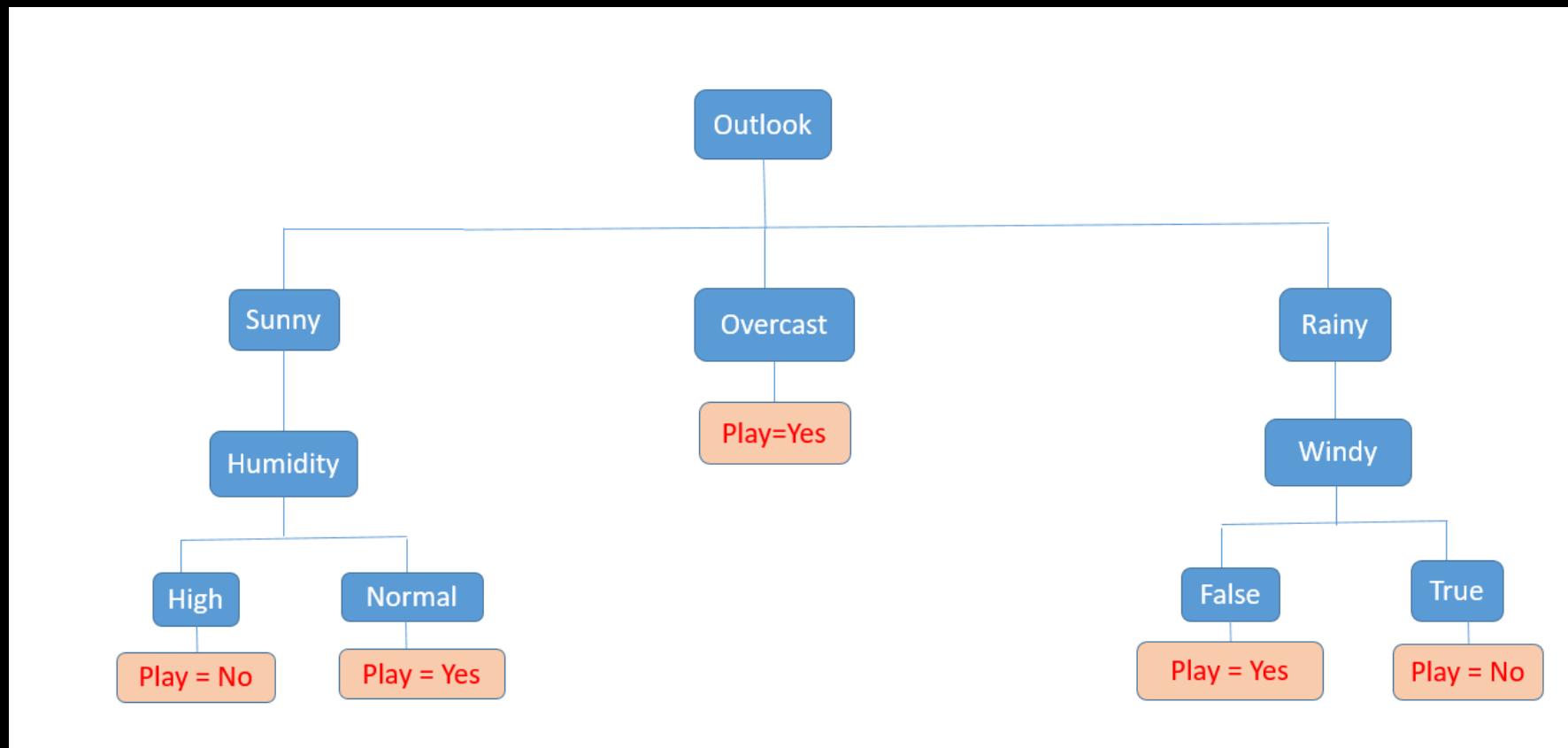
## CART (contd.)

Since, the information gain of the Outlook is highest, Outlook will be the root of the tree and we get tree as



# CART (contd.)

- Repeat the process until you get decision on leaf node as below:



# Pros and Cons of Decision Tree

## Pros

- Intuitive and easy to understand
- Non-parametric in nature and less number of data propagation required.

# Pros and Cons of Decision Tree

## Cons

- High chance of overfitting as it is a high variance algorithm.
- Unstable as small change in data can dramatically change the predictions produced by the model.
- Less effective in their ability to represent complex relationship between variables, particularly when dealing with nonlinear data.
- Although used for regression, but it is less appropriate for the prediction of continuous value.

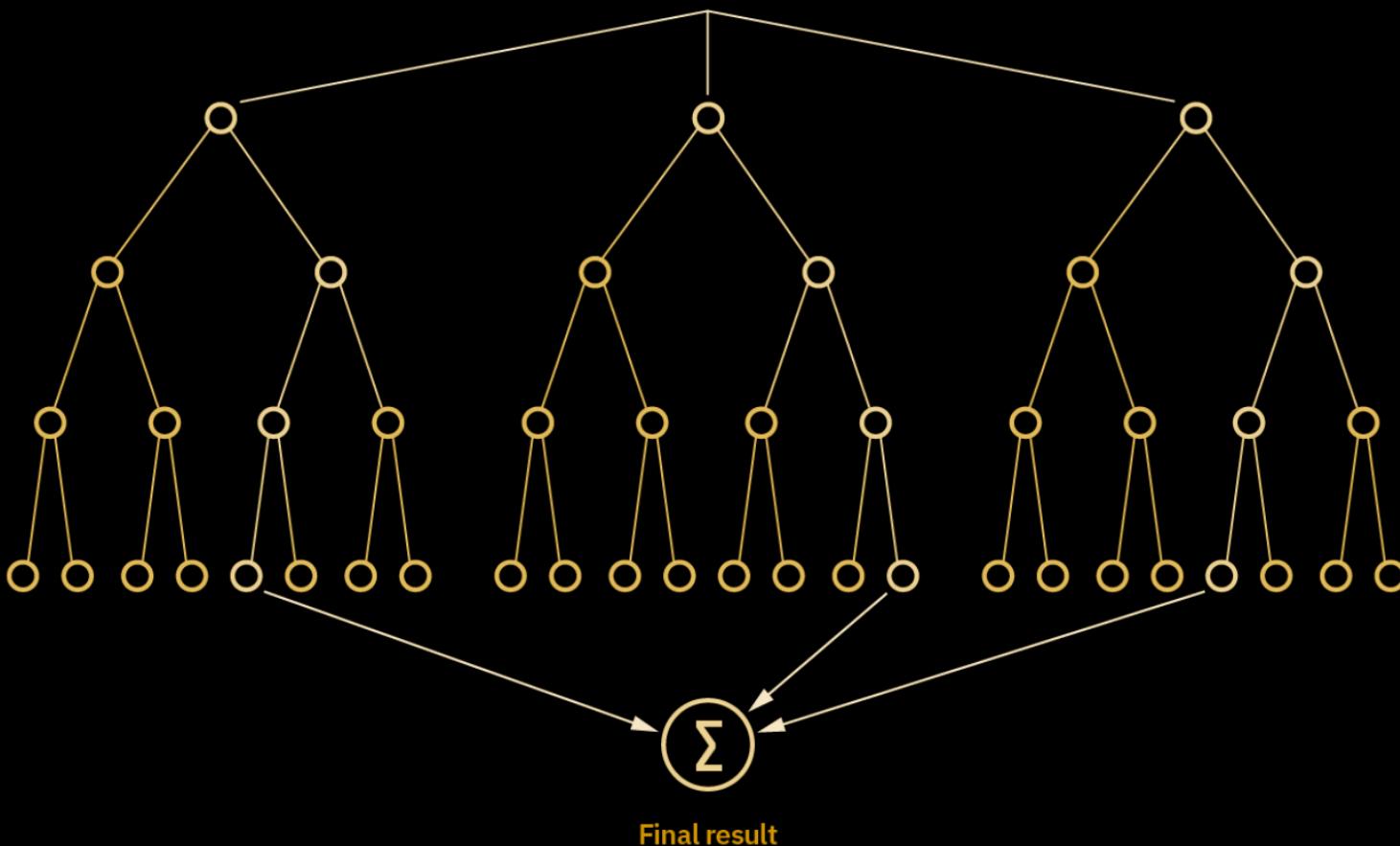
# Random Forest

- Random Forest is a commonly-used ML algorithm which gives the foundation to the rise of **ensemble method**.
- Forest basically means the multiple trees.
- Thus, Random Forest combines the output of multiple decision tree to determine single outcome with feature randomness to create an uncorrelated forest of decision trees.
- Feature randomness generates a random subset of features, which ensures low correlation among decision trees. (*This marks the key difference between decision trees and random forests.*)

# Random Forest (contd.)

- One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification.
- It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

# Random Forest (contd.)



# Essential Features of Random Forest

- **Miscellany:** Each tree has a unique attribute, variety and features concerning other trees. Not all trees are the same.
- **Immune to the curse of dimensionality:** Since a tree is a conceptual idea, it requires no features to be considered. Hence, the feature space is reduced.
- **Parallelization:** We can fully use the CPU to build random forests since each tree is created autonomously from different data and features.
- **Train-Test split:** In a Random Forest, we don't have to differentiate the data for train and test because the decision tree never sees 30% of the data.
- **Stability:** The final result is based on Bagging, meaning the result is based on majority voting or average.

# Steps Involved in Random Forest Algorithm

1. In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
2. Individual decision trees are constructed for each sample.
3. Each decision tree will generate an output.
4. Final output is considered based on **Majority Voting or Averaging** for Classification and regression, respectively.

# When to Avoid Using Random Forests?

Random Forests Algorithms are not ideal in the following situations:

- **Extrapolation:** Random Forest regression is not ideal in the extrapolation of data. Unlike linear regression, which uses existing observations to estimate values beyond the observation range.
- **Sparse Data:** Random Forest does not produce good results when the data is sparse. In this case, the subject of features and bootstrapped sample will have an invariant space. This will lead to unproductive spills, which will affect the outcome.

# Pros and Cons of Random Forest

## Pros

- **Reduced risk of overfitting:** Decision trees run the risk of overfitting as they tend to tightly fit all the samples within training data. However, when there's a robust number of decision trees in a random forest, the classifier won't overfit the model since the averaging of uncorrelated trees lowers the overall variance and prediction error.
- **Provides flexibility:** Since random forest can handle both regression and classification tasks with a high degree of accuracy, it is a popular method among data scientists. Feature bagging also makes the random forest classifier an effective tool for estimating missing values as it maintains accuracy when a portion of the data is missing.
- **Easy to determine feature importance:** Random forest makes it easy to evaluate variable importance, or contribution, to the model. There are a few ways to evaluate feature importance. **Gini importance** and mean decrease in impurity (MDI) are usually used to measure how much the model's accuracy decreases when a given variable is excluded. However, permutation importance, also known as **mean decrease accuracy (MDA)**, is another importance measure. MDA identifies the average decrease in accuracy by randomly permuting the feature values in oob samples.

# Pros and Cons of Random Forest

## Cons

- **Time-consuming process:** Since random forest algorithms can handle large data sets, they can provide more accurate predictions, but can be slow to process data as they are computing data for each individual decision tree.
- **Requires more resources:** Since random forests process larger data sets, they'll require more resources to store that data.
- **More complex:** The prediction of a single decision tree is easier to interpret when compared to a forest of them.

End of the Chapter

# Introduction to Big Data

Unit 5

# Introduction to Big Data

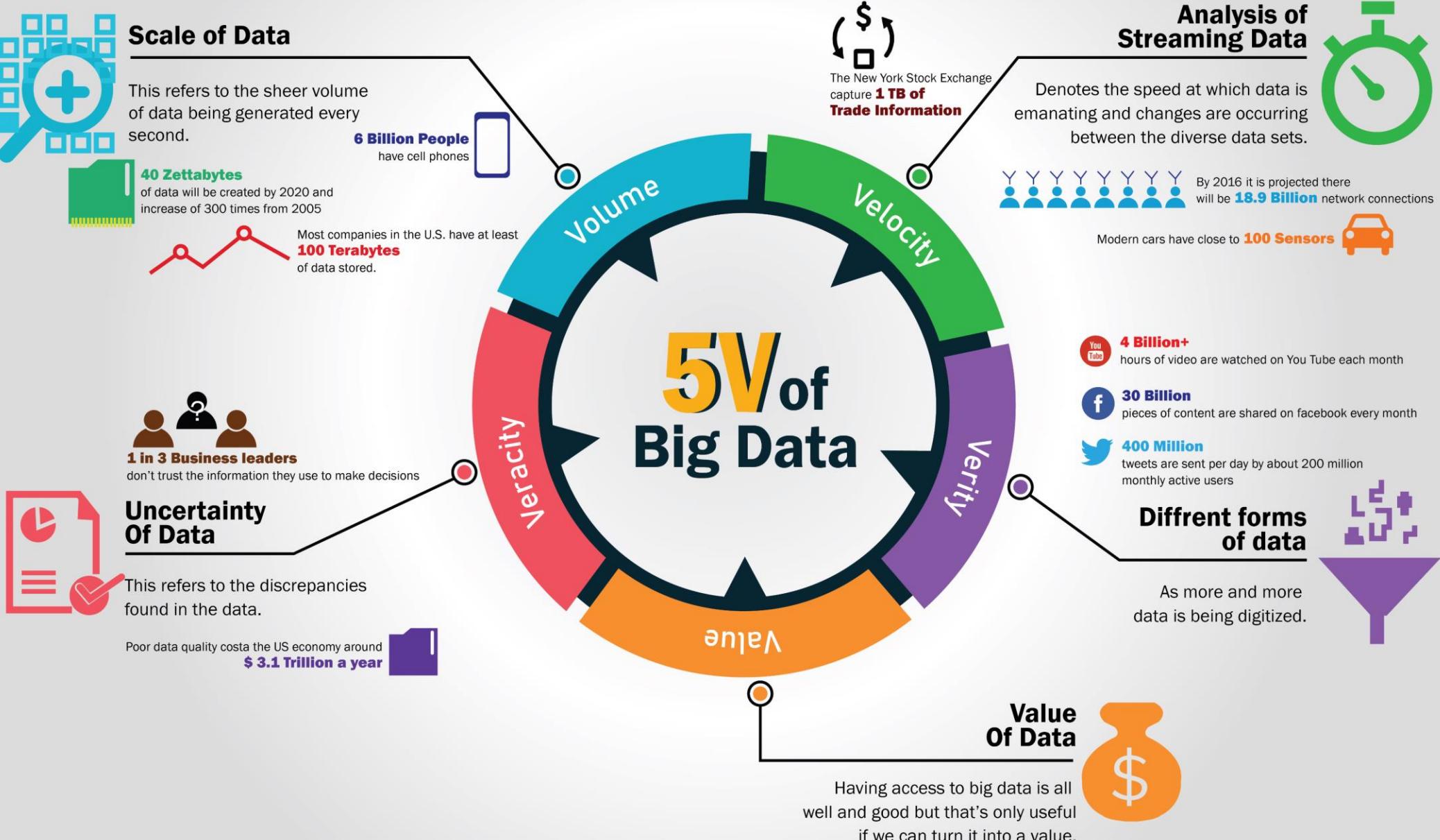
- Big data refers to a process that is used when traditional data mining and handling techniques isn't capable or sufficient enough uncover the insights and meaning of the underlying data.
- Collection of data sets so large and complex that it becomes difficult to process using on hand database management tools or traditional data processing applications.
- Big data is high-volume, high velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making.

# 5 V's of Big Data

- **Volume** - Data at rest (too big)
- **Variety** - Data in many forms (too complex)
- **Velocity** -Data in motion(too fast)
- **Veracity** - Data in doubt(uncertainty)
- **Value** - Data into money



# The Five V's of Big Data



# Volume: Scale of Data

- Refers to the vast amounts of data generated every second.
- This is where Big Data largely gets its name due to the sheer size of the data being collected.
- The actual size will vary based on the data being collected.
- For example, the user analytics of the Netflix database will be astronomical compared to e-commerce data for a small business, but both could be considered Big Data as it is a large amount of data which is being collected.
- We are not talking Terabytes but Brontobytes or Geopbytes.
- If we take all the data generated in the world between the beginning of time and 2008, the same amount of data will soon be generated every minute.

# Variety: Different Forms of Data

- This refers to the different types of data we can now use.
- In the past we focused on structured data that fits neatly into tables or relational databases, such as financial data.
- In fact, 80% of the world's data is unstructured (text, images, video, voice, etc.)
- Big data technology means we can now analyze and bring together data of different types such as messages, social media conversations, photos, sensor data, video or voice recordings.

# Velocity: Analysis of Streaming Data

- Refers to the speed at which new data is generated and the speed at which data moves around.
- Just think of social media messages going viral in seconds.
- Technology allows us now to analyze the data while it is being generated (in-memory analytics), without ever putting it into databases.
- Example: Google receives over 63,000 searches per second on any given day.

# Veracity: Uncertainty of Data

- Veracity is the quality or trustworthiness of the data.
- With many forms of big data, quality and accuracy are less controllable.
- Big data and analytics technology now allows us to work with these type of data.
- There is little point to collecting Big Data if you are not confident that the resulting analyze can be trusted.
- For example, if you are piping all order data in but also including fraudulent or cancelled orders, you can't trust the analysis of the e-commerce conversion rate because it will be artificially inflated.

# Value: Turning Big Data into Value

- Having access to big data is no good unless we can turn it into value.
- Companies are starting to generate amazing value from their big data.

# Scope of Big Data

- Increasing demand for Data Analytics
- Increasing enterprise adoption of Big Data
- Big Data finds application across various parallels of the industry
- Huge Job Opportunities & Meeting the Skill Gap
- Promises exponential salary growth
- Key Decision-Making Power

# Challenges of Handling Big Data

- Lack of understanding of Big Data
- Dealing with data growth
- Confusion while Big Data tool selection
- Generating insights in a timely manner
- Recruiting and retaining big data talent
- Integrating disparate data sources
- Securing big data
- Organizational resistance

# Commonly used tools for big data

- Hadoop
- Map-reduce programming
- HDFS
- Spark
- Apache Hive
- Apache Pig
- Apache Kafka
- Hbase
- NoSQL Database: MongoDB, Hbase, Cassandra etc.

# Hadoop

- The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models.
- It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.
- Rather than relying on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

# Hadoop (contd.)

The project includes these modules:

- **Hadoop Common:** The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS):** A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN:** A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce:** A YARN-based system for parallel processing of large data sets.

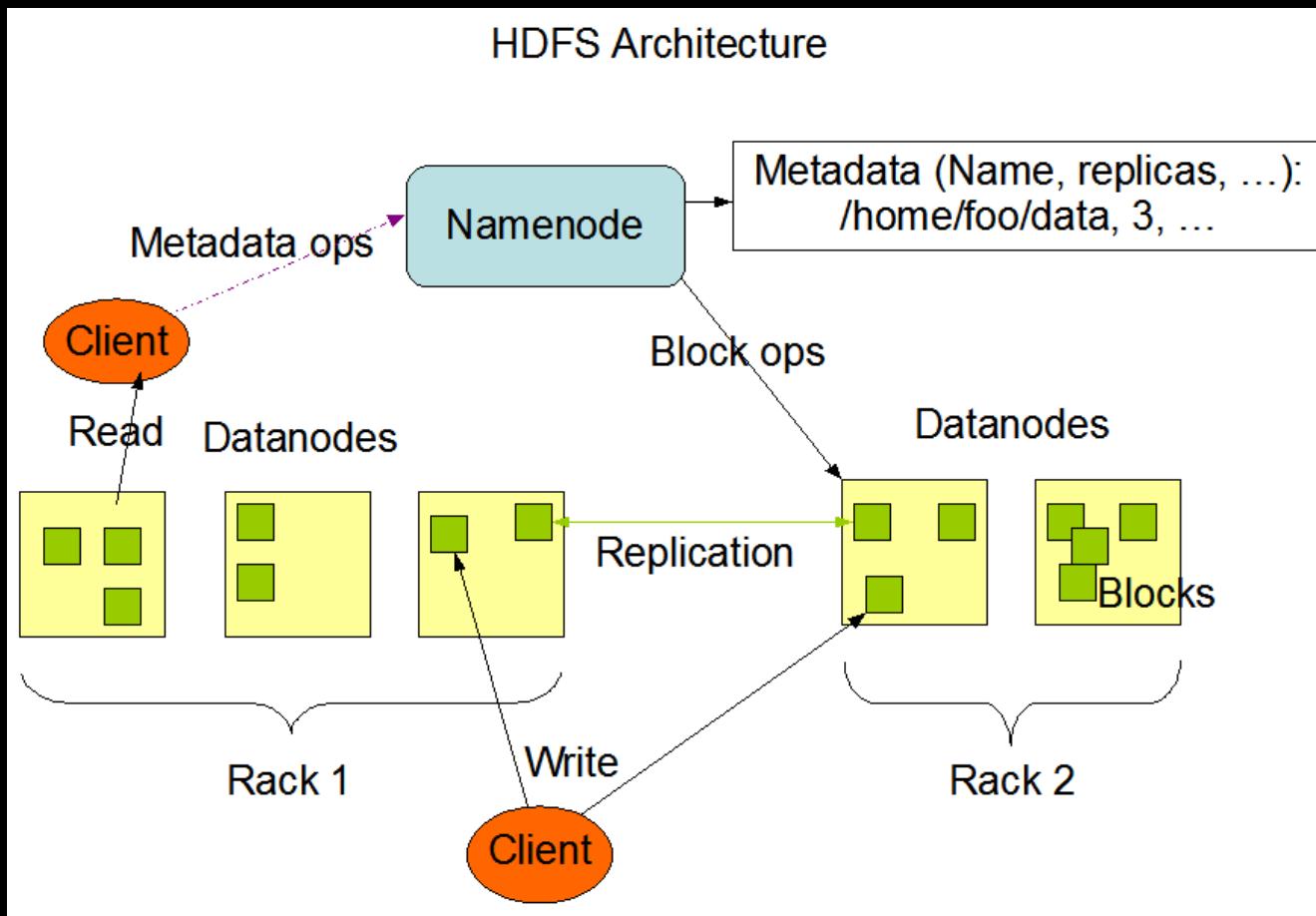
# HDFS – Hadoop File System

- The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware.
- It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant.
- HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.
- HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

## HDFS (contd.)

- HDFS has a master/slave architecture.
- An HDFS cluster consists of a single Name Node, a master server that manages the file system namespace and regulates access to files by clients.
- In addition, there are a number of Data Nodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on.
- HDFS exposes a file system namespace and allows user data to be stored in files.

# HDFS (contd.)



## HDFS (contd.)

- The Name Node and Data Node are pieces of software designed to run on commodity machines.
- These machines typically run a GNU/Linux operating system (OS). HDFS is built using the Java language; any machine that supports Java can run the Name Node or the Data Node software.
- Usage of the highly portable Java language means that HDFS can be deployed on a wide range of machines.
- A typical deployment has a dedicated machine that runs only the Name Node software. Each of the other machines in the cluster runs one instance of the Data Node software.

# Map reduce programming

- MapReduce is a framework using which we can write applications to process huge amounts of data, in parallel, on large clusters of commodity hardware in a reliable manner.
- MapReduce is a processing technique and a program model for distributed computing based on java.
- The MapReduce algorithm contains two important tasks, namely **Map** and **Reduce**.
- Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).

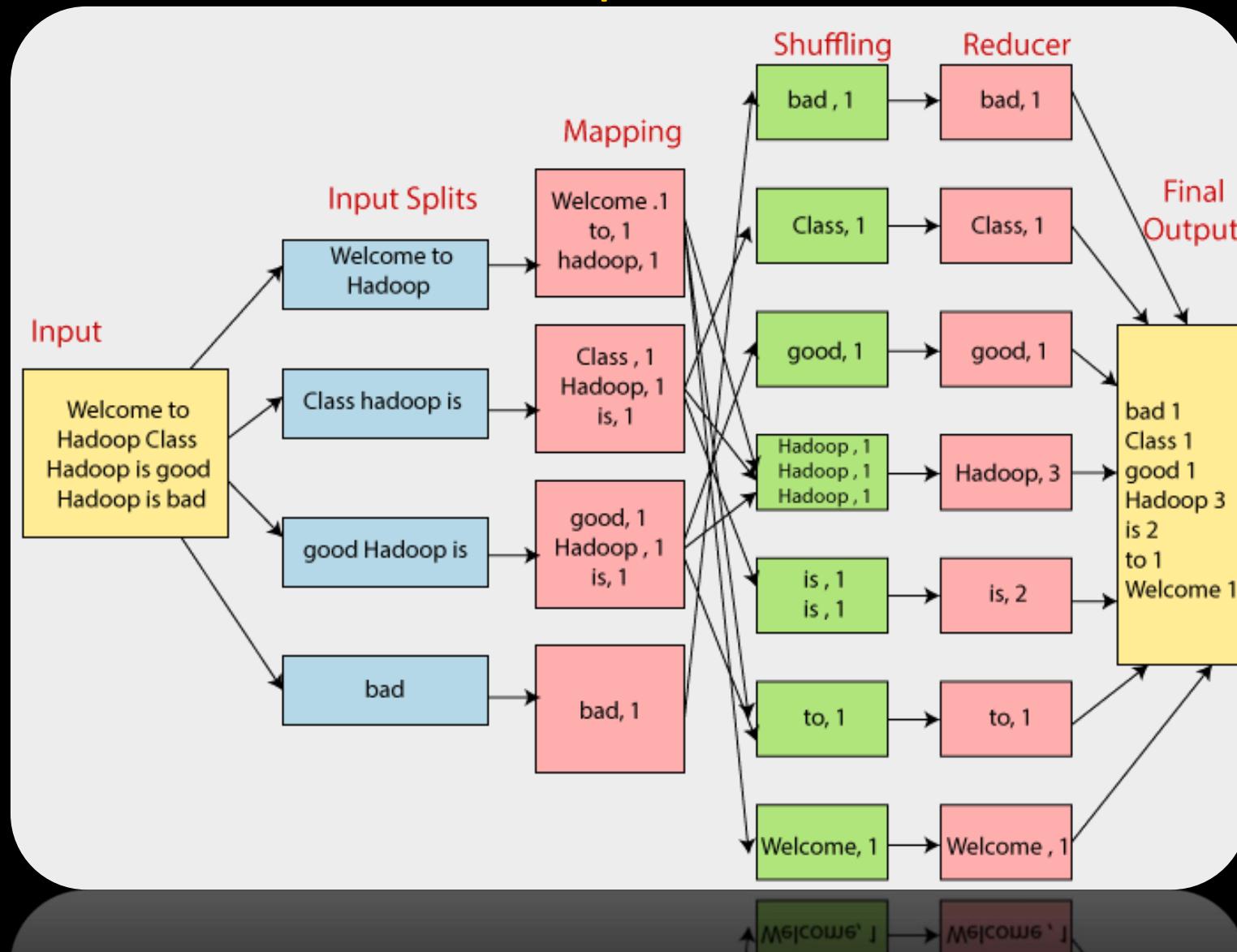
# Map reduce programming (contd.)

- Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples.
- As the sequence of the name MapReduce implies, the reduce task is always performed after the map job.
- The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes.
- Under the MapReduce model, the data processing primitives are called mappers and reducers.

# Map reduce programming (contd.)

- Decomposing a data processing application into mappers and reducers is sometimes nontrivial.
- But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change.
- This simple scalability is what has attracted many programmers to use the MapReduce model.

# Map Reduce - Example



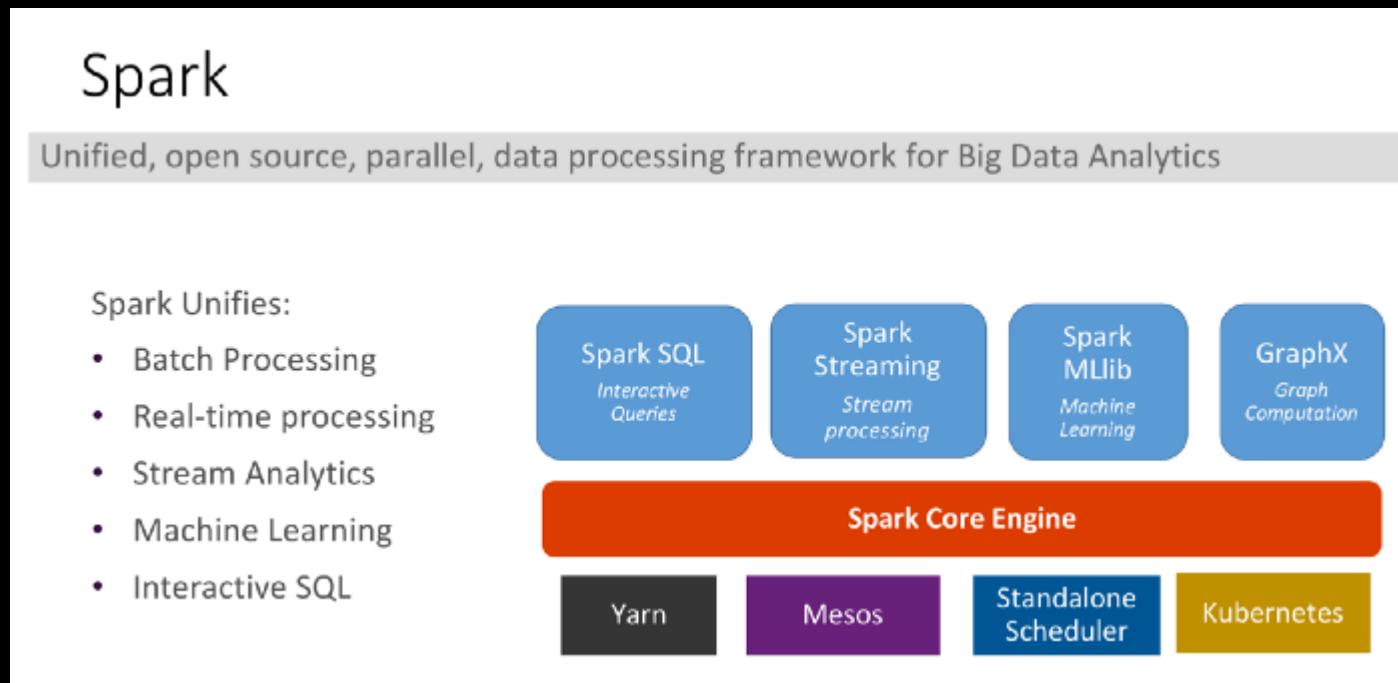
# Spark

- Apache Spark is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.
- Apache Spark is a unified analytics engine for large-scale data processing.
- It provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs.
- It also supports a rich set of higher-level tools including **Spark SQL** for SQL and structured data processing, pandas API on Spark for pandas workloads, **MLlib** for machine learning, **GraphX** for graph processing, and **Structured Streaming** for incremental computation and stream processing.

# Key features of Spark

1. **Batch/streaming data:** Unify the processing of your data in batches and real-time streaming, using your preferred language: Python, SQL, Scala, Java or R.
2. **SQL Analytics:** Execute fast, distributed ANSI SQL queries for dashboarding and ad-hoc reporting. Runs faster than most data warehouses.
3. **Data Science at scale:** Perform Exploratory Data Analysis (EDA) on petabyte-scale data without having to resort to downsampling.
4. **Machine Learning:** Train machine learning algorithms on a laptop and use the same code to scale to fault-tolerant clusters of thousands of machines.

# Spark Architecture



# Spark Architecture (contd.)

## Spark SQL and DataFrame

Spark SQL is a Spark module for structured data processing. It provides a programming abstraction called DataFrame and can also act as distributed SQL query engine.

## Streaming

Running on top of Spark, the streaming feature in Apache Spark enables powerful interactive and analytical applications across both streaming and historical data, while inheriting Spark's ease of use and fault tolerance characteristics.

# Spark Architecture (contd.)

## Mlib

Built on top of Spark, MLib is a scalable machine learning library that provides a uniform set of high-level APIs that help users create and tune practical machine learning pipelines.

```
# Every record of this DataFrame contains the label and
# features represented by a vector.
df = sqlContext.createDataFrame(data, ["label", "features"])

# Set parameters for the algorithm.
# Here, we limit the number of iterations to 10.
lr = LogisticRegression(maxIter=10)

# Fit the model to the data.
model = lr.fit(df)

# Given a dataset, predict each point's label, and show the results.
model.transform(df).show()
```

# Spark Architecture (contd.)

## Spark Core

Spark Core is the underlying general execution engine for the Spark platform that all other functionality is built on top of. It provides an RDD (Resilient Distributed Dataset) and in-memory computing capabilities.

# Spark Sample code

```
# Creates a DataFrame based on a table named "people"
# stored in a MySQL database.
url = \
    "jdbc:mysql://yourIP:yourPort/test?user=yourUsername;password=yourPassword"
df = sqlContext \
    .read \
    .format("jdbc") \
    .option("url", url) \
    .option("dbtable", "people") \
    .load()

# Looks the schema of this DataFrame.
df.printSchema()

# Counts people by age
countsByAge = df.groupBy("age").count()
countsByAge.show()

# Saves countsByAge to S3 in the JSON format.
countsByAge.write.format("json").save("s3a://...")
```

*Python code using spark*

# Hive

- Apache Hive is a distributed, fault-tolerant data warehouse system that enables analytics at a massive scale.
- It is a data warehouse software that facilitates reading, writing, and managing large datasets residing in distributed storage using SQL.
- Structure can be projected onto data already in storage.
- A command line tool and JDBC driver are provided to connect users to Hive.
- Hive provides standard SQL functionality, including many of the later SQL:2003, SQL:2011, and SQL:2016 features for analytics.
- Hive's SQL can also be extended with user code via user defined functions (UDFs), user defined aggregates (UDAFs), and user defined table functions (UDTFs).

# Hive (contd.)

- Built on top of Apache Hadoop, Hive provides the following features:
  - Tools to enable easy access to data via SQL, thus enabling data warehousing tasks such as extract/transform/load (ETL), reporting, and data analysis.
  - A mechanism to impose structure on a variety of data formats
  - Access to files stored either directly in Apache HDFS or in other data storage systems such as Apache HBase
  - Query execution via Apache Tez™, Apache Spark, or MapReduce
  - Procedural language with HPL-SQL

# Hive (contd.)

- There is not a single "Hive format" in which data must be stored. Hive comes with built in connectors for comma and tab-separated values (CSV/TSV) text files, Apache Parquet, Apache ORC, and other formats.
- Users can also extend Hive with connectors for other formats.
- Hive is not designed for online transaction processing (OLTP) workloads.
- It is best used for traditional data warehousing tasks.
- Hive is designed to maximize scalability (scale out with more machines added dynamically to the Hadoop cluster), performance, extensibility, fault-tolerance, and loose-coupling with its input formats.

# Hive (contd.)

## Creating Hive Tables

```
hive> CREATE TABLE pokes (foo INT, bar STRING);
```

creates a table called pokes with two columns, the first being an integer and the other a string.

```
hive> CREATE TABLE invites (foo INT, bar STRING) PARTITIONED BY (ds STRING);
```

creates a table called invites with two columns and a partition column called ds. The partition column is a virtual column. It is not part of the data itself but is derived from the partition that a particular dataset is loaded into.

## Browsing through Tables

```
hive> SHOW TABLES;
```

lists all the tables.

```
hive> SHOW TABLES '.*s';
```

lists all the table that end with 's'. The pattern matching follows Java regular expressions.

```
hive> DESCRIBE invites;
```

shows the list of columns.

# Real time Analytics with Apache Kafka

- Apache Kafka is the most popular open-source stream-processing software for collecting, processing, storing, and analyzing data at scale.
- Most known for its excellent performance, low latency, fault tolerance, and high throughput, it's capable of handling thousands of messages per second.

# Apache Kafka (contd.)

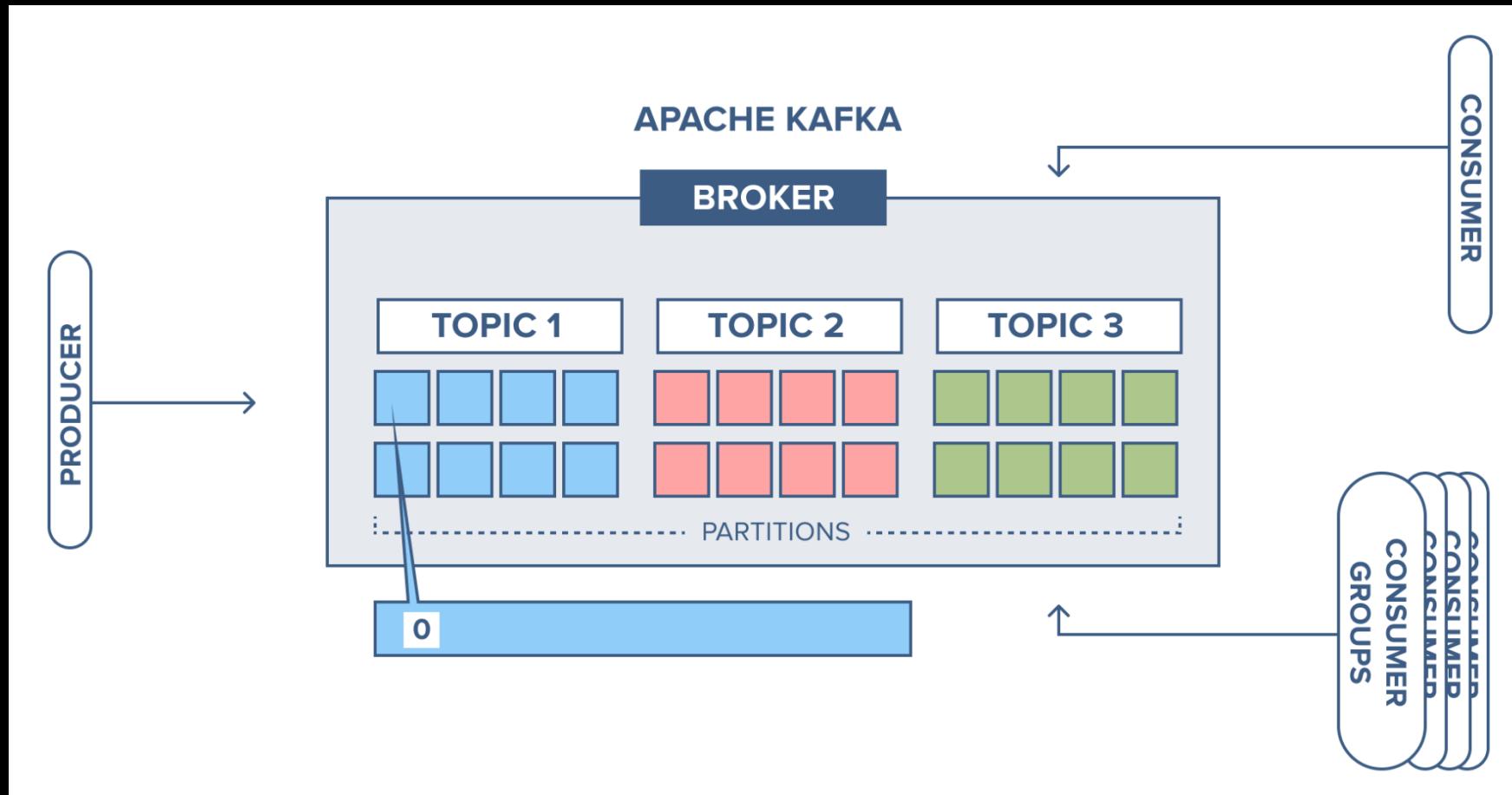
- Event streaming
  - is the practice of capturing data in real-time from event sources like databases, sensors, mobile devices, cloud services, and software applications in the form of streams of events;
  - storing these event streams durably for later retrieval; manipulating, processing, and reacting to the event streams in real-time as well as retrospectively;
  - and routing the event streams to different destination technologies as needed.
- Event streaming thus ensures a continuous flow and interpretation of data so that the right information is at the right place, at the right time.

# Apache Kafka (contd.)

Kafka combines three key capabilities so we can implement our use cases for event streaming end-to-end :

1. To **publish** (write) and subscribe to (read) streams of events, including continuous import/export of your data from other systems.
2. To **store** streams of events durably and reliably for as long as you want.
3. To **process** streams of events as they occur or retrospectively.

# Apache Kafka (contd.)



# Apache Kafka (contd.)

- And all this functionality is provided in a distributed, highly scalable, elastic, fault-tolerant, and secure manner.
- Kafka can be deployed on bare-metal hardware, virtual machines, and containers, and on-premises as well as in the cloud.

End of the Chapter

# Ethical Issues in Data Science

Chapter 6

# Ethics in Data Science

- Data Ethics encompasses moral obligations of collecting, storing, protecting and using data which sometimes are sensitive, personal, emotional and behavioral
- Studies and evaluates the moral problems related to data (including generation, recording, curation, processing, sharing, uses) and algorithms (including AI, machine learning and robotics) and related practices.
- We learn about ethical problems that occur during the usage of data.
- The modern technologies generates and make use of huge amount of data collected from huge varieties of sources.
- Almost every human activities and behaviors are transformed and translated to the data to make products, decisions, betterment of services etc.

# Ethics in Data Science (contd.)

- Advanced technologies like Machine Learning and AI has brought many innovation to life that makes human life better. Example use of ML to diagnosis a disease.
- Data Ethics is the center of concern for anyone who works/handles data like data analysts or data scientists or any IT professionals.

# Ethical Concerns

Data Ethics concerns during:

## 1. Data Collection

- Data is collected through various technique like Survey, Web scrapping, social medias etc.
- What data needs to be collected and what is the privacy concern related with those data? Is the utmost concern.
- Data can be theft, shared or downloaded via TOS violation concerning data.
- Data from secondary sources or secondary use of data should be well taken care of.
- Because data collection can be repetitious, time-consuming, and tedious there is a temptation to underestimate its importance.
- Those responsible for collecting data must be adequately trained and motivated.
- They should employ methods that limit or eliminate the effect of bias
- They should keep records of what was done by whom and when.

# Ethical Concerns (contd.)

## 2. Data Storage

- Third party storage - Is it safe? Is it secure? Who can access it? What is the mechanism of authentication and authorization?
- Hardware security - Accessibility to the computers or machines to access data. Accessibility and portability of storage devices. Can it be moved? Can it be accessed easily? Where is it stored?
- Data Security - What does the contract say? What is the level of privacy or severity?

# Ethical Concerns (contd.)

## 3. Data Usage, Sharing and Reproducibility

- How the public data is used? Who can access it? Can it be used or reused for different research? Can it be reproduced?
- What are the terms of use for public data?
- Can the data be shared? How is it shared?

# Ethical Concerns (contd.)

## 4. Re-identification and Consent

- Accessing through google, something published for public usage.
- Permission to reuse the data.
- What is the privacy level of data?

## 5. Data Security

### Limiting Access

- Locked Paper Records Offices
- Limiting access to Paper or Electronic records to appropriate personnel
- Password Protection of electronic records
- Defined privileges for electronic data users
- Firewalls to prevent outside access
- Regular Backups and proper archiving

# Bias and Fairness in Data

## Bias

- Machine learning model should produce a fair result.
- ML model based on data of human behavior can also have biased behaviors and tendencies.
- Cognitive biases are an obstacle when trying to interpret information
  - Can easily skew results
  - They are innate tendencies

## Fairness

- Model should treat people, group or community equally irrespective of caste, religion, gender, income level, education level etc.
- Free from unnecessary and undue weighting to certain groups or viewpoints.

# Common Biases

## In Group Favoritism and Outgroup Negativity

### In Group Favoritism

- Also called ingroup love
- Tendency to give preferential treatment to the same group they belong to.
- Very likely to occur during data collection.
- Also likely to occur during data filtering or removing irrelevant data.
- Highly impacts when data diversity is needed.

### Outgroup Negativity

- Also known as outgroup hate.
- Tendency to unlike the behavior, activities or people themselves who do not belong to the group they do.
- Very likely to occur during data collection
- Likely to have covered the most of negative aspects only of outgroup community

# Common Biases (contd.)

## Fundamental Attribution Error

- Tendency that the situational activity or behavior are attributed as intrinsic quality of someone's character.
- These are the judged or observed pattern and is very likely to occur during data collection.
- This feeds negative data to the machine learning model resulting in biased conclusion.

## Negativity Bias

- Tendency of emphasizing negative experiences over positives ones.
- This is very likely to occur during decision making.
- The negative thought about society may expect the negative conclusion from the data science projects.

# Common Biases (contd.)

## Stereotyping

- This is the tendency of expect a certain characteristics or behaviors without having actual information.
- This is the expectation set prior to the exploration.
- This is likely to occur during data wrangling and exploratory data analysis.

## Bandwagon Effect

- Tendency to follow others because
  - Some other top ranked researcher or people did.
  - All people are doing i.e. following the mass.
- Likely to occur during data collection like same sort of data is collected based on pre-collected data or research.
- Some might expect the same result as others has inferred.

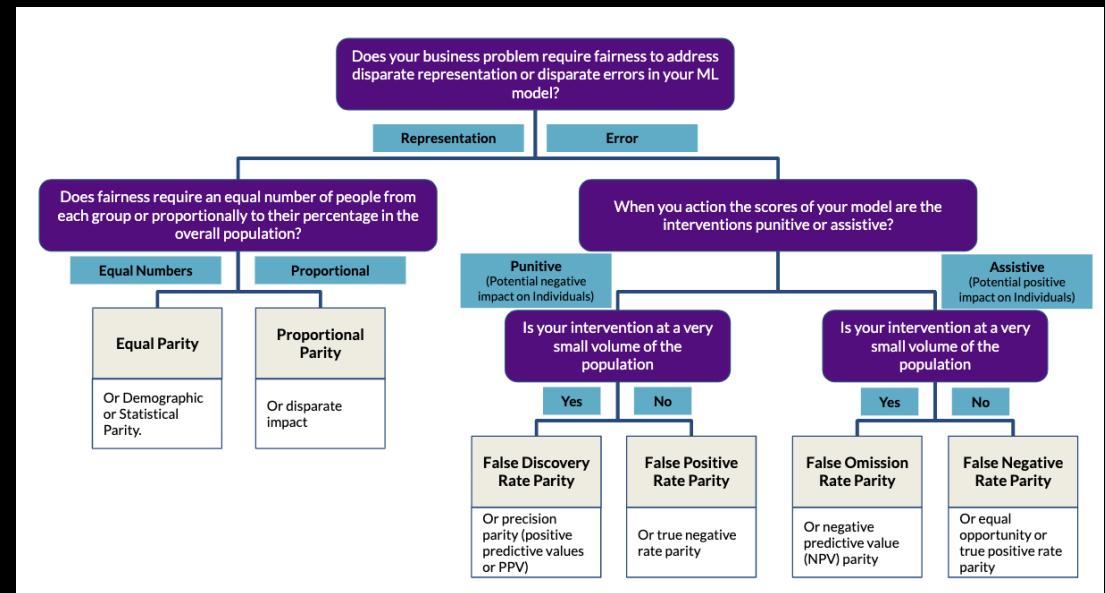
# Common Biases (contd.)

## Bias Blind Spot

- Our tendency not to see own personal biases.
- Likely to ignore or remain unnoticed where there are personal blind spot biases.
- Likely to occur from data collection to result analysis of data science process

# Addressing Bias

- Addressing bias in data science is an extremely complex topic and most importantly there are no universal solutions or silver bullets.
- Before any data scientist can work on the mitigation of biases we need to define fairness in the context of our business problem by consulting the following:
- *As an example, imagine you want to design some ML system to process mortgage loan applications and only a small fraction of applications are by women.*



# Addressing Biases (contd.)

## 1. Group unaware selection

- It's a preventive measure
- This is the process of preventing the bias by eliminating the factor that is likely to cause.
- For example, avoid the collection of gender to avoid bias by gender.

## 2. Adjusted group threshold

- Adjust any biased and unbalanced data
- Because historic biases make women appear less loan-worthy than men, e.g. work history and childcare responsibilities, we use different approval thresholds by group.

# Addressing Biases (contd.)

## 3. Demographic Parity

- The output of the machine learning model should not depend on the sensitive demographic attribute like gender, race, ethnicity, education level etc.

# Addressing Biases (contd.)

## 4. Equal Opportunity

- Equal opportunity fairness ensures that the proportion of people who should be selected by the model ("positives") that are correctly selected by the model is the same for each group. We refer to this proportion as the true positive rate (TPR) or sensitivity of the model.
- A doctor uses a tool to identify patients in need of extra care, who could be at risk for developing serious medical conditions. (This tool is used only to supplement the doctor's practice, as a second opinion.) It is designed to have a high TPR that is equal for each demographic group.
  - Provide equal opportunity to the diverse population.
  - Should be fair enough for the representation in sampling and treatment.
  - E.g. The representation of men and women should be same for granting loan in bank.

# Addressing Biases (contd.)

## 5. Precision Parity

- Tune the output of model to treat the group equally.
- Male and Female should get equal salary based on the position. If machine learning model suggests lesser salary to women compared to men in same post, then such model should be tuned so that both have similar earning.
- When building a ML model, keep de-biasing in mind.

# Common issues with privacy and data ethics

- Data privacy ethical issues are concerns about how data is collected, stored, shared, and used by organizations and individuals.
- Some of these issues include privacy, public safety, digital divide, working conditions, professional standard.

[Ethical Issues Related to Data Privacy and Security: Why We Must Balance Ethical and Legal Requirements in the Connected World - IEEE Digital Privacy](#)

[Ethics in Data Science and Proper Privacy and Usage of Data \(analyticsvidhya.com\)](#)

End of the Chapter