

Statistics and Machine Learning

Parameswari Ettiappan

A black and white photograph of a woman with curly hair, wearing a light-colored shirt, sitting at a desk and working on a laptop. She is looking down at the screen. The background is plain.

High performance. Delivered.

consulting | technology | outsourcing

Goals

- Statistics and Machine Learning



Statistics



Data Ingestion

- Data ingestion is the transportation of data from assorted sources to a storage medium where it can be accessed, used, and analyzed by an organization.
- The destination is typically a data warehouse, data mart, database, or a document store.
- Sources may be almost anything — including SaaS data, in-house apps, databases, spreadsheets, or even information scraped from the internet.

Data Ingestion

- In a broader sense, data ingestion can be understood as a directed dataflow between two or more systems that result in a smooth, and independent, operation (a definition which already implies some independence or automation).
- Ingestion can occur in real-time, as soon as the source produces it, or in batches, when data is input in specific chunks at set periods.
- Generally, three steps occur within data ingestion:
- Data extraction – retrieving data from sources
- Data transformation – validating, cleaning, and normalizing data to ensure accuracy and reliability (sometimes known as trustworthiness)
- Data loading – routing or placing the data in its correct silo or database for analysis



Reasons to automate data ingestion

- Automating data ingestion improves time-to-market goals.
- Automating data ingestion increases scalability.
- Automating data ingestion refocuses on necessary work.
- Automating data ingestion mitigates risk.

Data Ingestion Tools

- Apache NiFi (a.k.a. Hortonworks DataFlow)
- Stream Sets Data Collector (SDC)
- Gobblin
- Sqoop
- Flume
- Kafka
- Airflow
- Spark
- petl

Data Ingestion Tools

- Panoply
- Pandas
- Bubbles
- Bonobo
- Luigi
- etlalchemy

Difference between Structured, Semi-structured and Unstructured data



- Big Data includes huge volume, high velocity, and extensible variety of data.
- These are 3 types: Structured data, Semi-structured data, and Unstructured data.
- Structured data –
- Structured data is a data whose elements are addressable for effective analysis.
- It has been organised into a formatted repository that is typically a database.
- It concern all data which can be stored in database SQL in table with rows and columns.
- They have relational key and can easily mapped into pre-designed fields.
- Today, those data are most processed in development and simplest way to manage information. Example: Relational data.

Difference between Structured, Semi-structured and Unstructured data



- semi-structured data –
- Semi-structured data is information that does not reside in a rational database but that have some organizational properties that make it easier to analyze.
- With some process, you can store them in the relation database (it could be very hard for some kind of semi-structured data), but Semi-structured exist to ease space. Example: XML data.
- Unstructured data –
- Unstructured data is a data that is which is not organised in a pre-defined manner or does not have a pre-defined data model, thus it is not a good fit for a mainstream relational database.
- So for Unstructured data, there are alternative platforms for storing and managing, it is increasingly prevalent in IT systems and is used by organizations in a variety of business intelligence and analytics applications. Example: Word, PDF, Text, Media logs.

Difference between Structured, Semi-structured and Unstructured data



| PROPERTIES | STRUCTURED DATA | SEMI-STRUCTURED DATA | UNSTRUCTURED DATA |
|------------------------|---|--|--|
| Technology | It is based on Relational database table | It is based on XML/RDF | It is based on character and binary data |
| Transaction management | Matured transaction and various concurrency technique | Transaction is adapted from DBMS not matured | No transaction management and no concurrency |
| Version management | Versioning over tuples, row, tables | Versioning over tuples or graph is possible | Versioned as whole |
| Flexibility | It is schema dependent and less flexible | It is more flexible than structured data but less than flexible than unstructured data | It is very flexible and there is absence of schema |
| Scalability | It is very difficult to scale DB schema | Its scaling is simpler than structured data | It is very scalable |
| Robustness | Very robust | New technology, not very spread | — |
| Query performance | Structured query allow complex joining | Queries over anonymous nodes are possible | Only textual query are possible |

Data from Various Sources

- CSV file
- Flat File (tab, space, or any other separator)
- Text File (In a single file — reading data all at once)
- ZIP file
- Multiple Text Files (Data is split over multiple text files)
- Download File from Internet (File hosted on a server)
- Webpage (scraping)
- APIs (JSON)
- Text File (Reading data line by line)
- RDBMS (SQL Tables)

Common data problems

- Inconsistent column names
- Missing data
- Outliers
- Duplicate rows
- Untidy
- Need to process columns
- Column types can signal unexpected data values

Unclean data



| | Continent | Country | female literacy | fertility | population |
|---|-----------|-----------|-----------------|-----------|--------------|
| 0 | ASI | Chine | 90.5 | 1.769 | 1.324655e+09 |
| 1 | ASI | Inde | 50.8 | 2.682 | 1.139965e+09 |
| 2 | NAM | USA | 99.0 | 2.077 | 3.040600e+08 |
| 3 | ASI | Indonésie | 88.8 | 2.132 | 2.273451e+08 |
| 4 | LAT | Brésil | 90.2 | 1.827 | NaN |

- Column name inconsistencies
- Missing data
- Country names are in French



Data Cleansing Techniques

- Data forms the backbone of any data analytics you do.
- Regarding data, there are many things to go wrong – be it the construction, arrangement, formatting, spellings, duplication, extra spaces, and so on.
- To perform the data analytics properly we need various data cleaning techniques so that our data is ready for analysis. It's commonly said that,
- “Data scientists spend 80% of their time cleaning and manipulating data and only 20% of their time actually analyzing it.”

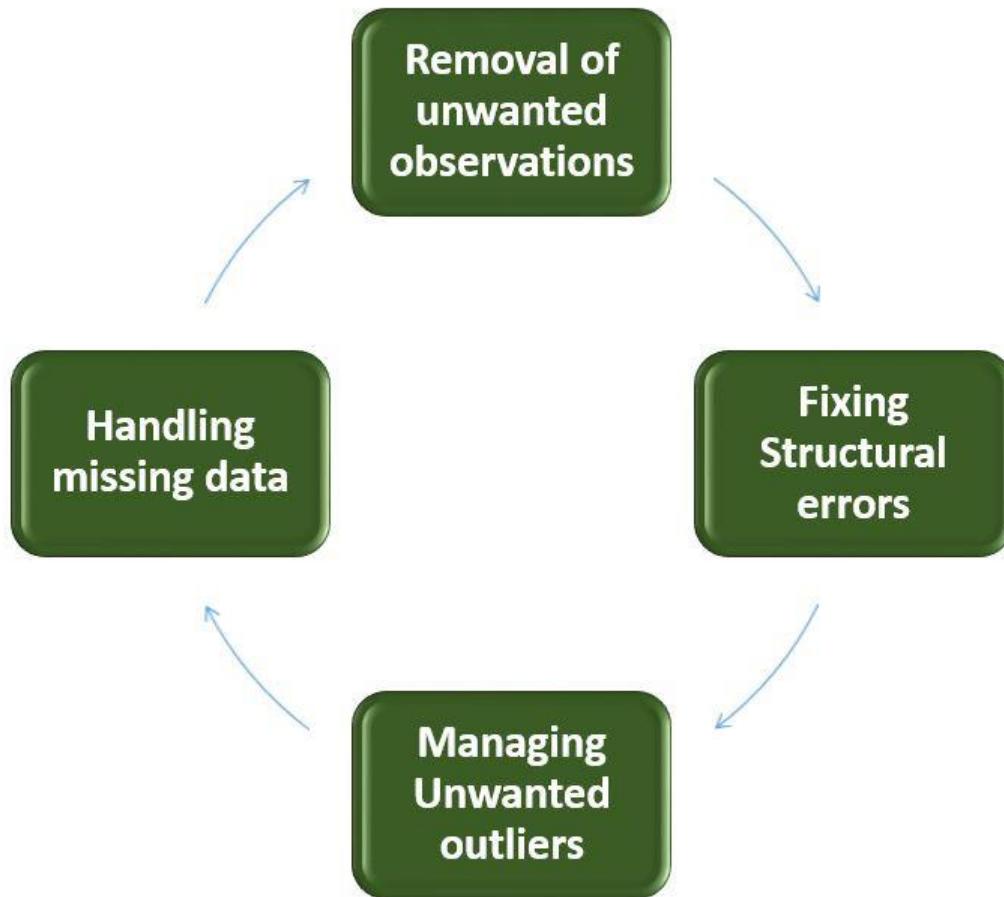
Data Cleansing Techniques

- Data cleansing or data cleaning is the process of identifying and removing (or correcting) inaccurate records from a dataset, table, or database and refers to recognizing unfinished, unreliable, inaccurate or non-relevant parts of the data and then restoring, remodeling, or removing the dirty or crude data.
- Data cleaning techniques may be performed as batch processing through scripting or interactively with data cleansing tools.



Data Cleansing Techniques

- After cleaning, a dataset should be uniform with other related datasets in the operation.
- The discrepancies identified or eliminated may have been basically caused by user entry mistakes, by corruption in storage or transmission, or by various data dictionary descriptions of similar items in various stores.





How Data Cleansing is useful?

- Improves Custom Acquisition-Related Activities
- Better Decision Making
- Streamlined Business Process
- Increased Productivity and Revenue

Steps Involved in Data Cleansing



- Removal of Unwanted Observations
 - Irrelevant Observations – These observations don't fit accurately to the specific problem that the user is trying to solve. During this step, user has to review charts from the Exploratory Analysis.
 - Duplicate Observations – This type of observation arises frequently during data collection and user associated processes to it such as scrape data, combination of datasets from multiple destinations and receive data from different departments or clients.

Steps Involved in Data Cleansing



- Fixing Structural Errors
 - The next step of data cleaning is fixation of structural errors.
 - These type of errors mostly arise during data transfer, measurement and poor data-keeping.
 - Structural errors include mislabelled classes, name feature typos, use of same attribute with different names, etc.

Steps Involved in Data Cleansing



- Managing Unwanted Outliers
 - Unwanted outliers can cause serious issues with certain types of data models.
 - When a user legitimately removes an outlier, it exceptionally improves the model's performance.
 - Thing to remember here is, unless the outlier is proven unwanted or include with suspicious measurements, the user should never remove it.

Steps Involved in Data Cleansing



- Handling Missing Data
 - This one is probably the most complex step of data cleansing. As most of the algorithms don't accept missing values, the user has to manage the missing data in some way. The two most commonly recommended ways to manage missing data are:
 - To drop observations for data that have missing values.
 - To impute the required missing values based on observations.

Steps Involved in Data Cleansing



- Data missingness is always informative in itself and the user requires to inform an algorithm if a value was missing.
- Even the user builds an effective model to impute the values, it will not add any real information as it will be like reinforcing the patterns that are already provided by other features.

Steps Involved in Data Cleansing



- Missing Categorical Data – As per data science, labelling the missing data for categorical features as ‘missing’ is the best way to handle them.
- This step includes essentially adding a new class for the feature. This also nullifies the technical requirement for no missing values.
- Missing Numeric Data – The user has to flag and fill for missing numeric data.
- To perform this, the user needs to flag the observation with a missingness indicator variable.
- Then, replace the missing values with zero to meet the technical requirement of missing values.

What are the Tools in Data Cleansing?



- OpenRefine
- Trifacta Wrangler
- Cloudingo
- IBM InfoSphere Quality Stage
- JASP
- RapidMiner
- Orange
- Talent Data Preparation



Data Cleansing Techniques in Excel

- Data Cleaning Techniques-Get Rid of Extra Spaces-TRIM
- Data Cleaning Techniques-Select and Treat All Blank Cells
- Data Cleaning Techniques-Remove Duplicates
- Data Cleaning Techniques-Highlight Errors
- Data Cleaning Techniques-Convert Numbers Stored as Text into Numbers
- Data Cleaning Techniques-Change Text to Lower/Upper/Proper Case
- Data Cleaning Techniques-Delete all Formatting

Data cleaning

- All data sources potentially include errors and missing values – data cleaning addresses these anomalies.
- Not cleaning data can lead to a range of problems, including linking errors, model misspecification, errors in parameter estimation and incorrect analysis leading users to draw false conclusions.
- The main data cleaning processes are editing, validation and imputation.



Data Removal

- This is the most frowned-upon method. For missing values, it is better to investigate the reason instead of simply eliminating the rows or columns that contain the missing values.
- This is not always avoidable, though. If an entire column is 85% missing and you cannot find another data source, you may not be able to use that column.
- Additionally, it's not optimal to remove outliers, as this is a kind of results doctoring. If you do remove datapoints, explain the reasoning for doing so (such as 85% of the data is irrecoverable) in the results and report.

Direct Correction

- For string consistency correction in smaller categorical sets, it can be trivial to run a unique values search and then write a couple of if-statements to replace errors.
- If you have something like city names, it may be difficult to go with explicit if-statements. You may want to use a fuzzy search and make corrections that way.

Direct Correction

- Numerical consistency errors, such as order of magnitude mismatches, are simple to fix by multiplication or division.
- Binary consistency issues can be corrected if you can accurately assign the non-binary input to one of the binary categories.
- In the set {on, off, broken}, you can probably safely map *broken* to *off*.
- Errors that arise from malfunctioning sensors or human input errors should also be corrected from the source, if possible.
- If you are using publicly available or large-scale, one-time-collection data sets, though, this won't be possible. In those cases, you may want to impute the values.

Scaling



- Scaling changes the ranges of data so some features do not dominate solely because they naturally produce larger values.
- For example, temperature for a city tends to have a much smaller range than the population for a city.
- Distance-based algorithms will assign much greater importance to the population variable, possibly entirely ignoring the temperature variable.
- Scaling brings variables in line with each other while retaining the proportional relationships within the variable. This is seen when you convert to percentages or baseline to 100.

Imputation

- This technique is most closely associated with filling in missing values, but it can be used for incorrect values, too, especially when a direct correction cannot be made.
- Imputation is a fancy way to say guess. However, since we are in the field of data science, this will be a data-driven guess, not just a random guess.
- You can impute values with statistical indicators (like mean, median, mode), hot-decking, stratification, and others.

Imputation

- One approach is to replace every missing value with a statistical indicator.
- In our missing building survey example above, if you just used the mean score for all missing data, you may overlook a strong negative sentiment in that building (which was why the building manager “forgot” to distribute the survey).
- Hot-decking fills in missing values by randomly selecting a value from the set of already-known values. Again, this can cause you to overlook important information believed by “missingness.”

Imputation

- Finally, stratification is beneficial if you already know some patterns in your data.
- The heights of women are, on average, shorter than the heights of men.
- You could split your data set into men and women, then use those sub-indicators for replacement or hot-deck from the subsets of men and women.
- Is it perfect? No, but it's better than using the indicators or hot-decking from the entire population.

Flagging



- This is particularly useful for missing values when you don't want to drop all of them.
- For numeric data, you can add another column to your data set and flag any missing values there.
- This will inform your algorithm of missing values, which may turn out to be influential.
- For categorical variables, simply create a “Missing” or “Unknown” class.

Editing

- Data editing can take place at different levels, and use different methods – the choice is known as the data editing strategy.
- Different data editing strategies are needed for each data type – there is no “one size fits all” solution for a S-DWH(Statistical Datawarehouse).

Macro- and micro-editing

- Macro-editing is generally subjective – eye-balling the output, in isolation and/or relative to similar outputs/previous time periods, or calculating measures of growth and applying rules of thumb to decide whether they are realistic or not.

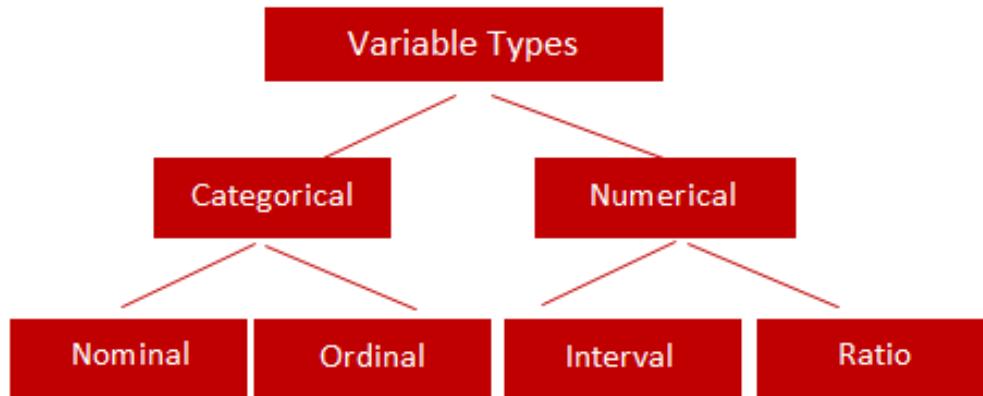
Macro- and micro-editing

- Micro-editing methods are numerous and well-established, and are appropriate for a S-DWH where editing should only take place in the sources layer.

Macro- and micro-editing

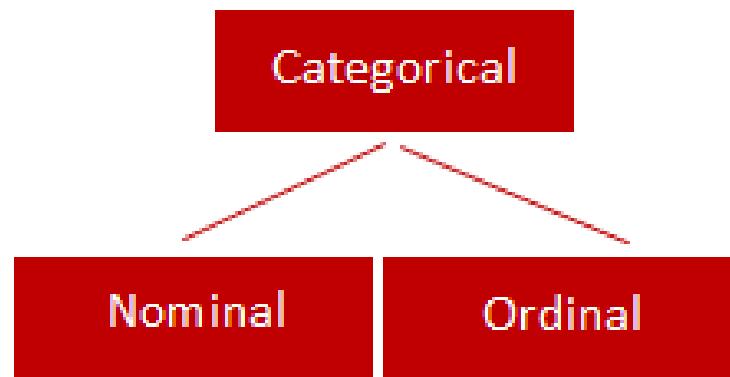
- Hard and soft edits
- Editing methods – known as rules – detect errors, but once a response fails the treatment varies dependent on the rule type.
- Hard edits (some validity, consistency, logical and statistical) do not require validation and can be treated automatically.
- Soft edits (all remaining) require external validation.

Types of Variables



Categorical

- Qualitative data are often termed **categorical data**.
- Data that can be added into **categories** according to their characteristics.
- Examples are gender, social class and blood types



Nominal Variable (Unordered list)

- A variable that has two or more categories, without any implied ordering.

- **Examples :**

Gender - Male, Female

Marital Status - Unmarried, Married, Divorcee

State - New Delhi, Haryana, Illinois, Michigan

Ordinal Variable (Ordered list)

- A variable that has two or more categories, with clear ordering.

Examples :

- Scale - Strongly Disagree, Disagree, Neutral, Agree, Strongly Agree
- Rating - Very low, Low, Medium, Great, Very great

Interval

- An interval variable is similar to an ordinal variable, except that the intervals between the values of the interval variable are equally spaced.
- In other words, it has order and equal intervals.

Examples :

- Temperature in Celsius - Temperature of 30°C is higher than 20°C , and temperature of 20°C is higher than 10°C . The size of these intervals is the same.
- Annual Income in Dollars - Three people who make \$5,000, \$10,000 and \$15,000. The second person makes \$5,000 more than the first person and \$5,000 less than the third person, and the size of these intervals is the same.

Ratio

- It is interval data with a natural zero point.
- When the variable equals 0.0, there is none of that variable.

Examples :

- Height
- Weight
- Temperature in Kelvin - It is a ratio variable, as 0.0 Kelvin really does mean 'no temperature.'
-

Central Location

(a) Arithmetic Mean

Of these “averages,” the most common and familiar is the arithmetic mean, defined by

$$\mu = \frac{1}{N} \sum_1^N x$$

(b) Geometric Mean

The geometric mean is defined as the nth root of the product of n observations:

$$G = \sqrt[n]{x_1 x_2 \cdots x_n},$$

Geometric Mean

- For example, if a strain of bacteria increases its population by 20% in the first hour, 30% in the next hour and 50% in the next hour, we can find out an estimate of the mean percentage growth in population using Geometric mean.
- Geometric mean of 2,3, and 6?
First, multiply the numbers together and then take the cubed root (because there are three numbers) =
 $(2*3*6)^{1/3} = 3.30$
- geometric mean of $1/2$, $1/4$, $1/5$, $9/72$ and $7/4$?
First, multiply the numbers together and then take the 5th root: $(1/2*1/4*1/5*9/72*7/4)^{(1/5)} = 0.35$.

Central Location

(c) Harmonic Mean

The **harmonic mean** involves inverses—i.e., one divided by each of the quantities. The harmonic mean is the inverse of the arithmetic mean of all the inverses.

$$\frac{1}{\frac{1}{x_1} + \frac{1}{x_2} + \dots}$$

Harmonic Mean

Using the weighted harmonic mean (correct):
price–earnings ratio (P/E),

$$P/E = \frac{0.3 + 0.7}{0.3/30 + 0.7/1000} \approx 93.46$$

Harmonic Mean

- You are a stock analyst in an investment bank. Your manager asked you to determine the P/E ratio of the index that tracks the stocks of Company A and Company B.
- Company A reports a market capitalization of \$1 billion and earnings of \$20 million while Company B reports a market capitalization of \$20 billion and earnings of \$5 billion. The index consists of 40% of Company A and 60% Company B.
- Firstly, we need to find the P/E ratios of each company. Remember that the P/E ratio is essentially the market capitalization divided by the earnings.

Harmonic Mean

- $P/E (\text{Company A}) = (\$1 \text{ billion}) / (\$20 \text{ million}) = 50$
 $P/E (\text{Company B}) = (\$20 \text{ billion}) / (\$5 \text{ billion}) = 4$
- We must use the weighted harmonic mean to calculate the P/E ratio of the index. Using the formula for the weighted harmonic mean, the P/E ratio of the index can be found in the following way:
- $P/E (\text{Index}) = (0.4+0.6) / (0.4/50 + 0.6/4) = 6.33$
- Note that if we calculate the P/E ratio of the index using the weighted **arithmetic mean**, it would be significantly overstated:
- $P/E (\text{Index}) = 0.4 \times 50 + 0.6 \times 4 = 22.4$

Median

- If all the items with which we are concerned are sorted in order of increasing magnitude (size), from the smallest to the largest, then the median is the middle item.
- Consider the five items: 12, 13, 21, 27, 31. Then 21 is the median.
- If the number of items is even, the median is given by the arithmetic mean of the two middle items. Consider the six items: 12, 13, 21, 27, 31, 33.
- The median is $(21 + 27) / 2 = 24$.

Mode

- If the frequency varies from one item to another, the mode is the value which appears most frequently.
- In the case of continuous variables the frequency depends upon how many digits are quoted, so the mode is more usefully considered as the midpoint of the class with the largest frequency.

Grouped Mean, Median and Mode

- You grew fifty baby carrots using special soil. You dig them up and measure their lengths (to the nearest mm) and group the results:

| Length (mm) | Frequency |
|-------------|-----------|
| 150 - 154 | 5 |
| 155 - 159 | 2 |
| 160 - 164 | 6 |
| 165 - 169 | 8 |
| 170 - 174 | 9 |
| 175 - 179 | 11 |
| 180 - 184 | 6 |
| 185 - 189 | 3 |

| Length (mm) | Midpoint x | Frequency f | fx |
|-------------|------------|-------------|-------------|
| 150 - 154 | 152 | 5 | 760 |
| 155 - 159 | 157 | 2 | 314 |
| 160 - 164 | 162 | 6 | 972 |
| 165 - 169 | 167 | 8 | 1336 |
| 170 - 174 | 172 | 9 | 1548 |
| 175 - 179 | 177 | 11 | 1947 |
| 180 - 184 | 182 | 6 | 1092 |
| 185 - 189 | 187 | 3 | 561 |
| Totals: | | 50 | 8530 |

**Estimated
Mean = 8530/50 = 170.6 mm**

Grouped Mean, Median and Mode

- Median

The Median is the mean of the 25th and the 26th length, so is in the **170 - 174** group:

L = 169.5 (the lower class boundary of the 170 - 174 group) \rightarrow $(167+172)/2$

n = 50

B = 5 + 2 + 6 + 8 = 21

G = 9 \rightarrow frequency

w = 5 \rightarrow Group width(150-154)

Estimated Median= $169.5 + (50/2) - 219 \times 5$

$$= 169.5 + 2.22\dots$$

$$= \mathbf{171.7 \text{ mm}} \text{ (to 1 decimal)}$$

Grouped Mean, Median and Mode

- Mode

The Modal group is the one with the highest frequency, which is **175 - 179**:

$L = 174.5$ (the lower class boundary of the 175 - 179 group)

$$f_{m-1} = 9$$

$$f_m = 11$$

$$f_{m+1} = 6$$

$$w = 5$$

$$\text{Estimated Mode} = 174.5 + \frac{11 - 9}{(11 - 9) + (11 - 6)} \times 5$$

$$= 174.5 + 1.42\dots$$

$$= 175.9 \text{ mm (to 1 decimal)}$$

Variability or Spread of the Data

- **Mean Deviation from the Mean**

The mean deviation from the mean, defined as –

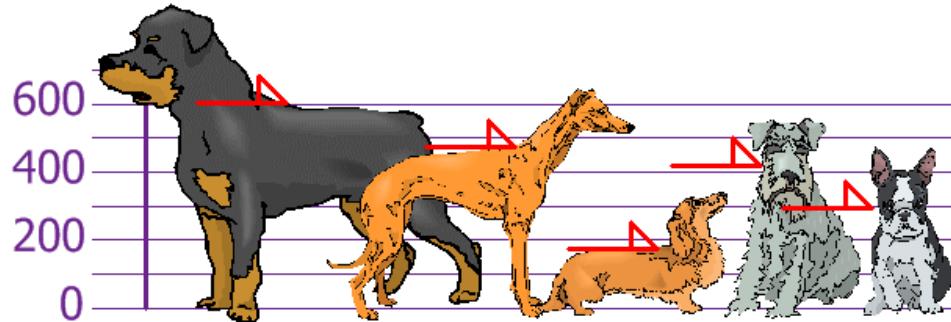
$$\sum_{i=1}^N (x_i - \bar{x}) / N$$

- **Mean Absolute Deviation from the Mean**

$$\sum_{i=1}^N |x_i - \bar{x}| / N$$

Mean Absolute Deviation

- Example: You and your friends have just measured the heights of your dogs (in millimeters):



The heights (at the shoulders) are: 600mm, 470mm, 170mm, 430mm and 300mm.

Mean Absolute Deviation

- Step 1: Find the **mean**:
- $\mu = 600 + 470 + 170 + 430 + 3005 = 19705 = 394$
- Step 2: Find the **Absolute Deviations**:

| x | $ x - \mu $ |
|-----|-------------------------|
| 600 | 206 |
| 470 | 76 |
| 170 | 224 |
| 430 | 36 |
| 300 | 94 |
| | $\Sigma x - \mu = 636$ |

Step 3. Find the **Mean Deviation**:

$$\text{Mean Deviation} = \frac{\Sigma|x - \mu|}{N} = \frac{636}{5} = 127.2$$

So, on average, the dogs' heights are **127.2 mm from the mean**.

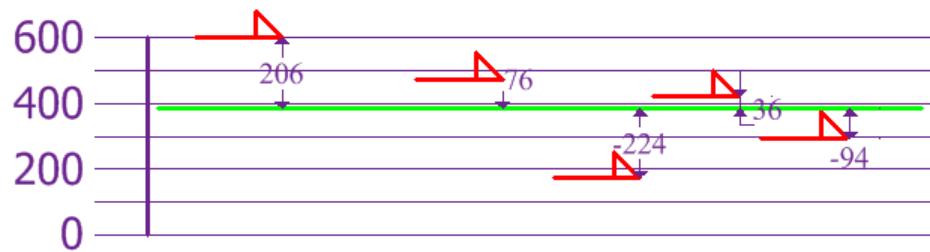
Variability or Spread of the Data

- **Variance**
- Since squares of both positive and negative real numbers are always positive, the variance is always positive.

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

Variability or Spread of the Data

- **Variance**
- To calculate the Variance, take each difference, square it, and then average the result:



Variance

σ^2

$$\begin{aligned}
 & (206^2 + 76^2 + \\
 & = (-224)^2 + 36^2 + \\
 & (-94)^2) / 5 \\
 & (42436 + 5776 + \\
 & = 50176 + 1296 + \\
 & 8836) / 5 \\
 & = 108520 / 5 \\
 & = 21704
 \end{aligned}$$

So the Variance is **21,704**

Variability or Spread of the Data

- Standard Deviation

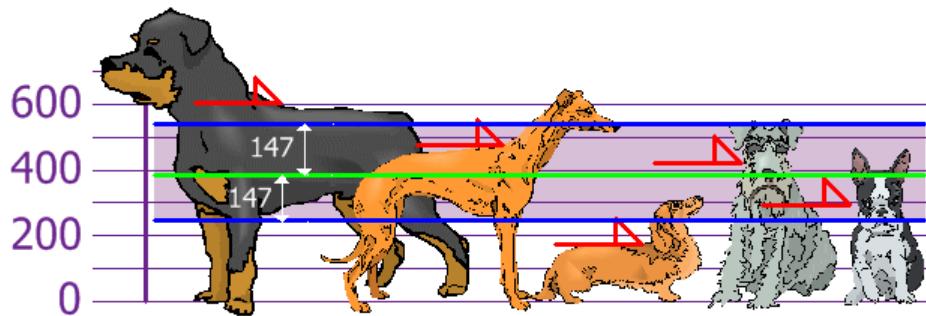
$$\sigma = \sqrt{\frac{\sum_{i=1}^N (x_i - \mu)^2}{N}}$$

Variability or Spread of the Data

- Standard Deviation
- And the Standard Deviation is just the square root of Variance, so:

Standard Deviation

$$\begin{aligned}
 \sigma &= \sqrt{21704} \\
 &= 147.32\ldots \\
 &= \mathbf{147} \text{ (to the nearest mm)}
 \end{aligned}$$



Multi Dimensional Array

- Create a multidimensional array as follows:
- In: `m = array([arange(2), arange(2)])`
- In: `m`
- Out:
- `array([[0, 1],`
- `[0, 1]])`
- Show the array shape as follows:

- In: `m.shape`
- Out: `(2, 2)`

NumPy numerical types

| Type | Description |
|---------|---|
| bool | Boolean (True or False) stored as a bit |
| int | Platform integer (normally either int32 or int64) |
| int8 | Byte (-128 to 127) |
| int16 | Integer (-32768 to 32767) |
| int32 | Integer (- 2^{31} to $2^{31} - 1$) |
| int64 | Integer (- 2^{63} to $2^{63} - 1$) |
| uint8 | Unsigned integer (0 to 255) |
| uint16 | Unsigned integer (0 to 65535) |
| uint32 | Unsigned integer (0 to $2^{32} - 1$) |
| uint64 | Unsigned integer (0 to $2^{64} - 1$) |
| float16 | Half precision float: sign bit, 5 bits exponent, and 10 bits mantissa |
| float32 | Single precision float: sign bit, 8 bits exponent, and 23 bits mantissa |

NumPy numerical types

| | |
|--------------------------|--|
| float64 or float | Double precision float: sign bit, 11 bits exponent, and 52 bits mantissa |
| complex64 | Complex number, represented by two 32-bit floats (real and imaginary components) |
| complex128 or complex | Complex number, represented by two 64-bit floats (real and imaginary components) |

NumPy numerical types

| Type | Character code |
|------------------------|----------------|
| integer | i |
| Unsigned integer | u |
| Single precision float | f |
| Double precision float | d |
| bool | b |
| complex | D |
| string | S |
| unicode | U |
| Void | V |

Manipulating array shapes

- Flattening is transforming a multidimensional array into a one-dimensional array.

Quartiles, Deciles, Percentiles, and Quantiles

- Quartiles, deciles, and percentiles divide a frequency distribution into a number of parts containing equal frequencies.
- **Quartiles** divide the range of values into four parts, each containing one quarter of the values.
- **Percentiles** divide into a hundred parts, each containing one hundredth of the total frequency.
- **Quantile** divides a frequency distribution into parts containing stated proportions of a distribution.

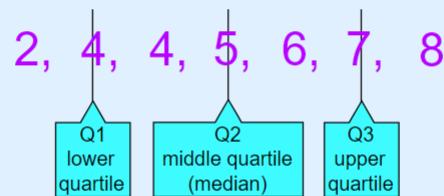
Quartiles

- Quartiles are the values that divide a list of numbers into quarters:
- Put the list of numbers **in order**
- Then cut the list into **four equal parts**
- The Quartiles are at the "cuts"

Example: 5, 7, 4, 4, 6, 2, 8

Put them in order: 2, 4, 4, 5, 6, 7, 8

Cut the list into quarters:



And the result is:

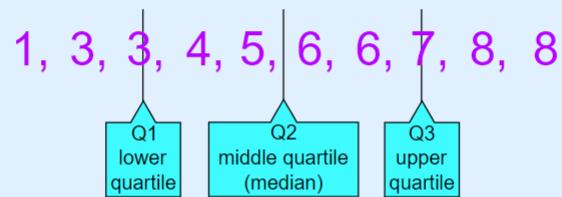
- Quartile 1 (Q1) = 4
- Quartile 2 (Q2), which is also the Median, = 5
- Quartile 3 (Q3) = 7

Quartiles

Example: 1, 3, 3, 4, 5, 6, 6, 7, 8, 8

The numbers are already in order

Cut the list into quarters:



In this case Quartile 2 is half way between 5 and 6:

$$Q2 = (5+6)/2 = 5.5$$

And the result is:

- Quartile 1 (Q1) = 3
- Quartile 2 (Q2) = 5.5
- Quartile 3 (Q3) = 7

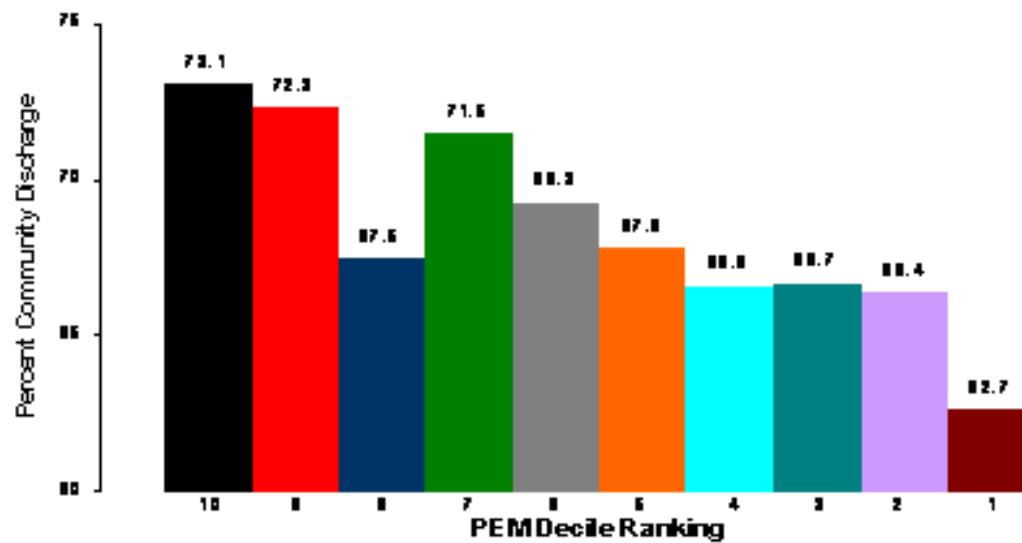
Decile

- What is a Decile used for in Real Life?
- Deciles and decile ranks are used more often in real life than in the classroom. For example, [Australia](#) uses decile ranks to report drought data. Deciles 1-2 represent the lowest 20% (“much below normal”). That means droughts that are “much below normal” don’t occur more than 20% of the time.
- Deciles are also commonly used for college admissions and high school rankings. For example, [this chart](#) from Roanoke College shows the high school decile rankings for the student body.

Decile

| Decile Rank | Percentile |
|-------------|------------|
| 1 | 10th |
| 2 | 20th |
| 3 | 30th |
| 4 | 40th |
| 5 | 50th |
| 6 | 60th |
| 7 | 70th |
| 8 | 80th |
| 9 | 90th |

Decile



Normal Distribution

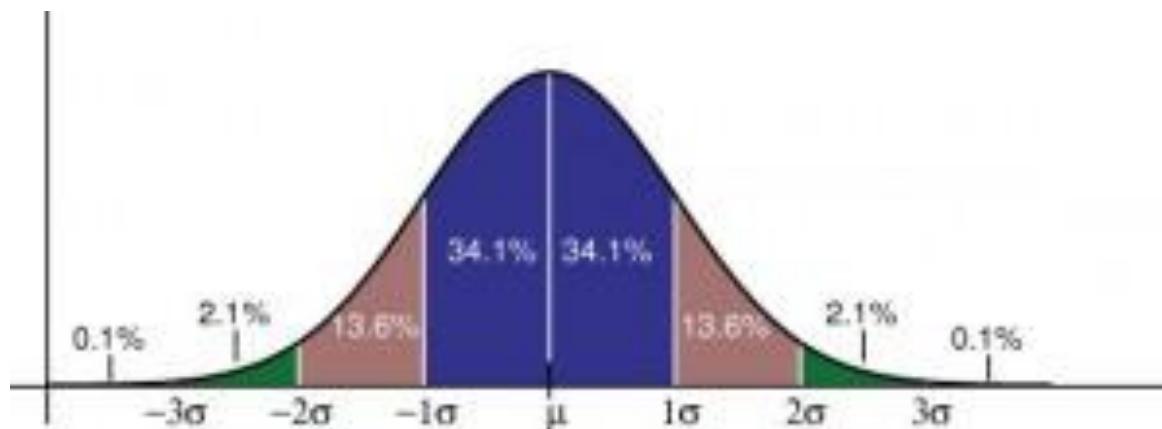
- A normal distribution, sometimes called the bell curve, is a distribution that occurs naturally in many situations.
- For example, the bell curve is seen in tests like the SAT and GRE.
- The bulk of students will score the average (C), while smaller numbers of students will score a B or D.
- An even smaller percentage of students score an F or an A.
- This creates a distribution that resembles a bell (hence the nickname).
- The bell curve is symmetrical. Half of the data will fall to the left of the mean; half will fall to the right.

Normal Distribution

- Many groups follow this type of pattern. That's why it's widely used in business, statistics and in government bodies like the [FDA](#):
- Heights of people.
- Measurement errors.
- Blood pressure.
- Points on a test.
- IQ scores.
- Salaries.

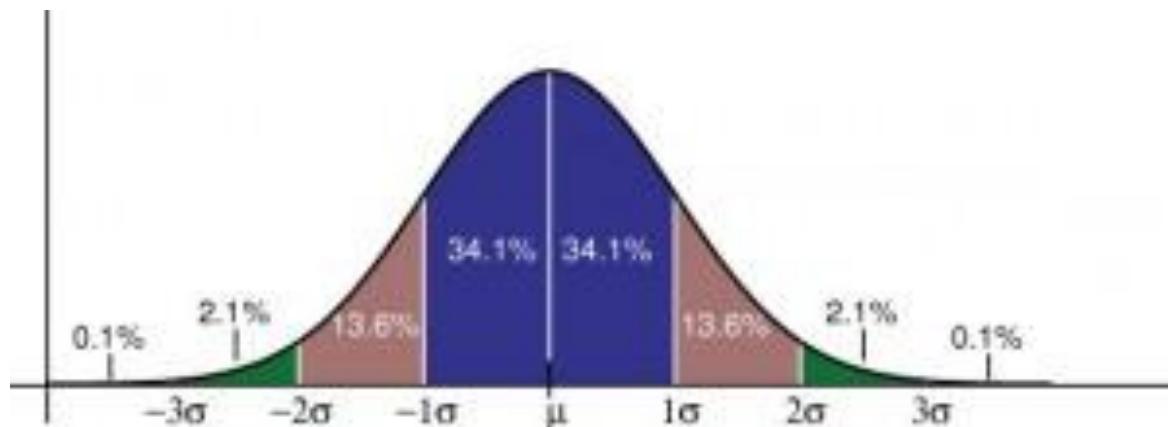
Normal Distribution

- The empirical rule tells you what percentage of your data falls within a certain number of standard deviations from the mean:
 - 68% of the data falls within one standard deviation of the mean.
 - 95% of the data falls within two standard deviations of the mean.
 - 99.7% of the data falls within three standard deviations of the mean.

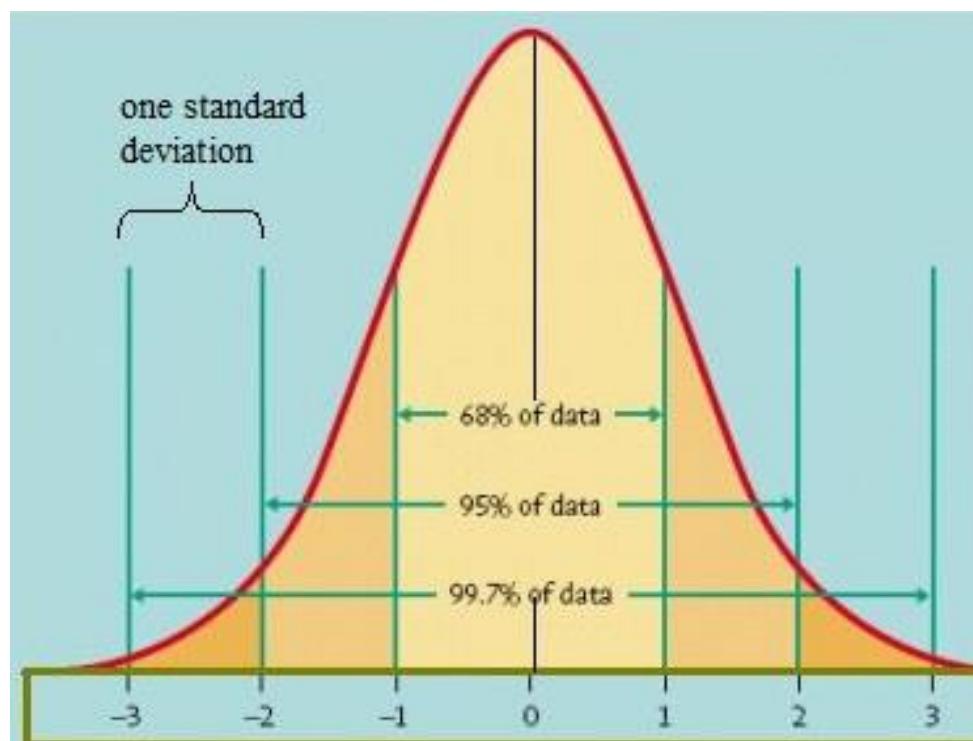


Normal Distribution

- The standard deviation controls the spread of the distribution.
- A smaller standard deviation indicates that the data is tightly clustered around the mean; the normal distribution will be taller.
- A larger standard deviation indicates that the data is spread out around the mean; the normal distribution will be flatter and wider.



Standard Normal Model



Normal Distribution

- Properties of a normal distribution
 - The mean, mode and median are all equal.
 - The curve is symmetric at the center (i.e. around the mean, μ).
 - Exactly half of the values are to the left of center and exactly half the values are to the right.
 - The total area under the curve is 1.

Word problems with normal distribution: “Between”: Steps



- **Step 1:** Identify the parts of the word problem. The word problem will identify:
 - The mean average or μ .
 - Standard deviation (σ).
 - Number selected (i.e. “choose one at random” or “select ten at random”).
 - X: the numbers associated with “between” (i.e. “between \$5,000 and \$10,000” would have X as 5,000 and as \$10,000).

Word problems with normal distribution: “Between”: Steps

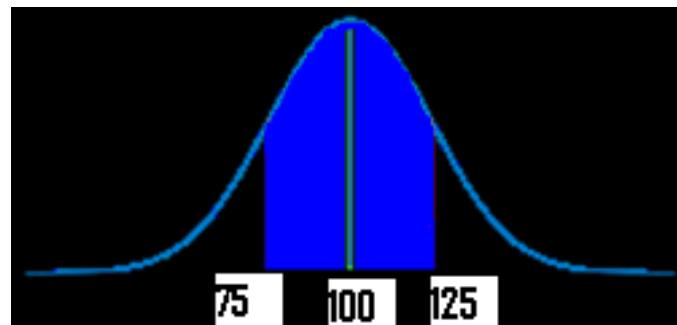


- **Step 1:** Identify the parts of the word problem. The word problem will identify:
 - The mean (average or μ).
 - Standard deviation (σ).
 - Number selected (i.e. “choose one at random” or “select ten at random”).
 - X: the numbers associated with “between” (i.e. “between \$5,000 and \$10,000” would have X as 5,000 and as \$10,000).

Word problems with normal distribution: “Between”: Steps



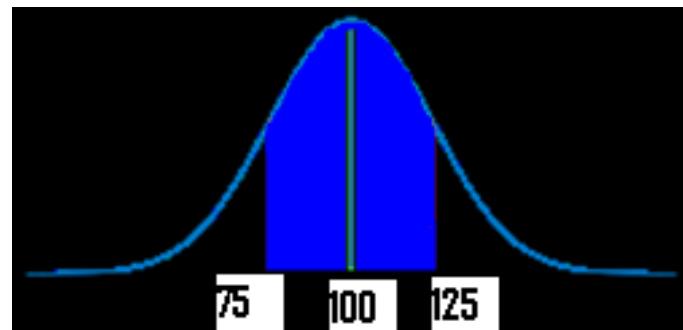
- **Step 2:** *Draw a graph.* Put the mean you identified in Step 1 in the center.
- Put the number associated with “between” on the graph (take a guess at where the numbers would fall—it doesn’t have to be exact).
- For example, if your mean was \$100, and you were asked for “hourly wages between \$75 and \$125”) your graph will look something like this:



Word problems with normal distribution: “Between”: Steps



- Step 3:Figure out the z-scores.
- Plug the first X value (in my graph above, it's 75) into the z value formula and solve. The μ (the mean), is 100 from the sample graph. You can get these figures (including σ , the standard deviation) from your answers in step 1 :
- z score formula
$$z = \frac{X - \mu}{\sigma} =$$
- *Note: if the formula confuses you, all this formula is asking you to do is:
- subtract the mean from X
- divide by the standard deviation.



Word problems with normal distribution: “Between”: Steps



- **Step 4:** Repeat step 3 for the second X.
- **Step 5:** Take the numbers from step 3 and 4 and use them to find the area in the z-table.
- **Step 6a:**
 - Convert the answer from step 5 into a percentage.
 - For example, 0.1293 is 12.93%.
 - That's it—skip step 6b!
- **Step 6b**
 - Multiply the sample size (found in step 1) by the z-value you found in step 4. For example, $0.300 * 100 = 30$.
 - That's it!

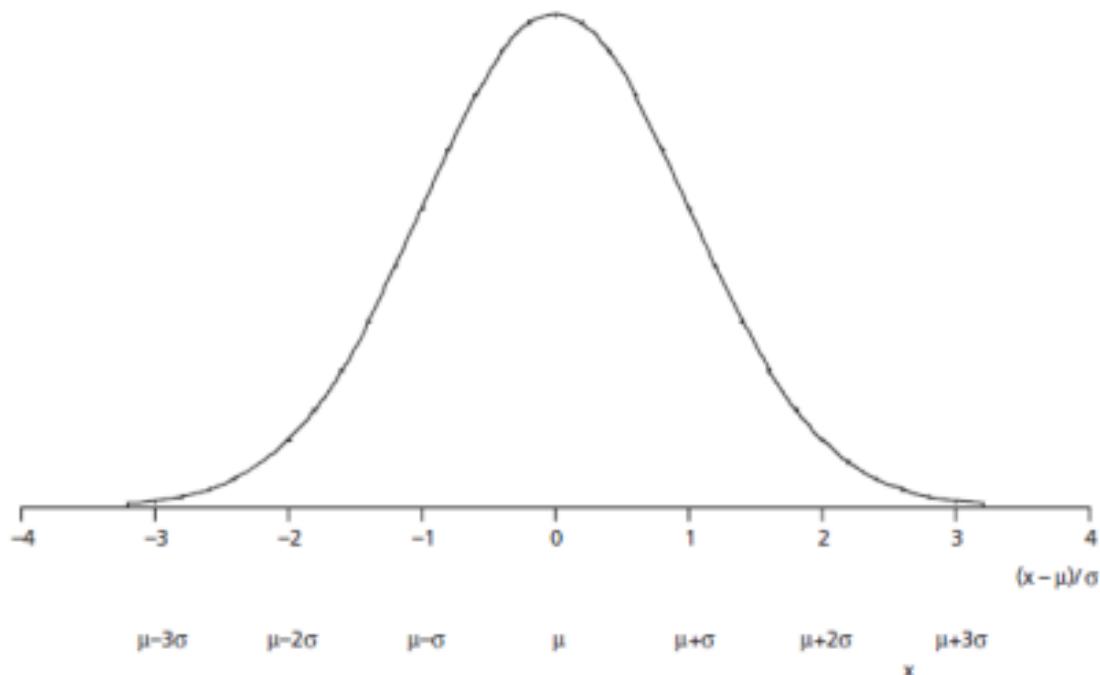
Normal Distribution

- They are approximately symmetrical, and the mode is close to the centre of the distribution.
- The mean, median, and mode are close together.
- The shape of the distribution can be approximated by a bell: nearly flat on top, then decreasing more quickly, then decreasing more slowly toward the tails of the distribution.
- This implies that values close to the mean are relatively frequent, and values farther from the mean tend to occur less frequently.

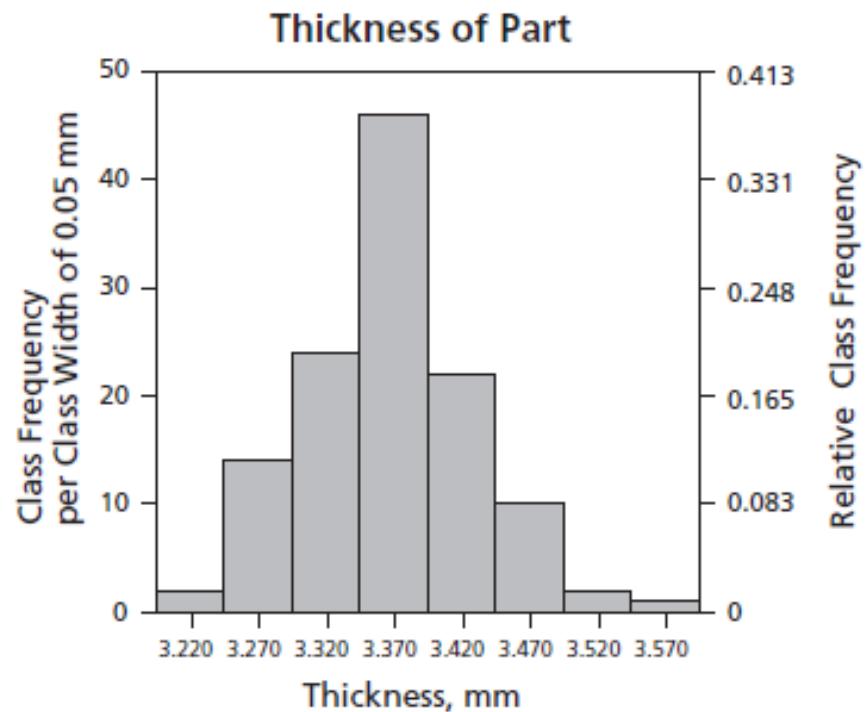
Probability Density Function

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

Normal Distribution



Normal Distribution



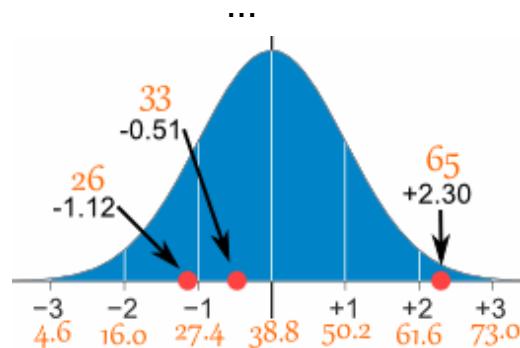
Normal Distribution

- A survey of daily travel time had these results (in minutes):
 - 26, 33, 65, 28, 34, 55, 25, 44, 50, 36, 26, 37, 43, 62, 35, 38, 45, 32, 28, 34
- The **Mean is 38.8 minutes**, and the **Standard Deviation is 11.4 minutes**
- To convert **26**:
 - first subtract the mean: $26 - 38.8 = -12.8$,
 - then divide by the Standard Deviation: $-12.8/11.4 = -1.12$
 - So **26 is -1.12 Standard Deviations from the Mean**

Normal Distribution

Here are the first three conversions

| Original Value | Calculation | Standard Score (z-score) |
|----------------|----------------------|-----------------------------|
| 26 | $(26-38.8) / 11.4 =$ | -1.12 |
| 33 | $(33-38.8) / 11.4 =$ | -0.51 |
| 65 | $(65-38.8) / 11.4 =$ | +2.30 |
| ... | ... | ... |



Feature Scaling



- StandardScaler
- MinMaxScaler
- RobustScaler
- Normalizer

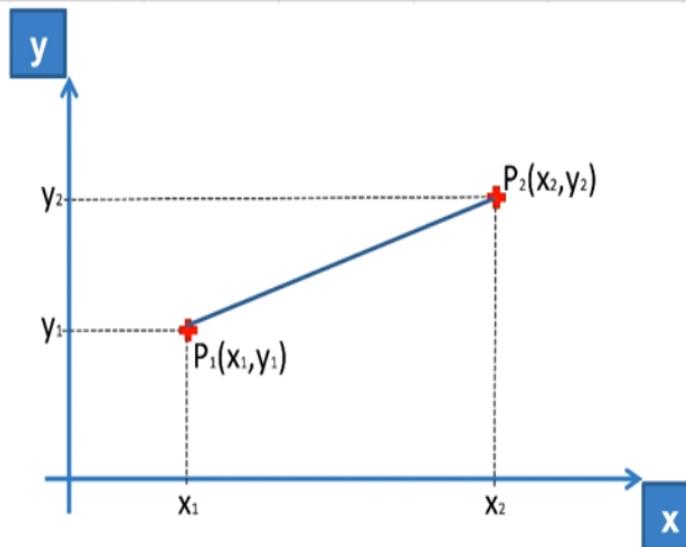
Standard Scaler

The `StandardScaler` assumes your data is normally distributed within each feature and will scale them such that the distribution is now centred around 0, with a standard deviation of 1.

The mean and standard deviation are calculated for the feature and then the feature is scaled based on:

$$\frac{x_i - \text{mean}(x)}{\text{stdev}(x)}$$

| | | | | |
|----|---------|-------------|-------------|-----------|
| 1 | Country | Age | Salary | Purchased |
| 2 | France | 44 | 72000 | No |
| 3 | Spain | 27 | 48000 | Yes |
| 4 | Germany | 30 | 54000 | No |
| 5 | Spain | 38 | 61000 | No |
| 6 | Germany | 40 | 63777.77778 | Yes |
| 7 | France | 35 | 58000 | Yes |
| 8 | Spain | 38.77777778 | 52000 | No |
| 9 | France | 48 | 79000 | Yes |
| 10 | Germany | 50 | 83000 | No |
| 11 | France | 37 | 67000 | Yes |
| 12 | | | | |
| 13 | | | | |
| 14 | | | | |
| 15 | | | | |
| 16 | | | | |
| 17 | | | | |
| 18 | | | | |
| 19 | | | | |
| 20 | | | | |
| 24 | | | | |



$$\text{Euclidean Distance between } P_1 \text{ and } P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Feature Scaling

| Standardisation | Normalisation |
|--|---|
| $x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation } (x)}$ | $x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$ |

Feature Scaling



- StandardScaler
- MinMaxScaler
- RobustScaler
- Normalizer

Min-Max Scaler

The `MinMaxScaler` is probably the most famous scaling algorithm, and follows the following formula for each feature:

$$\frac{x_i - \min(x)}{\max(x) - \min(x)}$$

It essentially shrinks the range such that the range is now between 0 and 1 (or -1 to 1 if there are negative values).

Feature Scaling

Robust Scaler

The `RobustScaler` uses a similar method to the Min-Max scaler but it instead uses the interquartile range, rather than the min-max, so that it is robust to outliers. Therefore it follows the formula:

$$\frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)}$$

For each feature.

Feature Scaling

Normalizer

The normalizer scales each value by dividing each value by its magnitude in n -dimensional space for n number of features.

Say your features were x, y and z Cartesian co-ordinates your scaled value for x would be:

$$\frac{x_i}{\sqrt{x_i^2 + y_i^2 + z_i^2}}$$

Each point is now within 1 unit of the origin on this Cartesian co-ordinate system.

Probability

- Many events can't be predicted with total certainty. The best we can say is how likely they are to happen, using the idea of probability.

Tossing a Coin



When a coin is tossed, there are two possible outcomes:

- heads (H) or
- tails (T)

We say that the probability of the coin landing **H** is $\frac{1}{2}$

And the probability of the coin landing **T** is $\frac{1}{2}$

Throwing Dice



When a single **die** is thrown, there are six possible outcomes: **1, 2, 3, 4, 5, 6**.

The probability of any one of them is $\frac{1}{6}$

Probability

Probability

In general:

$$\text{Probability of an event happening} = \frac{\text{Number of ways it can happen}}{\text{Total number of outcomes}}$$

Example: the chances of rolling a "4" with a die

Number of ways it can happen: 1 (there is only 1 face with a "4" on it)

Total number of outcomes: 6 (there are 6 faces altogether)

$$\text{So the probability} = \frac{1}{6}$$

Example: there are 5 marbles in a bag: 4 are blue, and 1 is red. What is the probability that a blue marble gets picked?

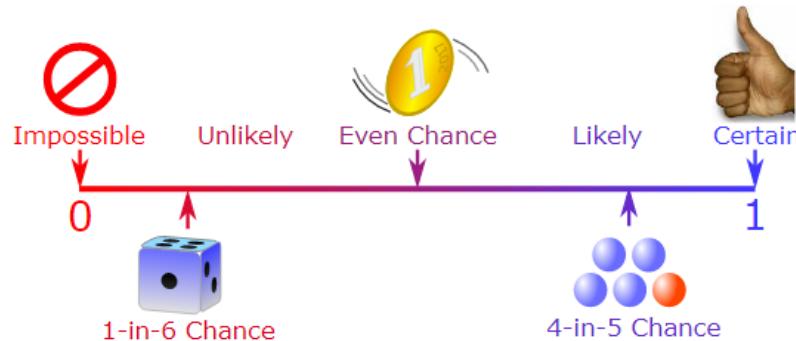
Number of ways it can happen: 4 (there are 4 blues)

Total number of outcomes: 5 (there are 5 marbles in total)

$$\text{So the probability} = \frac{4}{5} = 0.8$$

Probability Line

We can show probability on a [Probability Line](#):



Probability is always between 0 and 1

Probability is Just a Guide

Probability does not tell us exactly what will happen, it is just a guide

Example: toss a coin 100 times, how many Heads will come up?

Probability says that heads have a $\frac{1}{2}$ chance, so we can **expect 50 Heads**.

But when we actually try it we might get 48 heads, or 55 heads ... or anything really, but in most cases it will be a number near 50.

Probability is Just a Guide

Probability does not tell us exactly what will happen, it is just a guide

Example: toss a coin 100 times, how many Heads will come up?

Probability says that heads have a $\frac{1}{2}$ chance, so we can **expect 50 Heads**.

But when we actually try it we might get 48 heads, or 55 heads ... or anything really, but in most cases it will be a number near 50.

Learn more at [Probability_Index](#).

Words

Some words have special meaning in Probability:



Experiment: a repeatable procedure with a set of possible results.

Example: Throwing dice

We can throw the dice again and again, so it is repeatable.

The set of possible results from any single throw is $\{1, 2, 3, 4, 5, 6\}$



Conditional Probability



- Independent Events
 - Events can be "Independent", meaning each event is not affected by any other events.

Example: Tossing a coin.

Each toss of a coin is a perfect isolated thing.

What it did in the past will not affect the current toss.

The chance is simply 1-in-2, or 50%, just like ANY toss of the coin.

So each toss is an **Independent Event**.



Conditional Probability



- Dependent Events
 - But events can also be "dependent" ... which means they can be affected by previous events ...

Example: Marbles in a Bag

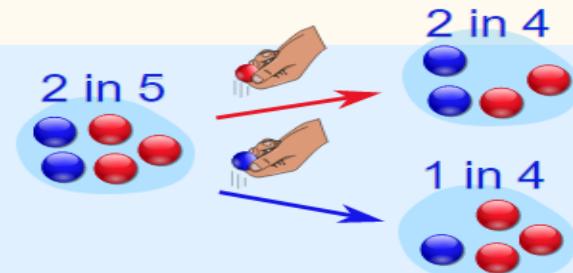
2 blue and 3 red marbles are in a bag.

What are the chances of getting a blue marble?

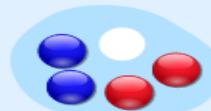
The chance is **2 in 5**

But after taking one out the chances change!

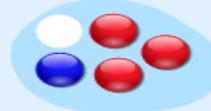
So the next time:



if we got a **red** marble before, then the chance of a blue marble next is **2 in 4**



if we got a **blue** marble before, then the chance of a blue marble next is **1 in 4**



Probability

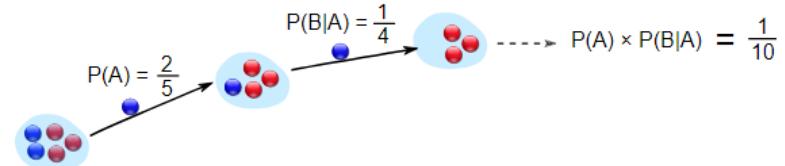
- We love notation in mathematics! It means we can then use the power of algebra to play around with the ideas. So here is the notation for probability:
- $P(A)$ means "Probability Of Event A"
- In our marbles example Event A is "get a Blue Marble first" with a probability of $2/5$:
- $P(A) = 2/5$
- And Event B is "get a Blue Marble second" ... but for that we have 2 choices:
 - If we got a Blue Marble first the chance is now $1/4$
 - If we got a Red Marble first the chance is now $2/4$
- So we have to say which one we want, and use the symbol " $|$ " to mean "given":
- $P(B|A)$ means "Event B given Event A"

Probability

- In other words, event A has already happened, now what is the chance of event B?
- $P(B|A)$ is also called the "Conditional Probability" of B given A.
- And in our case:
- $P(B|A) = 1/4$

$$P(B|A) = 1/4$$

So the probability of getting **2 blue marbles** is:



And we write it as

$$P(\text{Event A and Event B}) = P(\text{Event A}) \times P(\text{Event B} | \text{Event A})$$

"Probability Of" "Given"
 ↓ ↓
 P(A and B) = P(A) × P(B | A)
 ↑ ↑
 Event A Event B

"Probability of **event A and event B** equals
the probability of **event A** times the probability of **event B given event A**"

Probability

Big Example: Soccer Game

You are off to soccer, and want to be the Goalkeeper, but that depends who is the Coach today:

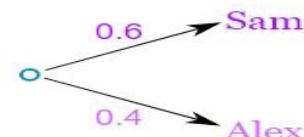
- with Coach Sam the probability of being Goalkeeper is **0.5**
- with Coach Alex the probability of being Goalkeeper is **0.3**



Sam is Coach more often ... about 6 out of every 10 games (a probability of **0.6**).

So, what is the probability you will be a Goalkeeper today?

Let's build a [tree diagram](#). First we show the two possible coaches: Sam or Alex:



The probability of getting Sam is 0.6, so the probability of Alex must be 0.4 (together the probability is 1)

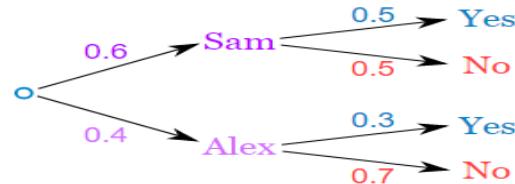
Now, if you get Sam, there is 0.5 probability of being Goalie (and 0.5 of not being Goalie):



If you get Alex, there is 0.3 probability of being Goalie (and 0.7 not):

Probability

If you get Alex, there is 0.3 probability of being Goalie (and 0.7 not):



The tree diagram is complete, now let's calculate the overall probabilities. Remember that:

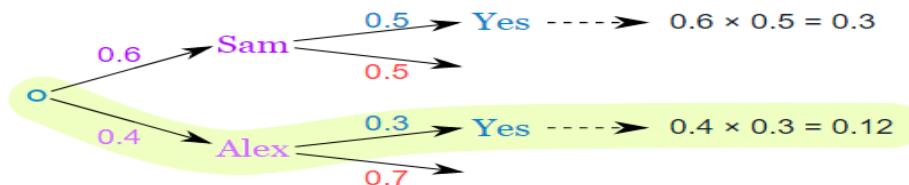
$$P(A \text{ and } B) = P(A) \times P(B|A)$$

Here is how to do it for the "Sam, Yes" branch:



(When we take the 0.6 chance of Sam being coach times the 0.5 chance that Sam will let you be Goalkeeper we end up with an 0.3 chance.)

But we are not done yet! We haven't included Alex as Coach:



An 0.4 chance of Alex as Coach, followed by an 0.3 chance gives 0.12

And the two "Yes" branches of the tree together make:

Probability

...

An 0.4 chance of Alex as Coach, followed by an 0.3 chance gives 0.12

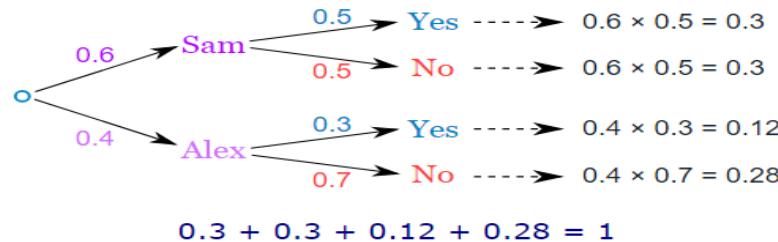
And the two "Yes" branches of the tree together make:

$0.3 + 0.12 = 0.42$ probability of being a Goalkeeper today

(That is a 42% chance)

Check

One final step: complete the calculations and make sure they add to 1:



Yes, they add to 1, so that looks right.

Friends and Random Numbers

Here is another quite different example of Conditional Probability.

4 friends (Alex, Blake, Chris and Dusty) each choose a random number between 1 and 5. What is the chance that any of them chose the same number?

Let's add our friends one at a time ...

Types of Learning

- **Supervised (inductive) learning**
 - Training data includes desired outputs
- **Unsupervised learning**
 - Training data does not include desired outputs
- **Semi-supervised learning**
 - Training data includes a few desired outputs
- **Reinforcement learning**
 - Rewards from sequence of actions



Types of Machine Learning

Machine Learning

Supervised

Task driven
(Regression /
Classification)

Unsupervised

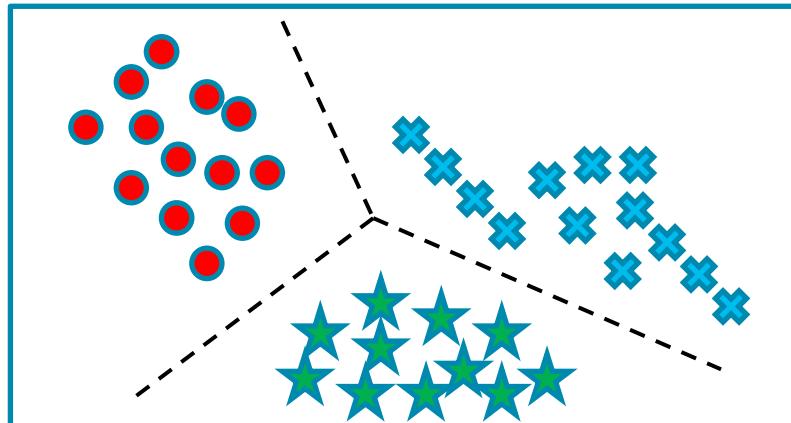
Data driven
(Clustering)

 Analytics Vidhya

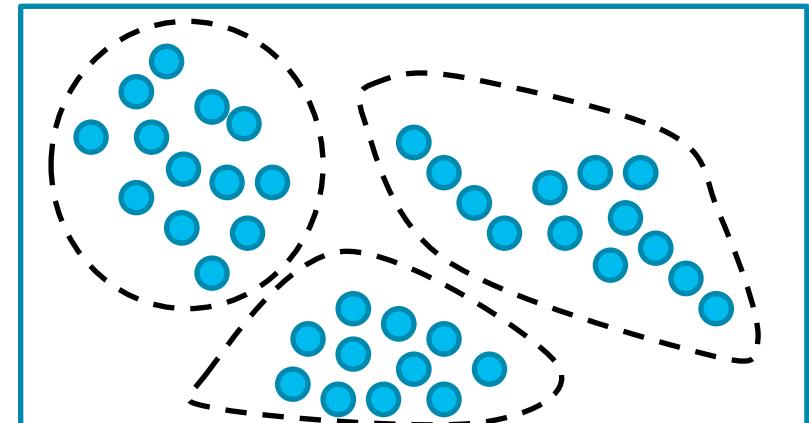
Reinforcement

Algorithm learns to
react to an
environment

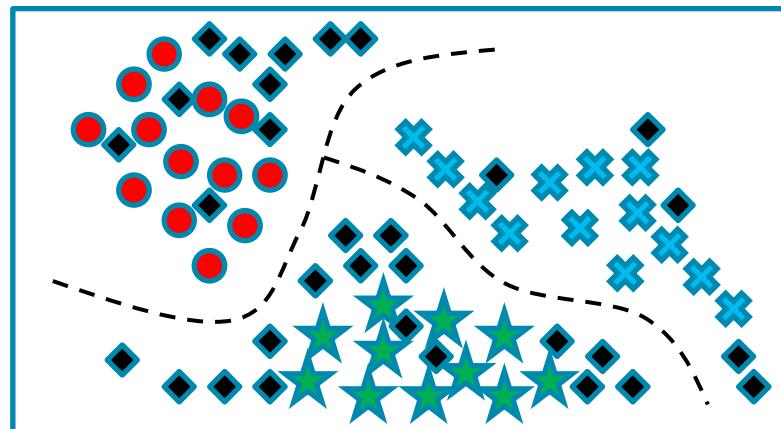
Algorithms



Supervised
learning



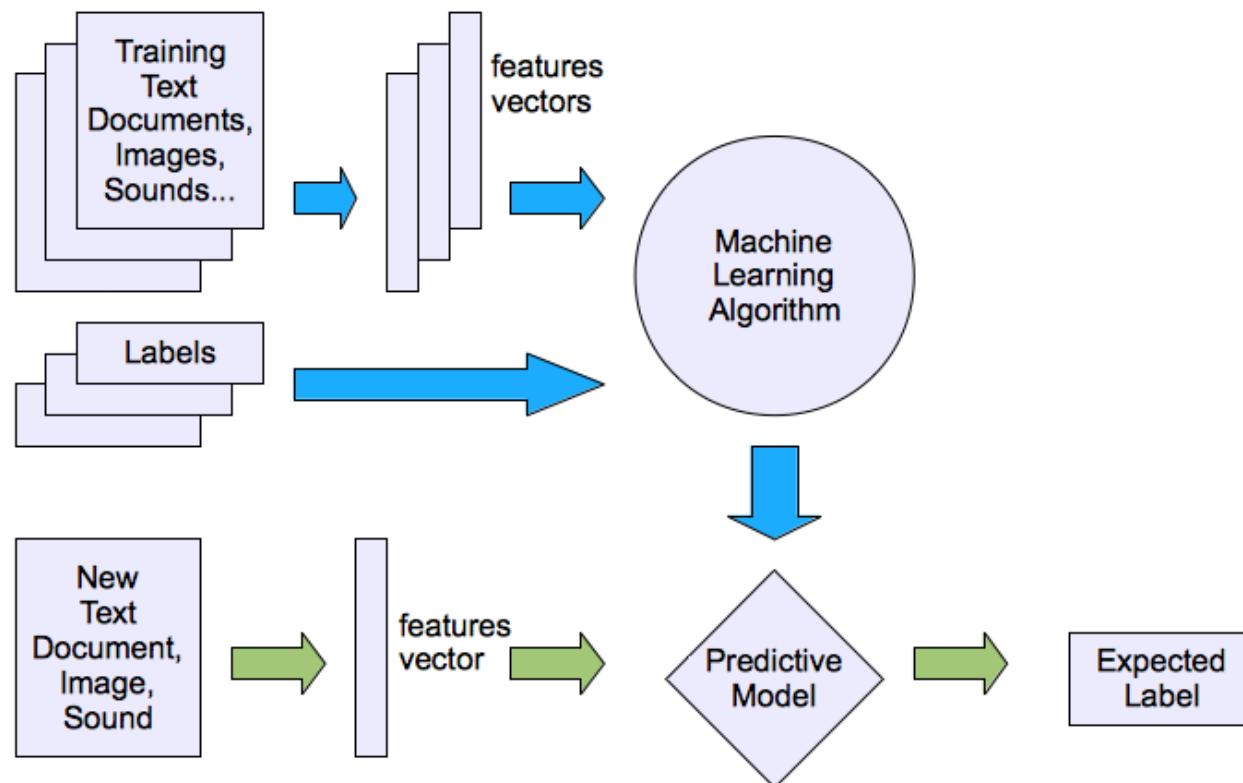
Unsupervised
learning



Semi-supervised
learning

Machine learning structure

- Supervised learning



What are we seeking?

- Supervised: Low E-out or maximize probabilistic terms

$$error = \frac{1}{N} \sum_{n=1}^N [y_n \neq g(x_n)]$$

E_{in} : for training set

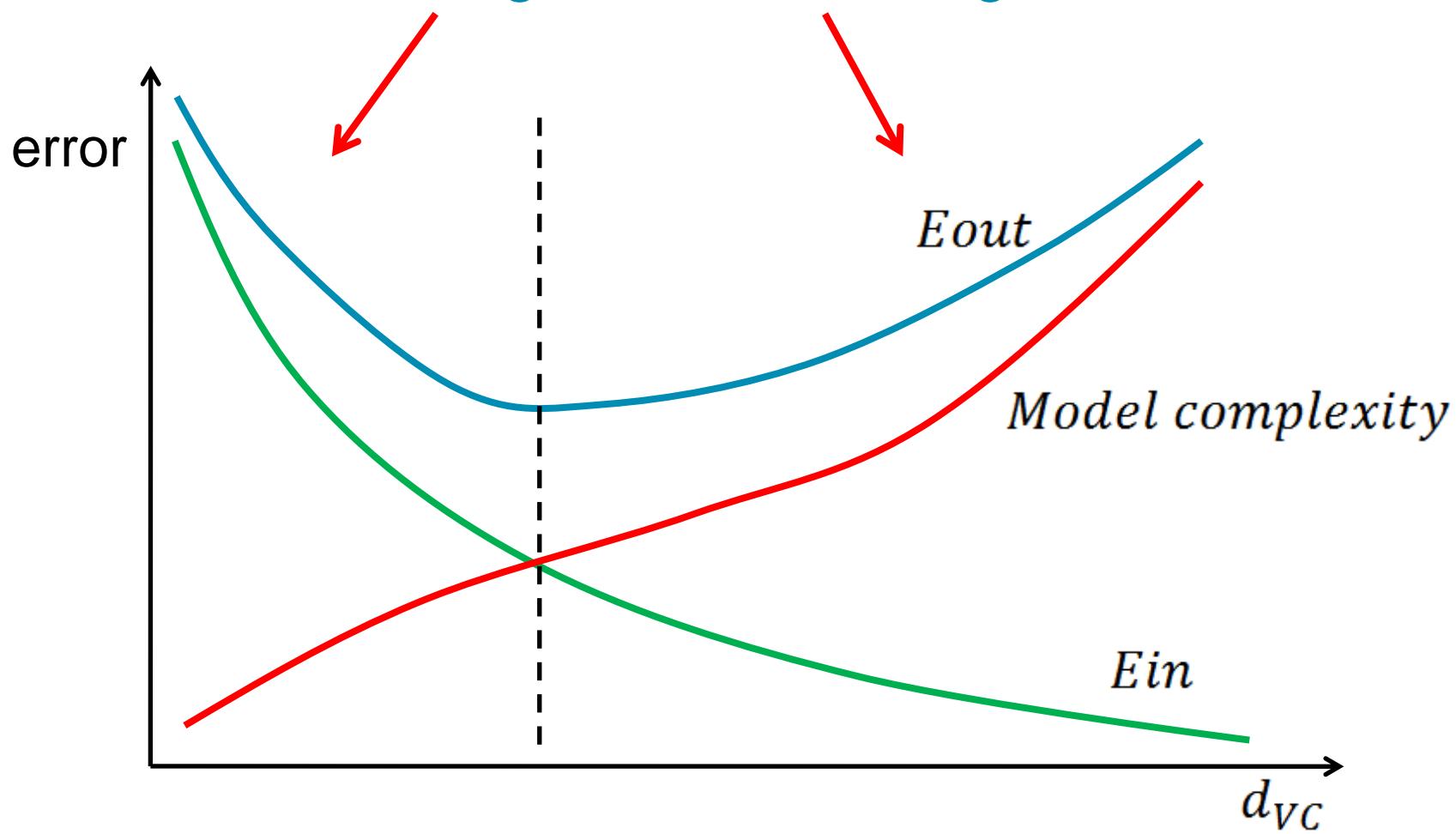
E_{out} : for testing

$$E_{out}(g) \leq E_{in}(g) \pm O\left(\sqrt{\frac{d_{VC}}{N} \ln N}\right)$$

- Unsupervised: Minimum quantization error, Minimum distance, MAP, MLE(maximum likelihood estimation)

What are we seeking?

Under-fitting VS. Over-fitting (fixed N)

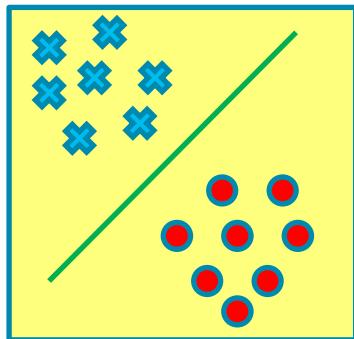


Learning techniques

- Supervised learning categories and techniques
 - **Linear classifier** (numerical functions)
 - **Parametric** (Probabilistic functions)
 - Naïve Bayes, Gaussian discriminant analysis (GDA), Hidden Markov models (HMM), Probabilistic graphical models
 - **Non-parametric** (Instance-based functions)
 - K -nearest neighbors, Kernel regression, Kernel density estimation, Local regression
 - **Non-metric** (Symbolic functions)
 - Classification and regression tree (CART), decision tree
 - **Aggregation**
 - Bagging (bootstrap + aggregation), Adaboost, Random forest

Learning techniques

- Linear classifier



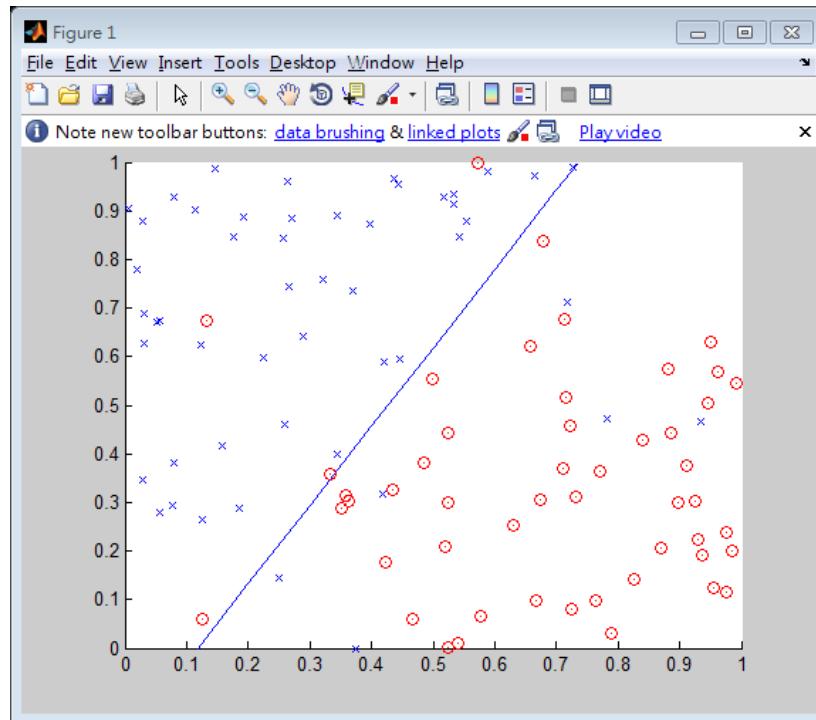
$$g(x_n) = \text{sign}(w^T x_n)$$

, where w is an d -dim vector (learned)

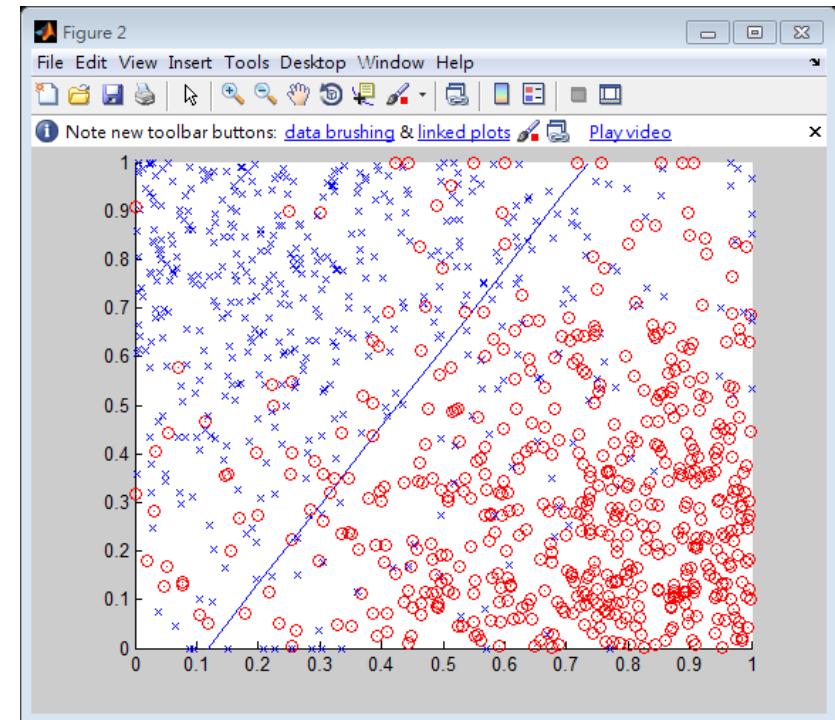
- Techniques:
 - Perceptron
 - Logistic regression
 - Support vector machine (SVM)
 - Ada-line
 - Multi-layer perceptron (MLP)

Learning techniques

Using perceptron learning algorithm(PLA)



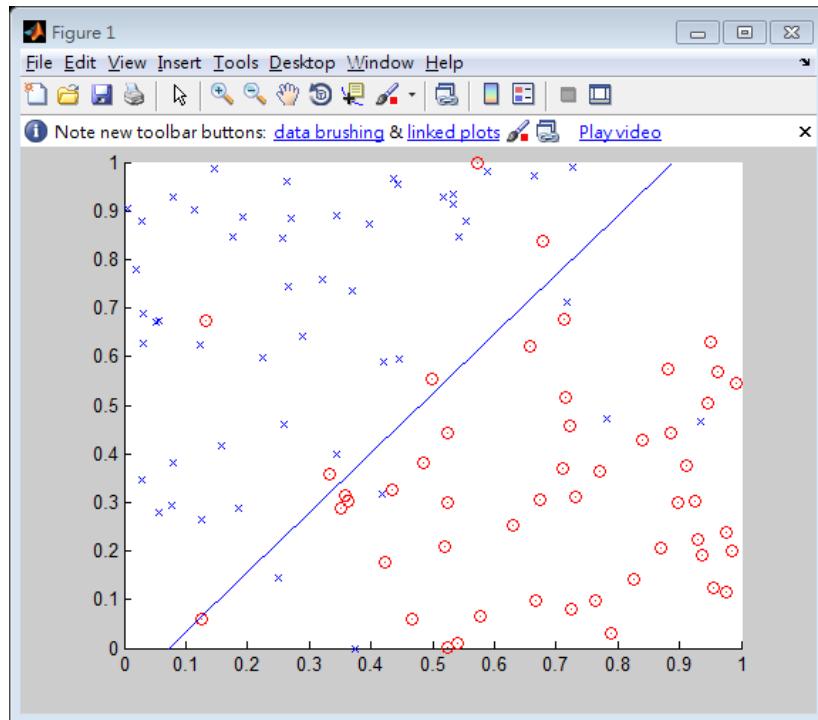
Training
Error rate:
0.10



Testing
Error rate:
0.156

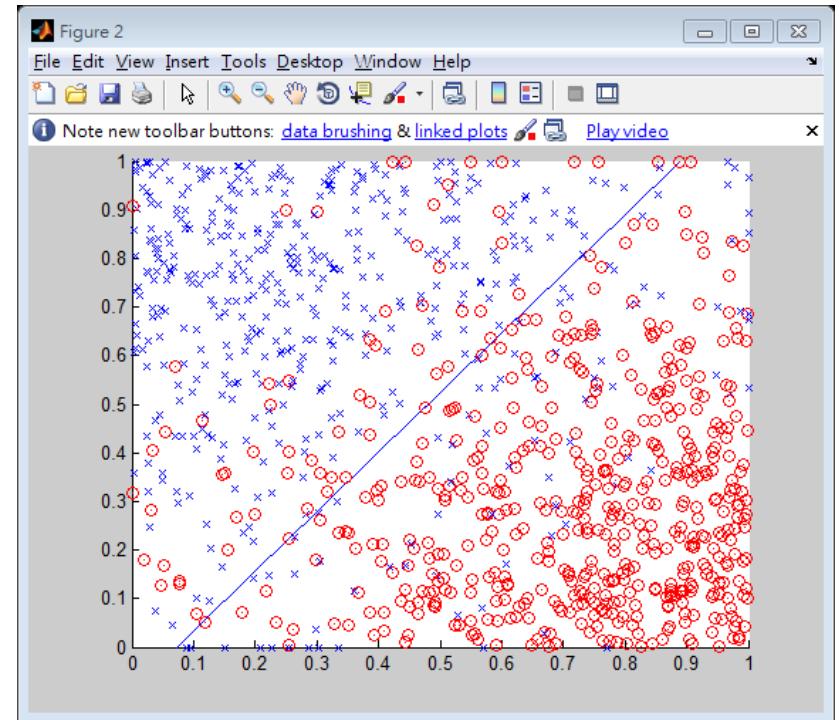
Learning techniques

Using logistic regression



Training

Error rate:
0.11

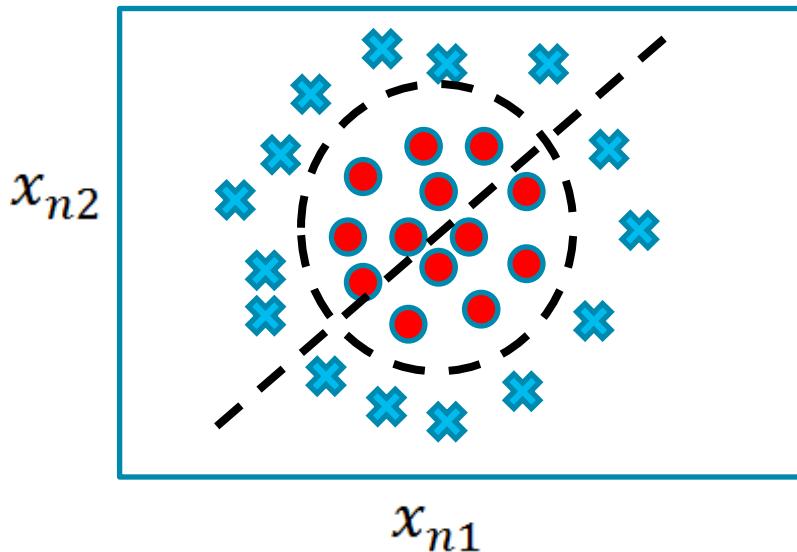


Testing

Error rate:
0.145

Learning techniques

- Non-linear case



$$x_n = [x_{n1}, x_{n2}]$$

↓

$$x_n = [x_{n1}, x_{n2}, x_{n1} * x_{n2}, x_{n1}^2, x_{n2}^2]$$

$$g(x_n) = \text{sign}(w^T x_n)$$

- Support vector machine (SVM):
 - Linear to nonlinear: **Feature transform** and **kernel function**

Learning techniques

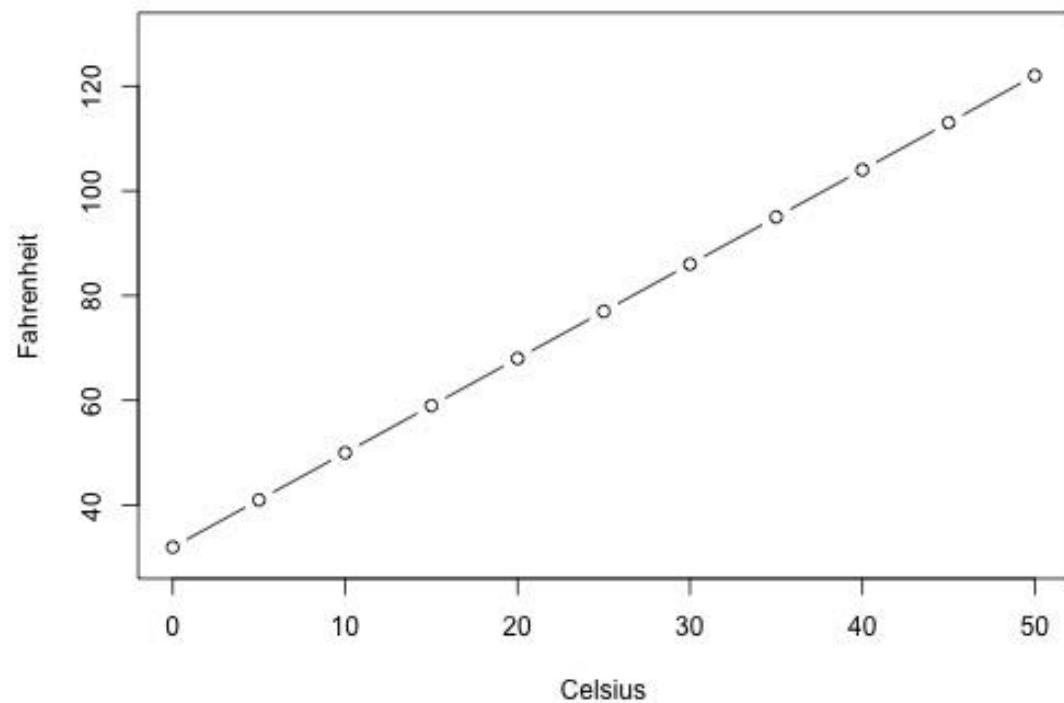
- Unsupervised learning categories and techniques
 - **Clustering**
 - K-means clustering
 - Spectral clustering
 - **Density Estimation**
 - Gaussian mixture model (GMM)
 - Graphical models
 - **Dimensionality reduction**
 - Principal component analysis (PCA)
 - Factor analysis

Supervised Learning

Simple Linear Regression

- **Simple linear regression** is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables:
- One variable, denoted x , is regarded as the **predictor, explanatory, or independent** variable.
- The other variable, denoted y , is regarded as the **response, outcome, or dependent** variable.

Simple Linear Regression



Types of Relationships

- Deterministic
 - Temperature conversion
 - Circumference = $\pi \times$ diameter
 - Hooke's Law: $Y = \alpha + \beta X$, where Y = amount of stretch in a spring, and X = applied weight.
 - Ohm's Law: $I = V/r$, where V = voltage applied, r = resistance, and I = current.
 - Boyle's Law: For a constant temperature, $P = \alpha/V$, where P = pressure, α = constant for each gas, and V = volume of gas.

Types of Relationships

- Statistical
- Height and weight — as height increases, you'd expect weight to increase, but not perfectly.
- Alcohol consumed and blood alcohol content — as alcohol consumption increases, you'd expect one's blood alcohol content to increase, but not perfectly.
- Vital lung capacity and pack-years of smoking — as amount of smoking increases (as quantified by the number of pack-years of smoking), you'd expect lung function (as quantified by vital lung capacity) to decrease, but not perfectly.
- Driving speed and gas mileage — as driving speed increases, you'd expect gas mileage to decrease, but not perfectly.

Real Time Example

- We have a dataset which contains information about relationship between ‘number of hours studied’ and ‘marks obtained’.
- Many students have been observed and their hours of study and grade are recorded.
- This will be our training data. Goal is to design a model that can predict marks if given the number of hours studied.
- Using the training data, a regression line is obtained which will give minimum error.
- This linear equation is then used for any new data.
- That is, if we give number of hours studied by a student as an input, our model should predict their mark with minimum error.

Simple Linear Regression

$$y = b_0 + b_1 * x_1$$

Constant Coefficient

Dependent variable (DV) Independent variable (IV)

Simple Linear Regression

- The values b_0 and b_1 must be chosen so that they minimize the error.
- If sum of squared error is taken as a metric to evaluate the model, then goal to obtain a line that best reduces the error.

$$\text{Error} = \sum_{i=1}^n (\text{actual_output} - \text{predicted_output}) ** 2$$

If we don't square the error, then positive and negative point will cancel out each other.

Simple Linear Regression

- For model with one predictor,

$$b_0 = \bar{y} - b_1 \bar{x}$$

$$b_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

Simple Linear Regression

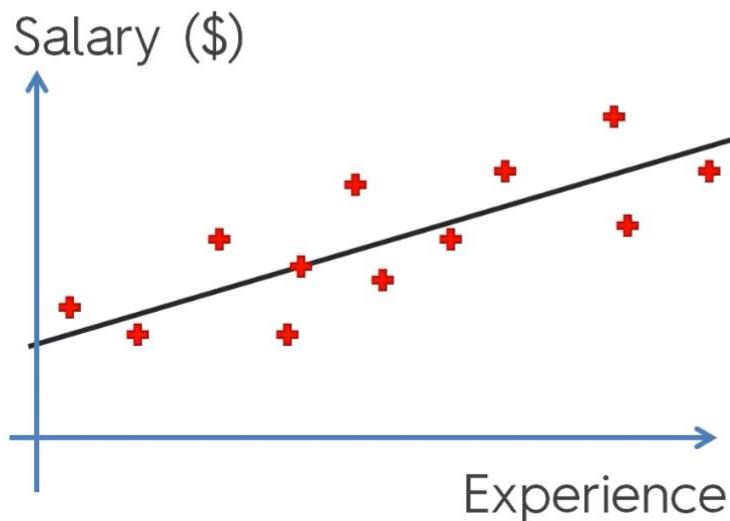
- ***Exploring 'b1'***
- If $b_1 > 0$, then x(predictor) and y(target) have a positive relationship. That is increase in x will increase y.
- If $b_1 < 0$, then x(predictor) and y(target) have a negative relationship. That is increase in x will decrease y.

Simple Linear Regression

- ***Exploring ‘b0’***
- If the model does not include $x=0$, then the prediction will become meaningless with only b_0 . For example, we have a dataset that relates height(x) and weight(y). Taking $x=0$ (that is height as 0), will make equation have only b_0 value which is completely meaningless as in real-time height and weight can never be zero. This resulted due to considering the model values beyond its scope.
- If the model includes value 0, then ‘ b_0 ’ will be the average of all predicted values when $x=0$. But, setting zero for all the predictor variables is often impossible.
- The value of b_0 guarantee that residual have mean zero. If there is no ‘ b_0 ’ term, then regression will be forced to pass over the origin. Both the regression co-efficient and prediction will be biased.

Regressions

Simple Linear Regression:



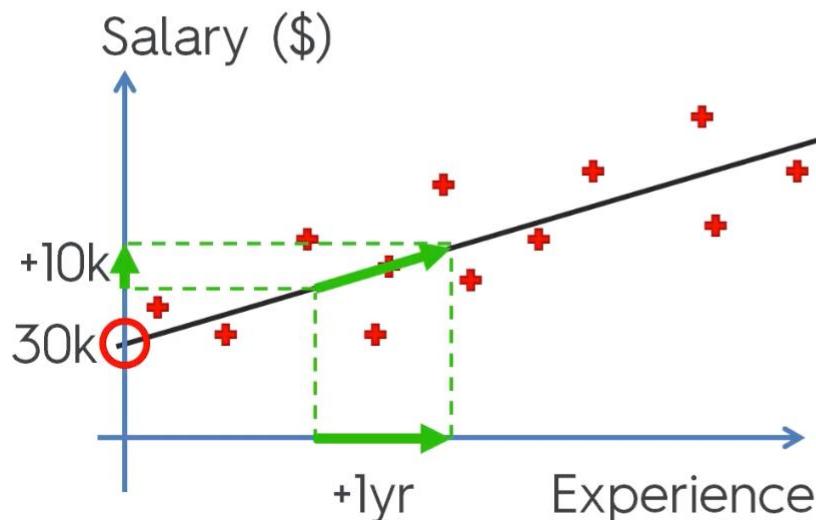
$$y = b_0 + b_1 * x$$



$$\text{Salary} = b_0 + b_1 * \text{Experience}$$

Regressions

Simple Linear Regression:

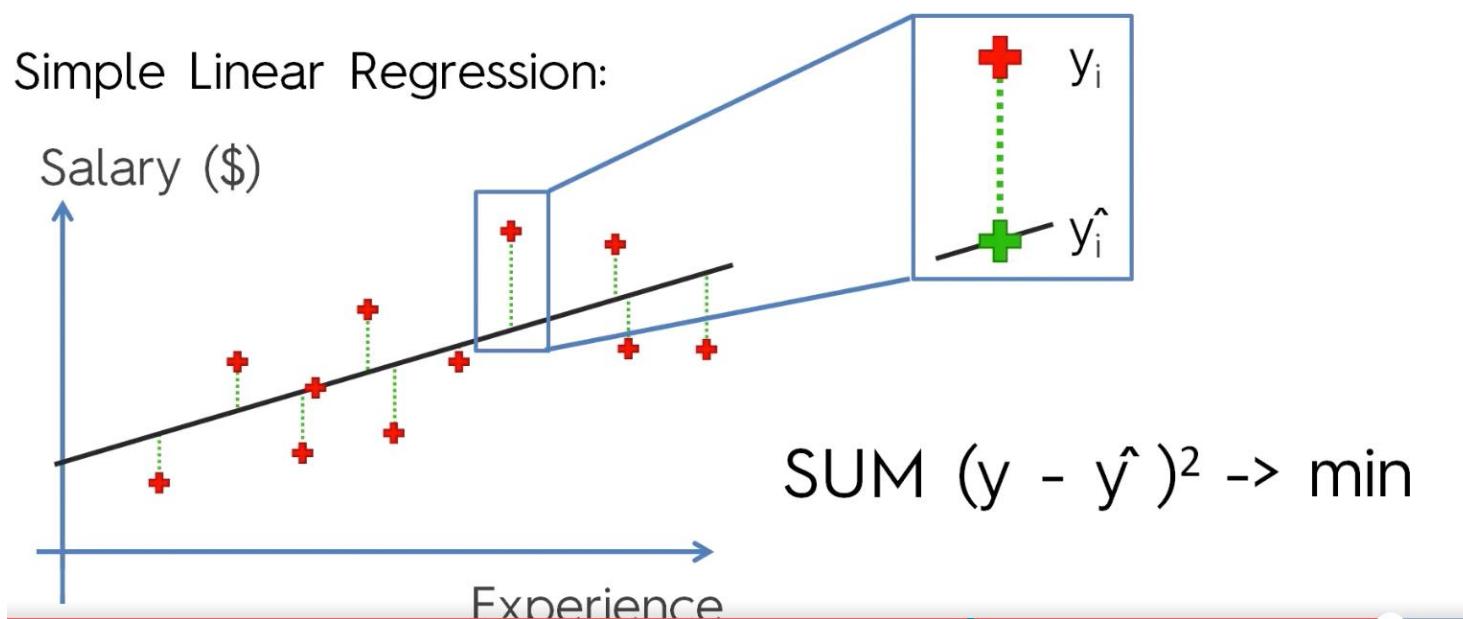


$$y = b_0 + b_1 * x$$

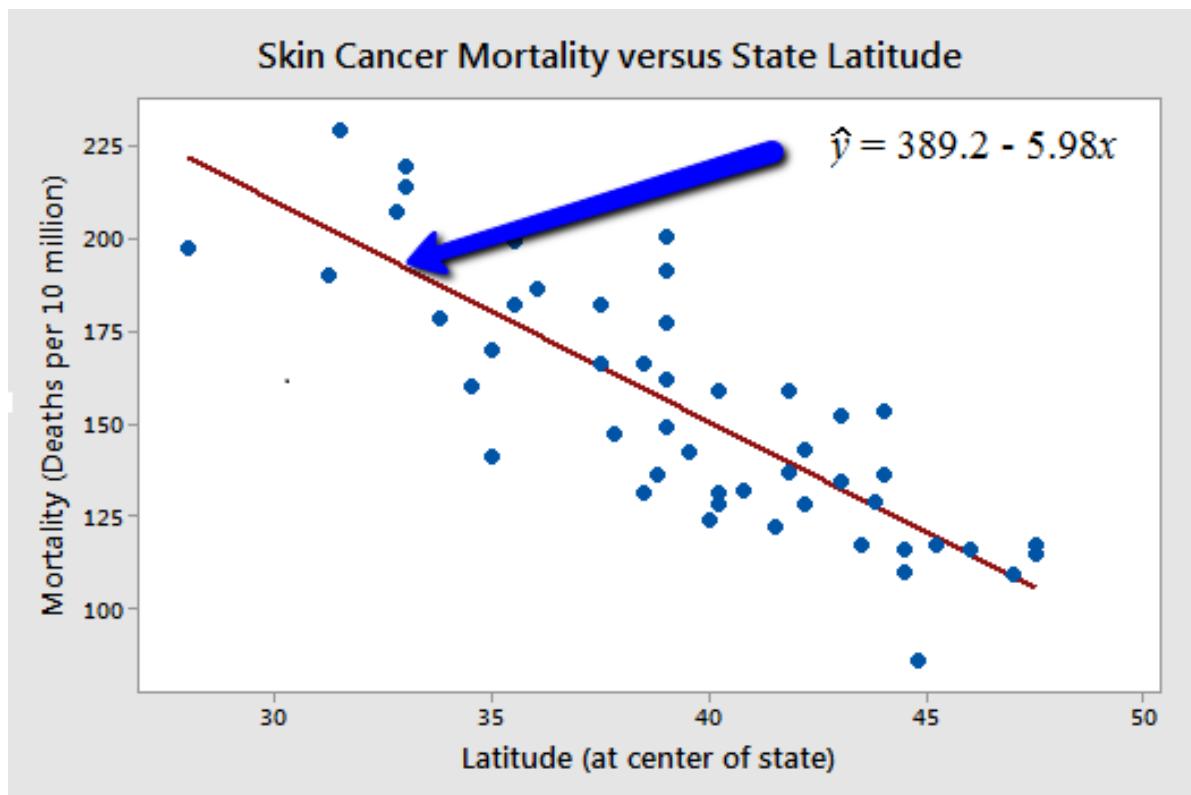

 Salary = b₀ + b₁ * Experience

Ordinary Least Square Method

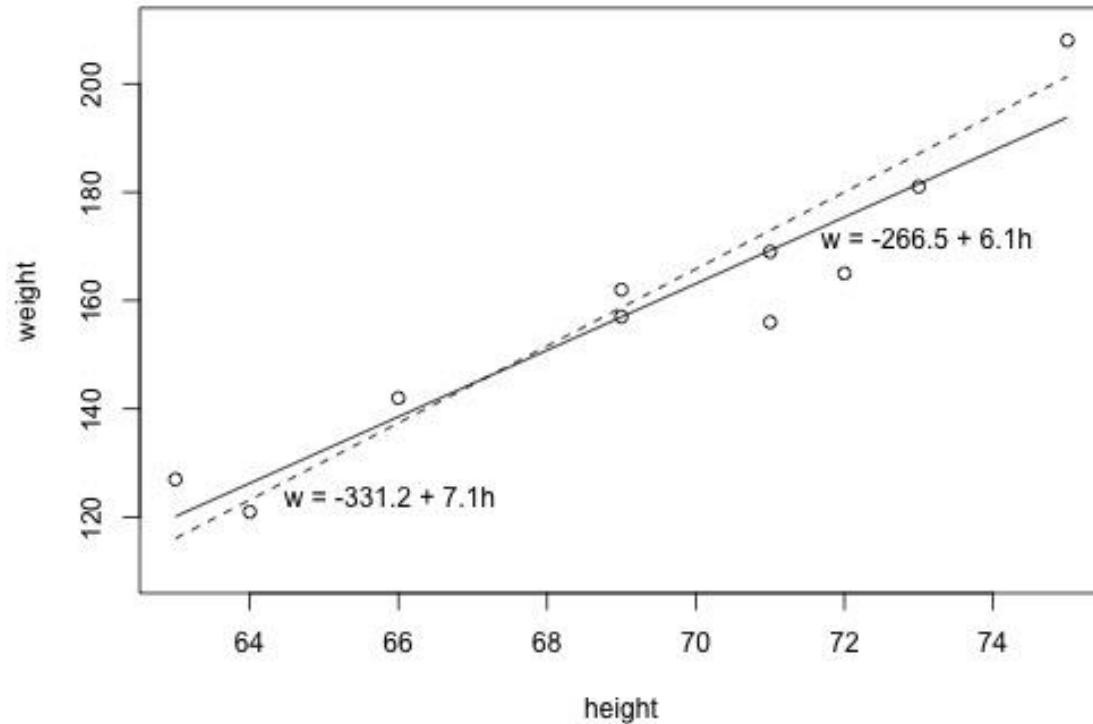
Simple Linear Regression:



Types of Relationships



What is the "Best Fitting Line"?



What is the "Best Fitting Line"?



- y_i denotes the observed response for experimental unit i
- x_i denotes the predictor value for experimental unit i
- \hat{y}_i is the predicted response (or fitted value) for experimental unit i

Then, the equation for the best fitting line is:

| i | x_i | y_i | \hat{y}_i |
|-----|-------|-------|-------------|
| 1 | 63 | 127 | 120.1 |
| 2 | 64 | 121 | 126.3 |
| 3 | 66 | 142 | 138.5 |
| 4 | 69 | 157 | 157.0 |
| 5 | 69 | 162 | 157.0 |
| 6 | 71 | 156 | 169.2 |
| 7 | 71 | 169 | 169.2 |
| 8 | 72 | 165 | 175.4 |
| 9 | 73 | 181 | 181.5 |
| 10 | 75 | 208 | 193.8 |

$$\hat{y}_i = b_0 + b_1 x_i$$

it has some "**prediction error**" (or "**residual error**").

In fact, the size of its prediction error is 127-120.1 or 6.9 pounds.

As you can see, the size of the prediction error depends on the data point.

If we didn't know the weight of student 5, the equation of the line would predict his or her weight to be $-266.53 + 6.1376(69)$ or 157 pounds.

The size of the prediction error here is 162-157, or 5 pounds.



What is the "Best Fitting Line"?

In general, when we use $\hat{y}_i = b_0 + b_1 x_i$ to predict the actual response y_i , we make a prediction error (or residual error) of size:

$$e_i = y_i - \hat{y}_i$$

A line that fits the data "**best**" will be one for which the **n prediction errors** — one for each observed data point — **are as small as possible in some overall sense**. One way to achieve this goal is to invoke the "**least squares criterion**," which says to "minimize the sum of the squared prediction errors." That is:

- The equation of the best fitting line is: $\hat{y}_i = b_0 + b_1 x_i$
- We just need to find the values b_0 and b_1 that make the sum of the squared prediction errors the smallest it can be.
- That is, we need to find the values b_0 and b_1 that minimize:

$$Q = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Here's how you might think about this quantity Q :

- The quantity $e_i = y_i - \hat{y}_i$ is the prediction error for data point i .
- The quantity $e_i^2 = (y_i - \hat{y}_i)^2$ is the squared prediction error for data point i .
- And, the symbol $\sum_{i=1}^n$ tells us to add up the squared prediction errors for all n data points.

What is the "Best Fitting Line"?

| $w = -331.2 + 7.1 h$ (the dashed line) | | | | | |
|--|-------|-------|-------------|---------------------|-----------------------|
| <i>i</i> | x_i | y_i | \hat{y}_i | $(y_i - \hat{y}_i)$ | $(y_i - \hat{y}_i)^2$ |
| 1 | 63 | 127 | 116.1 | 10.9 | 118.81 |
| 2 | 64 | 121 | 123.2 | -2.2 | 4.84 |
| 3 | 66 | 142 | 137.4 | 4.6 | 21.16 |
| 4 | 69 | 157 | 158.7 | -1.7 | 2.89 |
| 5 | 69 | 162 | 158.7 | 3.3 | 10.89 |
| 6 | 71 | 156 | 172.9 | -16.9 | 285.61 |
| 7 | 71 | 169 | 172.9 | -3.9 | 15.21 |
| 8 | 72 | 165 | 180.0 | -15.0 | 225.00 |
| 9 | 73 | 181 | 187.1 | -6.1 | 37.21 |
| 10 | 75 | 208 | 201.3 | 6.7 | 44.89 |
| | | | | | <u>766.5</u> |

| $w = -266.53 + 6.1376 h$ (the solid line) | | | | | |
|---|-------|-------|-------------|---------------------|-----------------------|
| <i>i</i> | x_i | y_i | \hat{y}_i | $(y_i - \hat{y}_i)$ | $(y_i - \hat{y}_i)^2$ |
| 1 | 63 | 127 | 120.139 | 6.8612 | 47.076 |
| 2 | 64 | 121 | 126.276 | -5.2764 | 27.840 |
| 3 | 66 | 142 | 138.552 | 3.4484 | 11.891 |
| 4 | 69 | 157 | 156.964 | 0.0356 | 0.001 |
| 5 | 69 | 162 | 156.964 | 5.0356 | 25.357 |
| 6 | 71 | 156 | 169.240 | -13.2396 | 175.287 |
| 7 | 71 | 169 | 169.240 | -0.2396 | 0.057 |
| 8 | 72 | 165 | 175.377 | -10.3772 | 107.686 |
| 9 | 73 | 181 | 181.515 | -0.5148 | 0.265 |
| 10 | 75 | 208 | 193.790 | 14.2100 | 201.924 |
| | | | | | <u>597.4</u> |

What is the "Best Fitting Line"?

- Based on the least squares criterion, which equation best summarizes the data?
- The sum of the squared prediction errors is 766.5 for the dashed line, while it is only 597.4 for the solid line.
- Therefore, of the two lines, the solid line, $w = -266.53 + 6.1376h$, best summarizes the data.



What is the "Best Fitting Line"?

If we used the above approach for finding the equation of the line that minimizes the sum of the squared prediction errors, we'd have our work cut out for us. We'd have to implement the above procedure for an infinite number of possible lines — clearly, an impossible task! Fortunately, somebody has done some dirty work for us by figuring out formulas for the **intercept** b_0 and the **slope** b_1 for the equation of the line that minimizes the sum of the squared prediction errors.

The formulas are determined using methods of calculus. We minimize the equation for the sum of the squared prediction errors:

$$Q = \sum_{i=1}^n (y_i - (b_0 + b_1 x_i))^2$$

(that is, take the derivative with respect to b_0 and b_1 , set to 0, and solve for b_0 and b_1) and get the "**least squares estimates**" for b_0 and b_1 :

$$b_0 = \bar{y} - b_1 \bar{x}$$

and:

$$b_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

Because the formulas for b_0 and b_1 are derived using the least squares criterion, the resulting equation — $\hat{y}_i = b_0 + b_1 x_i$ — is often referred to as the "**least squares regression line**," or simply the "**least squares line**." It is also sometimes called the "**estimated regression equation**." Incidentally, note that in deriving the above formulas, we made no assumptions about the data other than that they follow some sort of linear trend.

We can see from these formulas that the least squares line passes through the point (\bar{x}, \bar{y}) , since when $x = \bar{x}$, then $y = b_0 + b_1 \bar{x} = \bar{y} - b_1 \bar{x} + b_1 \bar{x} = \bar{y}$.

In practice, you won't really need to worry about the formulas for b_0 and b_1 . Instead, you are going to let statistical software, such as Minitab, find least squares lines for you. But, we can still learn something from the formulas — for b_1 in particular.

Slope and Intercept calculation

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$
$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

How to Find a Linear Regression Equation: Steps



Step 1: Make a chart of your data, filling in the columns in the same way as you would fill in the chart if you were finding the Pearson's Correlation Coefficient.

| SUBJECT | AGE X | GLUCOSE LEVEL Y | XY | X ² | Y ² |
|----------|-------|-----------------|-------|----------------|----------------|
| 1 | 43 | 99 | 4257 | 1849 | 9801 |
| 2 | 21 | 65 | 1365 | 441 | 4225 |
| 3 | 25 | 79 | 1975 | 625 | 6241 |
| 4 | 42 | 75 | 3150 | 1764 | 5625 |
| 5 | 57 | 87 | 4959 | 3249 | 7569 |
| 6 | 59 | 81 | 4779 | 3481 | 6561 |
| Σ | 247 | 486 | 20485 | 11409 | 40022 |

From the above table, $\Sigma x = 247$, $\Sigma y = 486$, $\Sigma xy = 20485$, $\Sigma x^2 = 11409$, $\Sigma y^2 = 40022$. n is the sample size (6, in our case).

How to Find a Linear Regression Equation: Steps



- **Step 2:** Use the following equations to find a and b.

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$\begin{aligned} a &= 65.1416 \\ b &= .385225 \end{aligned}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

How to Find a Linear Regression Equation: Steps



- **Step 2:** Use the following equations to find a and b.

Find a:

$$((486 \times 11,409) - ((247 \times 20,485)) / 6 (11,409) - 247^2)$$

$$484979 / 7445$$

$$=65.14$$

Find b:

$$(6(20,485) - (247 \times 486)) / (6 (11409) - 247^2)$$

$$(122,910 - 120,042) / 68,454 - 247^2$$

$$2,868 / 7,445$$

$$= .385225$$

How to Find a Linear Regression Equation: Steps

- **Step 3:** Insert the values into the equation.

$$y' = a + bx$$

$$y' = 65.14 + .385225x$$

- That's how to find a linear regression equation by hand!

Interpolation vs Regression

- In the mathematical field of numerical analysis, **interpolation** is a method of constructing new data points within the range of a discrete set of known data points.
- Interpolation is the process of deriving a simple function from a set of discrete data points so that the function passes through all the given data points (i.e. reproduces the data points exactly) and can be used to estimate data points in-between the given ones.

Interpolation vs Regression

- Newton method
- Regression is different from interpolation in that it allows us to approximate overdetermined system, which has more equations than unknowns.
- This is useful when the exact solution is too expensive or unnecessary due to errors in the data, such as measurement errors or random noise.

Interpolation vs Regression

- In interpolation you are given some data points, and you are supposed to find a curve which fits the input/output relationship perfectly. In case of interpolation, you don't have to worry about variance of the fitted curve. As long as your curve is giving perfect output(for a given dataset), you are done.
- In regression, your curve should not be over-fitting. It should give you a regularized model so that you can predict future values. Variance of your function should be as low as possible. Your model should be generalized so that you can accurately predict future values.

Interpolation vs Regression

- Interpolation is like an algorithm without a "brain": it tries to achieve perfect match to the given data
- Regression is the same algorithm with the power to **generalize**. It won't fit perfectly to your data but at least it will try to learn some insights from it.
-

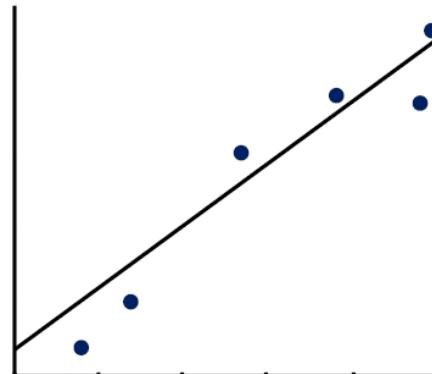
Example: Regression

- Given the following data:

| | | | | | | |
|-----|------|------|------|------|------|------|
| x | 0.8 | 1.4 | 2.7 | 3.8 | 4.8 | 4.9 |
| y | 0.69 | 1.00 | 2.02 | 2.39 | 2.34 | 2.83 |

Regression:

Obtain a straight line that
best fits the data



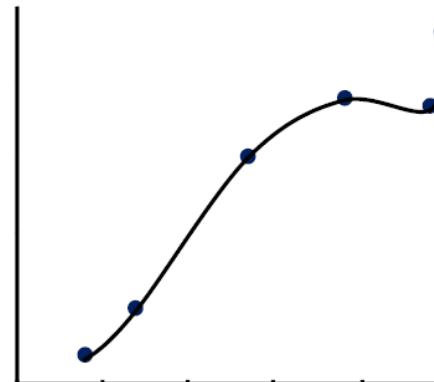
Example: Interpolation

- Given the following data:

| | | | | | | |
|-----|------|------|------|------|------|------|
| x | 0.8 | 1.4 | 2.7 | 3.8 | 4.8 | 4.9 |
| y | 0.69 | 1.00 | 2.02 | 2.39 | 2.34 | 2.83 |

Interpolation:

“Join the dots” and
find a curve passing
through the data.



Regression vs. Interpolation

- Given the following data:

| | | | | | | |
|-----|------|------|------|------|------|------|
| x | 0.8 | 1.4 | 2.7 | 3.8 | 4.8 | 4.9 |
| y | 0.69 | 1.00 | 2.02 | 2.39 | 2.34 | 2.83 |

- In **regression**, we are interested in fitting a chosen function to data

$$y = 0.45 + 0.47x$$

- In **interpolation**, given finite amount of data, we are interested in obtaining new data-points within this range.

At $x = 2.0, y = 1.87$

- **Linear Regression:** Fit a straight line to the given data
- **Newton's Interpolation:** For values at intermediate points

Example: Temperature variation in a day

| time | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| T | 25.6 | 25.4 | 25.1 | 24.9 | 24.9 | 25.2 | 25.9 | 26.3 | 27.1 | 29.3 | 30.8 | 31.2 | 32.1 |
| time | | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| T | | 31.0 | 30.3 | 31.4 | 30.6 | 31.8 | 29.6 | 28.4 | 28.1 | 28.2 | 27.4 | 26.8 | 26.1 |

We are interested in finding temperature at various times during the day, in addition to the ones where data is available.

We *interpolate* or “fill in” the missing data



Regressions

Multiple Linear Regression

Simple
Linear
Regression

$$y = b_0 + b_1 * x_1$$

Multiple
Linear
Regression

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n * x_n$$

Dependent variable (DV)

Independent variables (IVs)

Constant

Coefficients

```
graph TD; DV[Dependent variable (DV)] --> y; IVs[Independent variables (IVs)] --> terms["+ b1*x1 + b2*x2 + ... + bn*xn"]; Constant[Constant] --> b0[b0]; Coefficients[Coefficients] --> terms;
```



Dummy Variables

| Profit | R&D Spend | Admin | Marketing | State |
|------------|------------|------------|------------|------------|
| 192,261.83 | 165,349.20 | 136,897.80 | 471,784.10 | New York |
| 191,792.06 | 162,597.70 | 151,377.59 | 443,898.53 | California |
| 191,050.39 | 153,441.51 | 101,145.55 | 407,934.54 | California |
| 182,901.99 | 144,372.41 | 118,671.85 | 383,199.62 | New York |
| 166,187.94 | 142,107.34 | 91,391.77 | 366,168.42 | California |

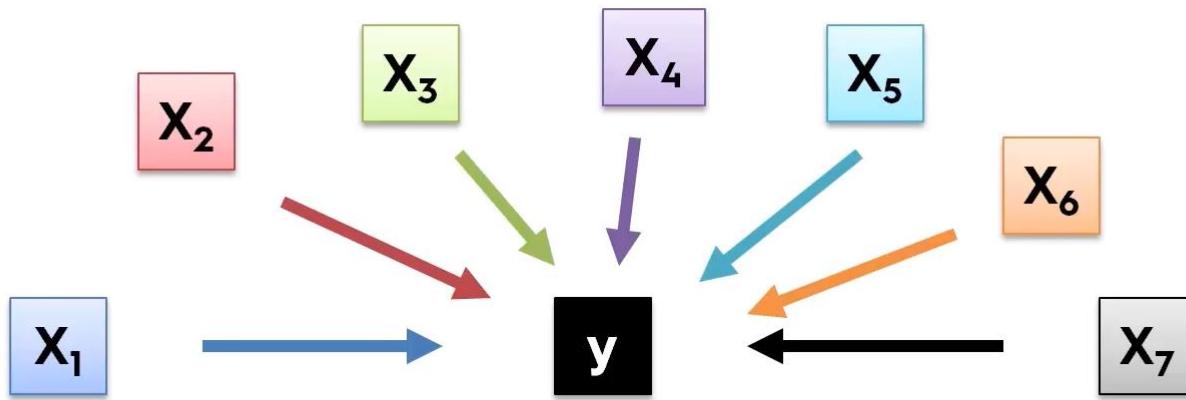
Dummy Variables

| New York | California |
|----------|------------|
| 1 | 0 |
| 0 | 1 |
| 0 | 1 |
| 1 | 0 |
| 0 | 1 |

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + b_3 * x_3 + b_4 * D_1$$

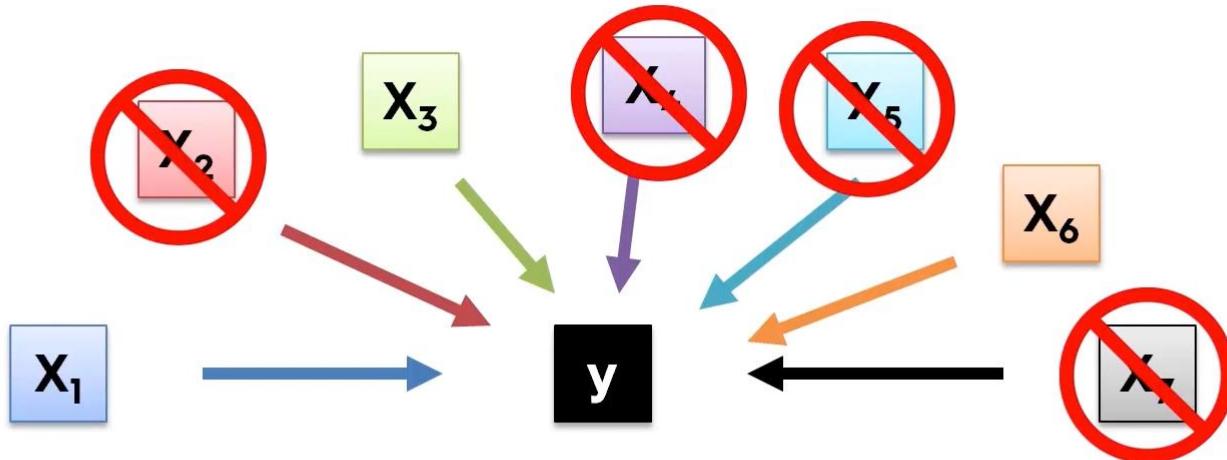


Building Model





Building A Model

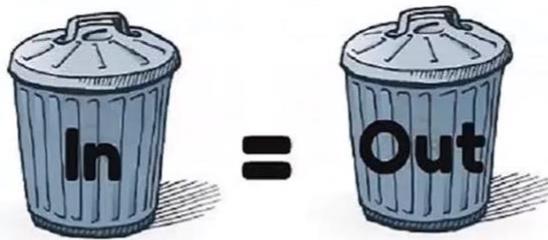


Why?



DATA SCIENCE

1)



2)

Building A Model

5 methods of building models:

1. All-in
 2. Backward Elimination
 3. Forward Selection
 4. Bidirectional Elimination
 5. Score Comparison
- 

Stepwise
Regression



Building A Model

Press Esc to exit full screen

Backward Elimination

STEP 1: Select a significance level to stay in the model (e.g. SL = 0.05)



STEP 2: Fit the full model with all possible predictors



STEP 3: Consider the predictor with the highest P-value. If $P > SL$, go to STEP 4, otherwise go to FIN



STEP 4: Remove the predictor



STEP 5: Fit model without this variable*



FIN: Your Model Is Ready

Building A Model

Forward Selection

STEP 1: Select a significance level to enter the model (e.g. SL = 0.05)



STEP 2: Fit all simple regression models $y \sim x_n$. Select the one with the lowest P-value



STEP 3: Keep this variable and fit all possible models with one extra predictor added to the one(s) you already have



STEP 4: Consider the predictor with the lowest P-value. If $P < SL$, go to STEP 3, otherwise go to FIN

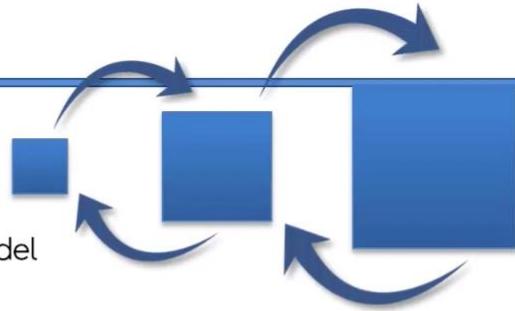


FIN: Keep the previous model

Building A Model

Bidirectional Elimination

STEP 1: Select a significance level to enter and to stay in the model
e.g.: SLENTER = 0.05, SLSTAY = 0.05



STEP 2: Perform the next step of Forward Selection (new variables must have: $P < \text{SLENTER}$ to enter)

STEP 3: Perform ALL steps of Backward Elimination (old variables must have $P < \text{SLSTAY}$ to stay)

STEP 4: No new variables can enter and no old variables can exit



FIN: Your Model Is Ready

Building A Model

All Possible Models

STEP 1: Select a criterion of goodness of fit (e.g. Akaike criterion)



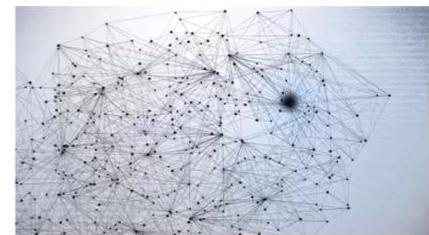
STEP 2: Construct All Possible Regression Models: $2^N - 1$ total combinations



STEP 3: Select the one with the best criterion



FIN: Your Model Is Ready



Example:
**10 columns means
1,023 models**



Building A Model

5 methods of building models:

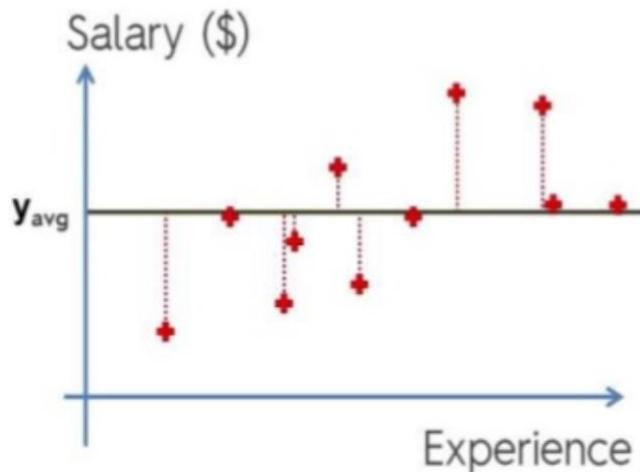
1. All-in
2. Backward Elimination
3. Forward Selection
4. Bidirectional Elimination
5. Score Comparison

```
import statsmodels.formula.api as sm
def backwardElimination(x, sl):
    numVars = len(x[0])
    for i in range(0, numVars):
        regressor_OLS = sm.OLS(y, x).fit()
        maxVar = max(regressor_OLS.pvalues).astype(float)
        if maxVar > sl:
            for j in range(0, numVars - i):
                if (regressor_OLS.pvalues[j].astype(float) == maxVar):
                    x = np.delete(x, j, 1)
            regressor_OLS.summary()
    return x

SL = 0.05
X_opt = X[:, [0, 1, 2, 3, 4, 5]]
X_Modeled = backwardElimination(X_opt, SL)
```

R SQUARED INTUITION

Simple Linear Regression:



$$SS_{res} = \text{SUM } (y_i - \hat{y}_i)^2$$

$$SS_{tot} = \text{SUM } (y_i - y_{avg})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

- Goal : find a line with SSres as low as possible
- Value of R^2 **generally** lies between 0 and 1
- The more closer the R^2 is to 1 the better our line is.
- Why we square ? To deal with positive values and to deal with outliers , however we can use power of 4 or 6 or etc. but square is convention and that is widely accepted hence we will use that for now.
- R^2 can take negative values (if our data fits worstly to our linear regression)

PROBLEMS WITH R²

As number of independent values increase the value of R² will either increase but will never decrease.

Hence we will not know how good of an influence does this newly added independent variable has on our dependent variable.

Reason for this is that any independent variable has tendency of slightly correlation with the dependent variable. This might help reducing the SSres value hence the value of R² increases.

To overcome this we use adjusted R²

Adjusted R²

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

R² - Goodness of fit
(greater is better)

$$y = b_0 + b_1 * x_1$$

Problem:

$$y = b_0 + b_1 * x_1 + b_2 * x_2$$

$$+ b_3 * x_3$$

SS_{res} → Min

R² will never decrease

ADJUSTED R²

Adjusted R² deals with additional independent variables.

This r squared value tends to penalize the value of the rsquare if our choice of independent variable wasn't good (i.e independent variable had no effect on dependent variable)

Also the bias of R SQUARE to not to decrease is handled pretty well in this adjusted R SQUARED method.

DIFFERENCE

The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases only if the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected by chance. The adjusted R-squared can be negative, but it's usually not. It is always lower than the R-squared.

R-squared or R² explains the degree to which your input variables explain the variation of your output / predicted variable. So, if R-square is 0.8, it means 80% of the variation in the output variable is explained by the input variables. So, in simple terms, higher the R squared, the more variation is explained by your input variables and hence better is your model.

However, the problem with R-squared is that it will either stay the same or increase with addition of more variables, even if they do not have any relationship with the output variables. This is where “Adjusted R square” comes to help. Adjusted R-square penalizes you for adding variables which do not improve your existing model.

Hence, if you are building Linear regression on multiple variable, it is always suggested that you use Adjusted R-squared to judge goodness of model. In case you only have one input variable, R-square and Adjusted R squared would be exactly same.

Typically, the more non-significant variables you add into the model, the gap in R-squared and Adjusted R-squared increases.

Ordinary Least Square Method

```
model1=sm.OLS(y_train,x_train)
```

```
In [221]: result=model1.fit()
```

```
In [222]: print(result.summary())
```

```
OLS Regression Results
=====
Dep. Variable:                      0   R-squared:                   0.959
Model:                            OLS   Adj. R-squared:             0.958
Method:                           Least Squares   F-statistic:                 711.8
Date:                Sun, 16 Apr 2017   Prob (F-statistic):        8.37e-263
Time:                    21:23:08   Log-Likelihood:            -1210.8
No. Observations:                  404   AIC:                         2448.
Df Residuals:                      391   BIC:                         2500.
Df Model:                           13
Covariance Type:            nonrobust
=====
```

| | coef | std err | t | P> t | [95.0% Conf. Int.] |
|----------|---------|----------------|--------|-------|--------------------|
| CRIM | -0.1077 | 0.039 | -2.779 | 0.006 | -0.184 -0.031 |
| ZN | 0.0484 | 0.016 | 2.952 | 0.003 | 0.016 0.081 |
| INDUS | -0.0232 | 0.073 | -0.317 | 0.751 | -0.167 0.121 |
| CHAS | 2.9930 | 1.062 | 2.819 | 0.005 | 0.906 5.080 |
| NOX | -2.1626 | 3.662 | -0.591 | 0.555 | -9.362 5.036 |
| RM | 5.9590 | 0.339 | 17.584 | 0.000 | 5.293 6.625 |
| AGE | -0.0169 | 0.015 | -1.094 | 0.274 | -0.047 0.013 |
| DIS | -1.0273 | 0.220 | -4.661 | 0.000 | -1.461 -0.594 |
| RAD | 0.1669 | 0.075 | 2.240 | 0.026 | 0.020 0.313 |
| TAX | -0.0105 | 0.004 | -2.368 | 0.018 | -0.019 -0.002 |
| PTRATIO | -0.3753 | 0.124 | -3.018 | 0.003 | -0.620 -0.131 |
| B | 0.0143 | 0.003 | 4.733 | 0.000 | 0.008 0.020 |
| LSTAT | -0.3463 | 0.057 | -6.129 | 0.000 | -0.457 -0.235 |
| Omnibus: | 151.837 | Durbin-Watson: | | | 1.804 |

We can drop few variables and select only those that have p values < 0.5 and then we can check improvement in the model.

A general approach to compare two different models is AIC(Akaike Information Criteria) and the model with minimum AIC is the best one.

```
model2=sm.OLS(y_train,x_train[['CRIM','ZN','CHAS','RM','DIS','RAD','TAX','PTRATIO','B','LSTAT']])
result2=model2.fit()
print(result2.summary())
```

```
OLS Regression Results
=====
Dep. Variable:                      O            R-squared:                   0.959
Model:                            OLS           Adj. R-squared:             0.958
Method:                           Least Squares   F-statistic:                 926.5
Date:                Sun, 16 Apr 2017   Prob (F-statistic):        1.08e-266
Time:                    20:40:51         Log-Likelihood:            -1212.1
No. Observations:                  404          AIC:                     2444.
Df Residuals:                      394          BIC:                     2484.
Df Model:                           10
Covariance Type:                nonrobust

=====
              coef    std err      t      P>|t|  [95.0% Conf. Int.]
-----
CRIM     -0.1040    0.039   -2.695    0.007    -0.180   -0.028
ZN        0.0521    0.016    3.229    0.001     0.020    0.084
CHAS      2.7772    1.046    2.655    0.008     0.721    4.834
RM        5.7093    0.271   21.103    0.000     5.177    6.241
DIS       -0.8541    0.188   -4.542    0.000    -1.224   -0.484
RAD        0.1845    0.071    2.607    0.009     0.045    0.324
TAX       -0.0125    0.004   -3.412    0.001    -0.020   -0.005
PTRATIO   -0.3939    0.123   -3.197    0.002    -0.636   -0.152
B          0.0138    0.003    4.640    0.000     0.008    0.020
LSTAT     -0.3920    0.048   -8.168    0.000    -0.486   -0.298
=====
```

OLS Summary Interpretation

- Omnibus/Prob(Omnibus) – a test of the skewness and kurtosis of the residual (characteristic #2). We hope to see a value close to zero which would indicate normalcy. The Prob (Omnibus) performs a statistical test indicating the probability that the residuals are normally distributed. We hope to see something close to 1 here. In this case Omnibus is relatively low and the Prob (Omnibus) is relatively high so the data is somewhat normal, but not altogether ideal. A linear regression approach would probably be better than random guessing but likely not as good as a nonlinear approach.
- Skew – a measure of data symmetry. We want to see something close to zero, indicating the residual distribution is normal. Note that this value also drives the Omnibus. This result has a small, and therefore good, skew.
- Kurtosis – a measure of "peakiness", or curvature of the data. Higher peaks lead to greater Kurtosis. Greater Kurtosis can be interpreted as a tighter clustering of residuals around zero, implying a better model with few outliers.

OLS Summary Interpretation

- Durbin-Watson – tests for homoscedasticity (characteristic #3). We hope to have a value between 1 and 2. In this case, the data is close, but within limits.
- Jarque-Bera (JB)/Prob(JB) – like the Omnibus test in that it tests both skew and kurtosis. We hope to see in this test a confirmation of the Omnibus test. In this case we do.
- Condition Number – This test measures the sensitivity of a function's output as compared to its input (characteristic #4). When we have multicollinearity, we can expect much higher fluctuations to small changes in the data, hence, we hope to see a relatively small number, something below 30. In this case we are well below 30, which we would expect given our model only has two variables and one is a constant.

OLS Summary Interpretation

- The AIC or BIC for a model is usually written in the form $[-2\log L + kp]$, where L is the likelihood function, p is the number of parameters in the model, and k is 2 for AIC and $\log(n)$ for BIC.
- AIC is an estimate of a constant plus the relative distance between the unknown true likelihood function of the data and the fitted likelihood function of the model, so that a lower AIC means a model is considered to be closer to the truth.
- BIC is an estimate of a function of the posterior probability of a model being true, under a certain Bayesian setup, so that a lower BIC means that a model is considered to be more likely to be the true model.

Regressions

Simple
Linear
Regression

$$y = b_0 + b_1 x_1$$

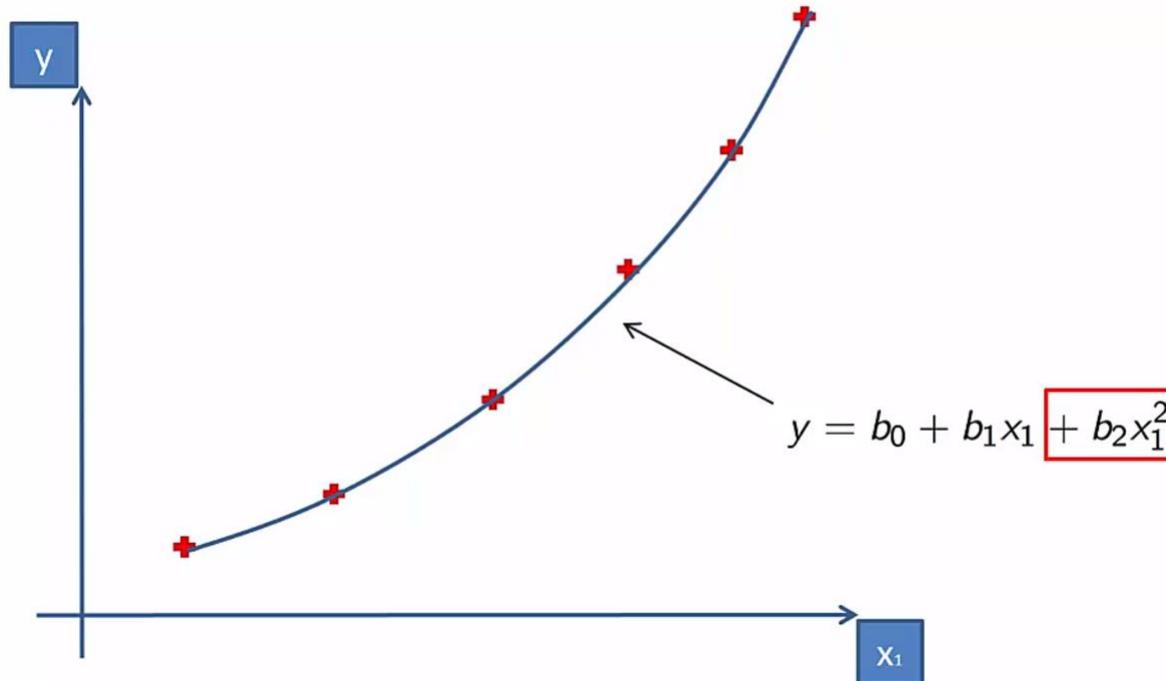
Multiple
Linear
Regression

$$y = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_n x_n$$

Polynomial
Linear
Regression

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \dots + b_n x_1^n$$

Polynomial Regression



Support Vector Regression

- **SVM is a *supervised* learning model**
- Problem:
- I have a business and I receive a lot of emails from customers every day. Some of these emails are complaints and should be answered very quickly. I would like a way to identify them quickly so that I answer these email in priority.

Support Vector Regression

- Approach: I can use a supervised machine learning algorithm.
- Step 1: I need a lot of emails, the more the better.
Step 2: I will read the title of each email and classify it by saying "it is a complaint" or "it is not a complaint". It put a **label** on each email.
Step 3: I will **train** a model on this dataset
Step 4: I will assess the quality of the prediction (using cross validation)
Step 5: I will use this model to **predict** if an email is a complaint or not.
- In this case, if I have trained the model with a lot of emails then it will perform well. SVM is just one among many models you can use to learn from this data and make predictions.

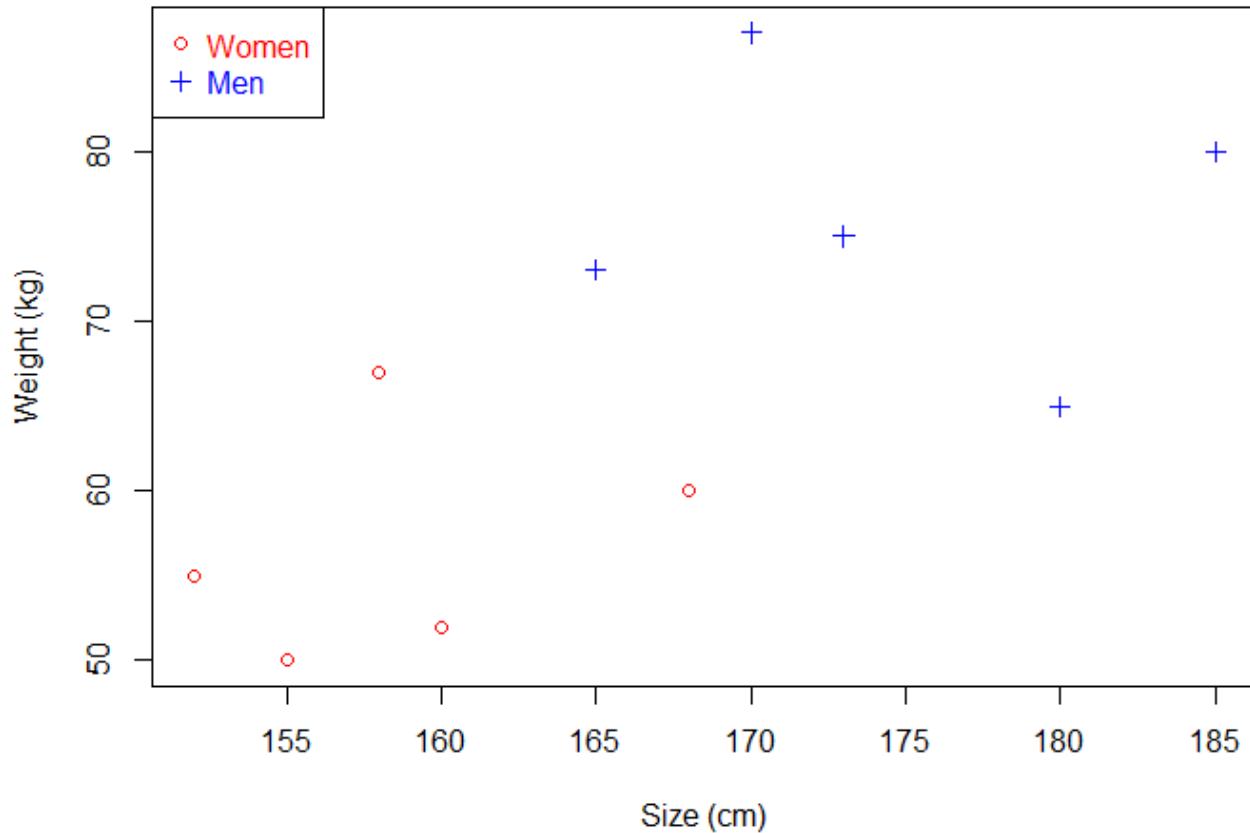
Support Vector Regression

- **SVMs - Support Vector Machines**
- Wikipedia tells us that SVMs can be used to do two things: classification or regression.
- **SVM** is used for classification
- **SVR (Support Vector Regression)** is used for regression

What is the goal of the Support Vector Machine (SVM)?



- The goal of a support vector machine is to find the optimal separating hyperplane which maximizes the margin of the training data.
- The first thing we can see from this definition, is that a SVM needs training data. Which means it is a supervised learning algorithm.
- It is also important to know that SVM is a classification algorithm. Which means we will use it to predict if something belongs to a particular class.



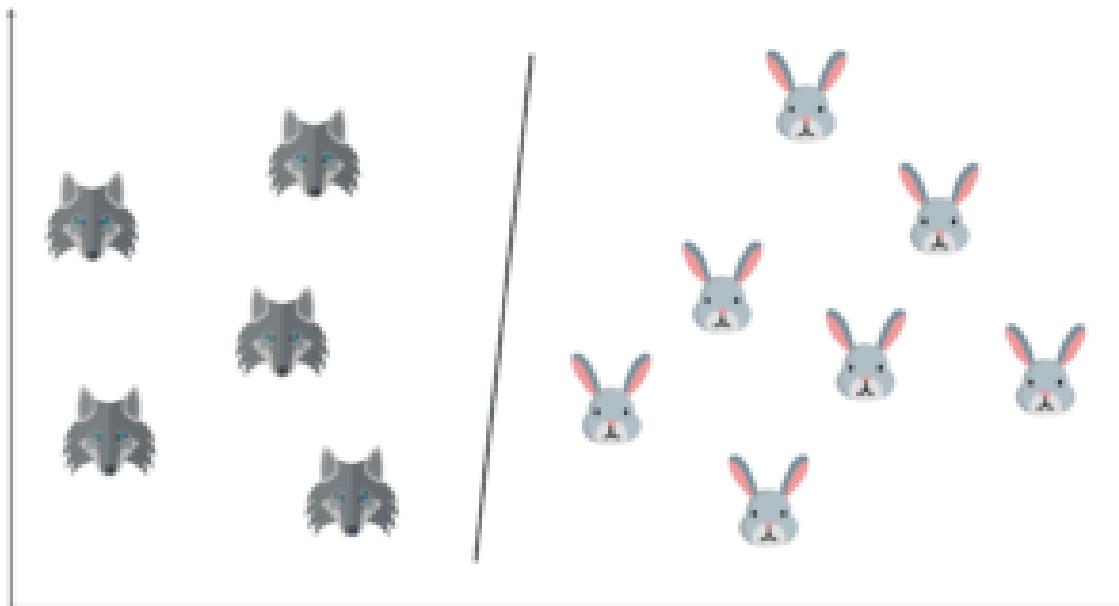
SVM

- We have plotted the size and weight of several people, and there is also a way to distinguish between men and women.
- With such data, using a SVM will allow us to answer the following question:
- Given a particular data point (weight and size), is the person a man or a woman ?
- For instance: if someone measures 175 cm and weights 80 kg, is it a man or a woman?

For a second, pretend you own a farm and you have a problem—you need to set up a fence to protect your rabbits from a pack of wolves. But where do you build your fence?

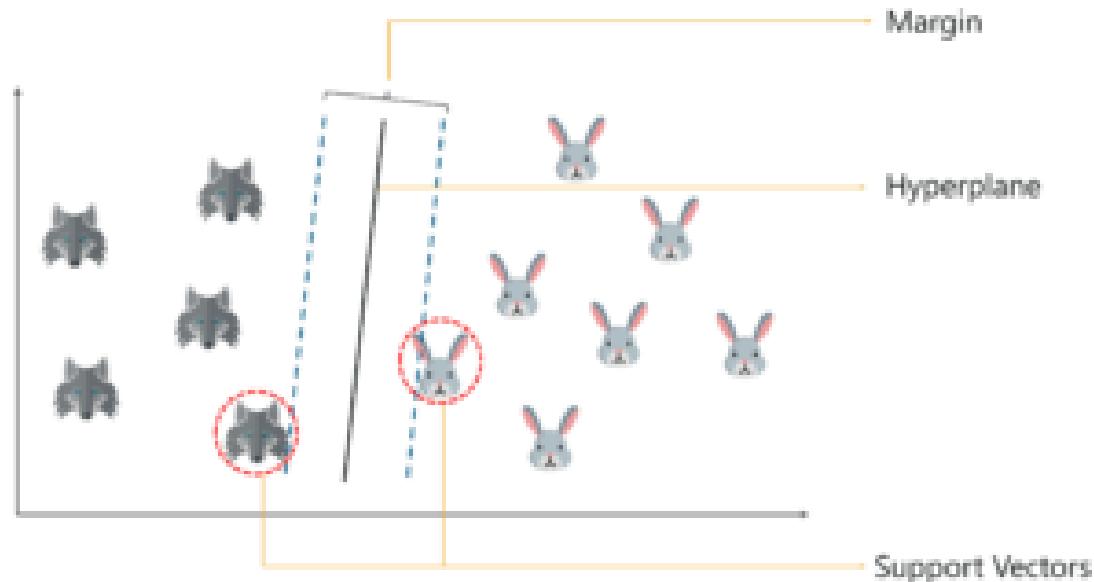


Hyperplane between any two classes in order to separate them or classify them

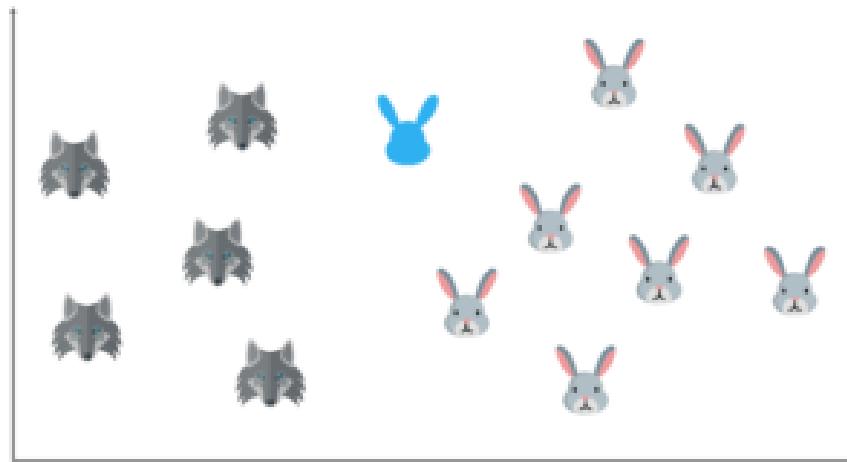


What is a Support Vector in SVM?

So, you start off by drawing a random hyperplane and then you check the distance between the hyperplane and the closest data points from each class. These closest data points to the hyperplane are known as support vectors. And that's where the name comes from, support vector machine.

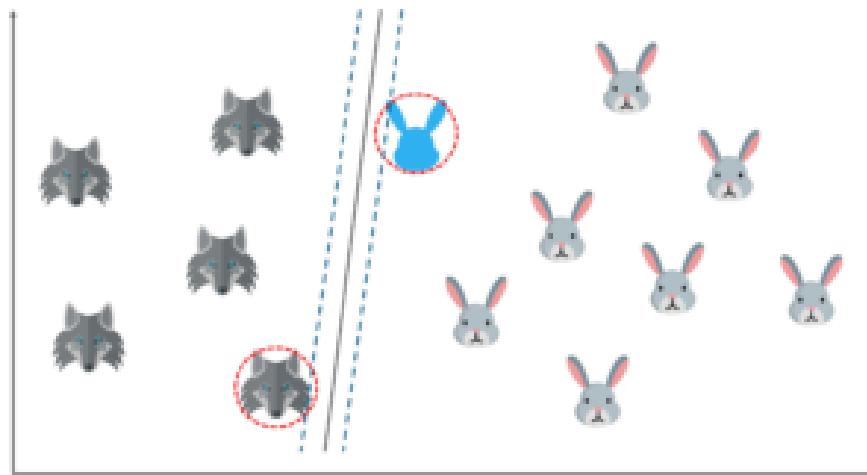


- The hyperplane is drawn based on these support vectors and an optimum hyperplane will have a maximum distance from each of the support vectors.
- And this distance between the hyperplane and the support vectors is known as the margin.
- To sum it up, SVM is used to classify data by using a hyperplane, such that the distance between the hyperplane and the support vectors is maximum.

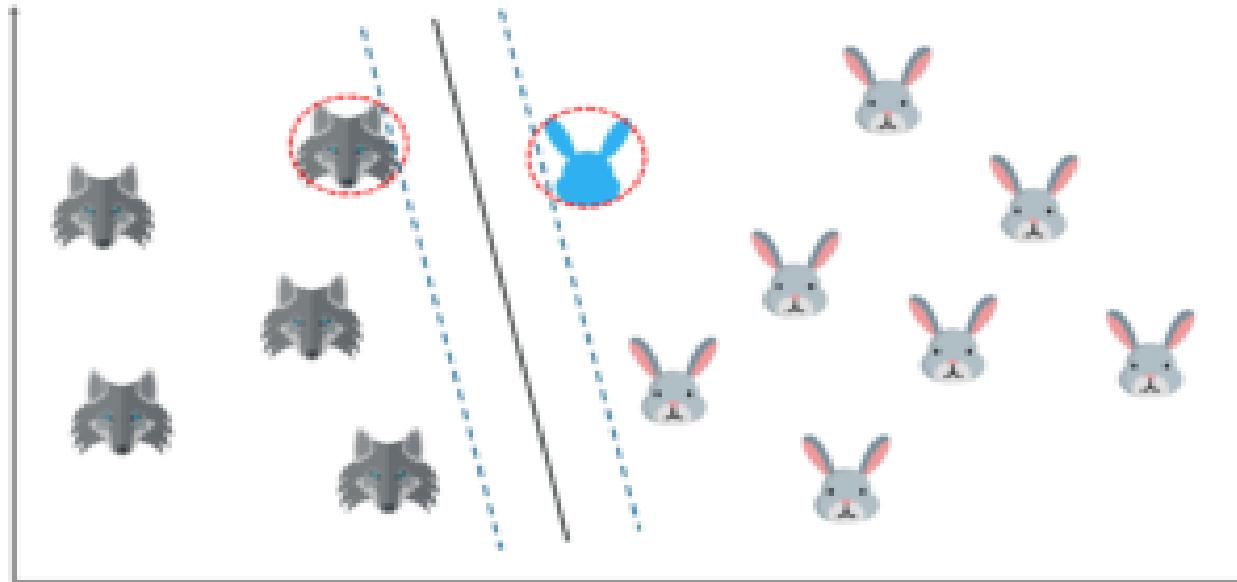


Let's say that I input a new data point and now I want to draw a hyperplane such that it best separates these two classes.

So, I start off by drawing a hyperplane and then I check the distance between the hyperplane and the support vectors. Here I'm basically trying to check if the margin is maximum for this hyperplane.



But what if I draw the hyperplane like this? The margin for this hyperplane is clearly more than the previous one. So, this is my optimal hyperplane.

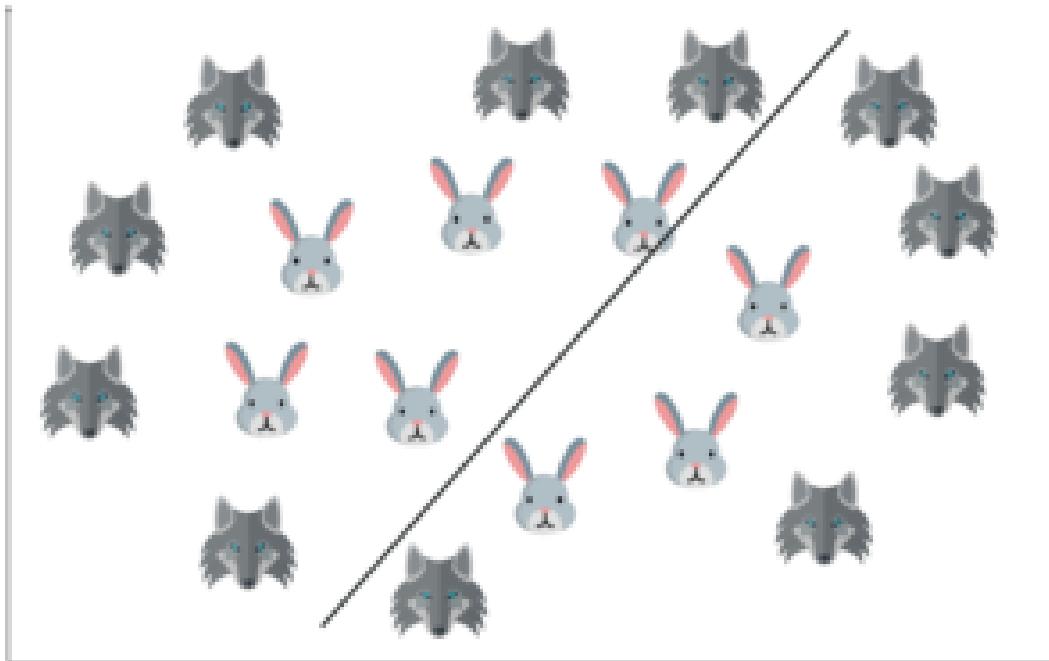


So far it was quite easy, our data was linearly separable, which means that you could draw a straight line to separate the two classes!

But what will you do if the data set is like this?



You possibly can't draw a hyperplane like this! It doesn't separate the two classes.

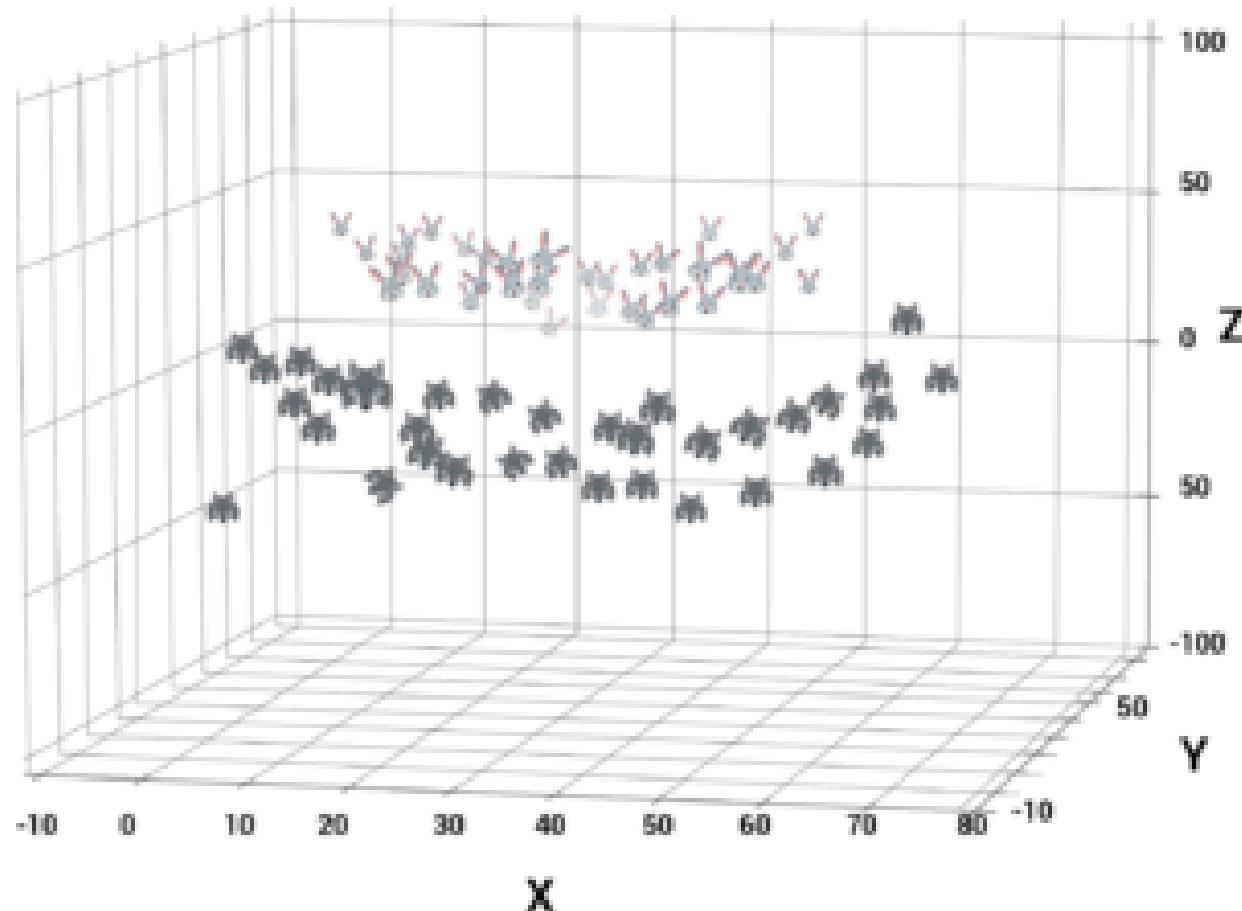


Non-Linear Support Vector Machine (SVM)



- Kernel functions offer the user the option of transforming nonlinear spaces into linear ones.
- Until this point, we were plotting our data on 2-dimensional space. So, we had only 2 variables, x and y.
- A simple trick would be transforming the two variables x and y into a new feature space involving a new variable z. Basically, we're visualizing the data on a 3-dimensional space.
- When you transform the 2D space into a 3D space you can clearly see a dividing margin between the 2 classes of data. And now you can go ahead and separate the two classes by drawing the best hyperplane between them.

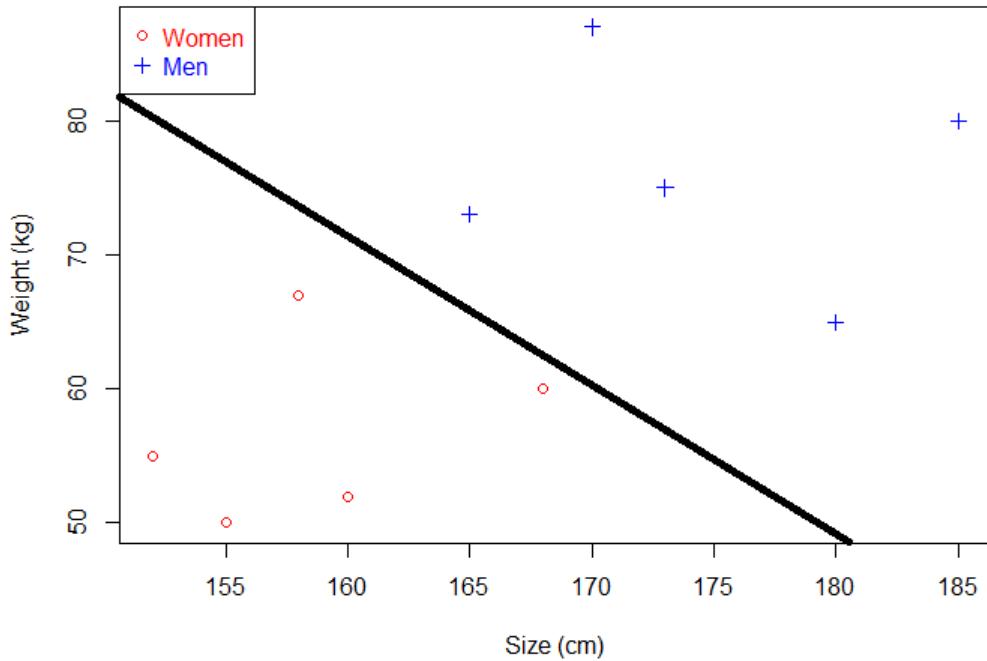
Non-Linear Support Vector Machine (SVM)



What is a separating hyperplane?



- For instance, we could trace a line and then all the data points representing men will be above the line, and all the data points representing women will be below the line.
- Such a line is called a **separating hyperplane** and is depicted below:



If it is just a line, why do we call it an hyperplane?

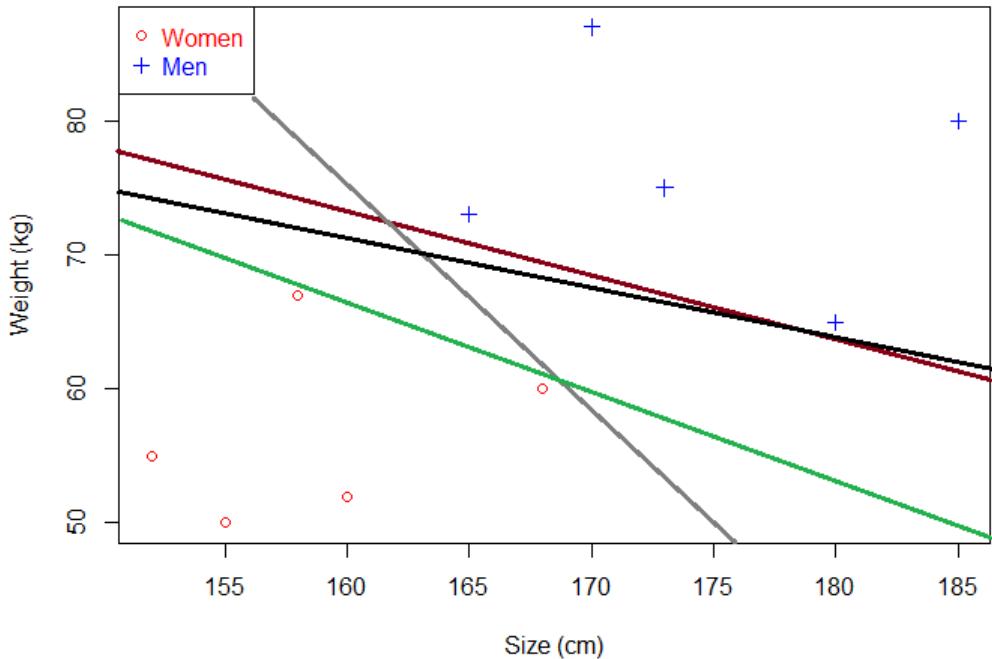


- An hyperplane is a generalization of a plane.
- in one dimension, an hyperplane is called a point
- in two dimensions, it is a line
- in three dimensions, it is a plane
- in more dimensions you can call it an hyperplane

What is the *optimal* separating hyperplane?

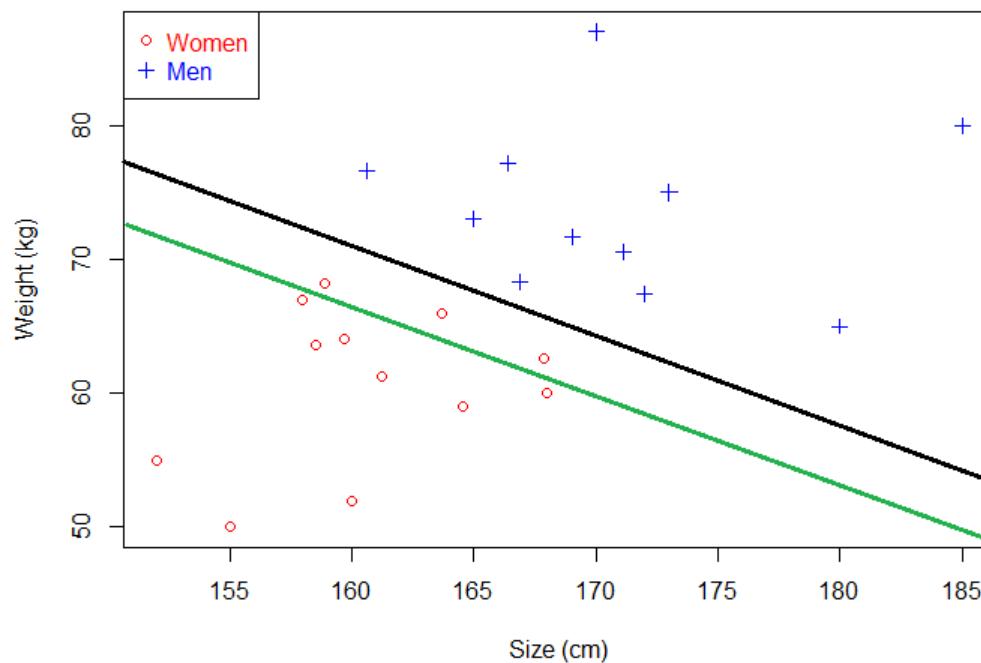


- The fact that you can find a **separating hyperplane**, does not mean it is the best one !
- In the example below there is several separating hyperplanes. Each of them is valid as it successfully separates our data set with men on one side and women on the other side.



Optimal Hyperplane

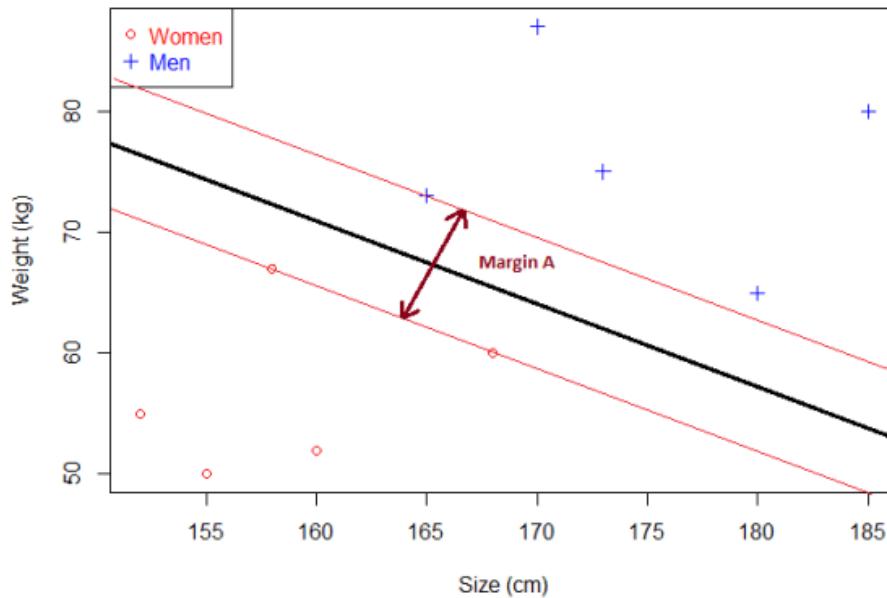
The black hyperplane classifies more accurately than the green one



What is the margin and how does it help choosing the optimal hyperplane?



- Given a particular hyperplane, we can compute the distance between the hyperplane and the closest data point.
- Once we have this value, if we double it we will get what is called the **margin**.
- Basically the margin is a no man's land. There will never be any data point inside the margin.**



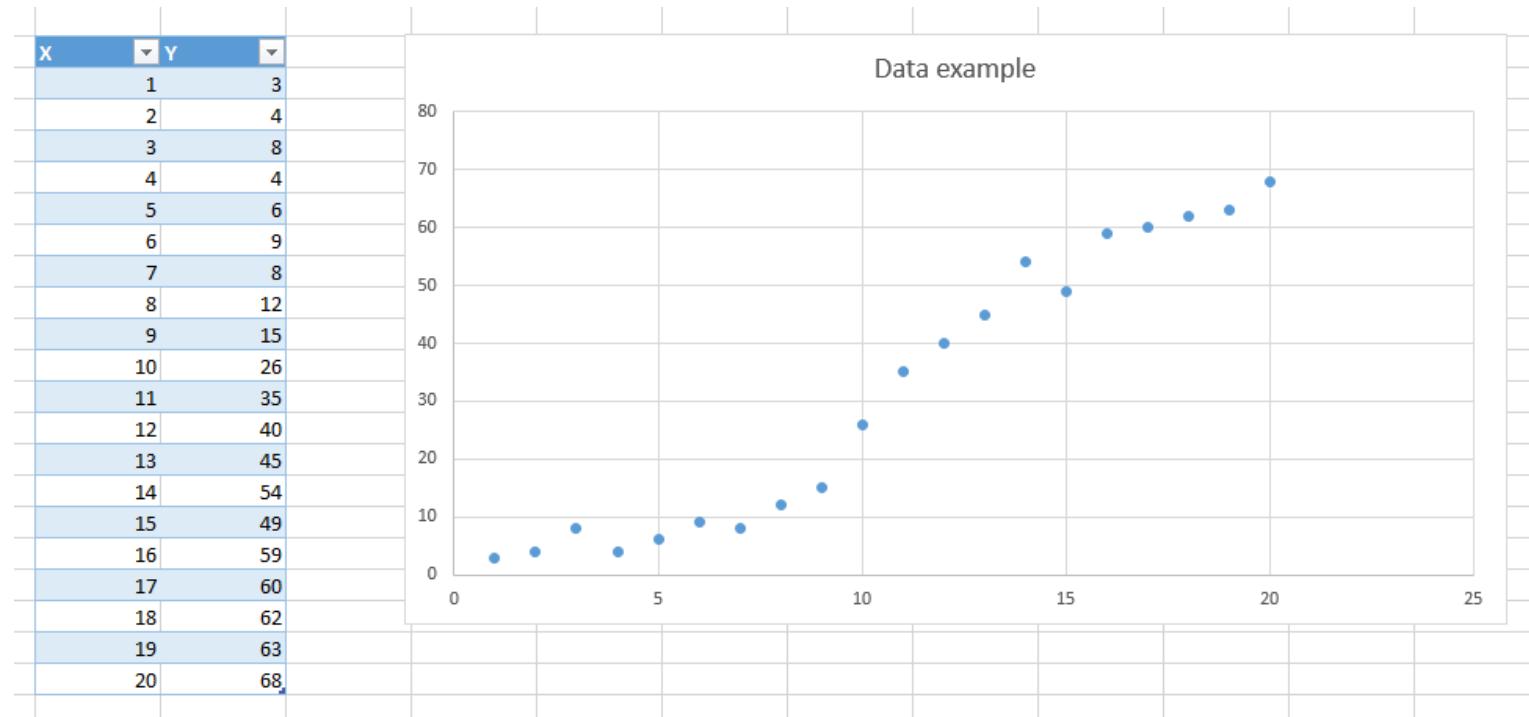
Optimal Hyperplane

We can make the following observations:

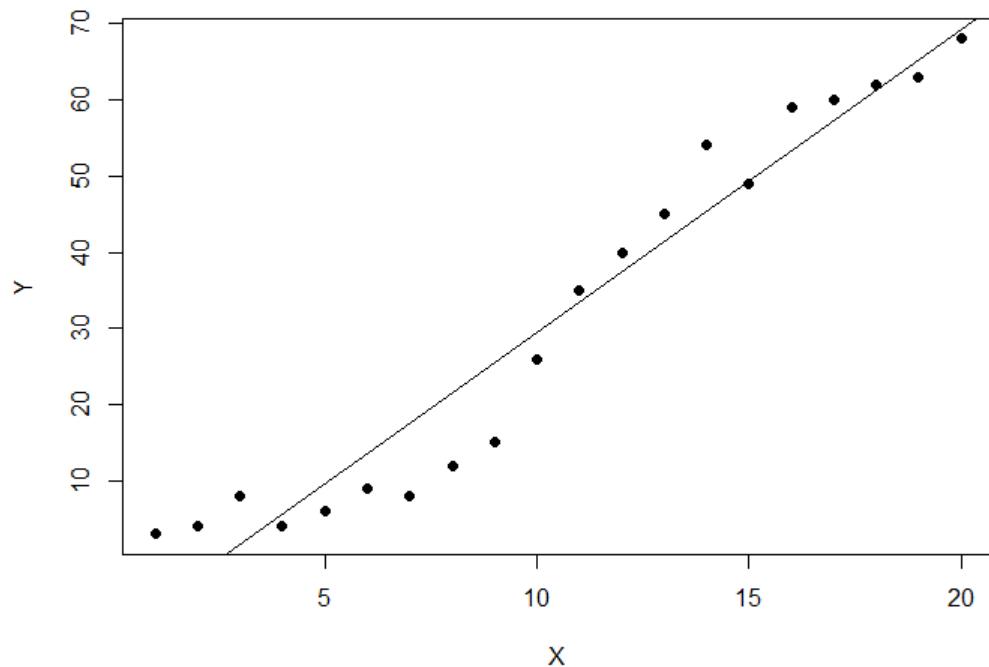
- If an hyperplane is very close to a data point, its margin will be small.
- The further an hyperplane is from a data point, the larger its margin will be.
- This means that **the optimal hyperplane will be the one with the biggest margin.**



Data Points



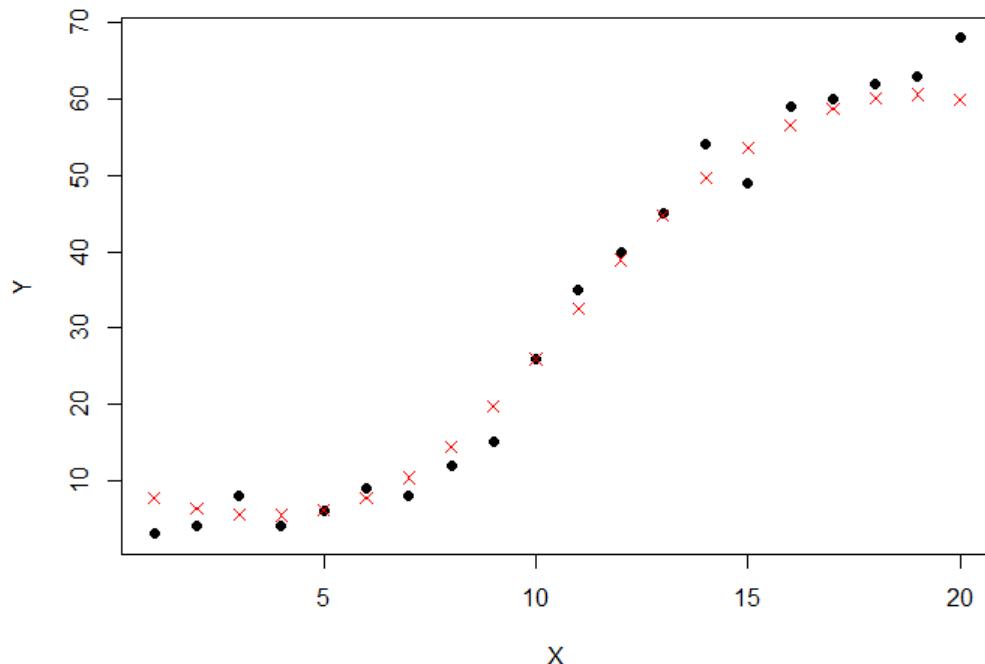
Linear Fit



Linear Fit

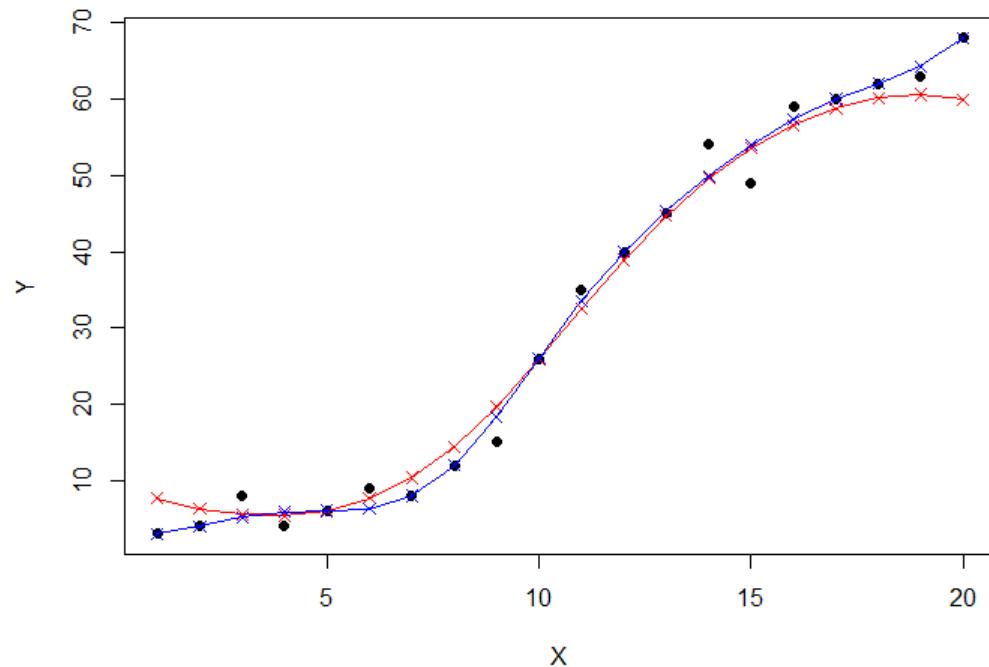
- rmse <- function(error)
- {
- sqrt(mean(error^2))
- }
-
- error <- model\$residuals # same as data\$Y - predictedY
- predictionRMSE <- rmse(error) # 5.703778
- How to Improve?

SVM Regression



```
# !\ this time svrModel$residuals is not  
# the same as data$Y - predictedY  
# so we compute the error like this  
error <- data$Y - predictedY  
svrPredictionRMSE <- rmse(error) #  
3.157061
```

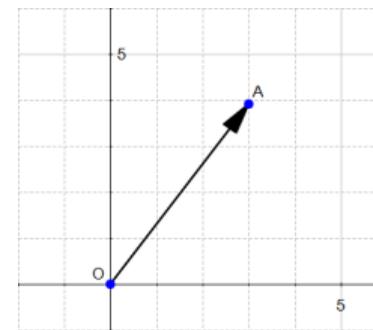
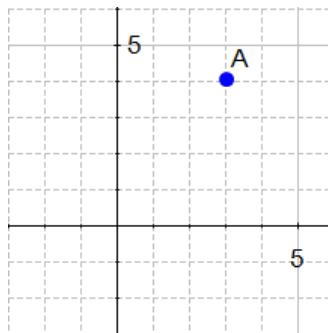
Improved SVM with eps factor adjustment



What is a vector?



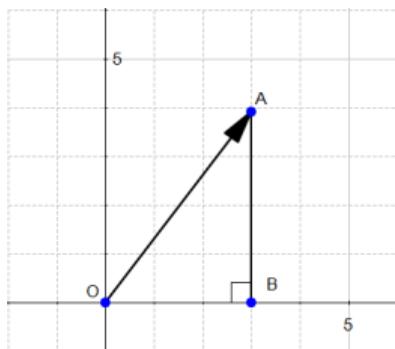
- A vector is an object that has both a magnitude and a direction.
- If we define a point $A(3,4)$ in R^2 we can plot it like this.
- Definition: Any point $x=(x_1,x_2), x \neq 0$, in R^2 specifies a vector in the plane, namely the vector starting at the origin and ending at x .
- If we say that the point at the origin is the point $O(0,0)$ then the vector above is the vector $OA \rightarrow$. We could also give it an arbitrary name such as u .



The magnitude



- The magnitude or length of a vector x is written $\|x\|$ and is called its norm.
- For our vector $OA \rightarrow$, $\|OA\|$ is the length of the segment OA



$$OA^2 = OB^2 + AB^2$$

$$OA^2 = 3^2 + 4^2$$

$$OA^2 = 25$$

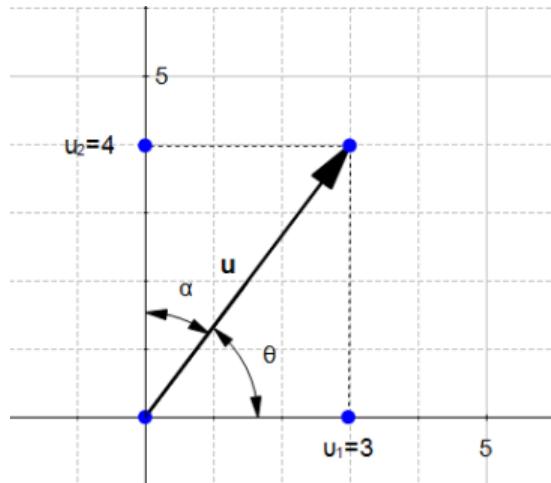
$$OA = \sqrt{25}$$

$$\|OA\| = OA = 5$$

The direction



- The direction is the second component of a vector.
- Definition : The direction of a vector $u(u_1, u_2)$ is the vector $w(u_1/\|u\|, u_2/\|u\|)$

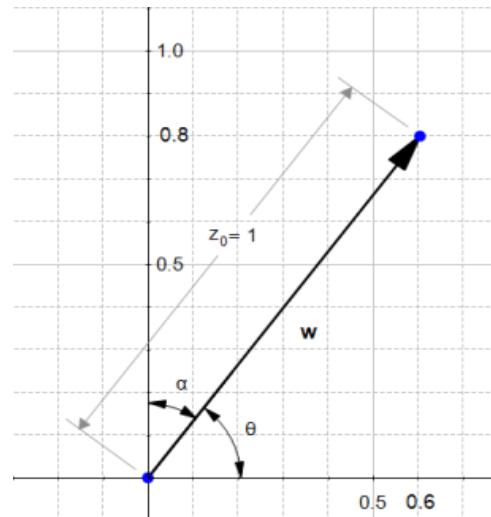


Computing the direction vector

- The direction of $u(3,4)$ is the vector $w(0.6,0.8)$

$$\cos(\theta) = \frac{u_1}{\|u\|} = \frac{3}{5} = 0.6$$

$$\cos(\alpha) = \frac{u_2}{\|u\|} = \frac{4}{5} = 0.8$$



The sum of two vectors

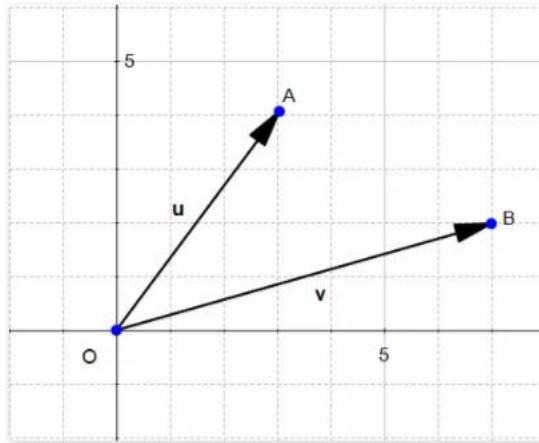


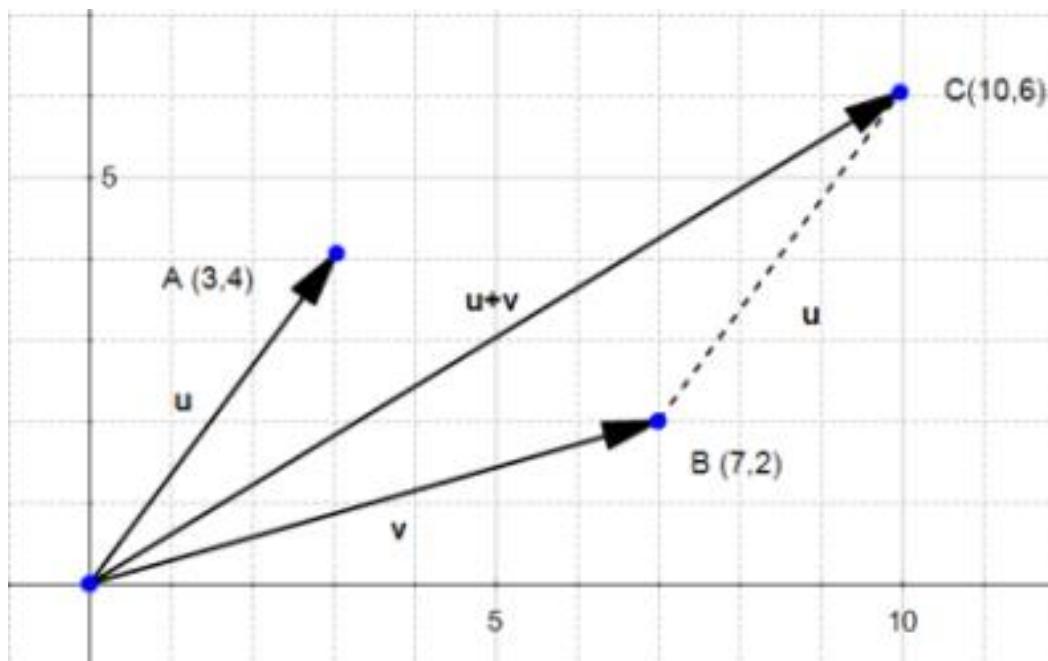
Figure 6: two vectors \mathbf{u} and \mathbf{v}

Given two vectors $\mathbf{u}(u_1, u_2)$ and $\mathbf{v}(v_1, v_2)$ then :

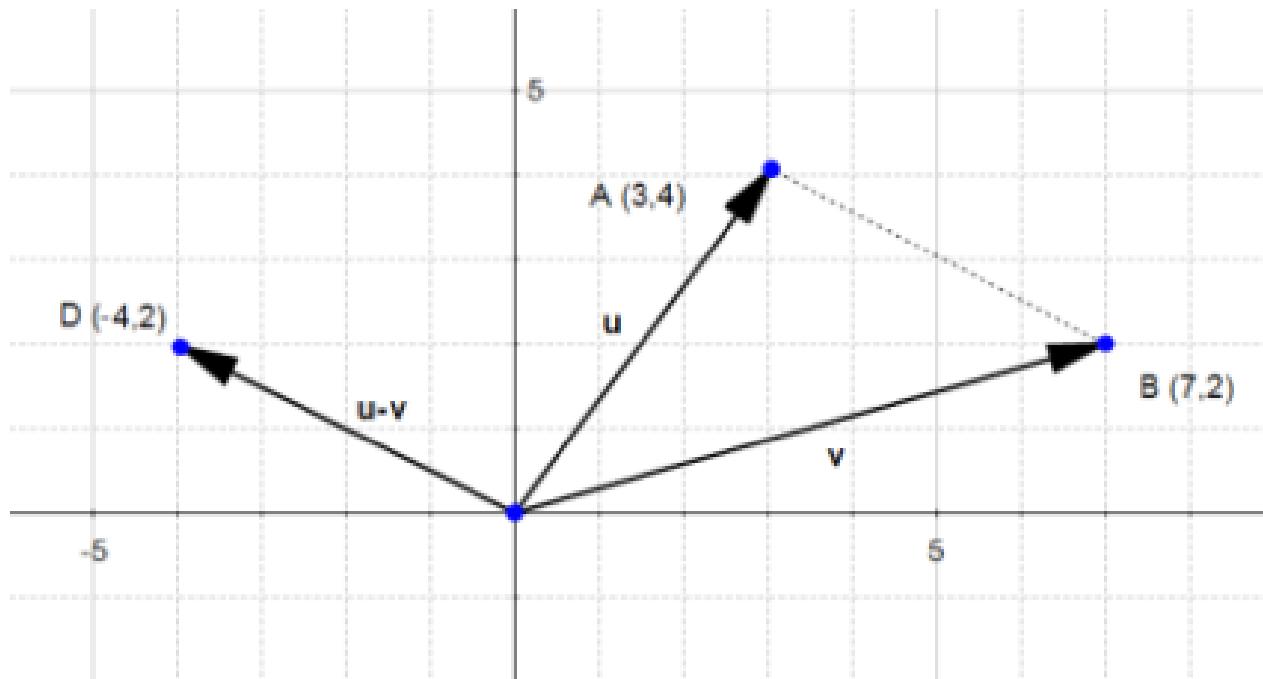
$$\mathbf{u} + \mathbf{v} = (u_1 + v_1, u_2 + v_2)$$



Third Vector



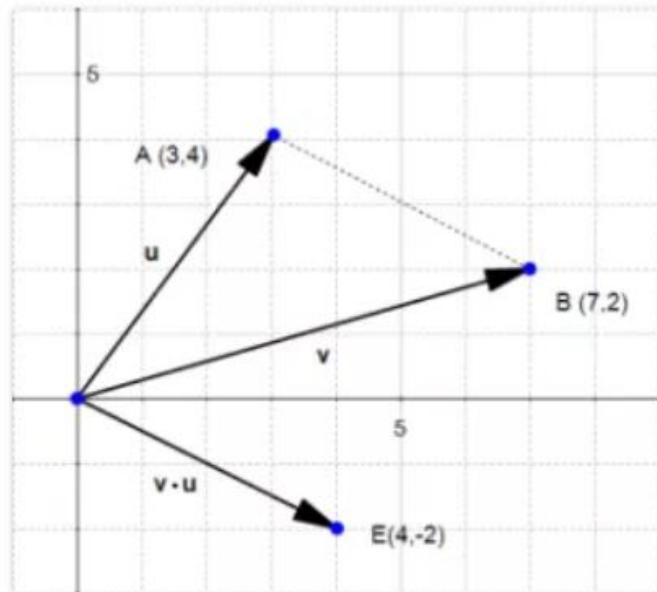
The difference between two vectors





The difference between two vectors

$$\mathbf{v} - \mathbf{u} = (v_1 - u_1, v_2 - u_2)$$



The dot product



- *Definition: Geometrically, it is the product of the Euclidian magnitudes of the two vectors and the cosine of the angle between them*

Which means if we have two vectors \mathbf{x} and \mathbf{y} and there is an angle θ (theta) between them, their dot product is :

$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\| \cos(\theta)$$

Why ?

To understand let's look at the problem geometrically.

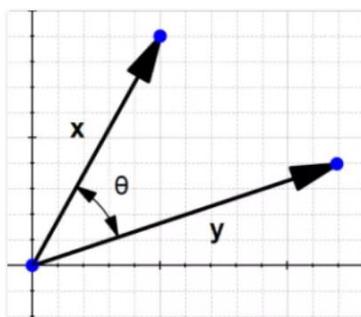
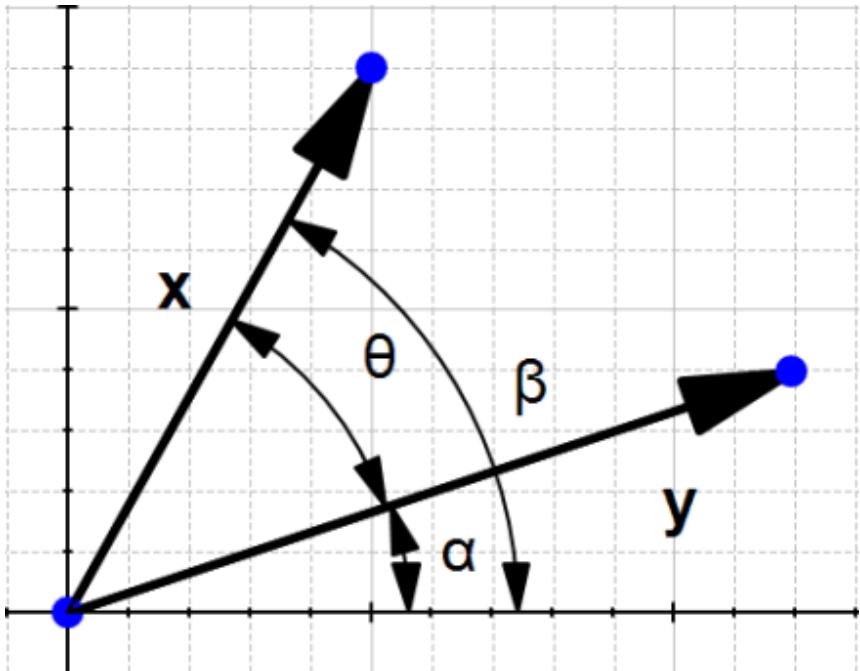


Figure 12

- We can see that
- $\theta = \beta - \alpha$
- So computing $\cos(\theta)$ is like computing $\cos(\beta - \alpha)$
- There is a special formula called the difference identity for cosine which says that:
- $\cos(\beta - \alpha) = \cos(\beta)\cos(\alpha) + \sin(\beta)\sin(\alpha)$



Let's use this formula!

$$\cos(\beta) = \frac{\text{adjacent}}{\text{hypotenuse}} = \frac{x_1}{\|x\|}$$

$$\sin(\beta) = \frac{\text{opposite}}{\text{hypotenuse}} = \frac{x_2}{\|x\|}$$

$$\cos(\alpha) = \frac{\text{adjacent}}{\text{hypotenuse}} = \frac{y_1}{\|y\|}$$

$$\sin(\alpha) = \frac{\text{opposite}}{\text{hypotenuse}} = \frac{y_2}{\|y\|}$$

So if we replace each term

$$\cos(\theta) = \cos(\beta - \alpha) = \cos(\beta)\cos(\alpha) + \sin(\beta)\sin(\alpha)$$

$$\cos(\theta) = \frac{x_1}{\|x\|} \frac{y_1}{\|y\|} + \frac{x_2}{\|x\|} \frac{y_2}{\|y\|}$$

$$\cos(\theta) = \frac{x_1 y_1 + x_2 y_2}{\|x\| \|y\|}$$

If we multiply both sides by $\|x\| \|y\|$ we get:

$$\|x\| \|y\| \cos(\theta) = x_1 y_1 + x_2 y_2$$

Which is the same as :

$$\|x\|\|y\|\cos(\theta) = \mathbf{x} \cdot \mathbf{y}$$

We just found the geometric definition of the dot product !

Eventually from the two last equations we can see that :

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 = \sum_{i=1}^2 (x_iy_i)$$

This is the algebraic definition of the dot product !

The SVM hyperplane



- Understanding the equation of the hyperplane
- You probably learnt that an equation of a line is : $y=ax+b$. However when reading about hyperplane, you will often find that the equation of an hyperplane is defined by :

$$\mathbf{w}^T \mathbf{x} = 0$$

The SVM hyperplane



- How does these two forms relate ?
- In the hyperplane equation you can see that the name of the variables are in bold. Which means that they are vectors !
- Moreover, $w^T x$ is how we compute the inner product of two vectors, and if you recall, the inner product is just another name for the dot product !

The SVM hyperplane



Note that

$$y = ax + b$$

is the same thing as

$$y - ax - b = 0$$

Given two vectors $\mathbf{w} \begin{pmatrix} -b \\ -a \\ 1 \end{pmatrix}$ and $\mathbf{x} \begin{pmatrix} 1 \\ x \\ y \end{pmatrix}$

$$\mathbf{w}^T \mathbf{x} = -b \times (1) + (-a) \times x + 1 \times y$$

$$\mathbf{w}^T \mathbf{x} = y - ax - b$$

The two equations are just different ways of expressing the same thing.

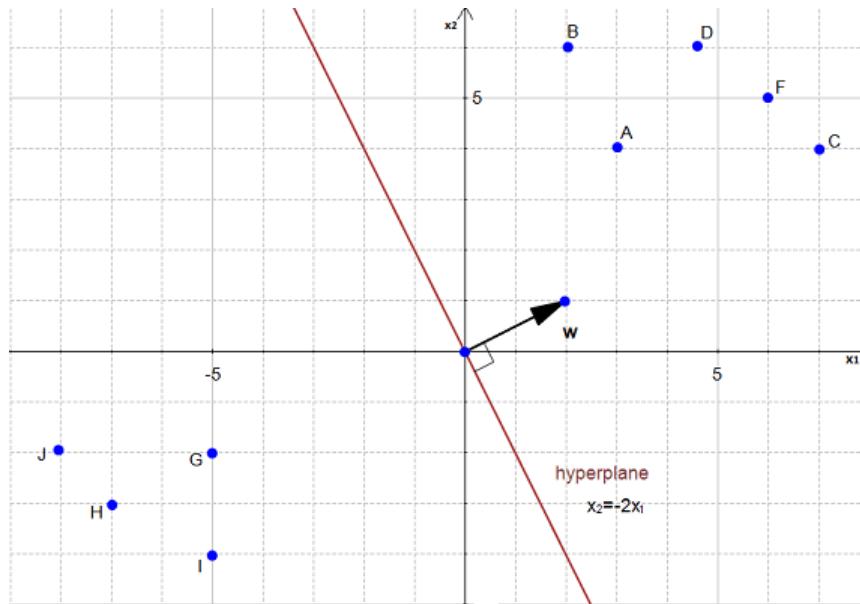
It is interesting to note that w_0 is $-b$, which means that this value determines the intersection of the line with the vertical axis.

Why do we use the hyperplane equation $\mathbf{w}^T \mathbf{x}$ instead of $y = ax + b$?

The SVM hyperplane



- For two reasons:
- it is easier to work in more than two dimensions with this notation,
- the vector w will always be normal to the hyperplane.



SVM hyperplane

- Our goal is to find the distance between the point $A(3,4)$ and the hyperplane.
- We start with two vectors, $w=(2,1)$ which is normal to the hyperplane, and $a=(3,4)$ which is the vector between the origin and A .

$$\|w\| = \sqrt{2^2 + 1^2} = \sqrt{5}$$

Let the vector **u** be the direction of **w**

$$\mathbf{u} = \left(\frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}} \right)$$

p is the orthogonal projection of **a** onto **w** so :

$$\mathbf{p} = (\mathbf{u} \cdot \mathbf{a})\mathbf{u}$$

$$\mathbf{p} = \left(3 \times \frac{2}{\sqrt{5}} + 4 \times \frac{1}{\sqrt{5}} \right) \mathbf{u}$$

$$\mathbf{p} = \left(\frac{6}{\sqrt{5}} + \frac{4}{\sqrt{5}} \right) \mathbf{u}$$

$$\mathbf{p} = \frac{10}{\sqrt{5}} \mathbf{u}$$

$$\mathbf{p} = \left(\frac{10}{\sqrt{5}} \times \frac{2}{\sqrt{5}}, \frac{10}{\sqrt{5}} \times \frac{1}{\sqrt{5}} \right)$$

$$\mathbf{p} = \left(\frac{20}{5}, \frac{10}{5} \right)$$

$$\mathbf{p} = (4, 2)$$

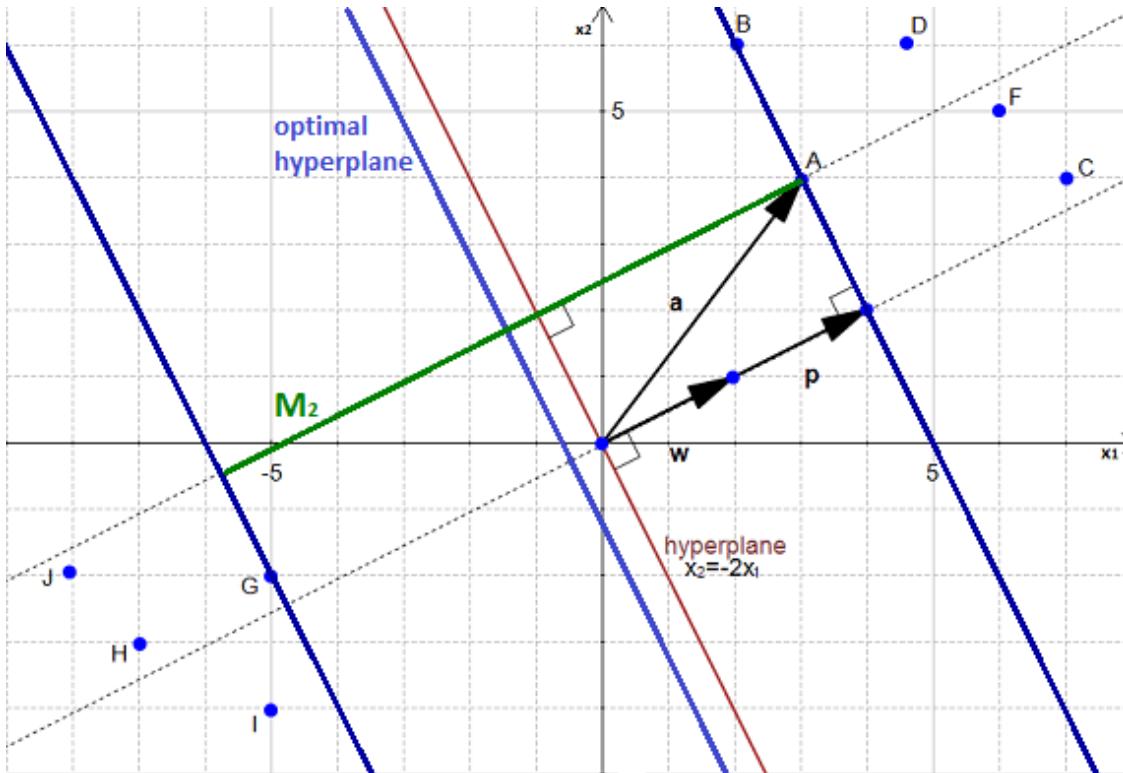
$$\|p\| = \sqrt{4^2 + 2^2} = 2\sqrt{5}$$

SVM Hyperplane

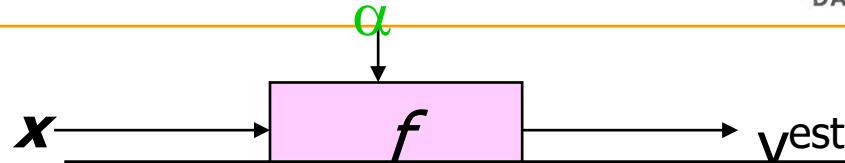
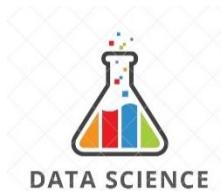
- Compute the margin of the hyperplane
- Now that we have the distance $\|p\|$ between A and the hyperplane, the margin is defined by :

$$\text{margin} = 2\|p\| = 4\sqrt{5}$$

Optimal hyperplane

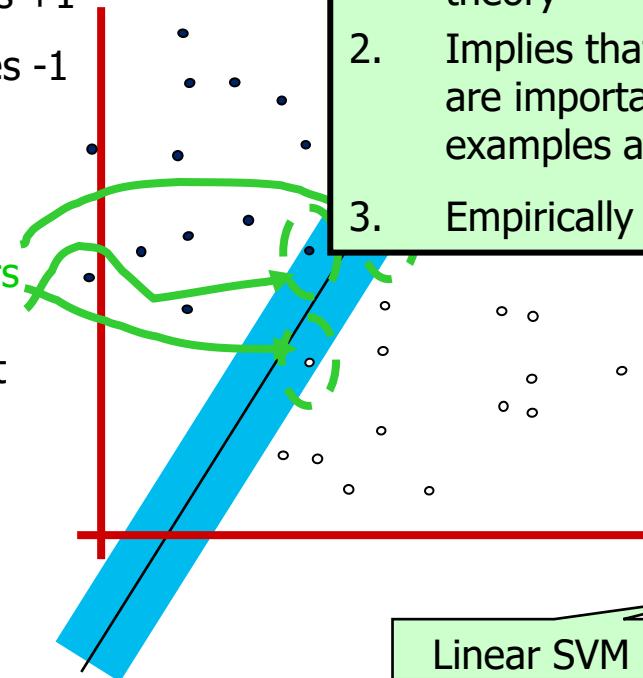


Maximum Margin



- denotes +1
- denotes -1

Support Vectors
are those
datapoints that
the margin
pushes up
against

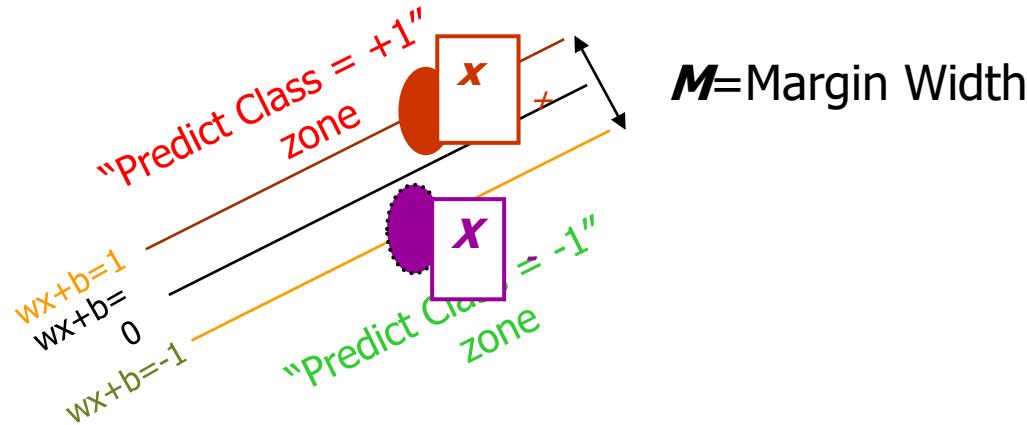


1. Maximizing the margin is good according to intuition and PAC theory
2. Implies that only support vectors are important; other training examples are ignorable.
3. Empirically it works very very well.

with the, um,
maximum margin.

This is the
simplest kind of
SVM (Called an
LSVM)

Linear SVM Mathematically



What we know:

- $w \cdot x^+ + b = +1$
- $w \cdot x^- + b = -1$
- $w \cdot (x^+ - x^-) = 2$

$$M = \frac{(x^+ - x^-) \cdot w}{|w|} = \frac{2}{|w|}$$

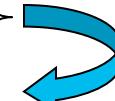
Linear SVM Mathematically



- Goal: 1) Correctly classify all training data

$$\begin{aligned} w\mathbf{x}_i + b &\geq 1 & \text{if } y_i = +1 \\ w\mathbf{x}_i + b &\leq -1 & \text{if } y_i = -1 \\ y_i(w\mathbf{x}_i + b) &\geq 1 & \text{for all } i \end{aligned}$$

2) Maximize the Margin $M = \frac{2}{|w|}$
same as minimize $\frac{1}{2} w^t w$



- We can formulate a Quadratic Optimization Problem and solve for w and b

- Minimize $\Phi(w) = \frac{1}{2} w^t w$

subject to $y_i(w\mathbf{x}_i + b) \geq 1 \quad \forall i$

Solving the Optimization Problem



Find w and b such that

$\Phi(w) = \frac{1}{2} w^T w$ is minimized;

and for all $\{(x_i, y_i)\}$: $y_i (w^T x_i + b) \geq 1$

- Need to optimize a *quadratic* function subject to *linear* constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems, and many (rather intricate) algorithms exist for solving them.
- The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:

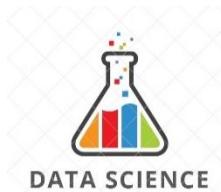
Find $\alpha_1 \dots \alpha_N$ such that

$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j x_i^T x_j$ is maximized and

$$(1) \sum \alpha_i y_i = 0$$

$$(2) \alpha_i \geq 0 \text{ for all } \alpha_i$$

The Optimization Problem Solution

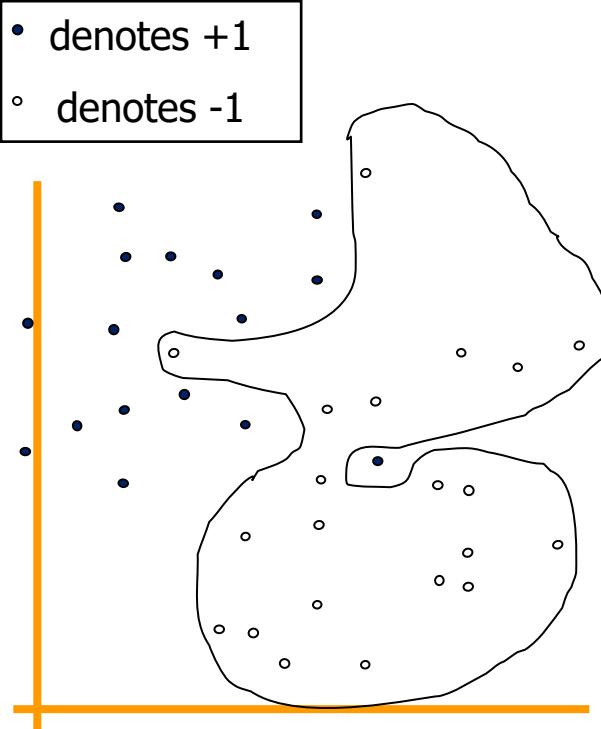


- The solution has the form:

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i \quad b = y_k - \mathbf{w}^T \mathbf{x}_k \text{ for any } \mathbf{x}_k \text{ such that } \alpha_k \neq 0$$

- Each non-zero α_i indicates that corresponding \mathbf{x}_i is a support vector.
- Then the classifying function will have the form:
$$f(\mathbf{x}) = \sum \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$
- Notice that it relies on an *inner product* between the test point \mathbf{x} and the support vectors \mathbf{x}_i – we will return to this later.
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x}_i^T \mathbf{x}_j$ between all pairs of training points.

Dataset with noise

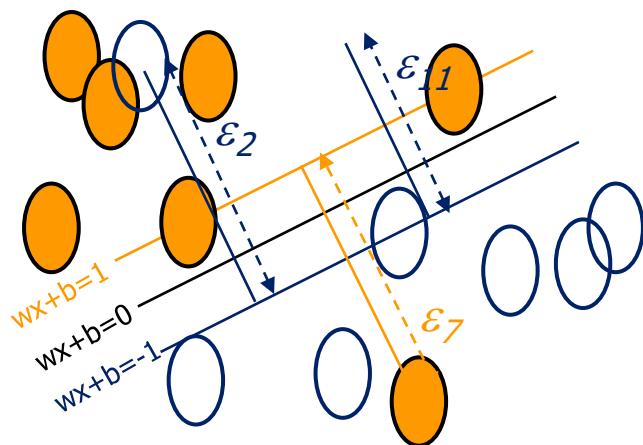


- **Hard Margin:** So far we require all data points be classified correctly
 - No training error
- What if the training set is noisy?
 - **Solution 1:** use very powerful kernels

OVERFITTING!

Soft Margin Classification

Slack variables ξ_i can be added to allow misclassification of difficult or noisy examples.



What should our quadratic optimization criterion be?

Minimize

$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R \xi_k$$

Hard Margin v.s. Soft Margin

- **The old formulation:**

Find \mathbf{w} and b such that

$$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} \text{ is minimized and for all } \{(\mathbf{x}_i, y_i)\}$$

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

- **The new formulation incorporating slack variables:**

Find \mathbf{w} and b such that

$$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum \xi_i \text{ is minimized and for all } \{(\mathbf{x}_i, y_i)\}$$

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0 \text{ for all } i$$

- **Parameter C can be viewed as a way to control overfitting.**

Linear SVMs: Overview

- The classifier is a *separating hyperplane*.
- Most “important” training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points x_i are support vectors with non-zero Lagrangian multipliers α_i .
- Both in the dual formulation of the problem and in the solution training points appear only inside dot products:

Find $\alpha_1 \dots \alpha_N$ such that

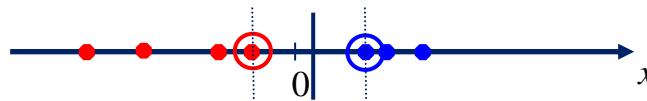
$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j x_i^T x_j$ is maximized and

- (1) $\sum \alpha_i y_i = 0$
- (2) $0 \leq \alpha_i \leq C$ for all α_i

$$f(x) = \sum \alpha_i y_i x_i^T x + b$$

Non-linear SVMs

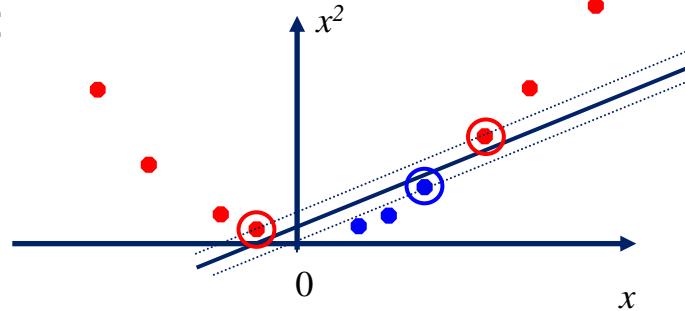
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?

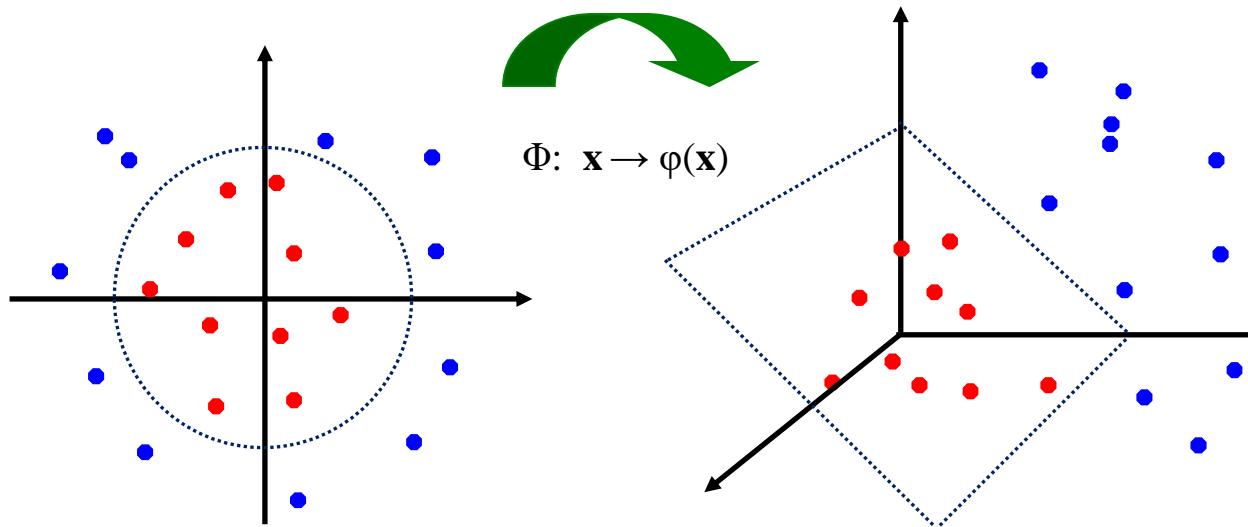


- How about... mapping data to a higher-dimensional space:



Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



The “Kernel Trick”

- The linear classifier relies on dot product between vectors $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- If every data point is mapped into high-dimensional space via some transformation $\Phi: \mathbf{x} \rightarrow \phi(\mathbf{x})$, the dot product becomes:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

- A *kernel function* is some function that corresponds to an inner product in some expanded feature space.
- Example:

2-dimensional vectors $\mathbf{x} = [x_1 \ x_2]$; let $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$,

Need to show that $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$:

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= (1 + \mathbf{x}_i^T \mathbf{x}_j)^2, \\ &= 1 + x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{j1} x_{i2} x_{j2} + x_{i2}^2 x_{j2}^2 + 2 x_{i1} x_{j1} + 2 x_{i2} x_{j2} \\ &= [1 \ x_{i1}^2 \ \sqrt{2} x_{i1} x_{i2} \ x_{i2}^2 \ \sqrt{2} x_{i2}]^T [1 \ x_{j1}^2 \ \sqrt{2} x_{j1} x_{j2} \ x_{j2}^2 \ \sqrt{2} x_{j1} \ \sqrt{2} x_{j2}] \\ &= \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \quad \text{where } \phi(\mathbf{x}) = [1 \ x_1^2 \ \sqrt{2} x_1 x_2 \ x_2^2 \ \sqrt{2} x_1 \ \sqrt{2} x_2] \end{aligned}$$

What Functions are Kernels?

- For some functions $K(\mathbf{x}_i, \mathbf{x}_j)$ checking that $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ can be cumbersome.
- Mercer's theorem:
Every semi-positive definite symmetric function is a kernel
- Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric Gram matrix:

$K =$

| | | | | |
|---------------------------------|---------------------------------|---------------------------------|-----|---------------------------------|
| $K(\mathbf{x}_1, \mathbf{x}_1)$ | $K(\mathbf{x}_1, \mathbf{x}_2)$ | $K(\mathbf{x}_1, \mathbf{x}_3)$ | ... | $K(\mathbf{x}_1, \mathbf{x}_N)$ |
| $K(\mathbf{x}_2, \mathbf{x}_1)$ | $K(\mathbf{x}_2, \mathbf{x}_2)$ | $K(\mathbf{x}_2, \mathbf{x}_3)$ | | $K(\mathbf{x}_2, \mathbf{x}_N)$ |
| ... | ... | ... | ... | ... |
| $K(\mathbf{x}_N, \mathbf{x}_1)$ | $K(\mathbf{x}_N, \mathbf{x}_2)$ | $K(\mathbf{x}_N, \mathbf{x}_3)$ | ... | $K(\mathbf{x}_N, \mathbf{x}_N)$ |

Examples of Kernel Functions

- Linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Polynomial of power p : $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$

- Gaussian (radial-basis function network):

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$

Non-linear SVMs Mathematically

- Dual problem formulation:

Find $\alpha_1 \dots \alpha_N$ such that

$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j K(x_i, x_j)$ is maximized and

$$(1) \sum \alpha_i y_i = 0$$

$$(2) \alpha_i \geq 0 \text{ for all } \alpha_i$$

- The solution is:

$$f(x) = \sum \alpha_i y_i K(x_i, x_j) + b$$

- Optimization techniques for finding α_i 's remain the same!

Nonlinear SVM - Overview

- SVM locates a separating hyperplane in the feature space and classify points in that space
- It does not need to represent the space explicitly, simply by defining a kernel function
- The kernel function plays the role of the dot product in the feature space.

Properties of SVM

- **Flexibility in choosing a similarity function**
- **Sparseness of solution when dealing with large data sets**
 - only support vectors are used to specify the separating hyperplane
- **Ability to handle large feature spaces**
 - complexity does not depend on the dimensionality of the feature space
- **Overfitting can be controlled by soft margin approach**
- **Nice math property:** a simple convex optimization problem which is guaranteed to converge to a single global solution
- **Feature Selection**

SVM Applications

- **SVM has been used successfully in many real-world problems**
 - text (and hypertext) categorization
 - image classification
 - bioinformatics (Protein classification,
Cancer classification)
 - hand-written character recognition

Application 1: Cancer Classification

- High Dimensional
 - $p > 1000$; $n < 100$

- Imbalanced
 - less positive samples

$$K[x, x] = k(x, x) + \lambda \frac{n^+}{N}$$

- Many irrelevant features
- Noisy

SVM is sensitive to noisy (mis-labeled) data \otimes

| Patients | Genes | | | |
|----------|-------|-----|-------|-----|
| | g-1 | g-2 | | g-p |
| P-1 | | | | |
| p-2 | | | | |
| | | | | |
| p-n | | | | |

FEATURE SELECTION

In the linear case,
 w_i^2 gives the ranking of dim i

Weakness of SVM

- **It is sensitive to noise**
 - A relatively small number of mislabeled examples can dramatically decrease the performance
- **It only considers two classes**
 - how to do multi-class classification with SVM?
 - Answer:
 - 1) with output arity m, learn m SVM's
 - SVM 1 learns "Output==1" vs "Output != 1"
 - SVM 2 learns "Output==2" vs "Output != 2"
 - :
 - SVM m learns "Output==m" vs "Output != m"
 - 2) To predict the output for a new input, just predict with each SVM and find out which one puts the prediction the furthest into the positive region.

Application 2: Text Categorization

- Task: The classification of natural text (or hypertext) documents into a fixed number of predefined categories based on their content.
 - email filtering, web searching, sorting documents by topic, etc..
- A document can be assigned to more than one category, so this can be viewed as a series of binary classification problems, one for each category

Representation of Text

IR's vector space model (aka bag-of-words representation)

- A doc is represented by a vector indexed by a pre-fixed set or dictionary of terms
- Values of an entry can be binary or weights

$$\phi_i(x) = \frac{\text{tf}_i \log (\text{idf}_i)}{\kappa},$$

- Normalization, stop words, word stems
- Doc $x \Rightarrow \varphi(x)$

Text Categorization using SVM

- The distance between two documents is $\varphi(x) \cdot \varphi(z)$
- $K(x,z) = \langle \varphi(x) \cdot \varphi(z) \rangle$ is a valid kernel, SVM can be used with $K(x,z)$ for discrimination.
- Why SVM?
 - High dimensional input space
 - Few irrelevant features (dense concept)
 - Sparse document vectors (sparse instances)
 - Text categorization problems are linearly separable

Some Issues

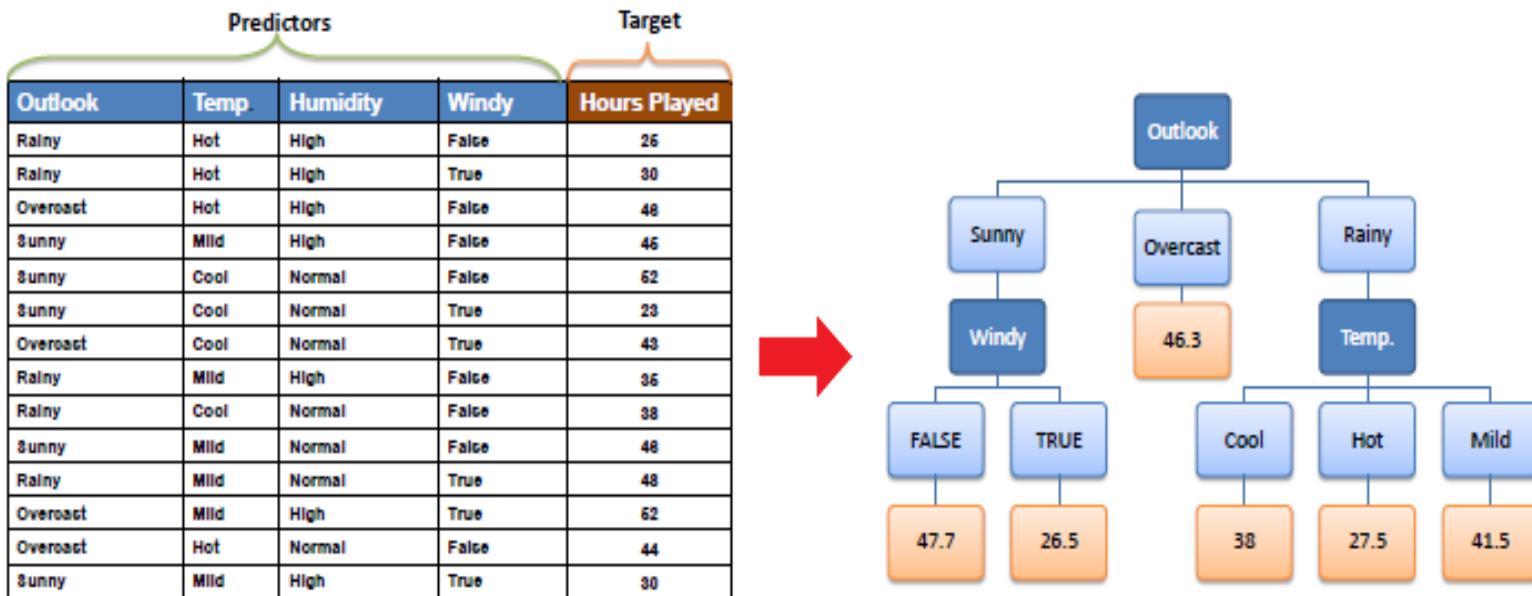
- **Choice of kernel**
 - Gaussian or polynomial kernel is default
 - if ineffective, more elaborate kernels are needed
 - domain experts can give assistance in formulating appropriate similarity measures
- **Choice of kernel parameters**
 - e.g. σ in Gaussian kernel
 - σ is the distance between closest points with different classifications
 - In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set such parameters.
- **Optimization criterion** – Hard margin v.s. Soft margin
 - a lengthy series of experiments in which various parameters are tested

Decision Tree Regression

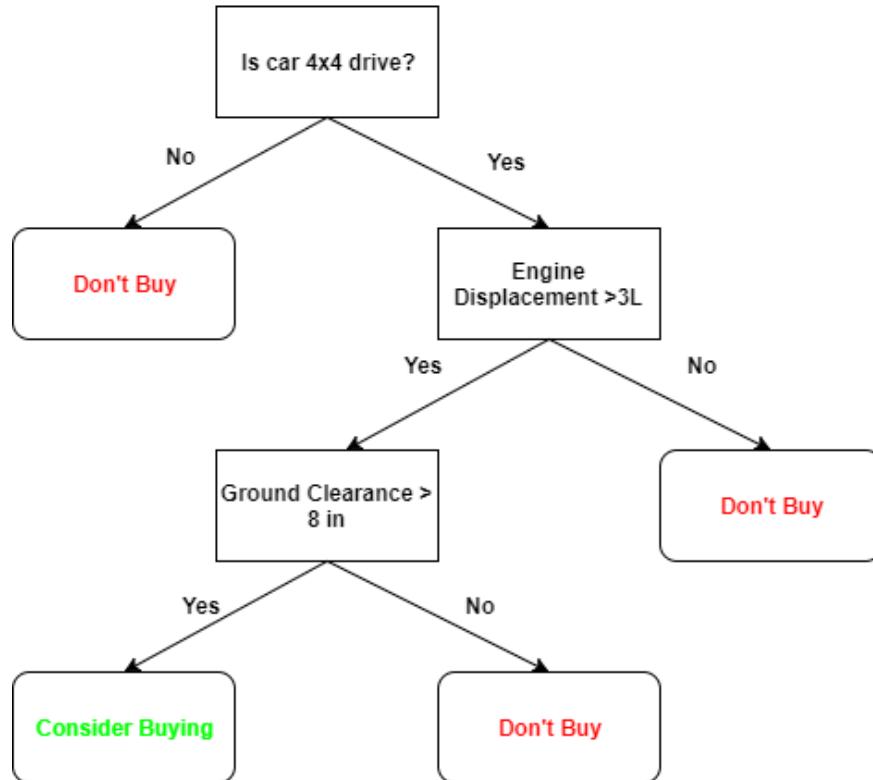


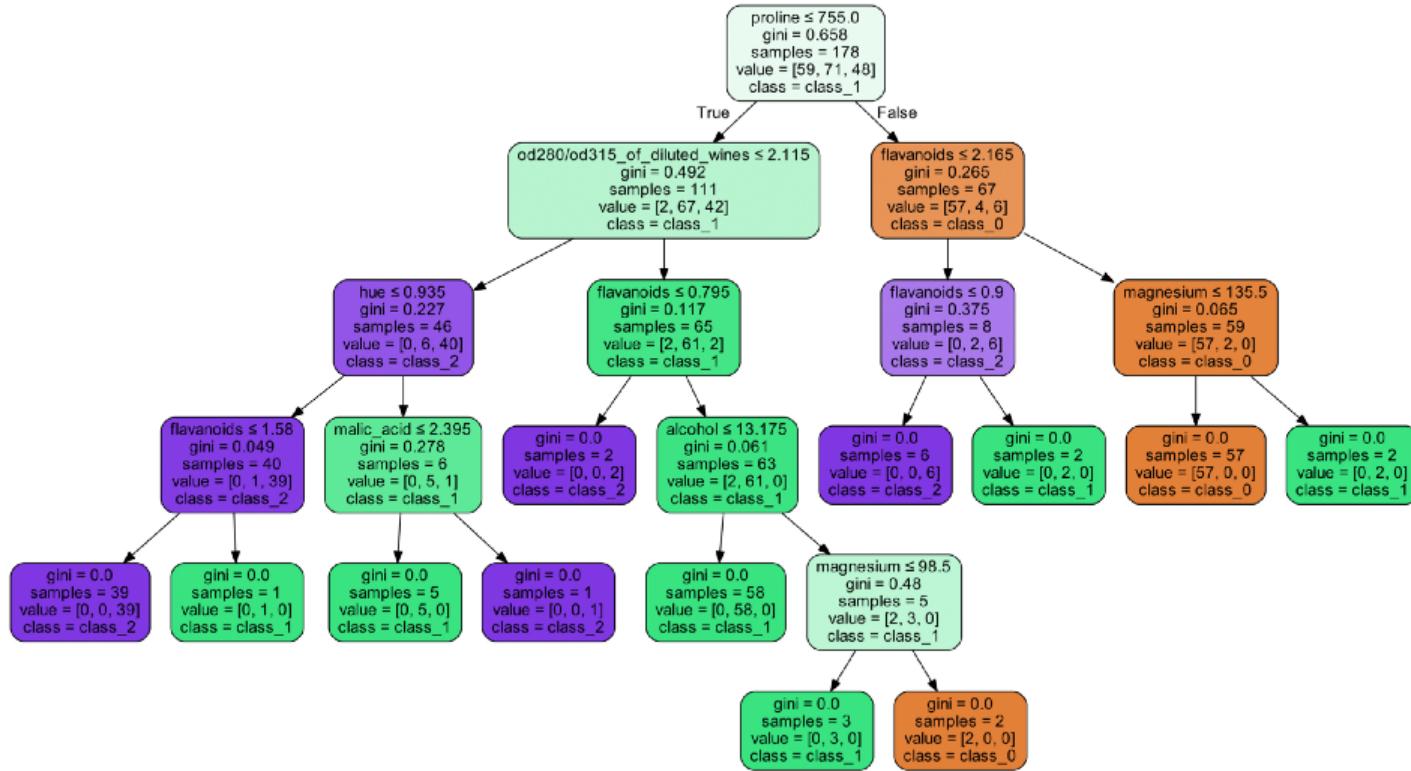
- Decision tree builds regression or classification models in the form of a tree structure.
- It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed.
- The final result is a tree with **decision nodes** and **leaf nodes**.
- A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested.
- Leaf node (e.g., Hours Played) represents a decision on the numerical target.
- The topmost decision node in a tree which corresponds to the best predictor called **root node**.
- Decision trees can handle both categorical and numerical data.

Decision Tree Regression



Decision Tree for Classification and Prediction





Decision Tree Regression

- The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking.
- The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with *Standard Deviation Reduction*.

Decision Tree Regression

Entropy

Entropy $H(S)$ is a measure of the amount of uncertainty in the (data) set S (i.e. entropy characterizes the (data) set S).

$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

Where,

- S – The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- C – Set of classes in S $C=\{ \text{yes}, \text{no} \}$
- $p(c)$ – The proportion of the number of elements in class c to the number of elements in set S

When $H(S) = 0$, the set S is perfectly classified (i.e. all elements in S are of the same class).

In ID3, entropy is calculated for each remaining attribute. The attribute with the **smallest** entropy is used to split the set S on this iteration. The higher the entropy, the higher the potential to improve the classification here.

Decision Tree Regression

Information gain

Information gain $IG(A)$ is the measure of the difference in entropy from before to after the set S is split on an attribute A . In other words, how much uncertainty in S was reduced after splitting set S on attribute A .

$$IG(A, S) = H(S) - \sum_{t \in T} p(t)H(t)$$

Where,

- $H(S)$ – Entropy of set S
- T – The subsets created from splitting set S by attribute A such that $S = \bigcup_{t \in T} t$
- $p(t)$ – The proportion of the number of elements in t to the number of elements in set S
- $H(t)$ – Entropy of subset t

In ID3, information gain can be calculated (instead of entropy) for each remaining attribute. The attribute with the **largest** information gain is used to split the set S on this iteration.

Decision Tree Regression

1. compute the entropy for data-set
2. for every attribute/feature:
 1. calculate entropy for all categorical values
 2. take average information entropy for the current attribute
 3. calculate gain for the current attribute
3. pick the highest gain attribute.
4. Repeat until we get the tree we desired.

Decision Tree Regression

Compute the entropy for the weather data set:



$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

$$C = \{\text{yes}, \text{no}\}$$

Out of 14 instances, 9 are classified as yes,
and 5 as no

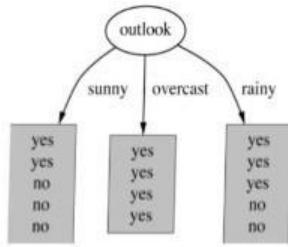
$$p_{\text{yes}} = -(9/14) * \log_2(9/14) = 0.41$$

$$p_{\text{no}} = -(5/14) * \log_2(5/14) = 0.53$$

$$H(S) = p_{\text{yes}} + p_{\text{no}} = 0.94$$

Decision Tree Regression

For every feature calculate the entropy and information gain



$$E(\text{Outlook}=\text{sunny}) = -\frac{2}{5} \log\left(\frac{2}{5}\right) - \frac{3}{5} \log\left(\frac{3}{5}\right) = 0.971$$

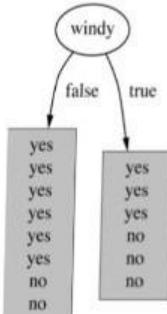
$$E(\text{Outlook}=\text{overcast}) = -1 \log(1) - 0 \log(0) = 0$$

$$E(\text{Outlook}=\text{rainy}) = -\frac{3}{5} \log\left(\frac{3}{5}\right) - \frac{2}{5} \log\left(\frac{2}{5}\right) = 0.971$$

Average Entropy information for Outlook

$$I(\text{Outlook}) = \frac{5}{14} * 0.971 + \frac{4}{14} * 0 + \frac{5}{14} * 0.971 = 0.693$$

$$\text{Gain}(\text{Outlook}) = E(S) - I(\text{outlook}) = 0.94 - 0.693 = 0.247$$



$$E(\text{Windy}=\text{false}) = -\frac{6}{8} \log\left(\frac{6}{8}\right) - \frac{2}{8} \log\left(\frac{2}{8}\right) = 0.811$$

$$E(\text{Windy}=\text{true}) = -\frac{3}{6} \log\left(\frac{3}{6}\right) - \frac{3}{6} \log\left(\frac{3}{6}\right) = 1$$

Average entropy information for Windy

$$I(\text{Windy}) = \frac{8}{14} * 0.811 + \frac{6}{14} * 1 = 0.892$$

$$\text{Gain}(\text{Windy}) = E(S) - I(\text{Windy}) = 0.94 - 0.892 = 0.048$$

$$H(S, \text{Outlook})$$

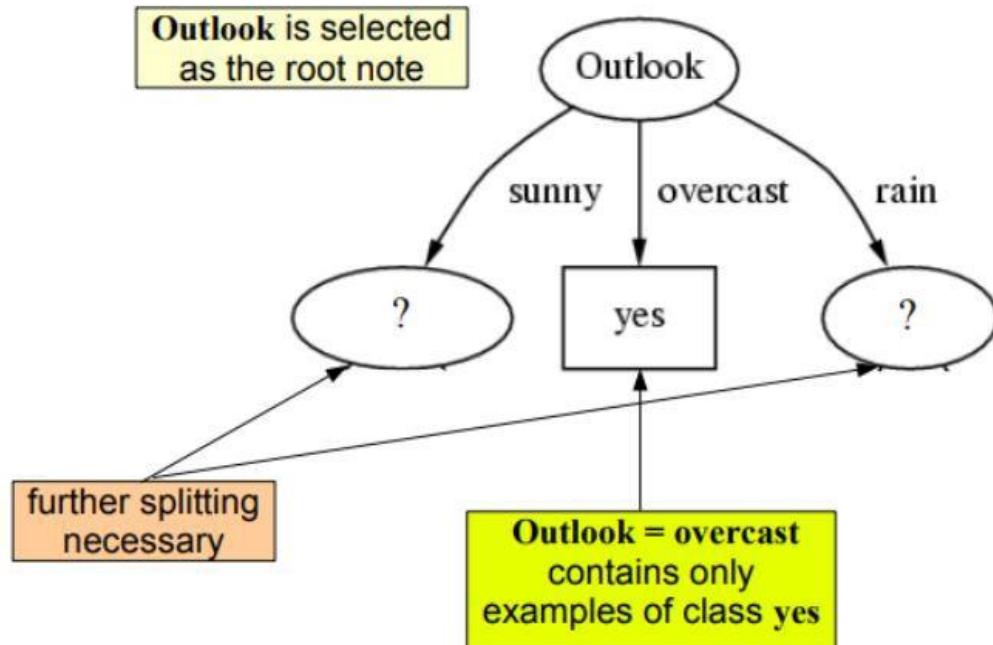
$$\sum_{t \in T} p(t) H(t)$$

$$IG(A, S) = H(S) - \sum_{t \in T} p(t) H(t)$$

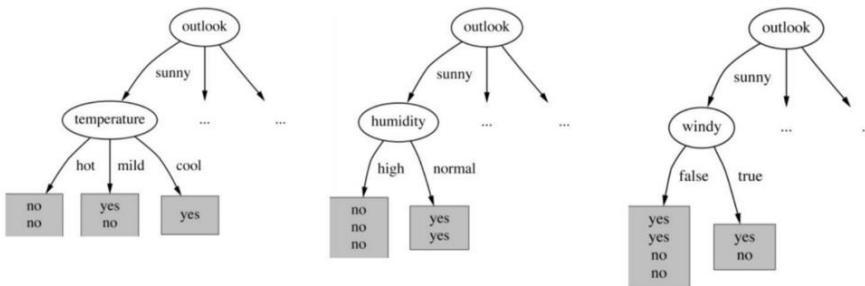
Pick the highest gain attribute.

| Outlook | | Temperature | |
|-------------------|-------|-------------------|-------|
| Info: | 0.693 | Info: | 0.911 |
| Gain: 0.940-0.693 | 0.247 | Gain: 0.940-0.911 | 0.029 |
| Humidity | | Windy | |
| Info: | 0.788 | Info: | 0.892 |
| Gain: 0.940-0.788 | 0.152 | Gain: 0.940-0.892 | 0.048 |

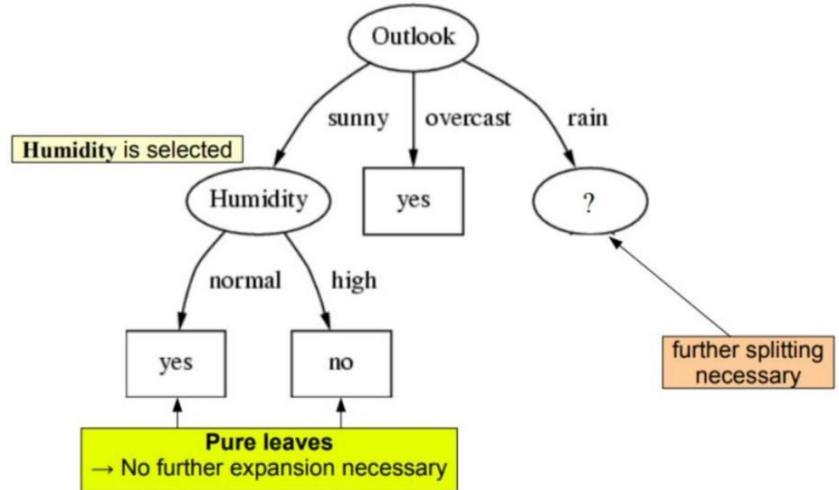
So our root node is Outlook.



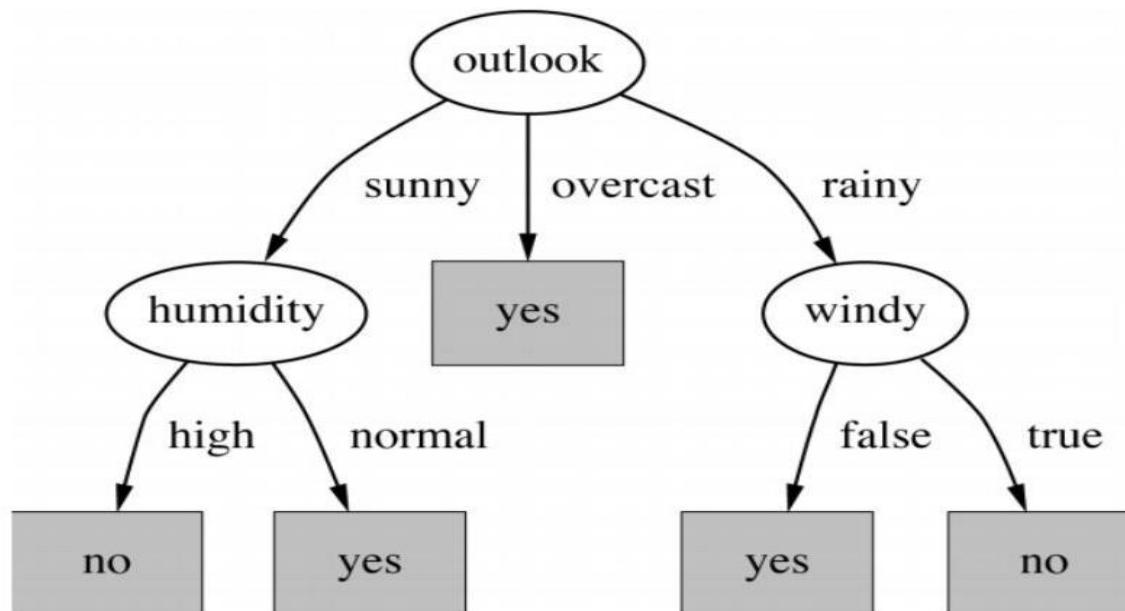
Repeat the same thing for sub-trees till we get the tree.



$$\begin{aligned}
 \text{Gain}(Temperature) &= 0.571 \text{ bits} \\
 \text{Gain}(Humidity) &= 0.971 \text{ bits} \\
 \text{Gain}(Windy) &= 0.020 \text{ bits}
 \end{aligned}
 \left. \right\} \text{Humidity is selected}$$



Final Decision Tree



Performance

- The Receiver Operating Characteristic (ROC) is a plot that can be used to determine the performance and robustness of a binary or multi-class classifier.
- The x-axis is the false positive rate (FPR) and the y-axis is the true positive rate (TPR).
- The ROC plot gives information about the true positive/negative rate and false positive/negative rate and something called the C-statistic or area under ROC curve (AUROC) for each class predicted by the classifier (there is one ROC for each class predicted by the classifier).

Performance

- The AUROC is defined as the probability that a randomly selected positive sample will have a higher prediction value than a randomly selected negative sample.
- *Interpretation of the AUC is easy: the higher the AUC, the better, with 0.50 indicating random performance and 1.00 denoting perfect performance.”*

GINI INDEX

GINI Index

$$Gini = \sum_{i \neq j} p(i)p(j)$$

i and j are levels of the target variable

GINI Index

- Example, banks and financial institutions grant credit facility after evaluating credit risk involved. Credit risk involved in credit decisions is evaluated using Credit Scorecard [Credit Score: What is it and how is it developed?]. Also, there are a few additional decisions involved in credit underwriting [Credit Underwriting: Minimize credit risk losses using Data Science and Analytics].
- Decision Tree: Non Technical Explanation
- The last 2 years of customer performance on meeting credit obligations is available with us. We want to understand the variable(s) explains high risk of customers who defaulted on a credit facility given to them.
- The sample has 24 customers. And for making it simple, only customer age and gender are considered. Age is a continuous variable and Gender is nominal variable.
- Input sample has 12 customers who have defaulted on the credit facility. So, default rate is 50%.
- We have an example in which input node, parent node, has equal number of Target variable values- “Yes” and “No”. Overall number of observations are 24.

GINI Index

Gender variable is considered to split the node. Gini Split value is calculated as below.

| Gender | Target Value | | Total |
|--------|--------------|-----|-------|
| | No | Yes | |
| Female | 6 | 2 | 8 |
| Male | 6 | 10 | 16 |
| Total | 12 | 12 | 24 |

Gini index for this node will be

$$= 1 - (1/2)^2 - (1/2)^2 \rightarrow 12/24 = 1/2$$

$$= 1 - 0.25 - 0.25$$

$$= 0.5$$

Now we want to split the code based on Gender Variable. After the split we will have following summary.

Now, let's calculate GINI index of the split using Gender variable.

$$\text{GINI}(s,t) = \text{GINI}(t) - P_L \text{GINI}(t_L) - P_R \text{GINI}(t_R)$$

$$\text{GINI}(t_L) = 1 - (6/8)^2 - (2/8)^2$$

$$= 1 - 0.5625 - 0.0625$$

$$= 0.375$$

GINI Index

- GINI (t_R) $= 1 - (6/16)^2 - (10/16)^2$
- $= 1 - 0.140625 - 0.390625$
- $= 0.469$
- GINI (s,t) $= 0.5 - (8/24)*0.375 - (16/24)*0.469$
- $= 0.5 - 0.125 - 0.313$
- $= 0.0625$
- **The attribute value that provides the smallest SPLIT Gini (T) is chosen to split the node.

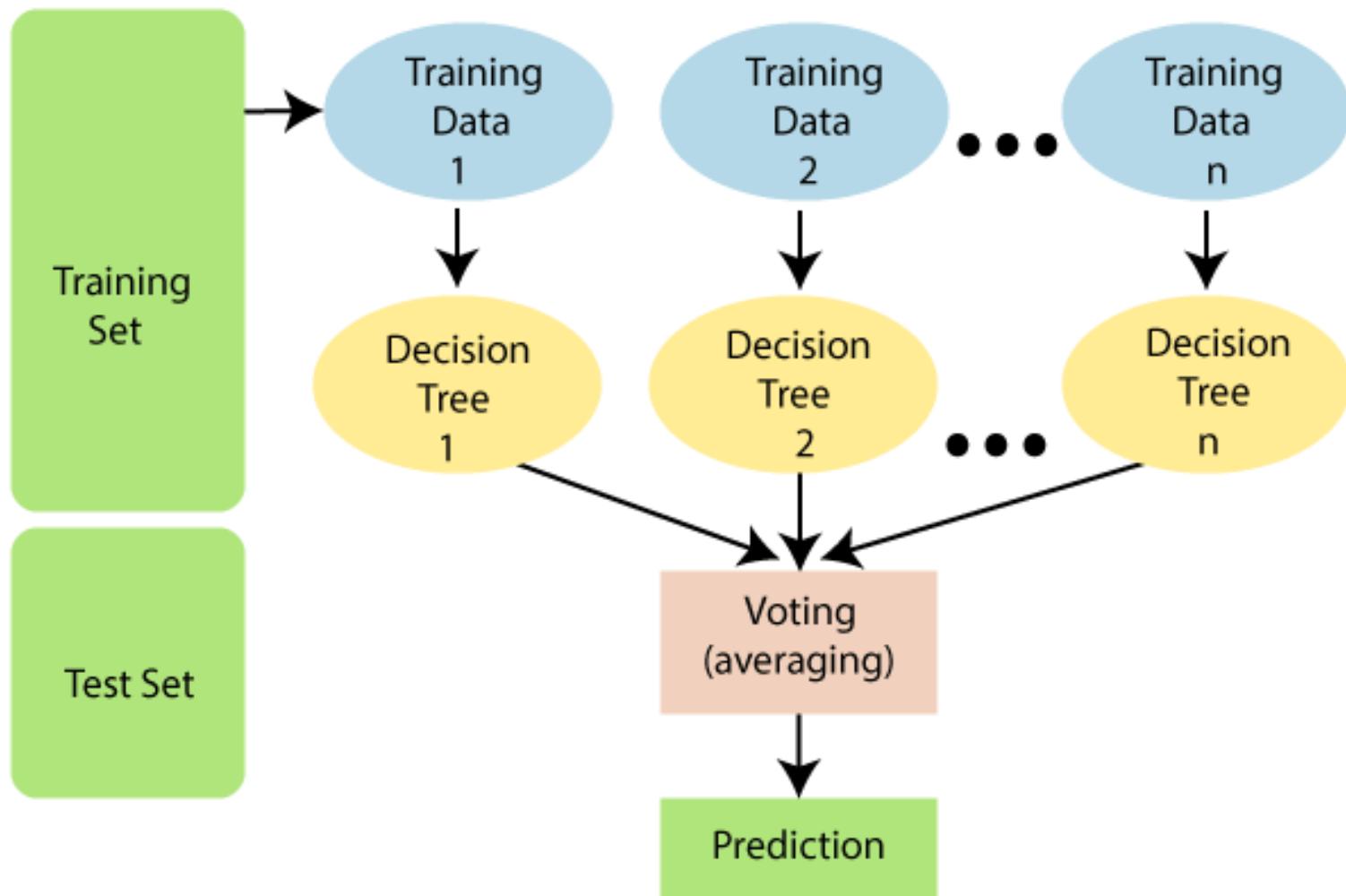
Random Forest Algorithm

- Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique.
- It can be used for both Classification and Regression problems in ML.
- It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

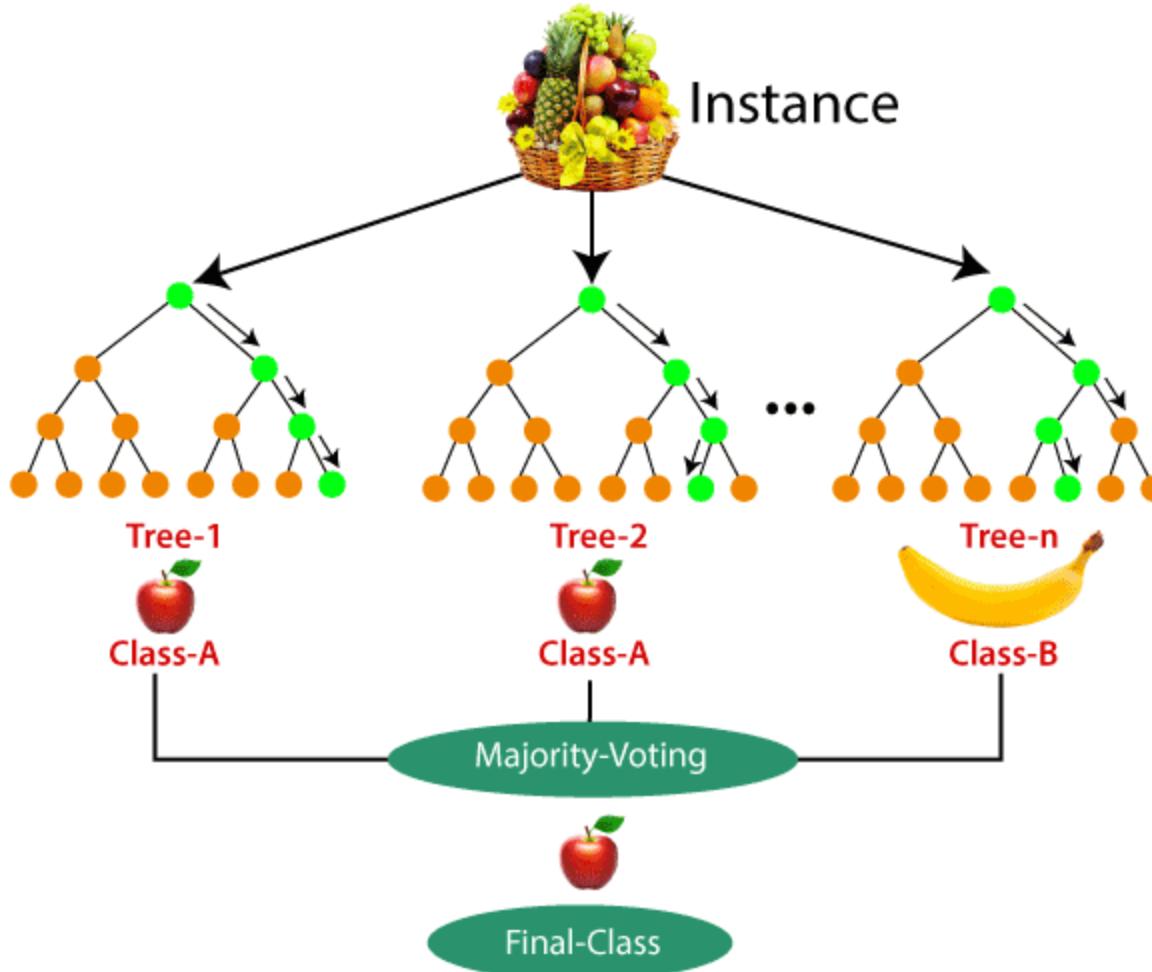
Random Forest Algorithm

- As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."
- Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

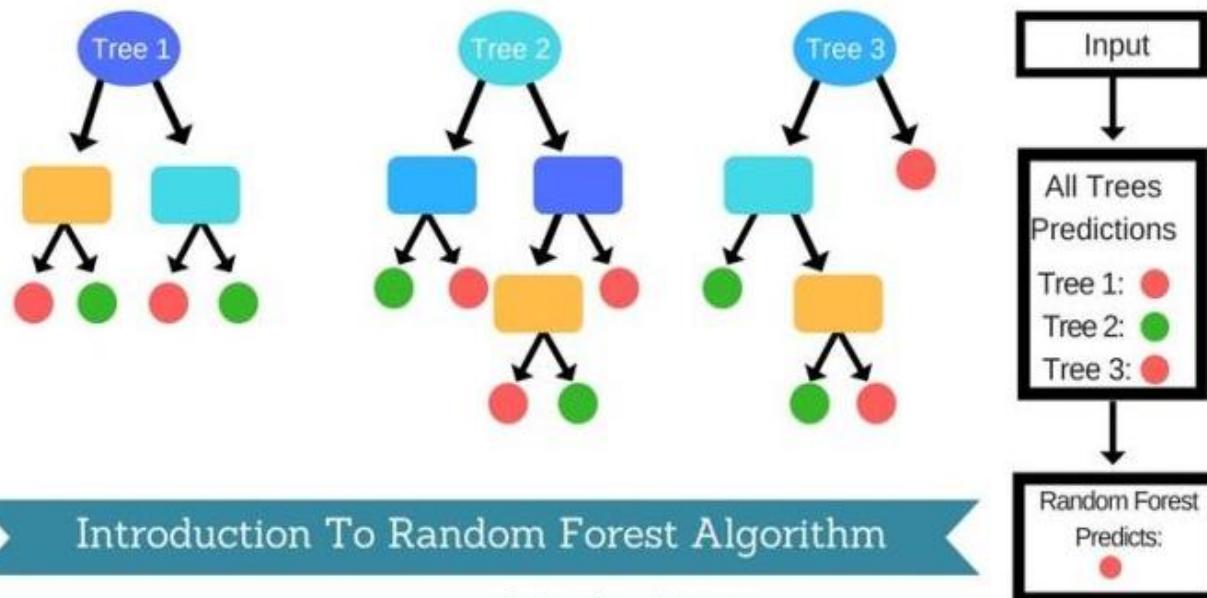
Random Forest Algorithm



Random Forest Algorithm



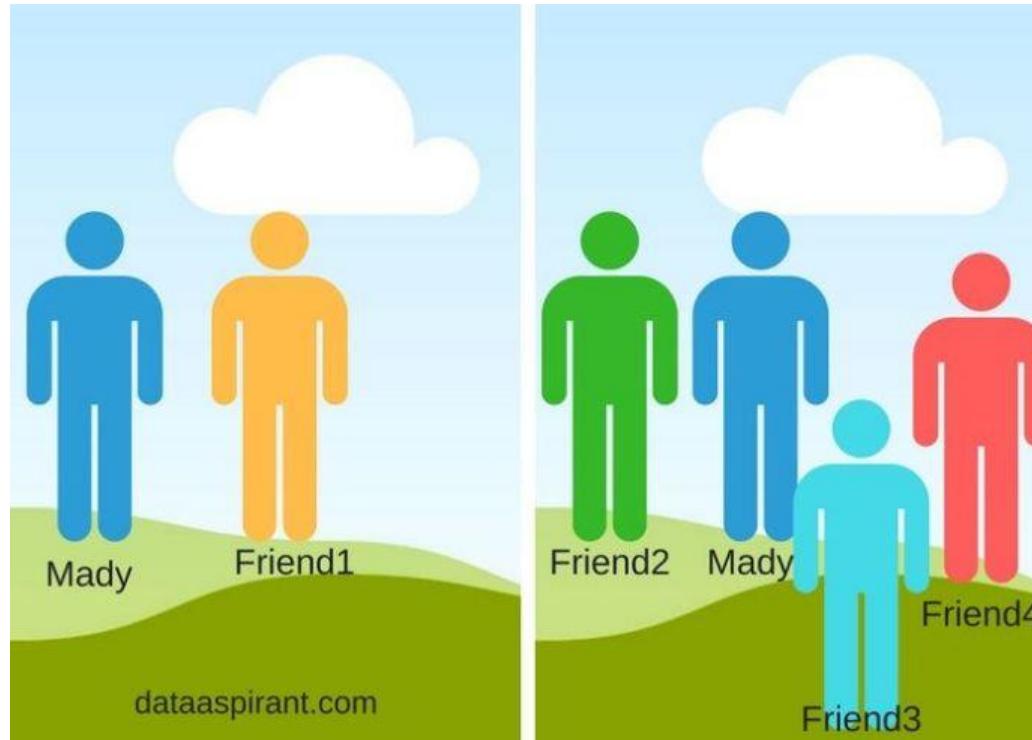
Random Forest Algorithm



Random Forest Algorithm

- Random forest algorithm is a supervised classification algorithm. As the name suggest, this algorithm creates the forest with a number of trees.
- In general, the **more trees in the forest** the more robust the forest looks like. In the same way in the random forest classifier, the **higher the number** of trees in the forest gives **the high accuracy** results.

Random forest algorithm real life example



Random forest algorithm real life example

- Suppose **Mady** somehow got 2 weeks leave from his office. He wants to spend his 2 weeks by traveling to the different place. He also wants to go to the place he may like.
- So he decided to ask his **best friend** about the places he may like. Then his friend started asking about his past trips. It's just like his best friend will ask, You have been visited the X place did you like it?
- Based on the answers which are given by Mady, his best start recommending the place Mady may like. Here his best formed the decision tree with the answer given by Mady.
- As his best friend may recommend his best place to Mady as a friend. The model will be **biased** with the closeness of their friendship. So he decided to ask few more friends to recommend the best place he may like.
- Now his friends asked some **random questions** and each one recommended one place to Mady. Now Mady considered the place which is **high votes** from his friends as the final place to visit.
- In the above Mady trip planning, two main interesting algorithms decision tree algorithm and random forest algorithm used. I hope you find it already. Anyhow, I would like to highlight it again.

Decision Tree

- To recommend the best place to Mady, his best friend asked some questions. Based on the answers given by mady, he recommended a place. This is **decision tree algorithm** approach.
- Mady friend used the answers given by mady to create rules. Later he used the created rules to recommend the best place which mady will like. These rules could be, mady like a place with lots of tree or waterfalls ..etc
- In the above approach mady best friend is the decision tree. The vote (recommended place) is the leaf of the decision tree (Target class). The target is finalized by a single person, In a technical way of saying, using an only single decision tree.

The data and the goal

- **Data:** A set of data records (also called examples, instances or cases) described by
 - k attributes: A_1, A_2, \dots, A_k
 - a class: Each example is labelled with a pre-defined class.
- **Goal:** To learn a classification model from the data that can be used to predict the classes of new (future, or test) cases/instances.

The data and the goal

An example: data (loan application)

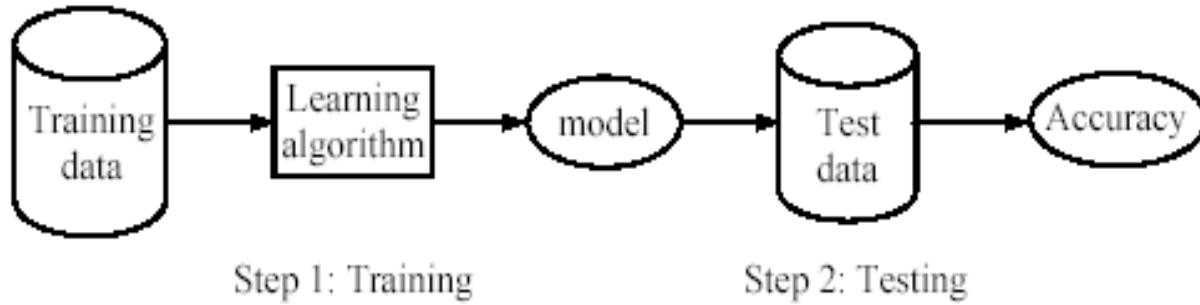
Approved or not

| ID | Age | Has_Job | Own_House | Credit_Rating | Class |
|----|--------|---------|-----------|---------------|-------|
| 1 | young | false | false | fair | No |
| 2 | young | false | false | good | No |
| 3 | young | true | false | good | Yes |
| 4 | young | true | true | fair | Yes |
| 5 | young | false | false | fair | No |
| 6 | middle | false | false | fair | No |
| 7 | middle | false | false | good | No |
| 8 | middle | true | true | good | Yes |
| 9 | middle | false | true | excellent | Yes |
| 10 | middle | false | true | excellent | Yes |
| 11 | old | false | true | excellent | Yes |
| 12 | old | false | true | good | Yes |
| 13 | old | true | false | good | Yes |
| 14 | old | true | false | excellent | Yes |
| 15 | old | false | false | fair | No |

Supervised learning process: two steps

- **Learning (training)**: Learn a model using the **training data**
- **Testing**: Test the model using **unseen test data** to assess the model accuracy

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



What do we mean by learning?

- Given
 - a data set D ,
 - a task T , and
 - a performance measure M ,
- a computer system is said to **learn** from D to perform the task T if after learning the system's performance on T improves as measured by M .
- In other words, the learned model helps the system to perform T better as compared to no learning.

An example

- **Data:** Loan application data
- **Task:** Predict whether a loan should be approved or not.
- **Performance measure:** accuracy.

No learning: classify all future applications (test data) to the majority class (i.e., Yes):

$$\text{Accuracy} = 9/15 = 60\%.$$

- We can do better than 60% with learning.

Fundamental assumption of learning

Assumption: The distribution of training examples is identical to the distribution of test examples (including future unseen examples).

- In practice, this assumption is often violated to certain degree.
- Strong violations will clearly result in poor classification accuracy.
- To achieve good accuracy on the test data, training examples must be sufficiently representative of the test data.

Introduction

- Decision tree learning is one of the most widely used techniques for classification.
 - Its classification accuracy is competitive with other methods, and
 - it is very efficient.
- The classification model is a tree, called **decision tree**.
- **C4.5** by Ross Quinlan is perhaps the best known system. It can be downloaded from the Web.

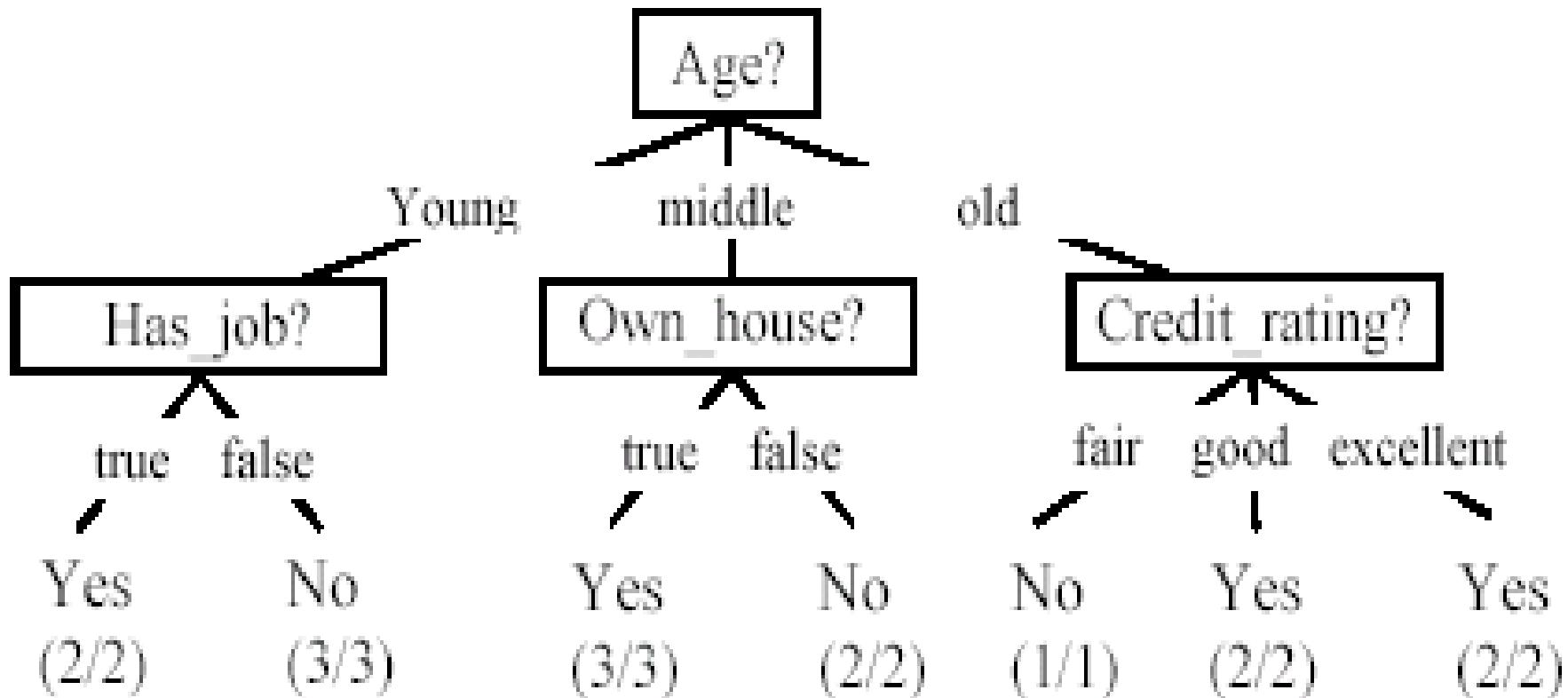
The loan data (reproduced)

Approved or not

| ID | Age | Has_Job | Own_House | Credit_Rating | Class |
|----|------------|----------------|------------------|----------------------|--------------|
| 1 | young | false | false | fair | No |
| 2 | young | false | false | good | No |
| 3 | young | true | false | good | Yes |
| 4 | young | true | true | fair | Yes |
| 5 | young | false | false | fair | No |
| 6 | middle | false | false | fair | No |
| 7 | middle | false | false | good | No |
| 8 | middle | true | true | good | Yes |
| 9 | middle | false | true | excellent | Yes |
| 10 | middle | false | true | excellent | Yes |
| 11 | old | false | true | excellent | Yes |
| 12 | old | false | true | good | Yes |
| 13 | old | true | false | good | Yes |
| 14 | old | true | false | excellent | Yes |
| 15 | old | false | false | fair | No |

A decision tree from the loan data

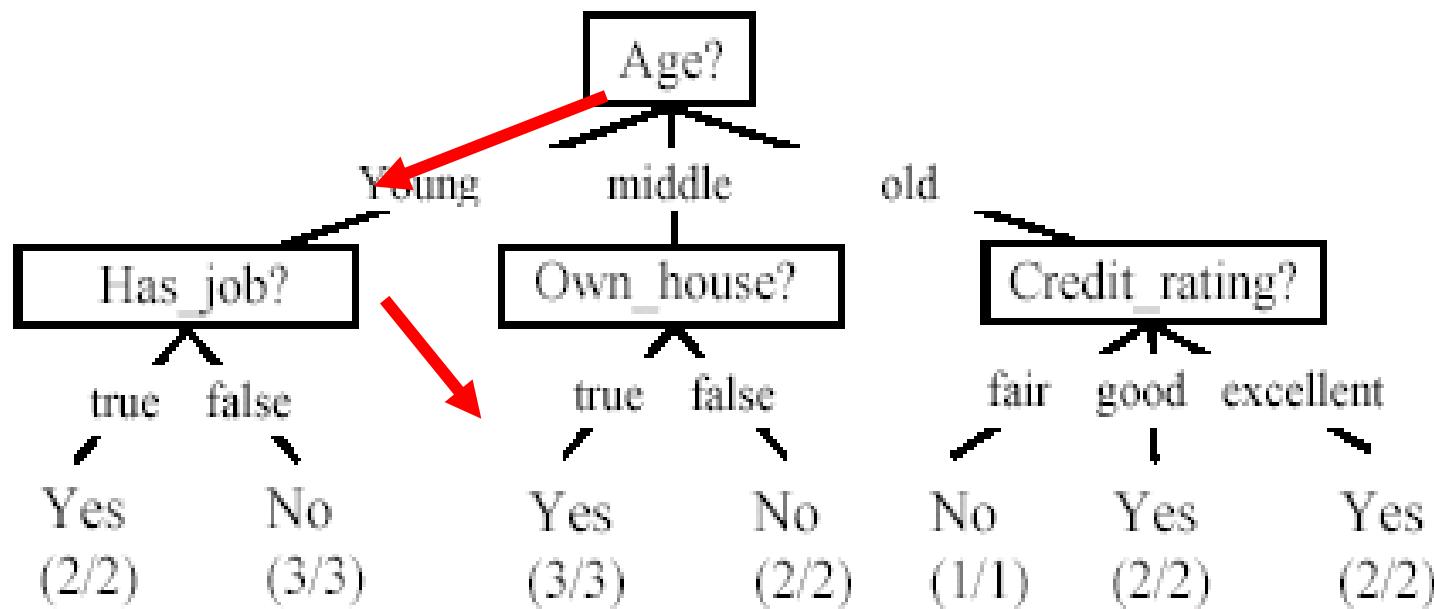
- Decision nodes and leaf nodes (classes)



Use the decision tree

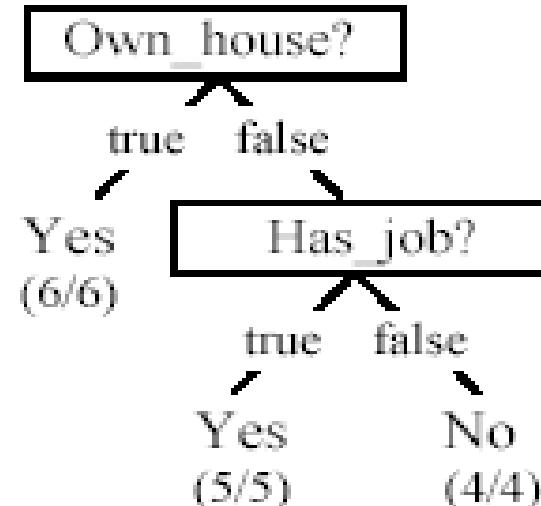
| Age | Has_Job | Own_house | Credit-Rating | Class |
|-------|---------|-----------|---------------|-------|
| young | false | false | good | ? |

No



From a decision tree to a set of rules

- A decision tree can be converted to a set of rules
- Each path from the root to a leaf is a rule.



$\text{Own_house} = \text{true} \rightarrow \text{Class} = \text{Yes}$ [sup=6/15, conf=6/6]
 $\text{Own_house} = \text{false}, \text{Has_job} = \text{true} \rightarrow \text{Class} = \text{Yes}$ [sup=5/15, conf=5/5]
 $\text{Own_house} = \text{false}, \text{Has_job} = \text{false} \rightarrow \text{Class} = \text{No}$ [sup=4/15, conf=4/4]

Algorithm for decision tree learning



- Basic algorithm (a greedy **divide-and-conquer** algorithm)
 - Assume attributes are categorical now (continuous attributes can be handled too)
 - Tree is constructed in a **top-down recursive manner**
 - At start, all the training examples are at the root
 - Examples are partitioned recursively based on selected attributes
 - Attributes are selected on the basis of an impurity function (e.g., **information gain**)
- Conditions for stopping partitioning
 - All examples for a given node belong to the same class
 - There are no remaining attributes for further partitioning – majority class is the leaf
 - There are no examples left

Decision tree learning algorithm



```
. Algorithm decisionTree( $D, A, T$ )
1   if  $D$  contains only training examples of the same class  $c_j \in C$  then
2       make  $T$  a leaf node labeled with class  $c_j$ ;
3   elseif  $A = \emptyset$  then
4       make  $T$  a leaf node labeled with  $c_j$ , which is the most frequent class in  $D$ 
5   else //  $D$  contains examples belonging to a mixture of classes. We select a single
6       // attribute to partition  $D$  into subsets so that each subset is purer
7        $p_0 = \text{impurityEval-1}(D)$ ;
8       for each attribute  $A_i \in \{A_1, A_2, \dots, A_k\}$  do
9            $p_i = \text{impurityEval-2}(A_i, D)$ 
10      end
11      Select  $A_g \in \{A_1, A_2, \dots, A_k\}$  that gives the biggest impurity reduction,
12          computed using  $p_0 - p_g$ 
13      if  $p_0 - p_g < \text{threshold}$  then //  $A_g$  does not significantly reduce impurity  $p_0$ 
14          make  $T$  a leaf node labeled with  $c_j$ , the most frequent class in  $D$ .
15      else //  $A_g$  is able to reduce impurity  $p_0$ 
16          Make  $T$  a decision node on  $A_g$ ;
17          Let the possible values of  $A_g$  be  $v_1, v_2, \dots, v_m$ . Partition  $D$  into  $m$ 
18          disjoint subsets  $D_1, D_2, \dots, D_m$  based on the  $m$  values of  $A_g$ .
19          for each  $D_j$  in  $\{D_1, D_2, \dots, D_m\}$  do
20              if  $D_j \neq \emptyset$  then
21                  create a branch (edge) node  $T_j$  for  $v_j$  as a child node of  $T$ ;
22                  decisionTree( $D_j, A-\{A_g\}, T_j$ ) //  $A_g$  is removed
23              end
24          end
25      end
26  end
```

Choose an attribute to partition data

- The *key* to building a decision tree - which attribute to choose in order to branch.
- The objective is to reduce impurity or uncertainty in data as much as possible.
 - A subset of data is *pure* if all instances belong to the same class.
- The *heuristic* in C4.5 is to choose the attribute with the maximum **Information Gain** or **Gain Ratio** based on information theory.

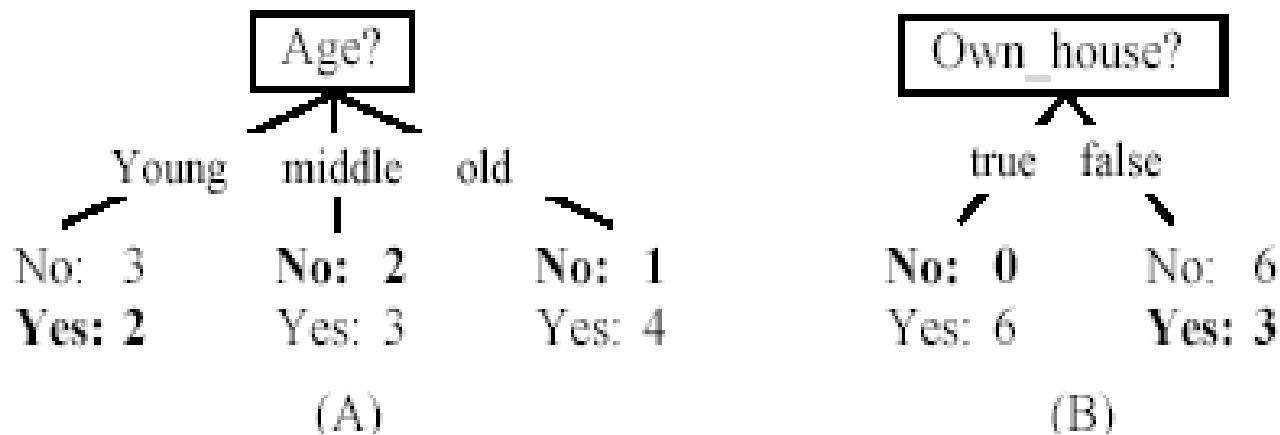
The loan data (reproduced)



Approved or not

| ID | Age | Has_Job | Own_House | Credit_Rating | Class |
|----|--------|---------|-----------|---------------|-------|
| 1 | young | false | false | fair | No |
| 2 | young | false | false | good | No |
| 3 | young | true | false | good | Yes |
| 4 | young | true | true | fair | Yes |
| 5 | young | false | false | fair | No |
| 6 | middle | false | false | fair | No |
| 7 | middle | false | false | good | No |
| 8 | middle | true | true | good | Yes |
| 9 | middle | false | true | excellent | Yes |
| 10 | middle | false | true | excellent | Yes |
| 11 | old | false | true | excellent | Yes |
| 12 | old | false | true | good | Yes |
| 13 | old | true | false | good | Yes |
| 14 | old | true | false | excellent | Yes |
| 15 | old | false | false | fair | No |

Two possible roots, which is better?



- Fig. (B) seems to be better.

Information theory

- **Information theory** provides a mathematical basis for measuring the information content.
- To understand the notion of information, think about it as providing the answer to a question, for example, whether a coin will come up heads.
 - If one already has a good guess about the answer, then the actual answer is less informative.
 - If one already knows that the coin is rigged so that it will come with heads with probability 0.99, then a message (advanced information) about the actual outcome of a flip is worth less than it would be for a honest coin (50-50).

Information theory (cont ...)

- For a fair (honest) coin, you have no information, and you are willing to pay more (say in terms of \$) for advanced information - less you know, the more valuable the information.
- **Information theory** uses this same intuition, but instead of measuring the value for information in dollars, it measures information contents in **bits**.
- One bit of information is enough to answer a yes/no question about which one has no idea, such as the flip of a fair coin

Information theory: Entropy measure

- The entropy formula,

$$\text{entropy}(D) = -\sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

$$\sum_{j=1}^{|C|} \Pr(c_j) = 1,$$

- $\Pr(c_j)$ is the probability of class c_j in data set D
- We use entropy as a **measure of impurity or disorder** of data set D . (Or, a measure of information in a tree)

Entropy measure: let us get a feeling

1. The data set D has 50% positive examples ($\Pr(\text{positive}) = 0.5$) and 50% negative examples ($\Pr(\text{negative}) = 0.5$).

$$\text{entropy}(D) = -0.5 \times \log_2 0.5 - 0.5 \times \log_2 0.5 = 1$$

2. The data set D has 20% positive examples ($\Pr(\text{positive}) = 0.2$) and 80% negative examples ($\Pr(\text{negative}) = 0.8$).

$$\text{entropy}(D) = -0.2 \times \log_2 0.2 - 0.8 \times \log_2 0.8 = 0.722$$

3. The data set D has 100% positive examples ($\Pr(\text{positive}) = 1$) and no negative examples, ($\Pr(\text{negative}) = 0$).

$$\text{entropy}(D) = -1 \times \log_2 1 - 0 \times \log_2 0 = 0$$

- As the data become purer and purer, the entropy value becomes smaller and smaller. This is useful to us!

Information gain

- Given a set of examples D , we first compute its entropy:

$$\text{entropy}(D) = - \sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

- If we make attribute A_i , with v values, the root of the current tree, this will partition D into v subsets D_1, D_2, \dots, D_v . The expected entropy if A_i is used as the current root:

$$\text{entropy}_{A_i}(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times \text{entropy}(D_j)$$

Information gain (cont ...)

- Information gained by selecting attribute A_i to branch or to partition the data is

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

- We choose the attribute with the highest gain to branch/split the current tree.

An example

Tested all the calculations

$$\text{entropy}(D) = \frac{6}{15} \times \log_2 \frac{6}{15} + \frac{9}{15} \times \log_2 \frac{9}{15} = 0.971$$

Entropy age → refer manual cal → $-2/5 \log(2/5) - 3/5 \log(3/5)$

$$\begin{aligned}\text{entropy}_{\text{Own_house}}(D) &= \frac{6}{15} \times \text{entropy}(D_1) + \frac{9}{15} \times \text{entropy}(D_2) \\ &= \frac{6}{15} \times 0 + \frac{9}{15} \times 0.918 \\ &= 0.551\end{aligned}$$

$$\begin{aligned}\text{entropy}_{\text{Age}}(D) &= \frac{5}{15} \times \text{entropy}(D_1) + \frac{5}{15} \times \text{entropy}(D_2) + \frac{5}{15} \times \text{entropy}(D_3) \\ &= \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.722 \\ &= 0.888\end{aligned}$$

- Own_house is the best choice for the root.

| ID | Age | Has_Job | Own_House | Credit_Rating | Class |
|----|--------|---------|-----------|---------------|-------|
| 1 | young | false | false | fair | No |
| 2 | young | false | false | excellent | No |
| 3 | young | true | false | good | Yes |
| 4 | young | true | true | good | Yes |
| 5 | young | false | false | fair | No |
| 6 | middle | false | false | fair | No |
| 7 | middle | false | false | good | No |
| 8 | middle | true | true | good | Yes |
| 9 | middle | false | true | excellent | Yes |
| 10 | middle | false | true | excellent | Yes |
| 11 | old | false | true | excellent | Yes |
| 12 | old | false | true | good | Yes |
| 13 | old | true | false | good | Yes |
| 14 | old | true | false | excellent | Yes |
| 15 | old | false | false | fair | No |

| Age | Yes | No | entropy(Di) |
|--------|-----|----|-------------|
| young | 2 | 3 | 0.971 |
| middle | 3 | 2 | 0.971 |
| old | 4 | 1 | 0.722 |

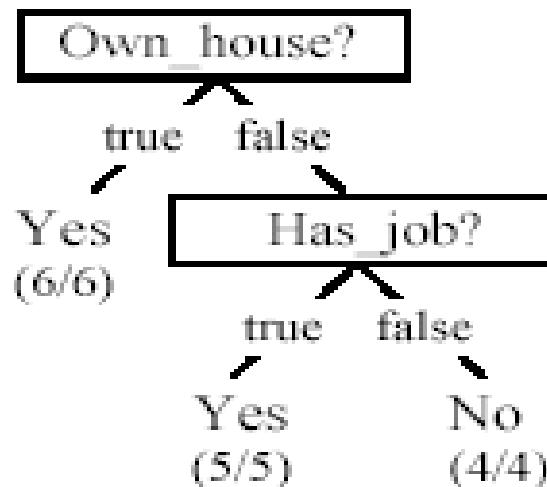
$$\text{gain}(D, \text{Age}) = 0.971 - 0.888 = 0.083$$

$$\text{gain}(D, \text{Own_house}) = 0.971 - 0.551 = 0.420$$

$$\text{gain}(D, \text{Has_Job}) = 0.971 - 0.647 = 0.324$$

$$\text{gain}(D, \text{Credit_Rating}) = 0.971 - 0.608 = 0.363$$

We build the final tree

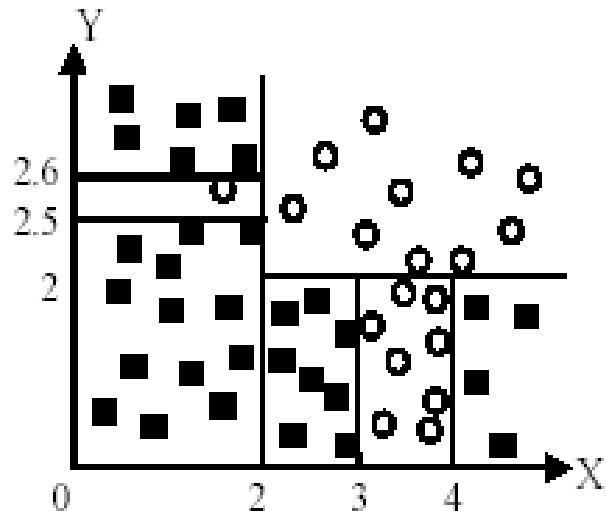


- We can use information gain ratio to evaluate the impurity as well (see the handout)

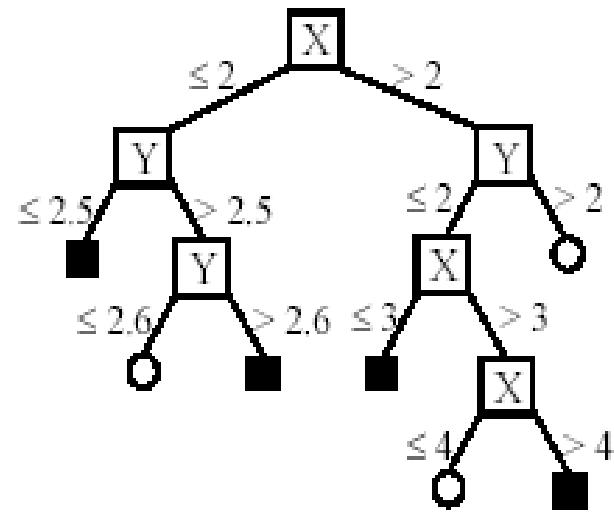
Handling continuous attributes

- Handle continuous attribute by splitting into two intervals (can be more) at each node.
- How to find the best threshold to divide?
 - Use information gain or gain ratio again
 - Sort all the values of an continuous attribute in increasing order $\{v_1, v_2, \dots, v_r\}$,
 - One possible threshold between two adjacent values v_i and v_{i+1} . Try all possible thresholds and find the one that maximizes the gain (or gain ratio).

An example in a continuous space



(A) A partition of the data space

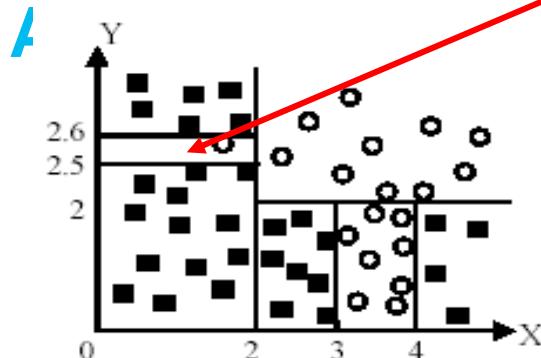


(B). The decision tree

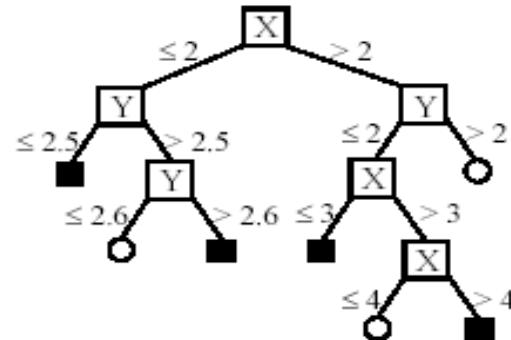
Avoid overfitting in classification

- **Overfitting:** A tree may overfit the training data
 - Good accuracy on training data but poor on test data
 - Symptoms: tree too deep and too many branches, some may reflect anomalies due to noise or outliers
- Two approaches to avoid overfitting
 - **Pre-pruning:** Halt tree construction early
 - Difficult to decide because we do not know what may happen subsequently if we keep growing the tree.
 - **Post-pruning:** Remove branches or sub-trees from a “fully grown” tree.
 - This method is commonly used. C4.5 uses a statistical method to estimates the errors at each node for pruning.
 - A validation set may be used for pruning as well.

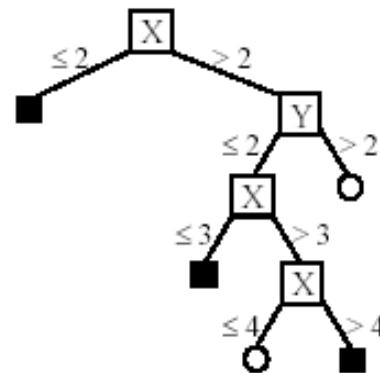
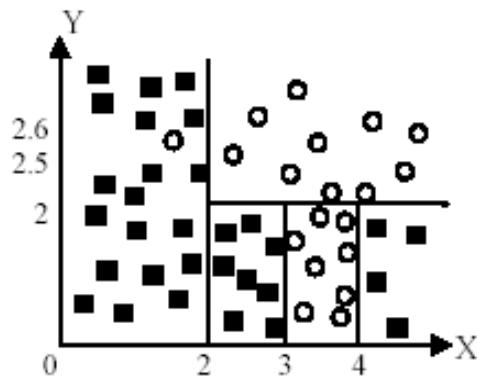
Likely to overfit the data



(A) A partition of the data space



(B). The decision tree



Other issues in decision tree learning

- From tree to rules, and rule pruning
- Handling of miss values
- Handing skewed distributions
- Handling attributes and classes with different costs.
- Attribute construction etc.

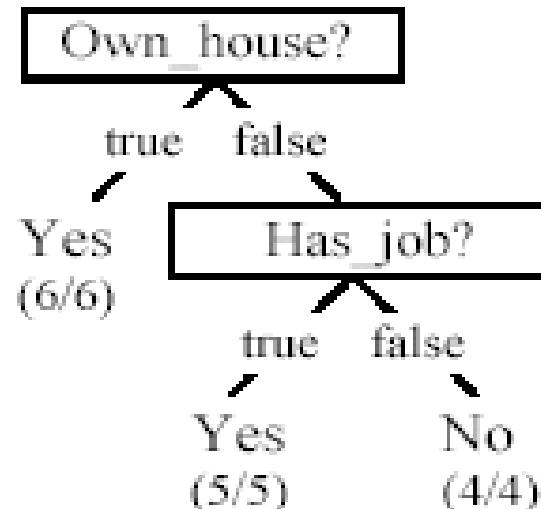
An example: the learning task

- Learn a classification model from the data
- Use the model to classify future loan applications into
 - Yes (approved) and
 - No (not approved)
- What is the class for following case/instance?

| Age | Has_Job | Own_house | Credit-Rating | <u>Class</u> |
|-------|---------|-----------|---------------|--------------|
| young | false | false | good | ? |

Is the decision tree unique?

- No. Here is a simpler tree.
- We want smaller tree and accurate tree.
 - Easy to understand and perform better.
- Finding the best tree is NP-hard.
- All current tree building algorithms are heuristic algorithms



Random Forest Algorithm:



- In the other case when mady asked his friends to recommend the best place to visit. Each friend asked him different questions and come up their recommend a place to visit. Later mady consider all the recommendations and calculated the votes. Votes basically is to pick the popular place from the recommend places from all his friends.
- Mady will consider each recommended place and if the same place recommended by some other place he will increase the count. At the end the high count place where mady will go.
- In this case, the recommended place (**Target Prediction**) is considered by many friends. Each friend is the tree and the combined all friends will form the forest. This forest is the random forest. As each friend asked random questions to recommend the best place visit.

How Random Forest Algorithm Works

| f11 | f12 | f13 | f14 | f15 | t1 |
|-----|-----|-----|-----|-----|----|
| f21 | f22 | f23 | f24 | f25 | t2 |
| f31 | f32 | f33 | f34 | f35 | t3 |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| fm1 | fm2 | fm3 | fm4 | fm5 | tm |

Dataset

| f11 | f12 | f13 | f14 | f15 | t1 |
|-----|-----|-----|-----|-----|----|
| f81 | f82 | f83 | f84 | f85 | t8 |
| f71 | f72 | f73 | f74 | f75 | t7 |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| fj1 | fj2 | fj3 | fj4 | fj5 | tj |

Random Dataset
for Tree-01

| f21 | f22 | f23 | f24 | f25 | t2 |
|-----|-----|-----|-----|-----|----|
| f51 | f52 | f53 | f54 | f55 | t5 |
| f31 | f32 | f33 | f34 | f35 | t3 |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| fm1 | fm2 | fm3 | fm4 | fm5 | tm |

Random Dataset
for Tree-02

| f31 | f32 | f33 | f34 | f35 | t3 |
|-----|-----|-----|-----|-----|----|
| f61 | f62 | f63 | f64 | f65 | t6 |
| f91 | f92 | f73 | f94 | f95 | t9 |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| fk1 | fk2 | fk3 | fk4 | fk5 | tk |

Random Dataset
for Tree-03

dataaspirant.com

Random Forest pseudocode:



- Randomly select “k” features from total “m” features.
- Where $k \ll m$
- Among the “k” features, calculate the node “d” using the best split point.
- Split the node into daughter nodes using the best split.
- Repeat 1 to 3 steps until “l” number of nodes has been reached.
- Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

Random Forest pseudocode:



- To perform prediction using the trained random forest algorithm uses the below pseudocode.
- Takes the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target)
- Calculate the votes for each predicted target.
- Consider the high voted predicted target as the final prediction from the random forest algorithm.

Random forest algorithm applications



- In the banking sector, random forest algorithm widely used in two main application. These are for finding the loyal customer and finding the fraud customers.
- The loyal customer means not the customer who pays well, but also the customer whom can take the huge amount as loan and pays the loan interest properly to the bank.
- As the growth of the bank purely depends on the loyal customers. The bank customers data highly analyzed to find the pattern for the loyal customer based the customer details.
- In the same way, there is need to identify the customer who are not profitable for the bank, like taking the loan and not paying the loan interest properly or find the outlier customers.
- If the bank can identify these kind of customer before giving the loan the customer.
- Bank will get a chance to not approve the loan to these kinds of customers. In this case, also random forest algorithm is used to identify the customers who are not profitable for the bank.

Random Forest Applications

2. Medicine

- In medicine field, random forest algorithm is used identify the correct combination of the components to validate the medicine. Random forest algorithm also helpful for identifying the disease by analyzing the patient's medical records.

3. Stock Market

- In the stock market, random forest algorithm used to identify the stock behavior as well as the expected loss or profit by purchasing the particular stock.

E-Commerce

- In e-commerce, the random forest used only in the small segment of the recommendation engine for identifying the likely hood of customer liking the recommend products base on the similar kinds of customers.
- Running random forest algorithm on very large dataset requires **high-end GPU** systems. If you are not having any GPU system.
- You can always run the machine learning models in cloud hosted desktop. You can use cloud desktop online platform to run high-end machine learning models from sitting any corner of the world.

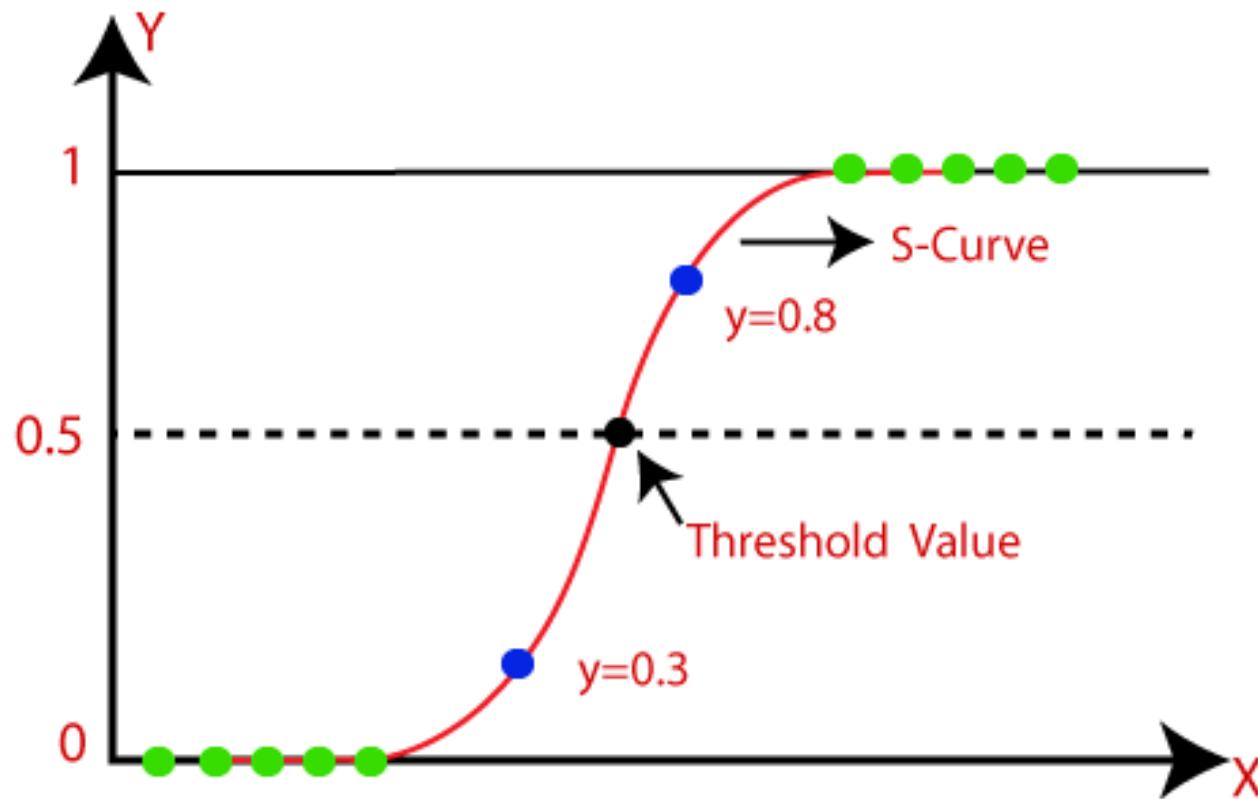
Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique.
- It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable.
- Therefore the outcome must be a categorical or discrete value.
- It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to the Linear Regression except that how they are used.
- Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

Logistic Regression

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

Logistic Regression



Logistic Function

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Logistic Function

- Assumptions for Logistic Regression:
 - The dependent variable must be categorical in nature.
 - The independent variable should not have multi-collinearity.

Logistic Function

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by $(1-y)$:

$$\frac{y}{1-y}; \text{ 0 for } y=0, \text{ and infinity for } y=1$$

- But we need range between $-[\infty]$ to $+[\infty]$, then take logarithm of the equation it will become:

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

Logistic Regression

- On the basis of the categories, **Logistic Regression can be classified into three types:**
- Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Logistic Regression

- Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome.
- The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).
- In logistic regression, the dependent variable is binary or dichotomous, i.e. it only contains data coded as 1 (TRUE, success, etc.) or 0 (FALSE, failure, etc.).

Logistic Regression

- The goal of logistic regression is to find the best fitting (yet biologically reasonable) model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables.

Logistic Regression

- Logistic regression generates the coefficients (and its standard errors and significance levels) of a formula to predict a logit transformation of the probability of presence of the characteristic of interest:

$$\text{logit}(p) = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_3 + \dots + b_k X_k$$

where p is the probability of presence of the characteristic of interest. The logit transformation is defined as the logged odds:

$$\text{odds} = \frac{p}{1-p} = \frac{\text{probability of presence of characteristic}}{\text{probability of absence of characteristic}}$$

and

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right)$$

Rather than choosing parameters that minimize the sum of squared errors (like in ordinary regression), estimation in logistic regression chooses parameters that maximize the likelihood of observing the sample values.

Logistic Regression



- Logistic regression predicts the probability of an outcome that can only have two values (i.e. a dichotomy).
- The prediction is based on the use of one or several predictors (numerical and categorical).
- A linear regression is not appropriate for predicting the value of a binary variable for two reasons:
- A linear regression will predict values outside the acceptable range (e.g. predicting probabilities outside the range 0 to 1)
- Since the dichotomous experiments can only have one of two possible values for each experiment, the residuals will not be normally distributed about the predicted line.

Logistic Regression



- On the other hand, a logistic regression produces a logistic curve, which is limited to values between 0 and 1.
- Logistic regression is similar to a linear regression, but the curve is constructed using the natural logarithm of the “odds” of the target variable, rather than the probability.
- Moreover, the predictors do not have to be normally distributed or have equal variance in each group.

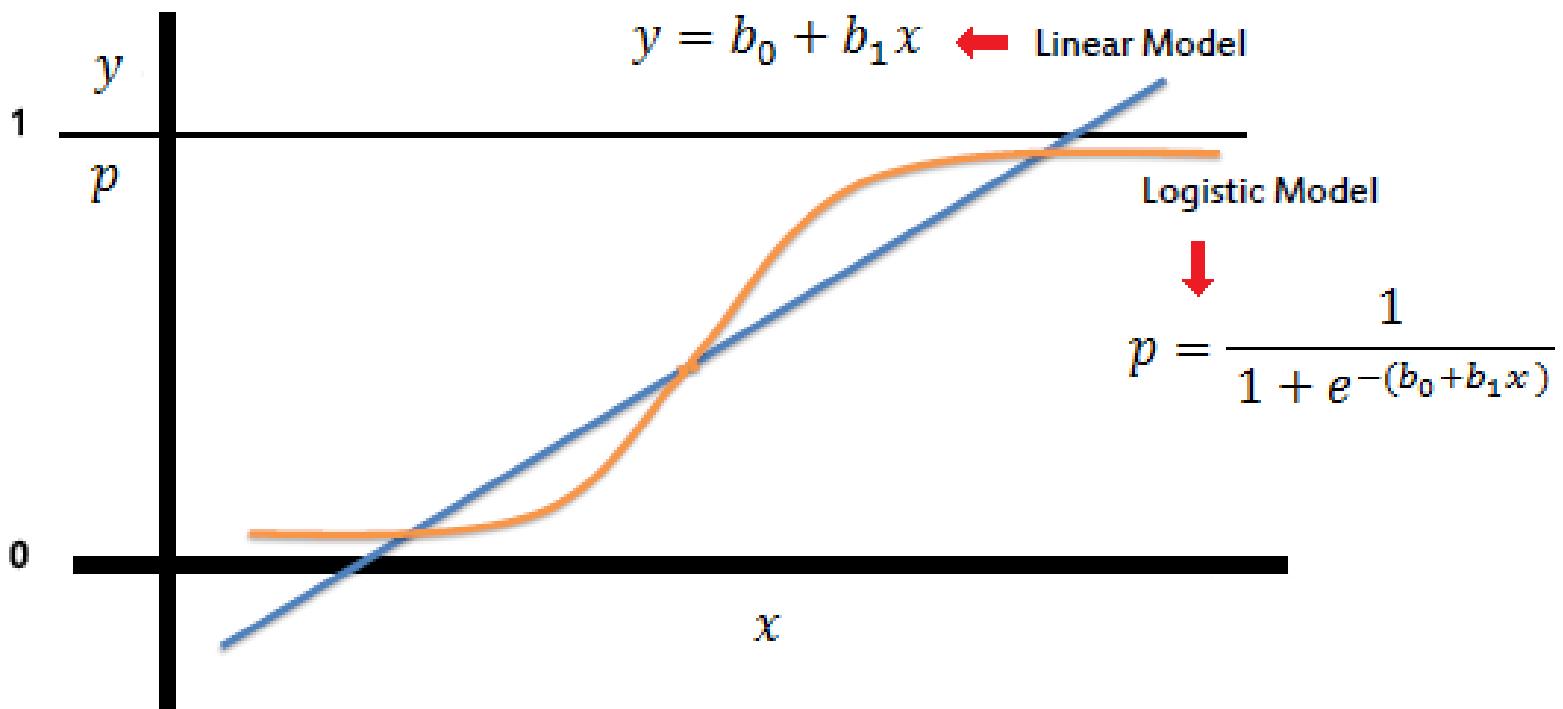
Example

- Example 1: Suppose that we are interested in the factors that influence whether a political candidate wins an election.
- The outcome (response) variable is binary (0/1); win or lose.
- The predictor variables of interest are the amount of money spent on the campaign, the amount of time spent campaigning negatively and whether or not the candidate is an incumbent(is he right person to hold the role/position).
- Example 2: A researcher is interested in how variables, such as GRE (Graduate Record Exam scores), GPA (grade point average) and prestige of the undergraduate institution, effect admission into graduate school. The response variable, admit/don't admit, is a binary variable.

Example

- Type of questions that a binary logistic regression can examine.
- How does the probability of getting lung cancer (yes vs. no) change for every additional pound a person is overweight and for every pack of cigarettes smoked per day?
- Do body weight, calorie intake, fat intake, and age have an influence on the probability of having a heart attack (yes vs. no)?

Logistic Regression



Logistic Regression



- In the logistic regression the constant (b_0) moves the curve left and right and the slope (b_1) defines the steepness of the curve. By simple transformation, the logistic regression equation can be written in terms of an odds ratio.

$$\frac{p}{1-p} = \exp(b_0 + b_1 x)$$

Logistic Regression



- Finally, taking the natural log of both sides, we can write the equation in terms of log-odds (logit) which is a linear function of the predictors. The coefficient (b_1) is the amount the logit (log-odds) changes with a one unit change in x .

$$\ln\left(\frac{p}{1-p}\right) = b_0 + b_1 x$$

Logistic Regression



- As mentioned before, logistic regression can handle any number of numerical and/or categorical variables.

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + \dots + b_p x_p)}}$$

$$\beta^1 = \beta^0 + [X^T W X]^{-1} \cdot X^T (y - \mu)$$

β is a vector of the logistic regression coefficients.

W is a square matrix of order N with elements $n_i \pi_i (1 - \pi_i)$ on the diagonal and zeros everywhere else.

μ is a vector of length N with elements $\mu_i = n_i \pi_i$.

Logistic Regression Assumptions



- Binary logistic regression requires the dependent variable to be binary.
- For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
- Only the meaningful variables should be included.
- The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.
- The independent variables are linearly related to the log odds.
- Logistic regression requires quite large sample sizes.

K-Nearest Neighbor(KNN) Algorithm for Machine Learning



- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suited category by using K-NN algorithm.

K-Nearest Neighbor(KNN) Algorithm for Machine Learning



- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- 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 and when it gets new data, then it classifies that data into a category that is much similar to the new data.

K-Nearest Neighbor(KNN) Algorithm for Machine Learning



- Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog.
- So for this identification, we can use the KNN algorithm, as it works on a similarity measure.
- Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

K-Nearest Neighbor(KNN) Algorithm for Machine Learning



KNN Classifier



K-Nearest Neighbor(KNN) Algorithm for Machine Learning



- The K-NN working can be explained on the basis of the below algorithm:
- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.

K-Nearest Neighbour Algorithm

| Type | Acidity | Strength | Class |
|--------------|---------|----------|-------|
| Type1_tissue | 7 | 7 | Bad |
| Type2_tissue | 7 | 4 | Bad |
| Type3_tissue | 3 | 4 | Good |
| Type4_tissue | 1 | 4 | Good |
| Test Data | 3 | 7 | ? |

X1 = Acid Durability (seconds)

Square Distance to query inst

(kg/square meter)

7

7

$$(7-3)^2 + (7-7)^2 = 16$$

7

4

$$(7-3)^2 + (4-7)^2 = 25$$

3

4

$$(3-3)^2 + (4-7)^2 = 9$$

1

4

$$(1-3)^2 + (4-7)^2 = 13$$

K-Nearest Neighbour Algorithm

| X1 = Acid Durability (seconds) | Strength (kg/square meter) | Square Distance to query instance (3, 7) | Rank minimum distance | Is it included in 3- Nearest neighbors? |
|-----------------------------------|----------------------------------|---|--------------------------|--|
| 7 | 7 | $(7-3)^2 + (7-7)^2 = 16$ | 3 | Yes |
| 7 | 4 | $(7-3)^2 + (4-7)^2 = 25$ | 4 | No |
| 3 | 4 | $(3-3)^2 + (4-7)^2 = 9$ | 1 | Yes |
| 1 | 4 | $(1-3)^2 + (4-7)^2 = 13$ | 2 | Yes |

Label it as Type_3 tissue closeset to rank 1

Naïve Bayes Classifier Algorithm

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Naïve Bayes Classifier Algorithm

- The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:
- Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features.
- Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple.
- Hence each feature individually contributes to identify that it is an apple without depending on each other.
- Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

Naïve Bayes Classifier Algorithm

- Bayes' Theorem:
- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge.
- It depends on the conditional probability.

Naïve Bayes Classifier Algorithm

Bayes Theorem

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

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes

- Working of Naïve Bayes' Classifier can be understood with the help of the below example:
- Suppose we have a dataset of weather conditions and corresponding target variable "Play".
- So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.
- So to solve this problem, we need to follow the below steps:
 - Convert the given dataset into frequency tables.
 - Generate Likelihood table by finding the probabilities of given features.
 - Now, use Bayes theorem to calculate the posterior probability.
- **Problem: If the weather is sunny, then the Player should play or not?**
- **Solution: To solve this, first consider the below dataset:**

Working of Naïve Bayes

- Naïve Bayes Working Model

Confusion Matrix

- The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data.
- It can only be determined if the true values for test data are known.
- The matrix itself can be easily understood, but the related terminologies may be confusing.
- Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix. Some features of Confusion matrix are given below:

Confusion Matrix

- For the 2 prediction classes of classifiers, the matrix is of 2×2 table, for 3 classes, it is 3×3 table, and so on.
- The matrix is divided into two dimensions, that are predicted values and actual values along with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.

Confusion Matrix

| $n =$ total predictions | Actual: No | Actual: Yes |
|-------------------------|----------------|----------------|
| Predicted: No | True Negative | False Positive |
| Predicted: Yes | False Negative | True Positive |

Confusion Matrix

- The above table has the following cases:
- True Negative: Model has given prediction No, and the real or actual value was also No.
- True Positive: The model has predicted yes, and the actual value was also true.
- False Negative: The model has predicted no, but the actual value was Yes, it is also called as Type-II error.
- False Positive: The model has predicted Yes, but the actual value was No. It is also called a Type-I error.

Confusion Matrix

- Need for Confusion Matrix in Machine learning
 - It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is.
 - It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error.
 - With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

Confusion Matrix

- Example: We can understand the confusion matrix using an example.
- Suppose we are trying to create a model that can predict the result for the disease that is either a person has that disease or not. So, the confusion matrix for this is given as:

| n = 100 | Actual: No | Actual: Yes | |
|----------------|------------|-------------|----|
| Predicted: No | TN: 65 | FP: 3 | 68 |
| Predicted: Yes | FN: 8 | TP: 24 | 32 |
| 73 | 27 | | |

Confusion Matrix

- The table is given for the two-class classifier, which has two predictions "Yes" and "NO." Here, Yes defines that patient has the disease, and No defines that patient does not have that disease.
- The classifier has made a total of 100 predictions. Out of 100 predictions, 89 are true predictions, and 11 are incorrect predictions.
- The model has given prediction "yes" for 32 times, and "No" for 68 times. Whereas the actual "Yes" was 27, and actual "No" was 73 times.

Confusion Matrix

- Calculations using Confusion Matrix
- We can perform various calculations for the model, such as the model's accuracy, using this matrix. These calculations are given below:
- Classification Accuracy: It is one of the important parameters to determine the accuracy of the classification problems.
- It defines how often the model predicts the correct output.
- It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers. The formula is given below:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Confusion Matrix

- Misclassification rate: It is also termed as Error rate, and it defines how often the model gives the wrong predictions.
- The value of error rate can be calculated as the number of incorrect predictions to all number of the predictions made by the classifier.
- The formula is given below:

$$\text{Error rate} = \frac{FP + FN}{TP + FP + FN + TN}$$

Confusion Matrix

- Precision: It can be defined as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true.
- It can be calculated using the below formula:

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

Confusion Matrix

- Recall: It is defined as the out of total positive classes, how our model predicted correctly.
- The recall must be as high as possible.

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

Confusion Matrix

- F-measure: If two models have low precision and high recall or vice versa, it is difficult to compare these models.
- So, for this purpose, we can use F-score.
- This score helps us to evaluate the recall and precision at the same time.
- The F-score is maximum if the recall is equal to the precision. It can be calculated using the below formula:

$$\text{F-measure} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Confusion Matrix

- Null Error rate: It defines how often our model would be incorrect if it always predicted the majority class.
- As per the accuracy paradox, it is said that "the best classifier has a higher error rate than the null error rate."
- ROC Curve: The ROC is a graph displaying a classifier's performance for all possible thresholds.
- The graph is plotted between the true positive rate (on the Y-axis) and the false Positive rate (on the x-axis).

Cumulative Accuracy Profile (CAP) Curve

- The Cumulative Accuracy Profile (CAP) is used as a tool in machine learning through which the discriminative power of a classification model is visualized.
- The CAP of a model represents the cumulative number of positive outcomes along the y-axis versus the corresponding cumulative number of a classifying parameter along the x-axis.
- The CAP is different from the Receiver Operator Characteristic (ROC) curves as ROC curves plot the true-positive rate against the false-positive rate of classification.

Cumulative Accuracy Profile (CAP) Curve

- In analyzing a classification model, the CAP curve analysis compares that model with a perfect classification model and a random classification model.
- It evaluates a model by comparing the curve to the perfect CAP in which the maximum number of positive outcomes is achieved directly and to the random CAP in which the positive outcomes are distributed equally.
- A good model will have a CAP between the perfect CAP and the random CAP with a better model tending to the perfect CAP.

Cumulative Accuracy Profile (CAP) Curve

- The four classification models used are Random Forest Model, Logistic Regression Model, K-Nearest Neighbor Model and Naive-Bayes Model.
- Once these models are trained then they are tested on prediction with new data.
- This prediction performance on new test data has been analyzed using the CAP curve analysis.

K-Means Clustering Algorithm

- K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.
- K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.
- Here K defines the number of pre-defined clusters that need to be created in the process, as if $K=2$, there will be two clusters, and for $K=3$, there will be three clusters, and so on.

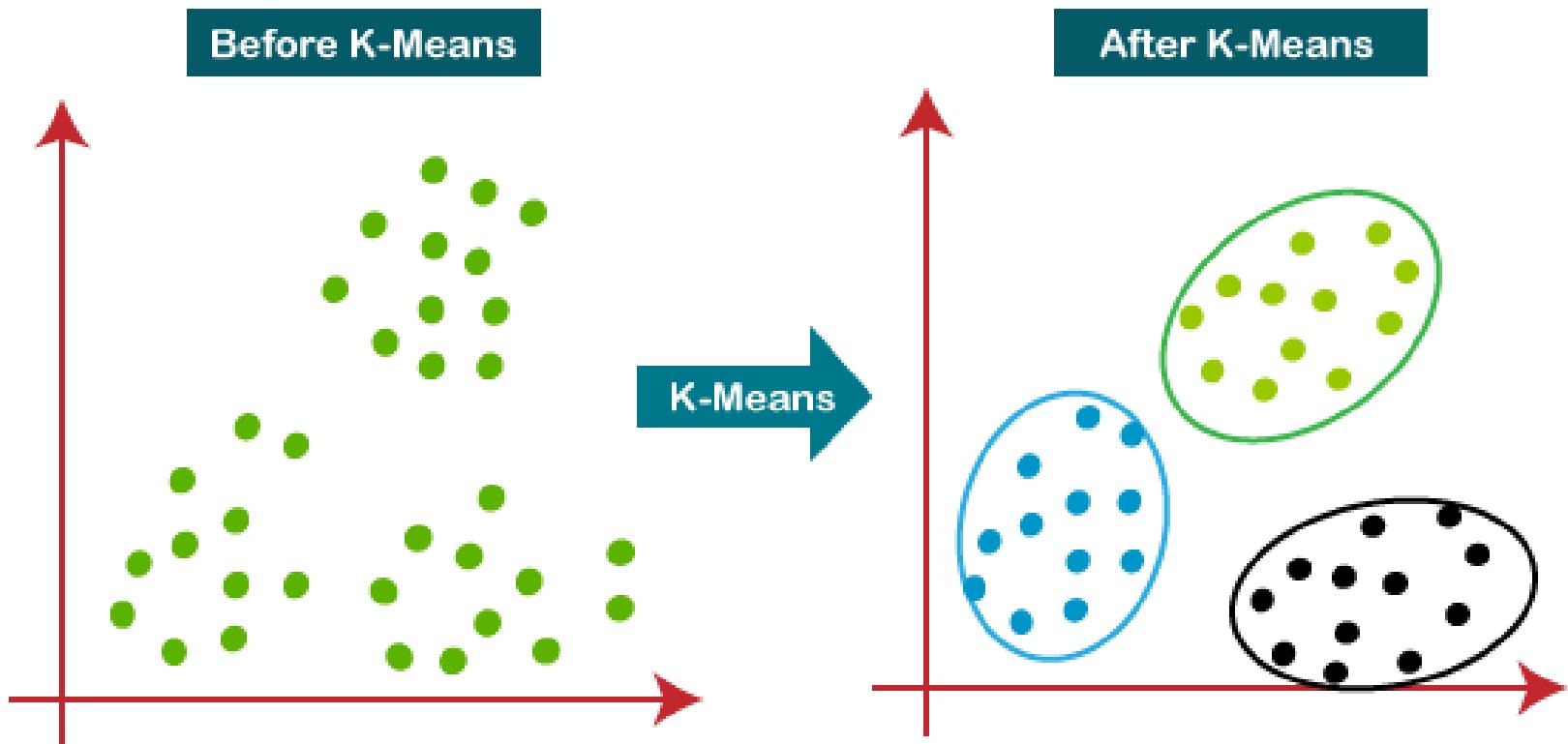
K-Means Clustering Algorithm

- It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.
- It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.
- It is a centroid-based algorithm, where each cluster is associated with a centroid.
- The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

K-Means Clustering Algorithm

- The k-means clustering algorithm mainly performs two tasks:
- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

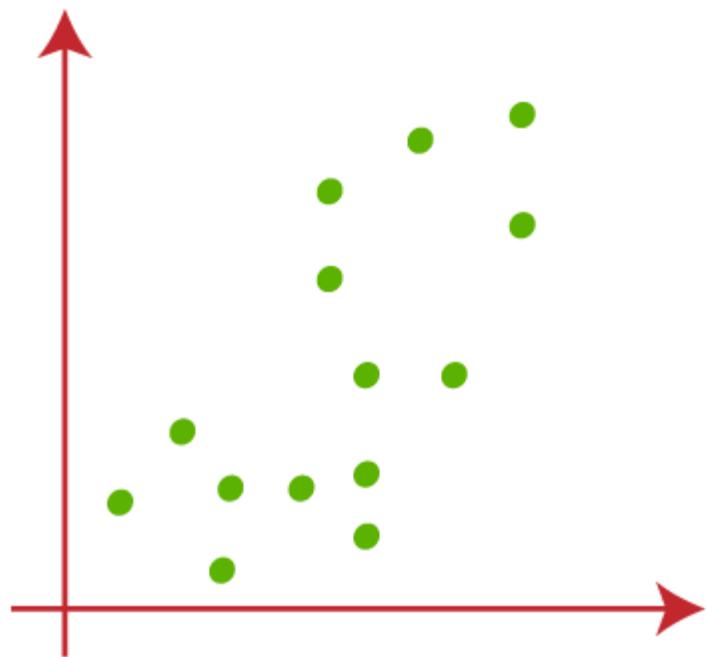
K-Means Clustering Algorithm



K-Means Clustering Algorithm

- The working of the K-Means algorithm is explained in the below steps:
 - Step-1: Select the number K to decide the number of clusters.
 - Step-2: Select random K points or centroids. (It can be other from the input dataset).
 - Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
 - Step-4: Calculate the variance and place a new centroid of each cluster.
 - Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
 - Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
 - Step-7: The model is ready.

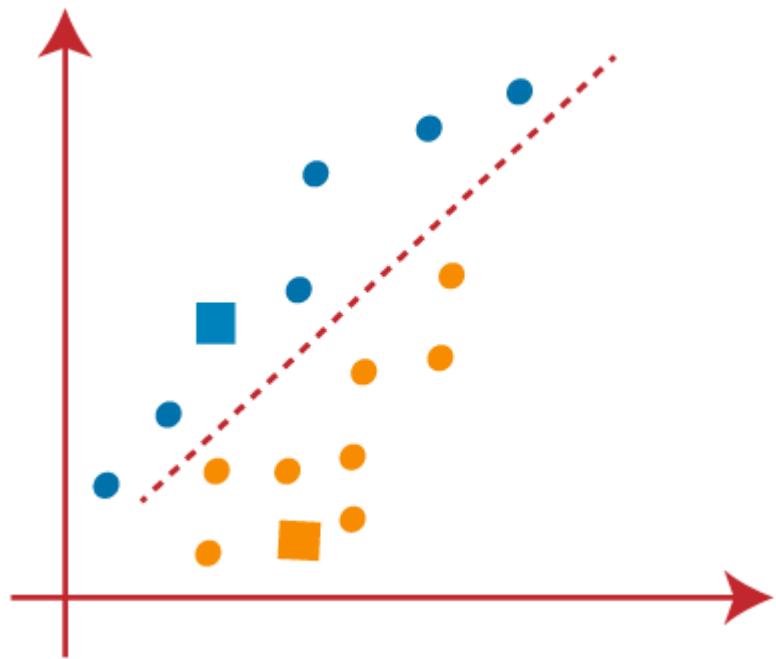
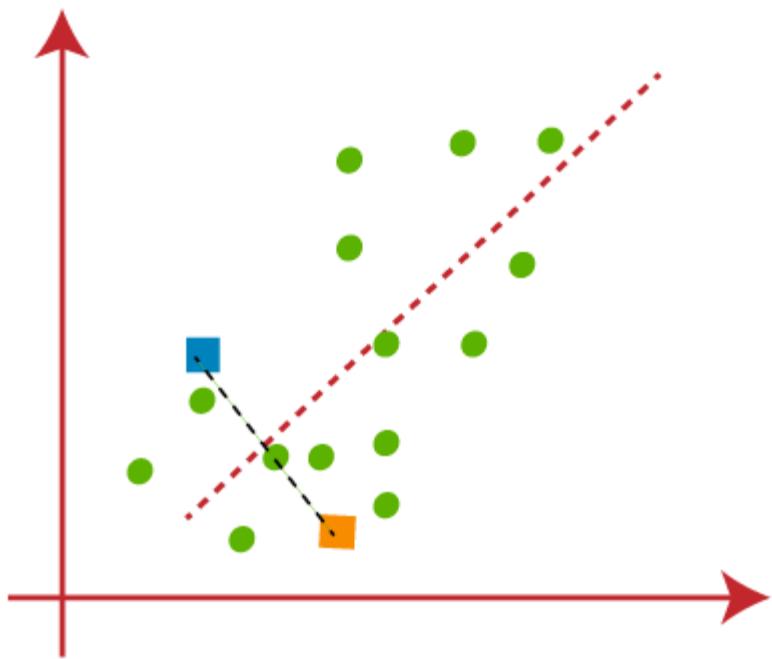
K-Means Clustering Algorithm



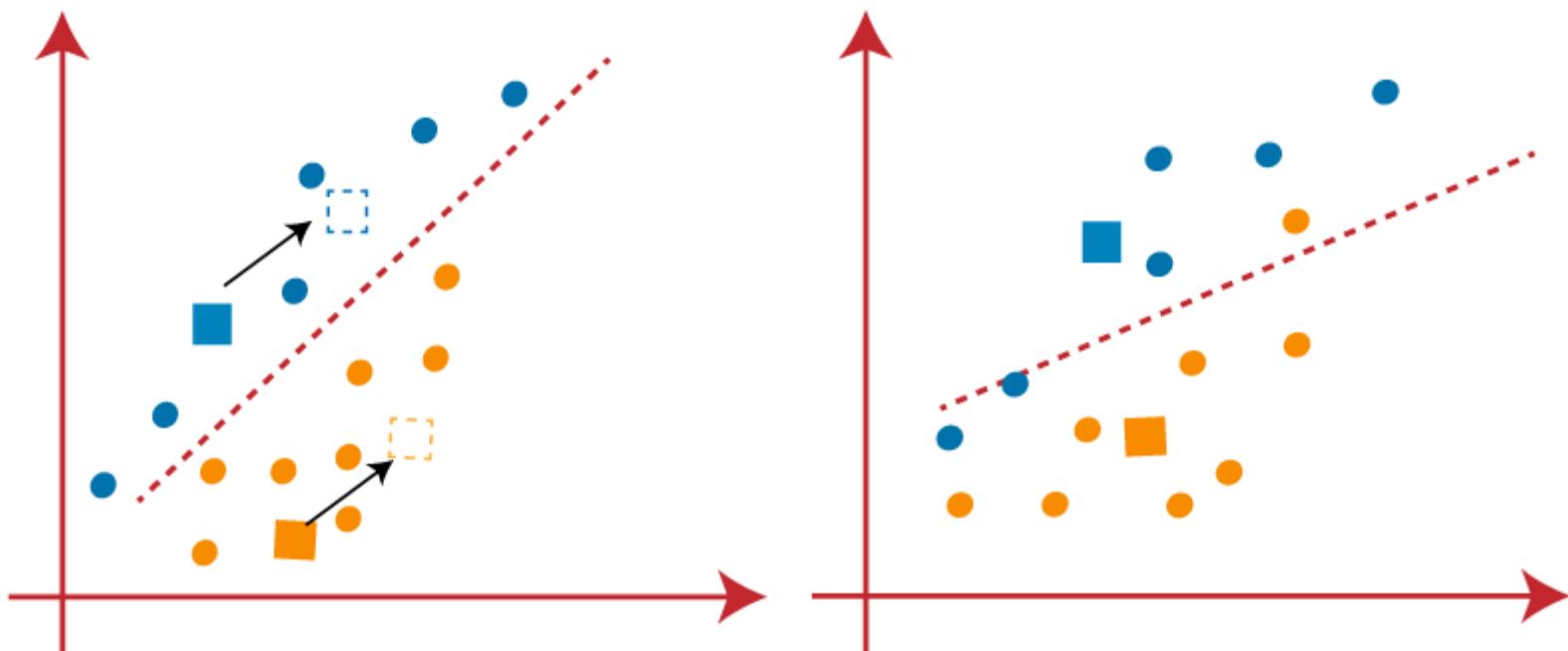
Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters.



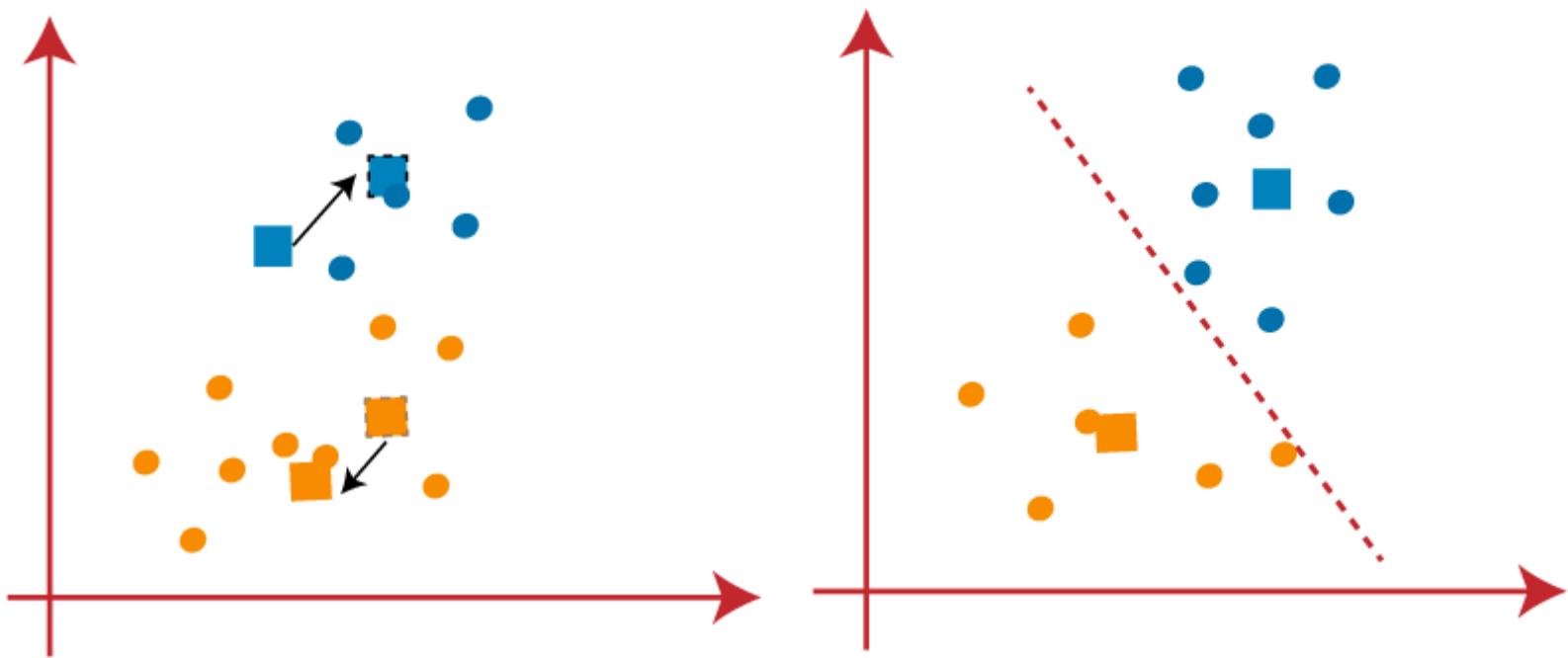
K-Means Clustering Algorithm



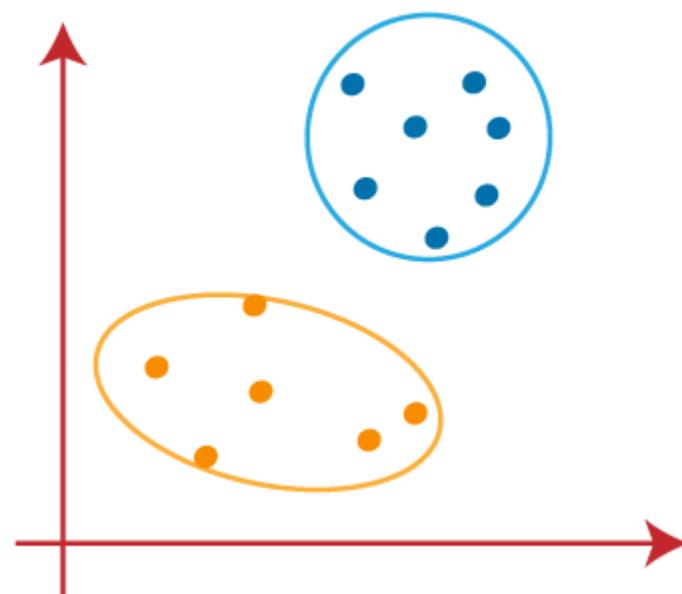
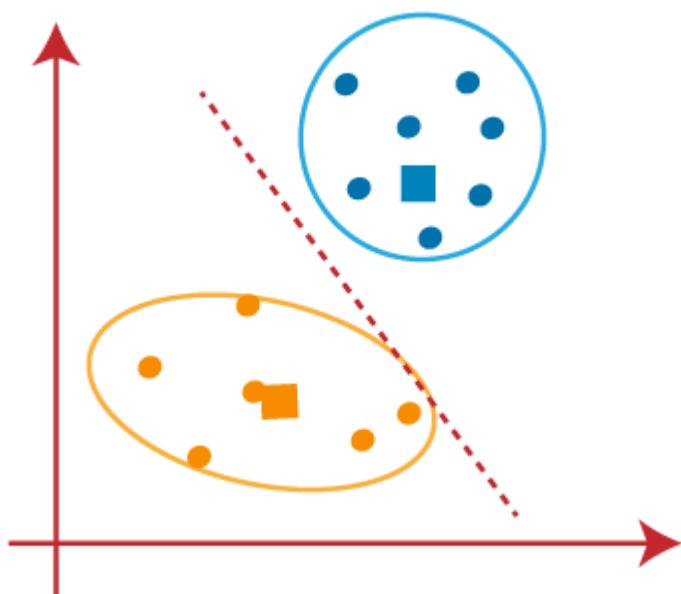
K-Means Clustering Algorithm



K-Means Clustering Algorithm



K-Means Clustering Algorithm



Elbow Method

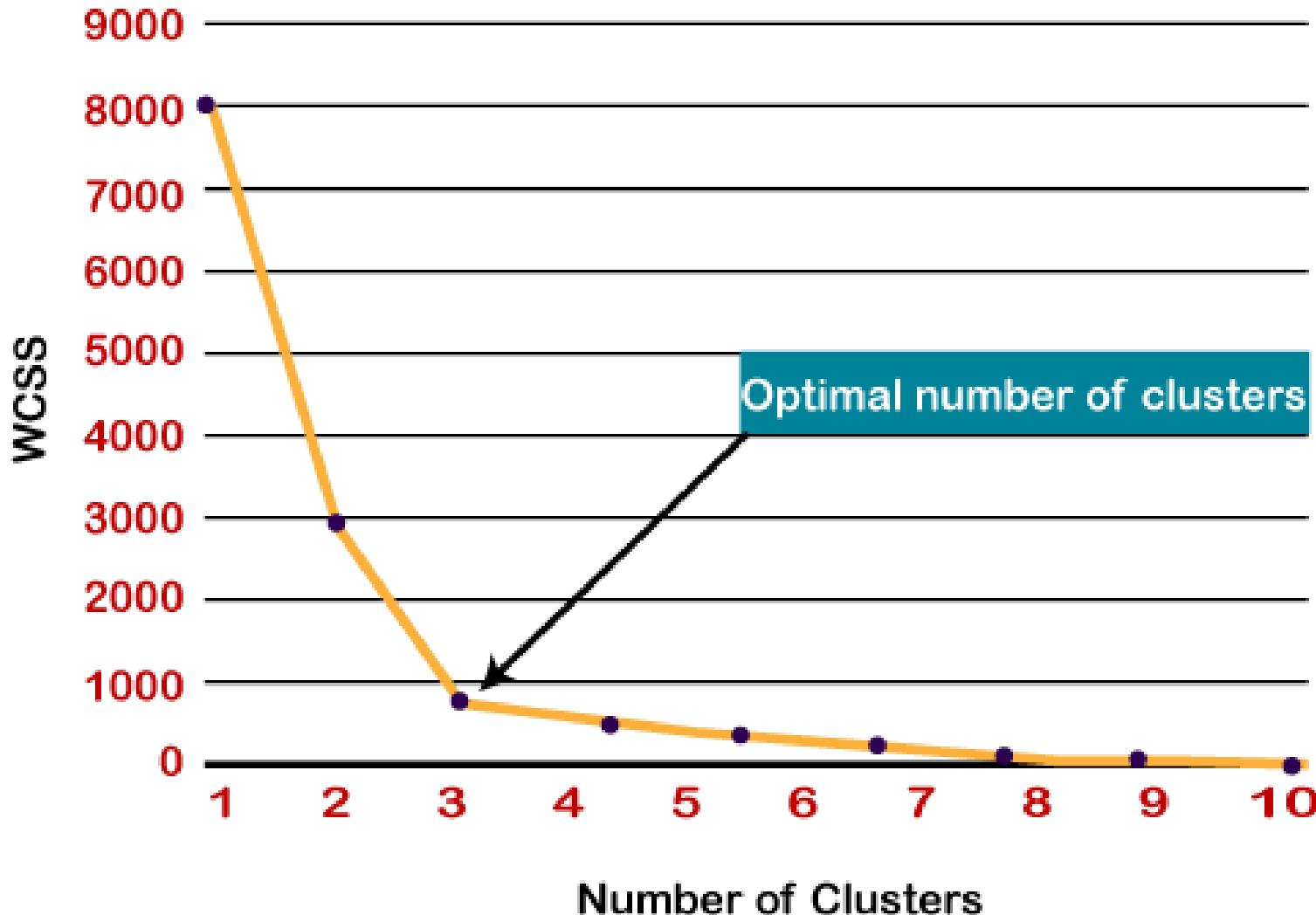
- The Elbow method is one of the most popular ways to find the optimal number of clusters.
- This method uses the concept of WCSS value. WCSS stands for Within Cluster Sum of Squares, which defines the total variations within a cluster.
- The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$\text{WCSS} = \sum_{P_i \text{ in Cluster1}} \text{distance}(P_i | C_1)^2 + \sum_{P_i \text{ in Cluster2}} \text{distance}(P_i | C_2)^2 + \sum_{P_i \text{ in Cluster3}} \text{distance}(P_i | C_3)^2$$

Elbow Method

- In the above formula of WCSS,
- $\sum_{i \in \text{Cluster1}} \text{distance}(P_i | C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.
- To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.
- To find the optimal value of clusters, the elbow method follows the below steps:
 - It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
 - For each value of K, calculates the WCSS value.
 - Plots a curve between calculated WCSS values and the number of clusters K.
 - The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Elbow Method



Elbow Method

```
#finding optimal number of clusters using the elbow method
from sklearn.cluster import KMeans
wcss_list= [] #Initializing the list for the values of WCSS

#Using for loop for iterations from 1 to 10.
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
    kmeans.fit(x)
    wcss_list.append(kmeans.inertia_)

mtp.plot(range(1, 11), wcss_list)
mtp.title('The Elbow Method Graph')
mtp.xlabel('Number of clusters(k)')
mtp.ylabel('wcss_list')
mtp.show()
```

Hierarchical Clustering

- Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as hierarchical cluster analysis or HCA.
- In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the dendrogram.
- Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work.
- As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

Hierarchical Clustering

- The hierarchical clustering technique has two approaches:
- Agglomerative: Agglomerative is a bottom-up approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
- Divisive: Divisive algorithm is the reverse of the agglomerative algorithm as it is a top-down approach.

Why Hierarchical Clustering

- As we already have other clustering algorithms such as K-Means Clustering, then why we need hierarchical clustering?
- So, as we have seen in the K-means clustering that there are some challenges with this algorithm, which are a predetermined number of clusters, and it always tries to create the clusters of the same size.
- To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

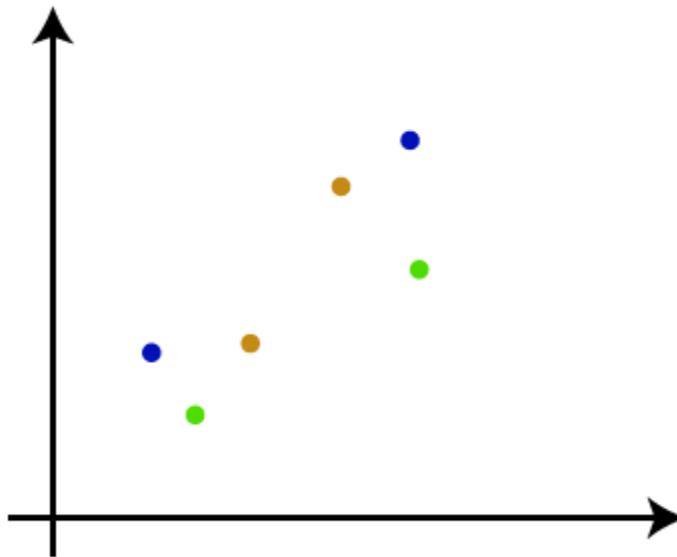
Agglomerative Hierarchical clustering

- The agglomerative hierarchical clustering algorithm is a popular example of HCA.
- To group the datasets into clusters, it follows the bottom-up approach.
- It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together.
- It does this until all the clusters are merged into a single cluster that contains all the datasets.
- This hierarchy of clusters is represented in the form of the dendrogram.

How the Agglomerative Hierarchical clustering Work?



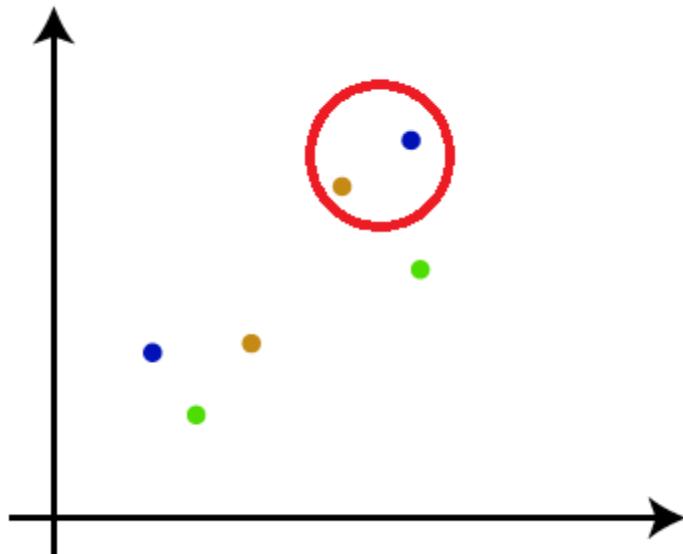
- The working of the AHC algorithm can be explained using the below steps:
- Step-1: Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N .



How the Agglomerative Hierarchical clustering Work?



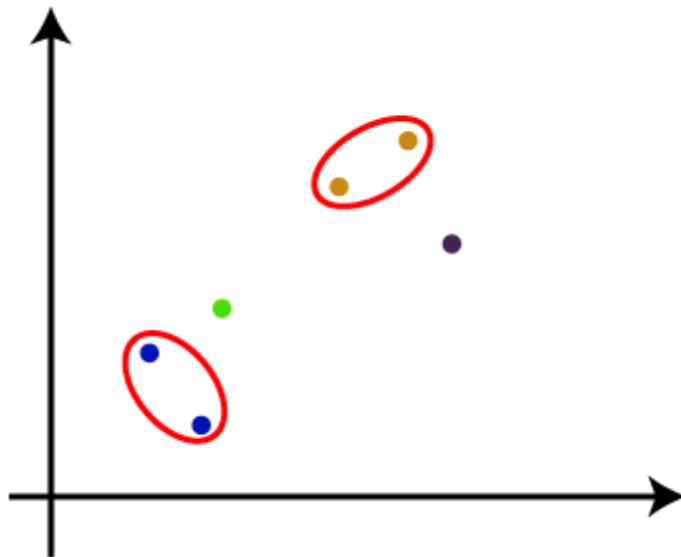
- Step-2: Take two closest data points or clusters and merge them to form one cluster. So, there will now be $N-1$ clusters.



How the Agglomerative Hierarchical clustering Work?



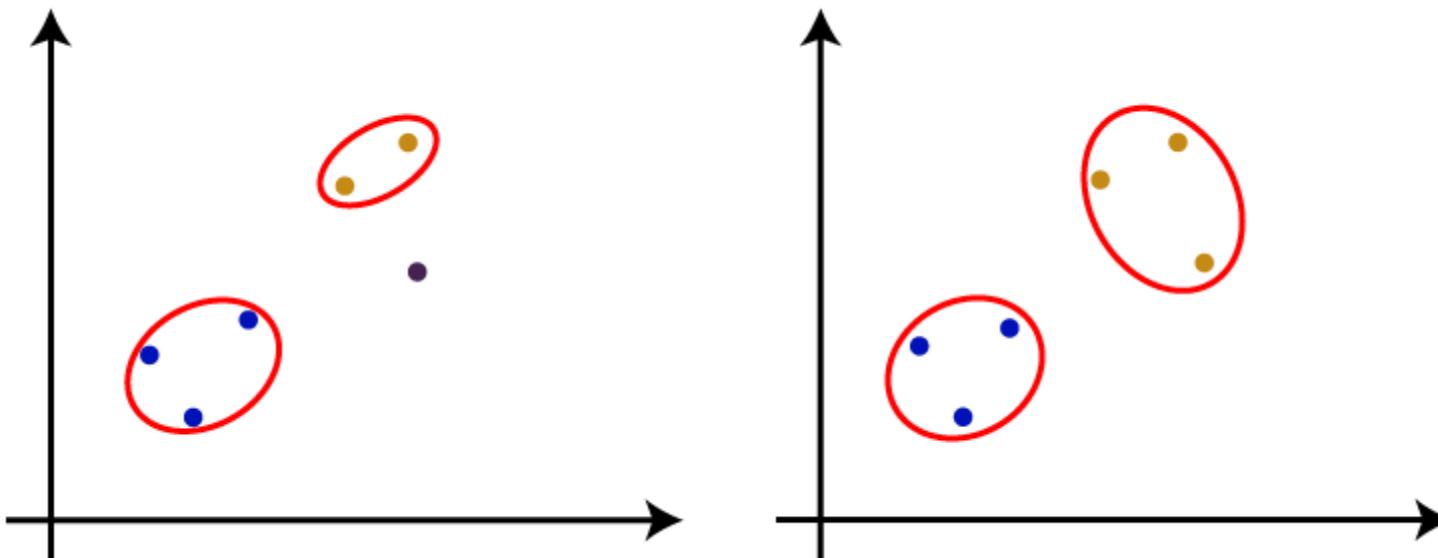
- Step-3: Again, take the two closest clusters and merge them together to form one cluster. There will be $N-2$ clusters.



How the Agglomerative Hierarchical clustering Work?



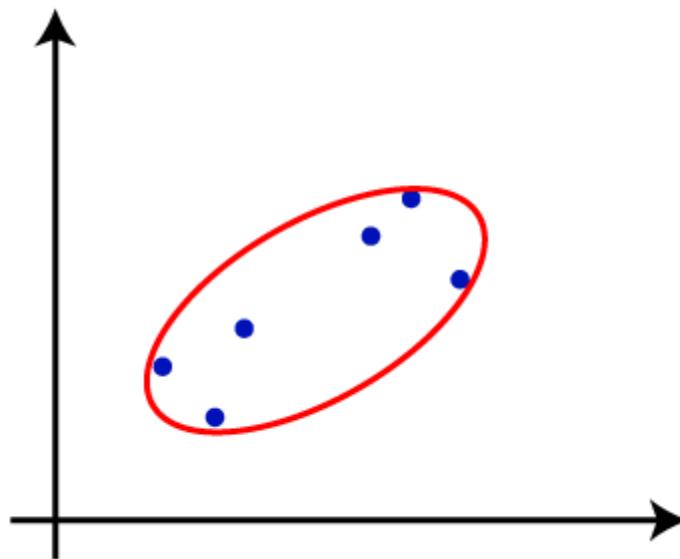
- Step-4: Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:



How the Agglomerative Hierarchical clustering Work?

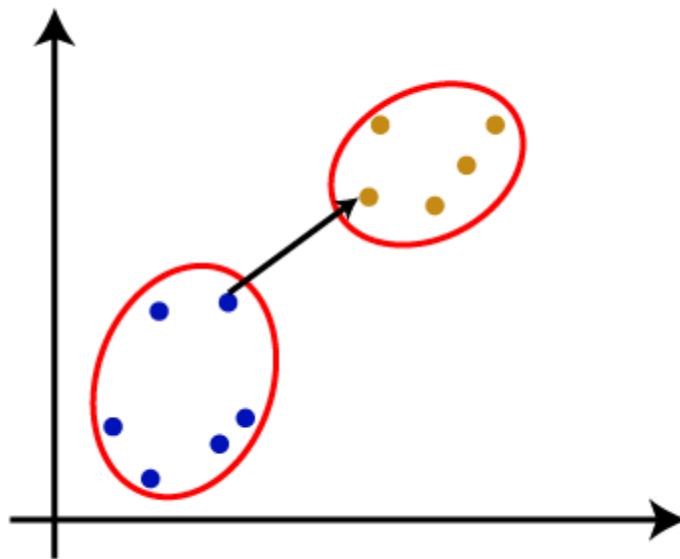


- Step-5: Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.



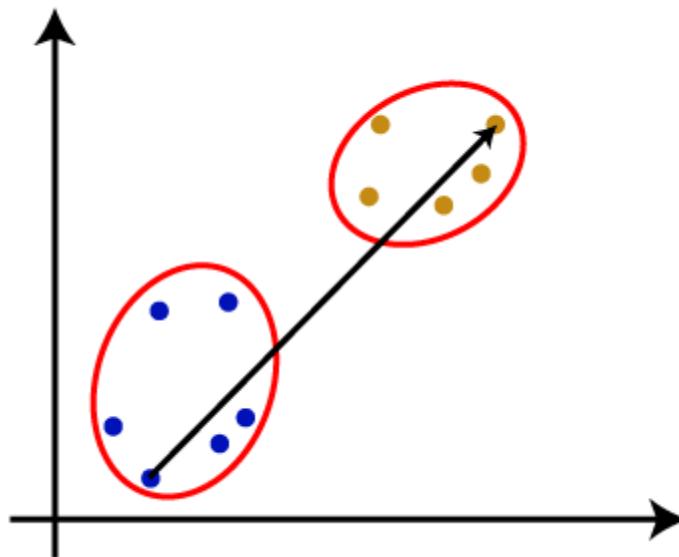
Measure for the distance between two clusters

- Single Linkage: It is the Shortest Distance between the closest points of the clusters. Consider the below image:



Measure for the distance between two clusters

- Complete Linkage: It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.

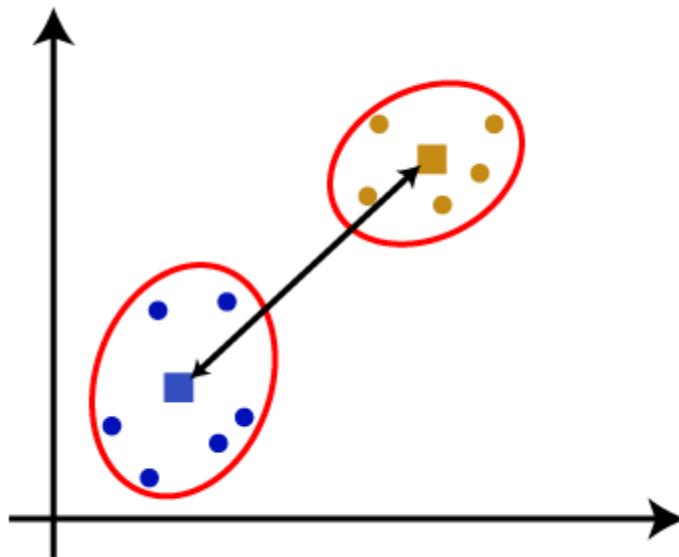


Measure for the distance between two clusters

- Average Linkage: It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.

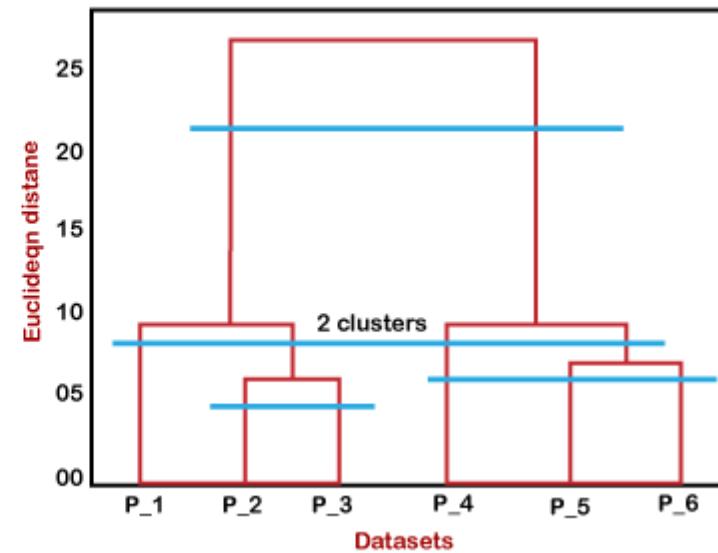
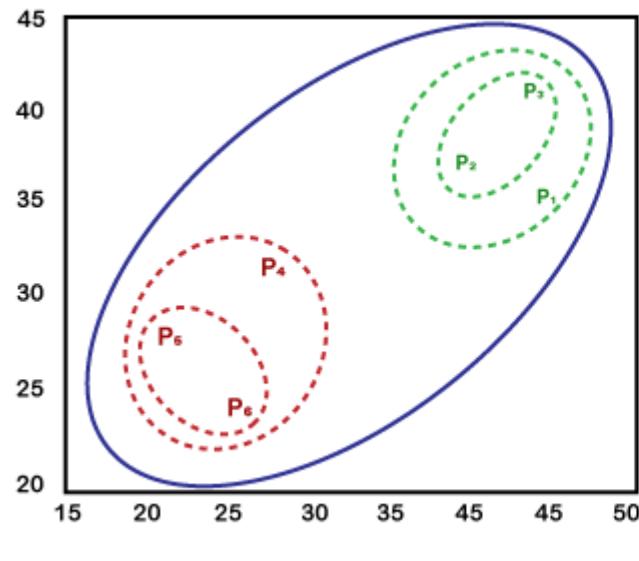
Measure for the distance between two clusters

- Centroid Linkage: It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:



Working of Dendrogram in Hierarchical clustering

- The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.



Association Rule Learning



- Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable.
- It tries to find some interesting relations or associations among the variables of dataset.
- It is based on different rules to discover the interesting relations between variables in the database.

Association Rule Learning



- The association rule learning is one of the very important concepts of machine learning, and it is employed in Market Basket analysis, Web usage mining, continuous production, etc.
- Here market basket analysis is a technique used by the various big retailer to discover the associations between items.
- We can understand it by taking an example of a supermarket, as in a supermarket, all products that are purchased together are put together.

Association Rule Learning



- For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored within a shelf or mostly nearby. Consider the below diagram:



Customer 1



Customer 2



Customer 3



Customer n

Association Rule Learning



- Association rule learning can be divided into three types of algorithms:
- Apriori
- Eclat
- F-P Growth Algorithm

How does Association Rule Learning work?



- Association rule learning works on the concept of If and Else Statement, such as if A then B.



- Here the If element is called antecedent, and then statement is called as Consequent.
- These types of relationships where we can find out some association or relation between two items is known as single cardinality.
- It is all about creating rules, and if the number of items increases, then cardinality also increases accordingly.
- So, to measure the associations between thousands of data items, there are several metrics. These metrics are given below:
 - Support
 - Confidence
 - Lift

Association Rule Learning

- Measure 1: Support.** This says how popular an itemset is, as measured by the proportion of transactions in which an itemset appears. In Table 1 below, the support of {apple} is 4 out of 8, or 50%. Itemsets can also contain multiple items. For instance, the support of {apple, beer, rice} is 2 out of 8, or 25%.

| | | | | |
|---------------|---|---|---|---|
| Transaction 1 | 🍎 | 🍺 | 🥣 | 🍗 |
| Transaction 2 | 🍎 | 🍺 | 🥣 | |
| Transaction 3 | 🍎 | 🍺 | | |
| Transaction 4 | 🍎 | 🍐 | | |
| Transaction 5 | 🍼 | 🍺 | 🥣 | 🍗 |
| Transaction 6 | 🍼 | 🍺 | 🥣 | |
| Transaction 7 | 🍼 | 🍺 | | |
| Transaction 8 | 🍼 | 🍐 | | |

$$\text{Supp}(X) = \frac{\text{Freq}(X)}{T}$$

$$\text{Support } \{ \text{🍎} \} = \frac{4}{8}$$

Association Rule Learning

- If you discover that sales of items beyond a certain proportion tend to have a significant impact on your profits, you might consider using that proportion as your *support threshold*.
- You may then identify item sets with support values above this threshold as significant item sets.

$$\text{Support } \{\text{apple}\} = \frac{4}{8}$$

Association Rule Learning



- **Measure 2: Confidence.** This says how likely item Y is purchased when item X is purchased, expressed as $\{X \rightarrow Y\}$. This is measured by the proportion of transactions with item X, in which item Y also appears. In Table 1, the confidence of $\{\text{apple} \rightarrow \text{beer}\}$ is 3 out of 4, or 75%.

$$\text{Confidence} = \frac{Freq(X,Y)}{Freq(X)}$$

$$\text{Confidence } \{\text{🍎} \rightarrow \text{🍺}\} = \frac{\text{Support } \{\text{🍎}, \text{🍺}\}}{\text{Support } \{\text{🍎}\}}$$

Association Rule Learning

- **Measure 3: Lift.** This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is. In Table 1, the lift of {apple -> beer} is 1, which implies no association between items. A lift value greater than 1 means that item Y is *likely* to be bought if item X is bought, while a value less than 1 means that item Y is *unlikely* to be bought if item X is bought.

$$\text{Lift} = \frac{\text{Supp}(X,Y)}{\text{Supp}(X) \times \text{Supp}(Y)}$$

$$\text{Lift } \{\text{🍎} \rightarrow \text{🍺}\} = \frac{\text{Support }\{\text{🍎}, \text{🍺}\}}{\text{Support }\{\text{🍎}\} \times \text{Support }\{\text{🍺}\}}$$

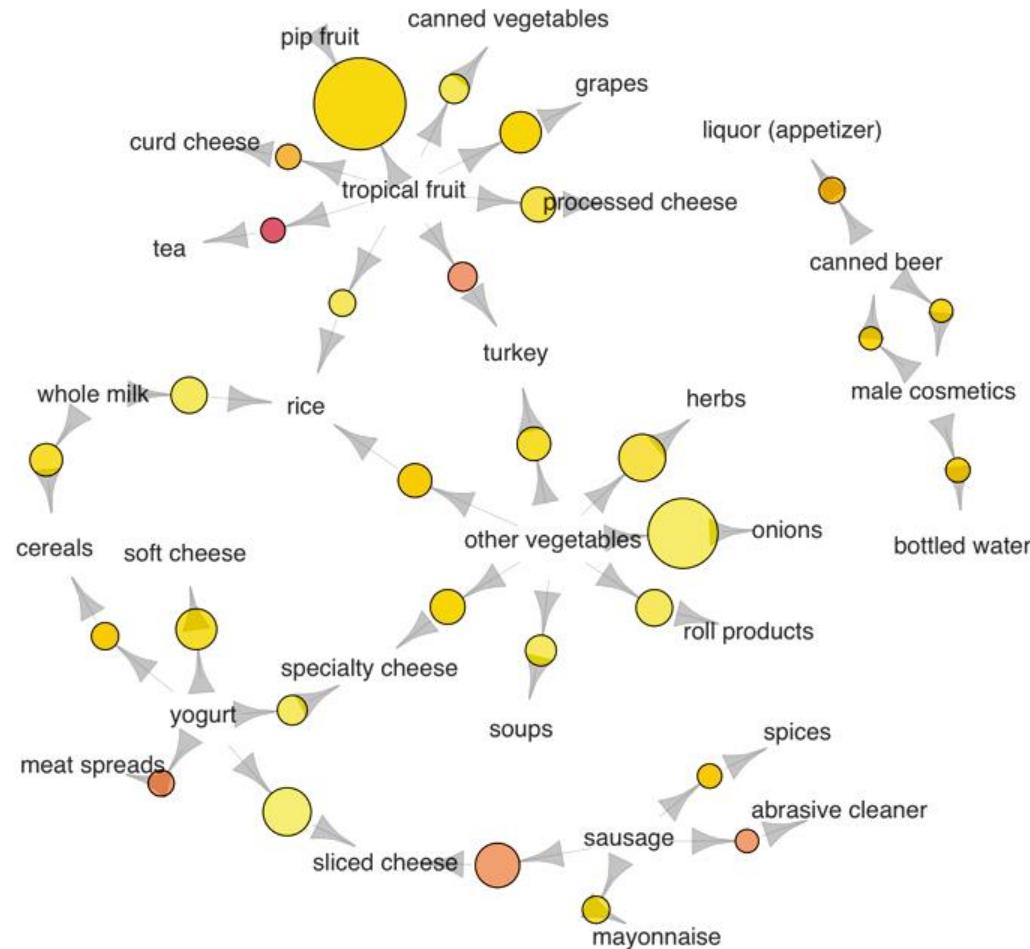
Association Rule Learning

- It is the ratio of the observed support measure and expected support if X and Y are independent of each other. It has three possible values:
- If Lift= 1: The probability of occurrence of antecedent and consequent is independent of each other.
- Lift>1: It determines the degree to which the two itemsets are dependent to each other.
- Lift<1: It tells us that one item is a substitute for other items, which means one item has a negative effect on another.

Association Rule Learning



Associations between selected items. Visualized using the arulesViz R library.



Association Rule Learning

| Transaction | Support | Confidence | Lift |
|------------------------------|---------|------------|------|
| Canned Beer → Soda | 1% | 20% | 1.0 |
| Canned Beer → Berries | 0.1% | 1% | 0.3 |
| Canned Beer → Male Cosmetics | 0.1% | 1% | 2.6 |

| Transaction | Support |
|----------------|---------|
| Canned Beer | 10% |
| Soda | 20% |
| Berries | 3% |
| Male Cosmetics | 0.5% |

Apriori

- With the quick growth in e-commerce applications, there is an accumulation vast quantity of data in months not in years.
- Data Mining, also known as Knowledge Discovery in Databases(KDD), to find anomalies, correlations, patterns, and trends to predict outcomes.
- Apriori algorithm is a classical algorithm in data mining.
- It is used for mining frequent item sets and relevant association rules.
- It is devised to operate on a database containing a lot of transactions, for instance, items brought by customers in a store.

Apriori

- Apriori algorithm refers to the algorithm which is used to calculate the association rules between objects.
- It means how two or more objects are related to one another.
- In other words, we can say that the apriori algorithm is an association rule leaning that analyzes that people who bought product A also bought product B.

Apriori

- The primary objective of the apriori algorithm is to create the association rule between different objects.
- The association rule describes how two or more objects are related to one another.
- Apriori algorithm is also called frequent pattern mining.
- Generally, you operate the Apriori algorithm on a database that consists of a huge number of transactions.
- Let's understand the apriori algorithm with the help of an example; suppose you go to Big Bazar and buy different products.
- It helps the customers buy their products with ease and increases the sales performance of the Big Bazar.

- We take an example to understand the concept better.
- You must have noticed that the Pizza shop seller makes a pizza, soft drink, and breadstick combo together.
- He also offers a discount to their customers who buy these combos.
- Do you ever think why does he do so?
- He thinks that customers who buy pizza also buy soft drinks and breadsticks.
- However, by making combos, he makes it easy for the customers. At the same time, he also increases his sales performance.

- Similarly, you go to Big Bazar, and you will find biscuits, chips, and Chocolate bundled together.
- It shows that the shopkeeper makes it comfortable for the customers to buy these products in the same place.

Components of Apriori algorithm

- The given three components comprise the apriori algorithm.
 - Support
 - Confidence
 - Lift

Components of Apriori algorithm

- Suppose you have 4000 customers transactions in a Big Bazar.
- You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate.
- This is because customers frequently buy these two items together.
- Out of 4000 transactions, 400 contain Biscuits, whereas 600 contain Chocolate, and these 600 transactions include a 200 that includes Biscuits and chocolates.
- Using this data, we will find out the support, confidence, and lift

Components of Apriori algorithm

- Support
 - Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get
 - Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)
 - $= 400/4000 = 10$ percent.

Components of Apriori algorithm

- Confidence
 - Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.
 - Hence,
 - Confidence = (Transactions relating both biscuits and Chocolate) / (Total transactions involving Biscuits)
 - = 200/400
 - = 50 percent.
 - It means that 50 percent of customers who bought biscuits bought chocolates also.

Components of Apriori algorithm

- Lift
 - Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.
 - $\text{Lift} = (\text{Confidence (Biscuits - chocolates}) / (\text{Support (Biscuits)})$
 - $= 50/10 = 5$
 - It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.

How does the Apriori Algorithm work in Data Mining?



- Consider a Big Bazar scenario where the product set is $P = \{\text{Rice}, \text{Pulse}, \text{Oil}, \text{Milk}, \text{Apple}\}$.
- The database comprises six transactions where 1 represents the presence of the product and 0 represents the absence of the product.

How does the Apriori Algorithm work in Data Mining?



| Transaction ID | Rice | Pulse | Oil | Milk | Apple | |
|----------------|------|-------|-----|------|-------|---|
| t1 | 1 | 1 | 1 | 0 | 0 | 0 |
| t2 | 0 | 1 | 1 | 1 | 1 | 0 |
| t3 | 0 | 0 | 0 | 1 | 1 | 1 |
| t4 | 1 | 1 | 0 | 1 | 1 | 0 |
| t5 | 1 | 1 | 1 | 0 | 0 | 1 |
| t6 | 1 | 1 | 1 | 1 | 1 | 1 |

How does the Apriori Algorithm work in Data Mining?



- The Apriori Algorithm makes the given assumptions
 - All subsets of a frequent itemset must be frequent.
 - The subsets of an infrequent item set must be infrequent.
 - Fix a threshold support level. In our case, we have fixed it at 50 percent.

How does the Apriori Algorithm work in Data Mining?



- Step 1
 - Make a frequency table of all the products that appear in all the transactions.
 - Now, short the frequency table to add only those products with a threshold support level of over 50 percent.
 - We find the given frequency table.

How does the Apriori Algorithm work in Data Mining?



| Product | Frequency (Number of transactions) |
|----------|------------------------------------|
| Rice (R) | 4 |
| Pulse(P) | 5 |
| Oil(O) | 4 |
| Milk(M) | 4 |

The above table indicated the products frequently bought by the customers.

How does the Apriori Algorithm work in Data Mining?



Step 2

Create pairs of products such as RP, RO, RM, PO, PM, OM. You will get the given frequency table..

| Itemset | Frequency (Number of transactions) |
|---------|------------------------------------|
| RP | 4 |
| RO | 3 |
| RM | 2 |
| PO | 4 |
| PM | 3 |
| OM | 2 |

How does the Apriori Algorithm work in Data Mining?



Step 3

Implementing the same threshold support of 50 percent and consider the products that are more than 50 percent. In our case, it is more than 3

Thus, we get RP, RO, PO, and PM

Step 4

Now, look for a set of three products that the customers buy together. We get the given combination.

RP and RO give RPO
PO and PM give POM

How does the Apriori Algorithm work in Data Mining?



Step 5

Calculate the frequency of the two itemsets, and you will get the given frequency table.

| Itemset | Frequency (Number of transactions) |
|---------|------------------------------------|
| RPO | 4 |
| POM | 3 |

How to improve the efficiency of the Apriori Algorithm?



- There are various methods used for the efficiency of the Apriori algorithm
- Hash-based itemset counting
 - In hash-based itemset counting, you need to exclude the k-itemset whose equivalent hashing bucket count is least than the threshold is an infrequent itemset.
- Transaction Reduction
 - In transaction reduction, a transaction not involving any frequent X itemset becomes not valuable in subsequent scans.

Advantages of Apriori Algorithm

- It is used to calculate large itemsets.
- Simple to understand and apply.

Disadvantages of Apriori Algorithms

- Apriori algorithm is an expensive method to find support since the calculation has to pass through the whole database.
- Sometimes, you need a huge number of candidate rules, so it becomes computationally more expensive.

What is Reinforcement Learning?

- Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions.
- For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.
- In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning.

What is Reinforcement Learning?

- Since there is no labeled data, so the agent is bound to learn by its experience only.
- RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive rewards.

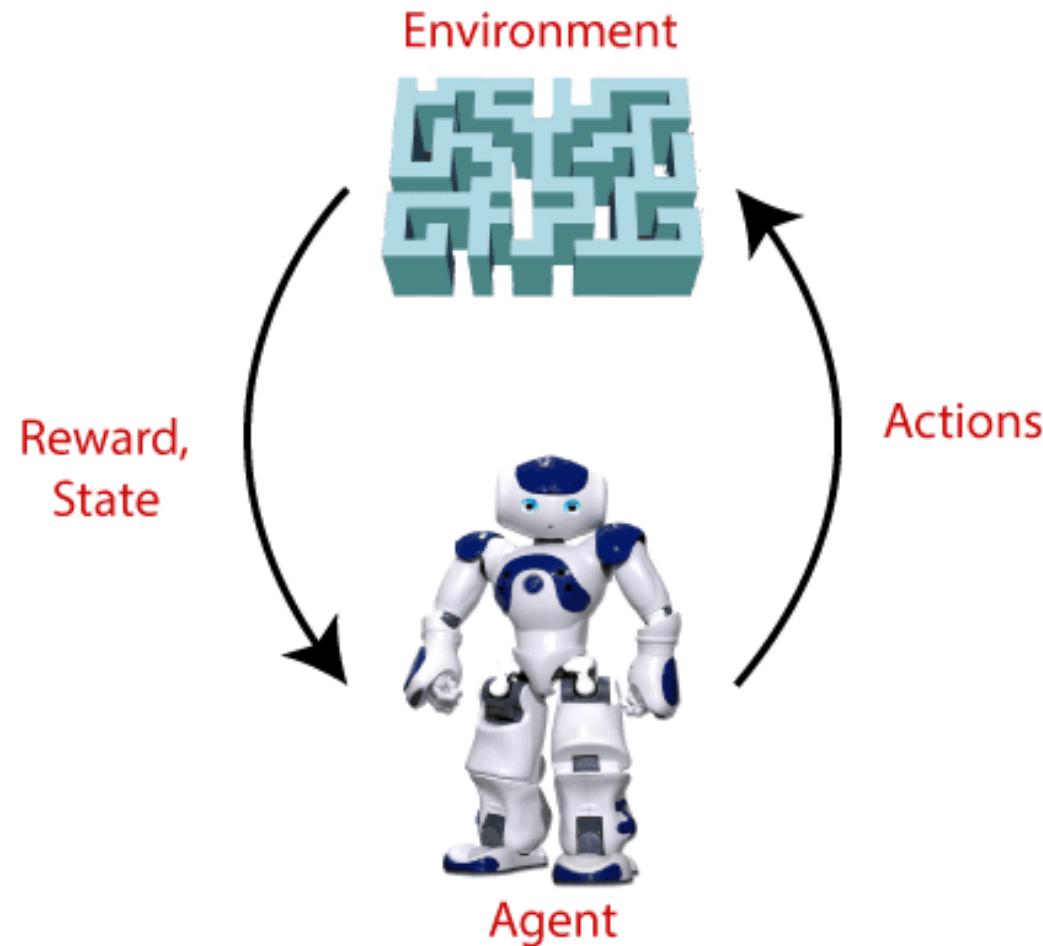
What is Reinforcement Learning?

- The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way.
- Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that."
- How a Robotic dog learns the movement of his arms is an example of Reinforcement learning.
- It is a core part of Artificial intelligence, and all AI agent works on the concept of reinforcement learning.
- Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention.

What is Reinforcement Learning?

- Example: Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond.
- The agent interacts with the environment by performing some actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.
- The agent continues doing these three things (take action, change state/remain in the same state, and get feedback), and by doing these actions, he learns and explores the environment.
- The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty.
- As a positive reward, the agent gets a positive point, and as a penalty, it gets a negative point.

What is Reinforcement Learning?



Terms Used in Reinforcement Learning?

- Agent(): An entity that can perceive/explore the environment and act upon it.
- Environment(): A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
- Action(): Actions are the moves taken by an agent within the environment.
- State(): State is a situation returned by the environment after each action taken by the agent.
- Reward(): A feedback returned to the agent from the environment to evaluate the action of the agent.
- Policy(): Policy is a strategy applied by the agent for the next action based on the current state.
- Value(): It is expected long-term return with the discount factor and opposite to the short-term reward.
- Q-value(): It is mostly similar to the value, but it takes one additional parameter as a current action (a).

Approaches to implement Reinforcement Learning

- There are mainly three ways to implement reinforcement-learning in ML, which are:
- Value-based:
 - The value-based approach is about to find the optimal value function, which is the maximum value at a state under any policy. Therefore, the agent expects the long-term return at any state(s) under policy π .
- Policy-based:
 - Policy-based approach is to find the optimal policy for the maximum future rewards without using the value function. In this approach, the agent tries to apply such a policy that the action performed in each step helps to maximize the future reward.
- The policy-based approach has mainly two types of policy:
 - Deterministic: The same action is produced by the policy (π) at any state.
 - Stochastic: In this policy, probability determines the produced action.

Approaches to implement Reinforcement Learning

- Model-based: In the model-based approach, a virtual model is created for the environment, and the agent explores that environment to learn it.
- There is no particular solution or algorithm for this approach because the model representation is different for each environment.

Elements of Reinforcement Learning

- There are four main elements of Reinforcement Learning, which are given below:
 - Policy
 - Reward Signal
 - Value Function
 - Model of the environment

Elements of Reinforcement Learning

- 1) Policy: A policy can be defined as a way how an agent behaves at a given time. It maps the perceived states of the environment to the actions taken on those states.
- A policy is the core element of the RL as it alone can define the behavior of the agent. In some cases, it may be a simple function or a lookup table, whereas, for other cases, it may involve general computation as a search process.
- It could be deterministic or a stochastic policy:
- For deterministic policy: $a = \pi(s)$
- For stochastic policy: $\pi(a | s) = P[A_t = a | S_t = s]$

Elements of Reinforcement Learning

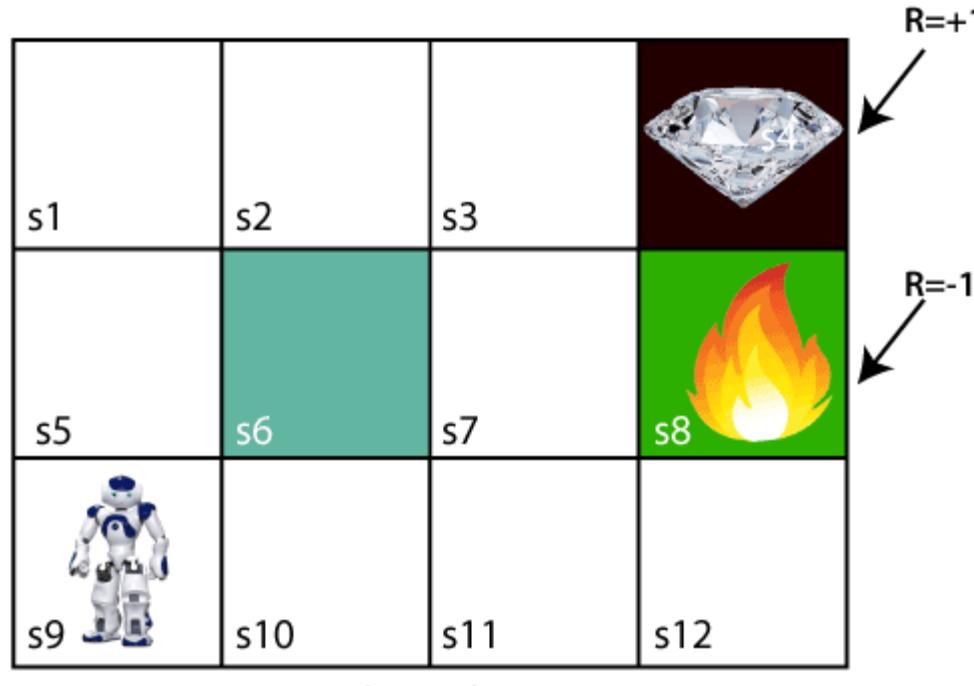
- 2) Reward Signal: The goal of reinforcement learning is defined by the reward signal.
- At each state, the environment sends an immediate signal to the learning agent, and this signal is known as a reward signal.
- These rewards are given according to the good and bad actions taken by the agent.
- The agent's main objective is to maximize the total number of rewards for good actions.
- The reward signal can change the policy, such as if an action selected by the agent leads to low reward, then the policy may change to select other actions in the future.

Elements of Reinforcement Learning

- 3) Value Function: The value function gives information about how good the situation and action are and how much reward an agent can expect.
- A reward indicates the immediate signal for each good and bad action, whereas a value function specifies the good state and action for the future.
- The value function depends on the reward as, without reward, there could be no value. The goal of estimating values is to achieve more rewards.
- 4) Model: The last element of reinforcement learning is the model, which mimics the behavior of the environment.
- With the help of the model, one can make inferences about how the environment will behave. Such as, if a state and an action are given, then a model can predict the next state and reward.

How does Reinforcement Learning Work?

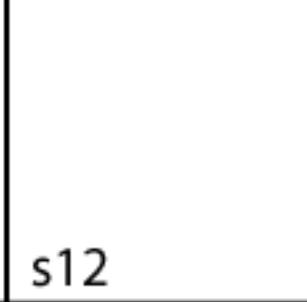
- To understand the working process of the RL, we need to consider two main things:
 - Environment: It can be anything such as a room, maze, football ground, etc.
 - Agent: An intelligent agent such as AI robot.



How does Reinforcement Learning Work?

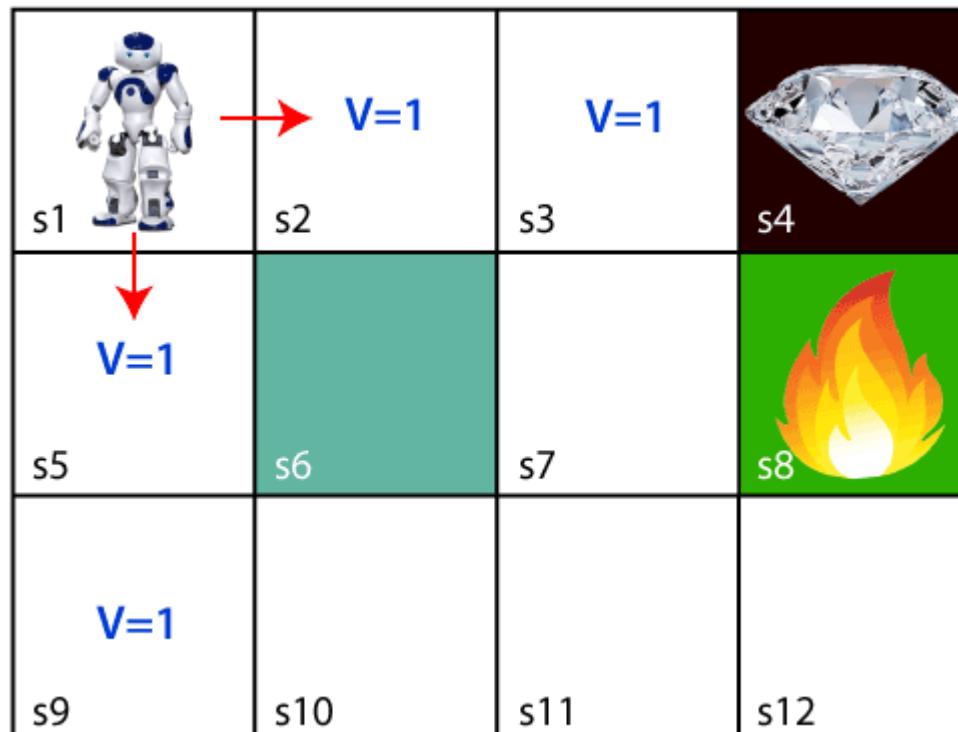
- In the above image, the agent is at the very first block of the maze.
- The maze is consisting of an S6 block, which is a wall, S8 a fire pit, and S4 a diamond block.
- The agent cannot cross the S6 block, as it is a solid wall. If the agent reaches the S4 block, then get the +1 reward; if it reaches the fire pit, then gets -1 reward point. It can take four actions: move up, move down, move left, and move right.
- The agent can take any path to reach to the final point, but he needs to make it in possible fewer steps.
- Suppose the agent considers the path S9-S5-S1-S2-S3, so he will get the +1-reward point.
- The agent will try to remember the preceding steps that it has taken to reach the final step.
- To memorize the steps, it assigns 1 value to each previous step.

How does Reinforcement Learning Work?

| | | | |
|--|--|---|--|
| $V=1$ s1 | $V=1$ s2 | $V=1$ s3 |  s4 |
| $V=1$ s5 |  s6 |  s7 |  s8 |
|  $V=1$ s9 |  s10 |  s11 |  s12 |

How does Reinforcement Learning Work?

- Now, the agent has successfully stored the previous steps assigning the 1 value to each previous block.
- But what will the agent do if he starts moving from the block, which has 1 value block on both sides? Consider the below diagram:





How does Reinforcement Learning Work?

- It will be a difficult condition for the agent whether he should go up or down as each block has the same value. So, the above approach is not suitable for the agent to reach the destination.
- Hence to solve the problem, we will use the Bellman equation, which is the main concept behind reinforcement learning.

Difference between Reinforcement Learning and Supervised Learning



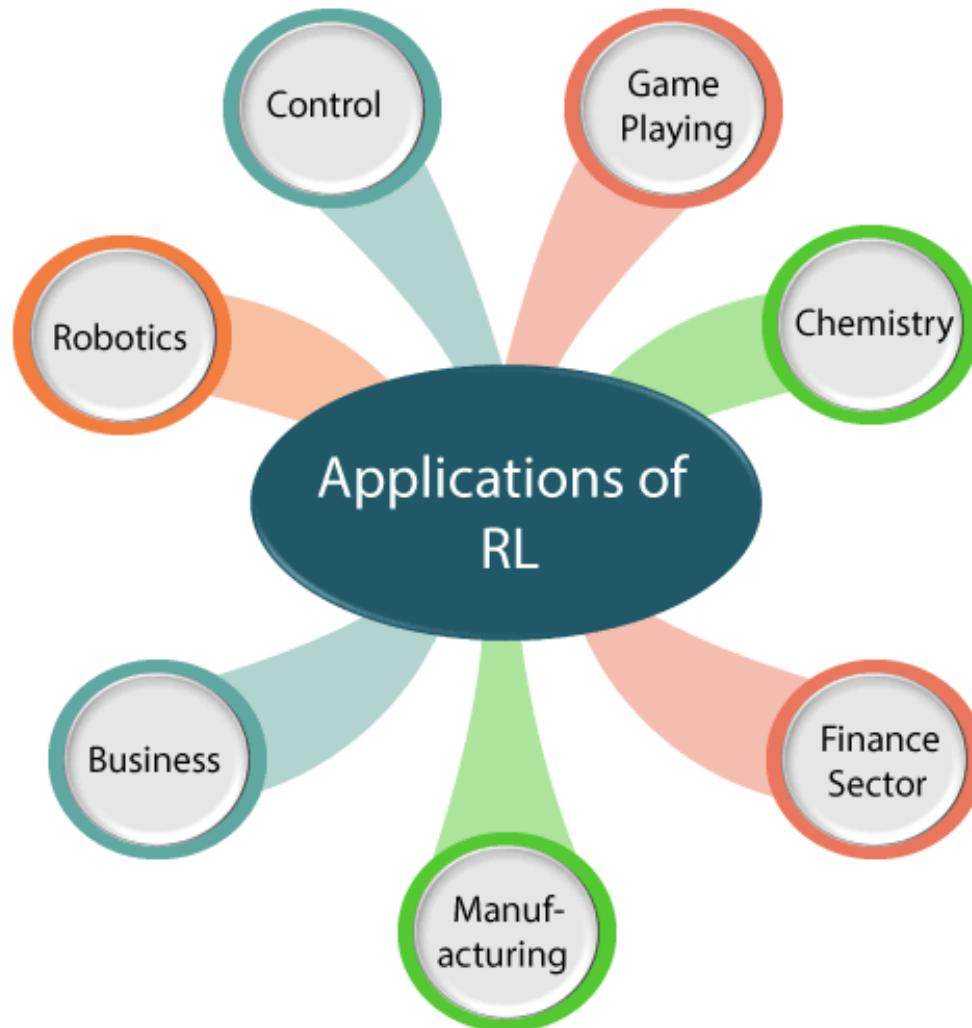
- The Reinforcement Learning and Supervised Learning both are the part of machine learning, but both types of learnings are far opposite to each other.
- The RL agents interact with the environment, explore it, take action, and get rewarded.
- Whereas supervised learning algorithms learn from the labeled dataset and, on the basis of the training, predict the output.

Difference between Reinforcement Learning and Supervised Learning

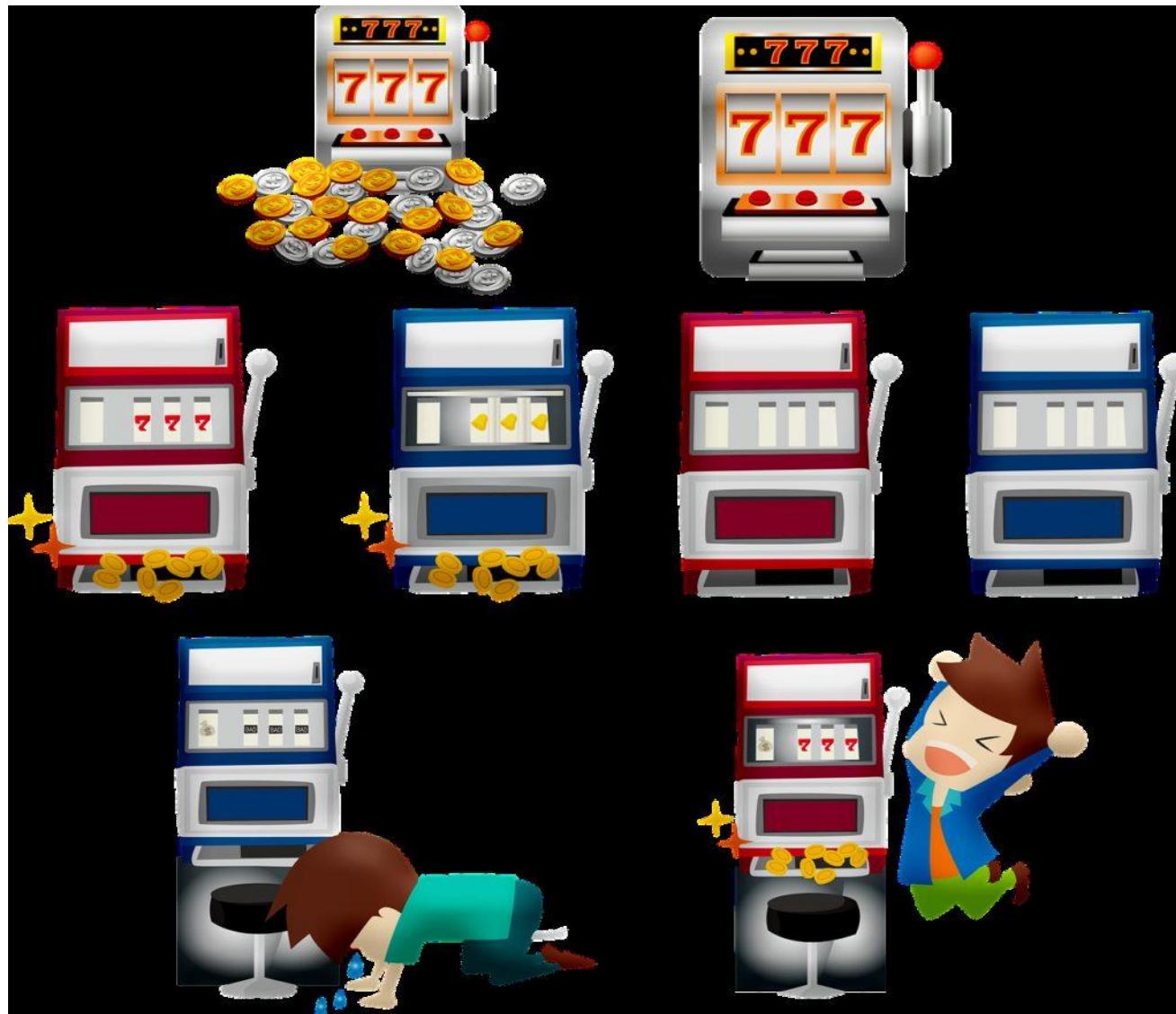


| Reinforcement Learning | Supervised Learning |
|---|--|
| RL works by interacting with the environment. | Supervised learning works on the existing dataset. |
| The RL algorithm works like the human brain works when making some decisions. | Supervised Learning works as when a human learns things in the supervision of a guide. |
| There is no labeled dataset is present | The labeled dataset is present. |
| No previous training is provided to the learning agent. | Training is provided to the algorithm so that it can predict the output. |
| RL helps to take decisions sequentially. | In Supervised learning, decisions are made when input is given. |

Reinforcement Applications



Multi-Armed Bandit Problem



Multi-Armed Bandit Problem

- Multi or K-Armed bandit problem is a learner.
- The learner takes some action and the environment returns some reward value.
- The learner has to find a policy that leads to maximum rewards.
- Suppose, you have to promote some advertisement for any product.
- And you want to test various types of advertisements for it.
- Your aim is to select the advertisement that generates more clicks.

Multi-Armed Bandit Problem

- Obviously, you don't know in advance about the number of clicks each advertisement will get.
- So, you need an algorithm to find the best advertisement.
- The traditional approach is A/B Testing.

Multi-Armed Bandit Problem

- A/B Testing
- In A/B testing, before you start the experiment, you must decide what level of statistical significance is enough.
- After that, you start sending visitors to all advertisements simultaneously. And show these ads to visitors randomly. All advertisements get an equal share of visitors.
- For example, if you have three types of advertisements for any product. So each advertisement gets one-third of all visitors.

Multi-Armed Bandit Problem

- A/B Testing
- When the level of significance is achieved, you look at which advertisement is the best performer.
- So you choose the winner advertisement and remove others.

Multi-Armed Bandit Problem

- A/B Testing
- A/B Testing takes long enough time to conduct this experiment.
- You can't interrupt it in between experiments.
- And you can't change anything.
- Only you wait patiently.
- You are not only waiting for long, moreover, you are also wasting your money.
- Because you are testing on all advertisements.
- Most of them are ineffective. But you can't do anything.

Multi-Armed Bandit Problem

- Frustrating Right?
- That's why the Multi-Armed Bandit Problem comes into the scene.

Multi-Armed Bandit Problem

- Multi-Armed Bandit Problem was introduced by William R. Thompson in 1933.
- He wanted to solve the problem of Medical drug experiments.
- Because it was hard to give ineffective drugs to the patient.
- So clinical trials are one of the first intended applications of the K-Armed Bandit Problem.
- The name multi-armed bandit itself comes from studies of these two guys. Frederick Mosteller and Robert Bush.

Multi-Armed Bandit Problem

- Explore-Exploit Dilemma
- Multi or K-Armed Bandit problem work on the explore-exploit scenario.
- Explore— An option that looks inferior.
- Exploit— Currently the best looking option.

What is the K-Armed Bandit Problem?

- The multi or K-armed bandit problem works on the exploitation-exploration dilemma.
- To understand the multi-armed bandit problem, first, see a one-armed bandit problem.
- Suppose we have a slot machine, which has one lever and a screen. The screen displays three or more wheels.
- When you pull the lever, the game is activated. This single lever represents the single-arm or one-arm bandit.



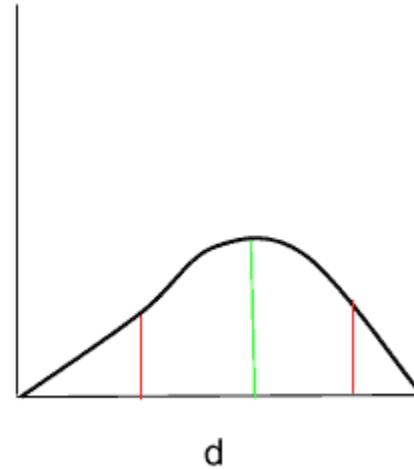
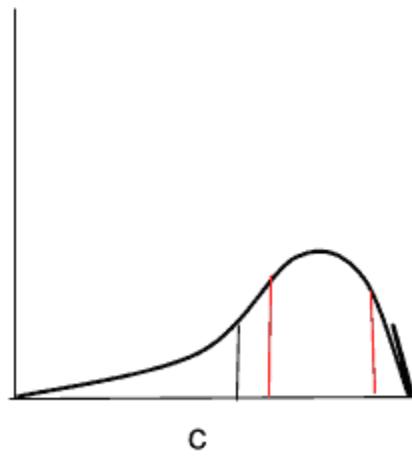
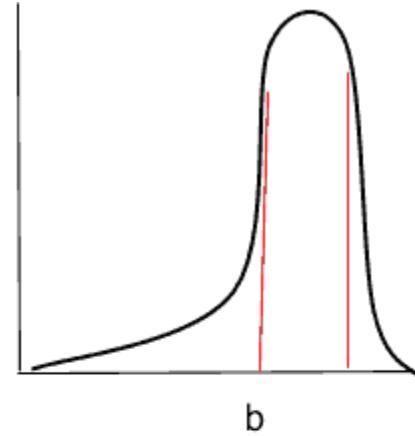
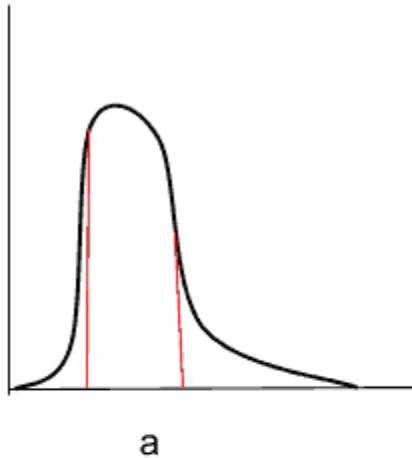
What is the K-Armed Bandit Problem?

- So what does a bandit represent here?
- The people who play on these machines, they lose more money, than winning. These slot machines have a high probability of taking your money than giving you back. That's why it is known as Bandit.
- So what does Multi-Armed Bandit mean now?
- Multi-Armed Bandit is a scenario when you have multiple slot machines. It may be 5, 10, or more.

What is the K-Armed Bandit Problem?

- One important thing to consider here is that each machine has a different probability distribution of success.
- As a player, you don't know before the probability distribution of these machines.
- Our aim is to find out which slot machines have the best distribution.
- Let's replace these 4 slot machines with their associated distribution.

What is the K-Armed Bandit Problem?



What is the K-Armed Bandit Problem?

- So, do you know, which one is the best distribution among four?.
- The answer is the b part distribution. Because it is left-skewed. It has a high mean, median, and mode.
- But most of the time, we found a different distribution that is similar to the optimal one.
- And this is not of goal. Our goal is to find the optimal distribution among all slot machines.
- So, to find out the optimal distribution, we need to do lots of exploration.
- And if we don't do much exploration, then we may settle down with other similar distribution.
- We think that this is the optimal solution but that is not optimal in reality.

What is the K-Armed Bandit Problem?

- So, to solve this Multi-Armed Bandit problem, there are two algorithms.
 - Upper Confidence Bound (UCB).
 - Thompson Sampling.

UCB

- Upper-Confidence Bound action selection uses uncertainty in the action-value estimates for balancing exploration and exploitation.
- Since there is inherent uncertainty in the accuracy of the action-value estimates when we use a sampled set of rewards thus UCB uses uncertainty in the estimates to drive exploration.

$$A_t = \operatorname{argmax}_a (Q_{t(a)} + c \sqrt{\frac{\ln(t)}{N_t(a)}})$$

Exploit

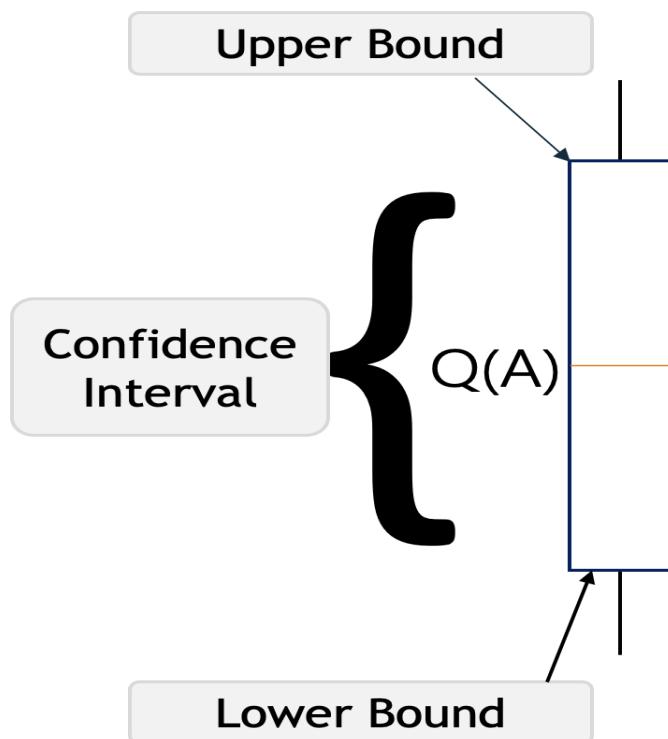
Explore

t = timesteps

$N_t(a)$ = no. of times action (a) is taken

UCB

- $Q_t(a)$ here represents the current estimate for action a at time t . We select the action that has the highest estimated action-value plus the upper-confidence bound exploration term.

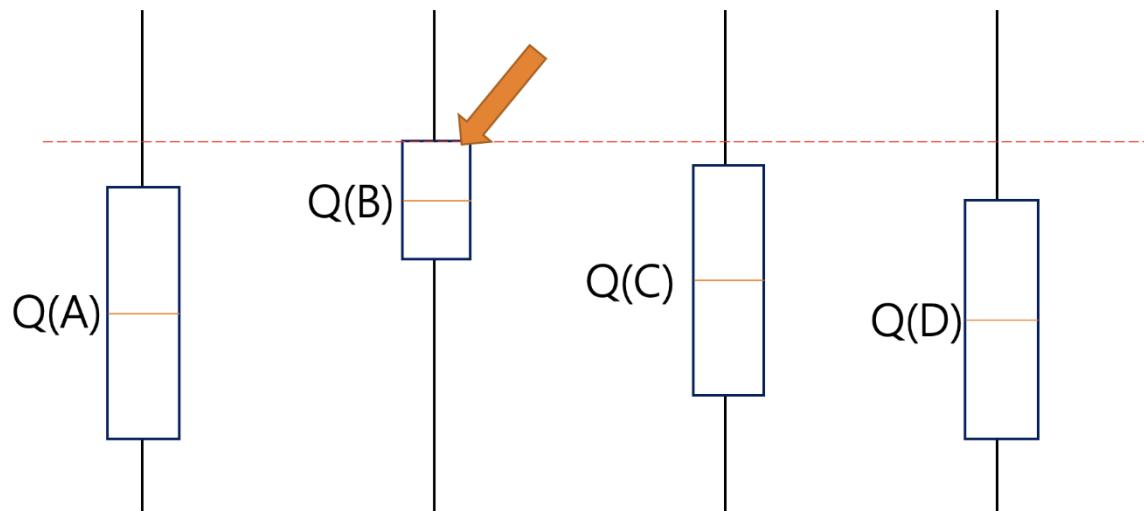


- $Q(A)$ in the above picture represents the current action-value estimate for action A.
- The brackets represent a confidence interval around $Q^*(A)$ which says that we are confident that the actual action-value of action A lies somewhere in this region.
- The lower bracket is called the lower bound, and the upper bracket is the upper bound.
- The region between the brackets is the confidence interval which represents the uncertainty in the estimates.
- If the region is very small, then we become very certain that the actual value of action A is near our estimated value.
- On the other hand, if the region is large, then we become uncertain that the value of action A is near our estimated value.

- The Upper Confidence Bound follows the principle of optimism in the face of uncertainty which implies that if we are uncertain about an action, we should optimistically assume that it is the correct action.
- For example, let's say we have these four actions with associated uncertainties in the picture below, our agent has no idea which is the best action.
- So according to the UCB algorithm, it will optimistically pick the action that has the highest upper bound i.e. A.
- By doing this either it will have the highest value and get the highest reward, or by taking that we will get to learn about an action we know least about.

UCB

- Let's assume that after selecting the action A we end up in a state depicted in the picture below.
- This time UCB will select the action B since $Q(B)$ has the highest upper-confidence bound because its action-value estimate is the highest, even though the confidence interval is small.



UCB

- The UCB algorithm is aptly named because we are only concerned with the upper bound, given that we are trying to find the arm with the highest reward rate.

$$\mu_i + \sqrt{\frac{2\ln(n)}{n_i}}$$

- where μ_i represents the current reward return average of arm i at the current round, n represents the number of trials passed, and n_i represents the number of pulls given to arm i in the playthrough history.

- The above formulation is simple but yet has several interesting implications as explained in the following:
- The upper boundary is proportional to the squared root of $\ln(n)$, which means that when the experiment progresses, all arms have their upper boundaries increases by a factor of squared root of $\ln(n)$.
- This upper boundary is inversely proportional to the squared root of n_i .
- The more times the specific arm has been engaged before in the past, the greater the confidence boundary reduces towards the point estimate.

- The time complexity between the numerator and denominator provides a tension between exploration and exploitation.
- For any increase in n , the UCB increases only by logarithmic time, while for any increase in n_i , the UCB decreases by n_i .
- Thus, an arm that has not been explored as often as other arms will have a bigger UCB component.
- Depending on its current average mean, the overall UCB function representation of that specific arm may be greater than other arms with higher return but smaller components, and consequently enable that arm to be picked.

- The Upper Confidence Bound (UCB) algorithm is often phrased as “optimism in the face of uncertainty”.
- To understand why, consider at a given round that each arm’s reward function can be perceived as a point estimate based on the average rate of reward as observed.
- Drawing intuition from confidence intervals, for each point estimate, we can also incorporate some form of uncertainty boundary around the point estimate.
- In that sense, we have both lower boundary and upper boundary for each arm.

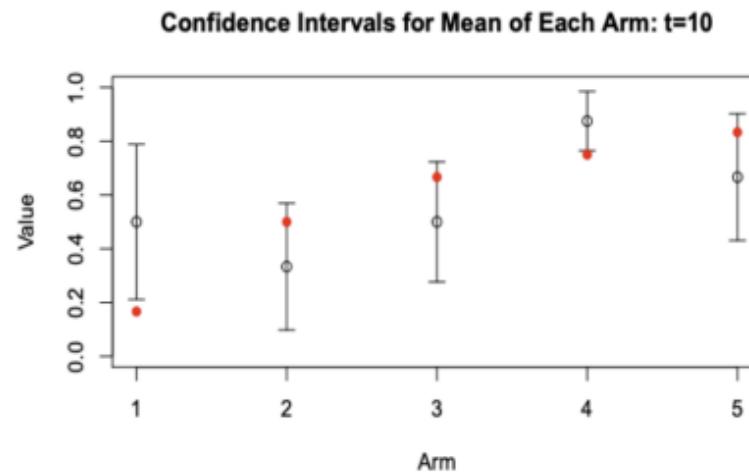
- A great motto for this algorithm would be “optimism in the face of uncertainty”.
- The idea of the greedy algorithm was simple: at each time step, choose the best arm (arm with highest \hat{p}_a).
- The algorithm we discuss now is very similar, but turns out to work a lot better: construct a confidence interval for \hat{p}_a for each arm, and choose the one with the highest POTENTIAL to be best.
- That is, suppose we had three arms with the following estimates and confidence intervals at some time t:
- Arm 1: Estimate is $\hat{p}_1 = 0.75$. Confidence interval is $[0.75 - 0.10, 0.75 + 0.10] = [0.65, 0.85]$.
- Arm 2: Estimate is $\hat{p}_2 = 0.33$. Confidence interval is $[0.33 - 0.25, 0.33 + 0.25] = [0.08, 0.58]$.
- Arm 3: Estimate is $\hat{p}_3 = 0.60$. Confidence interval is $[0.60 - 0.29, 0.60 + 0.29] = [0.31, 0.89]$.

UCB

Algorithm 4 UCB1 Algorithm (Upper Confidence Bound) for Bernoulli Bandits

- 1: **for** $i = 1, 2, \dots, K$ **do**
 - 2: Pull arm i once, observing $r_i \sim \text{Ber}(p_i)$.
 - 3: Estimate $\hat{p}_i = r_i/1$. ▷ Each estimate \hat{p}_i will *initially* either be 1 or 0.
 - 4: **for** $t = K + 1, K + 2, \dots, T$ **do:**
 - 5: Pull arm $a_t = \arg \max_{i \in \{1, 2, \dots, K\}} \left(\hat{p}_i + \sqrt{\frac{2 \ln(t)}{N_t(i)}} \right)$, where $N_t(i)$ is the number of times arm i was pulled before time t .
 - 6: Receive reward $r_t \sim \text{Ber}(p_{a_t})$.
 - 7: Update $N_t(a_t)$ and \hat{p}_{a_t} (using newly observed reward r_t).
-

Suppose we have $K = 5$ arms. The following picture depicts at time $t = 10$ what the confidence intervals may look like. The horizontal lines at the top of each arm represent the upper confidence bound, and the red dots represent the TRUE (unknown) means. The center of each confidence interval are the ESTIMATED means.



Thompson Sampling

- Thompson Sampling is an algorithm that can be used to analyze multi-armed bandit problems.
- Imagine you're in a casino standing in front of three slot machines.
- You have 10 free plays.
- Each machine pays \$1 if you win or \$0 if you lose.
- Each machine pays out according to a different probability distribution and these distributions are unknown to you.
- Your goal is to win as much money as possible.

Thompson Sampling

- It's unlikely you'll ever have to analyze a slot machine, but the scenario maps to many real-world problems.
- For example, suppose you have three different Internet advertising strategies and you want to determine which of them is the best as quickly as possible.
- Or suppose you work for a medical company and you want to determine which of three new drugs is the most effective.

Thompson Sampling

- There are many different algorithms for multi-armed bandit problems.
- You could use your first six plays to see which machine paid the most, and then use your remaining four plays on the best machine found.
- This is sometimes called the epsilon-first algorithm.
- Thompson Sampling is a sophisticated technique that simultaneously balances exploration, where you're trying to find the best machine, with exploitation, where you're playing the best machine known at any point in time.

Thompson Sampling

- A good way to see where this article is headed is to look at the screenshot of a demo run in Figure 1.
- The demo sets up three machine that pay out with probabilities (0.3, 0.7, 0.5) respectively.
- Therefore, machine [0] is the worst and machine [1] is the best.
- These probabilities would be unknown to you in a real problem.
- On each of the 10 trials, three probability values are generated, and the machine with the highest probability is played.
- Generating these probabilities uses the mathematical beta distribution and is the key to Thompson Sampling.

Thompson Sampling

- On each trial, the selected machine is played and either wins (a success) or loses (a failure).
- The current number of successes and failures for each machine are used to determine the probability values.
- At the end of the demo run, machine [0] won 0 times and lost 1 time; machine [1] won 5 times and lost 1 time; machine [2] won 2 times and lost 1 time.
- In this simulation, the best machine was correctly identified, and with 7 wins and 3 losses, there was a net gain of 4.

Thompson Sampling

Let's look at the steps we need in Thompson sampling:

Step 1. At each round n , we consider two numbers for each ad i :

- $N_i^1(n)$ - the number of times the ad i got reward 1 up to round n ,
- $N_i^0(n)$ - the number of times the ad i got reward 0 up to round n .

Step 2. For each ad i , we take a random draw from the distribution below:

$$\theta_i(n) = \beta(N_i^1(n) + 1, N_i^0(n) + 1)$$

Step 3. We select the ad that has the highest $\theta_i(n)$.

At the first step, the total number of rewards(either 0 or 1) is counted.

To understand the second step, we need to look at the Bayesian inference rules-

Algorithm Comparison: Upper Confidence Bound vs. Thompson Sampling:

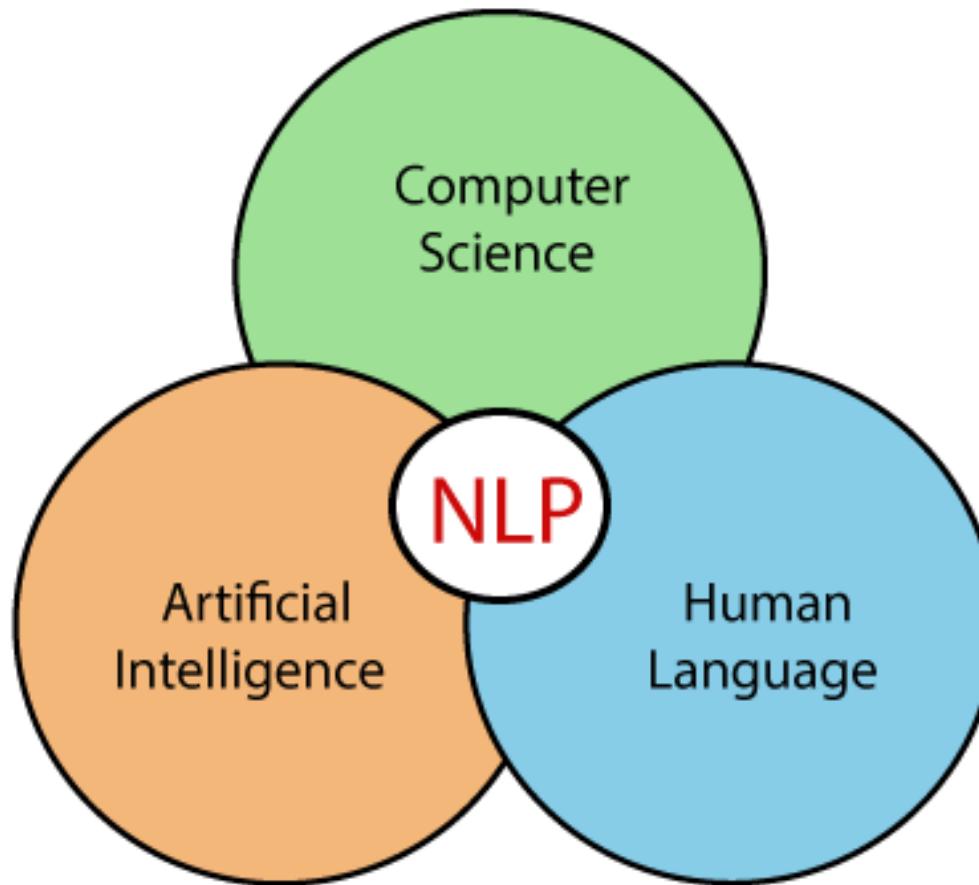


- There is a significant difference between UCB and Thompson sampling.
- Firstly UCB is a deterministic algorithm whereas Thompson sampling is a probabilistic algorithm.
- In UCB you must incorporate the value at every round, you cannot proceed to the next round without adjusting the value.
- In Thompson, you can accommodate delayed feedback
- This means you can update the dataset for your multi-armed bandit problem in a batch manner, that will save your additional computing resource or cost of updating the dataset each time.
- This is the main advantage of Thompson sampling over UCB.

Natural Language Processing

- NLP stands for Natural Language Processing, which is a part of Computer Science, Human language, and Artificial Intelligence.
- It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages.
- It helps developers to organize knowledge for performing tasks such as translation, automatic summarization, Named Entity Recognition (NER), speech recognition, relationship extraction, and topic segmentation.

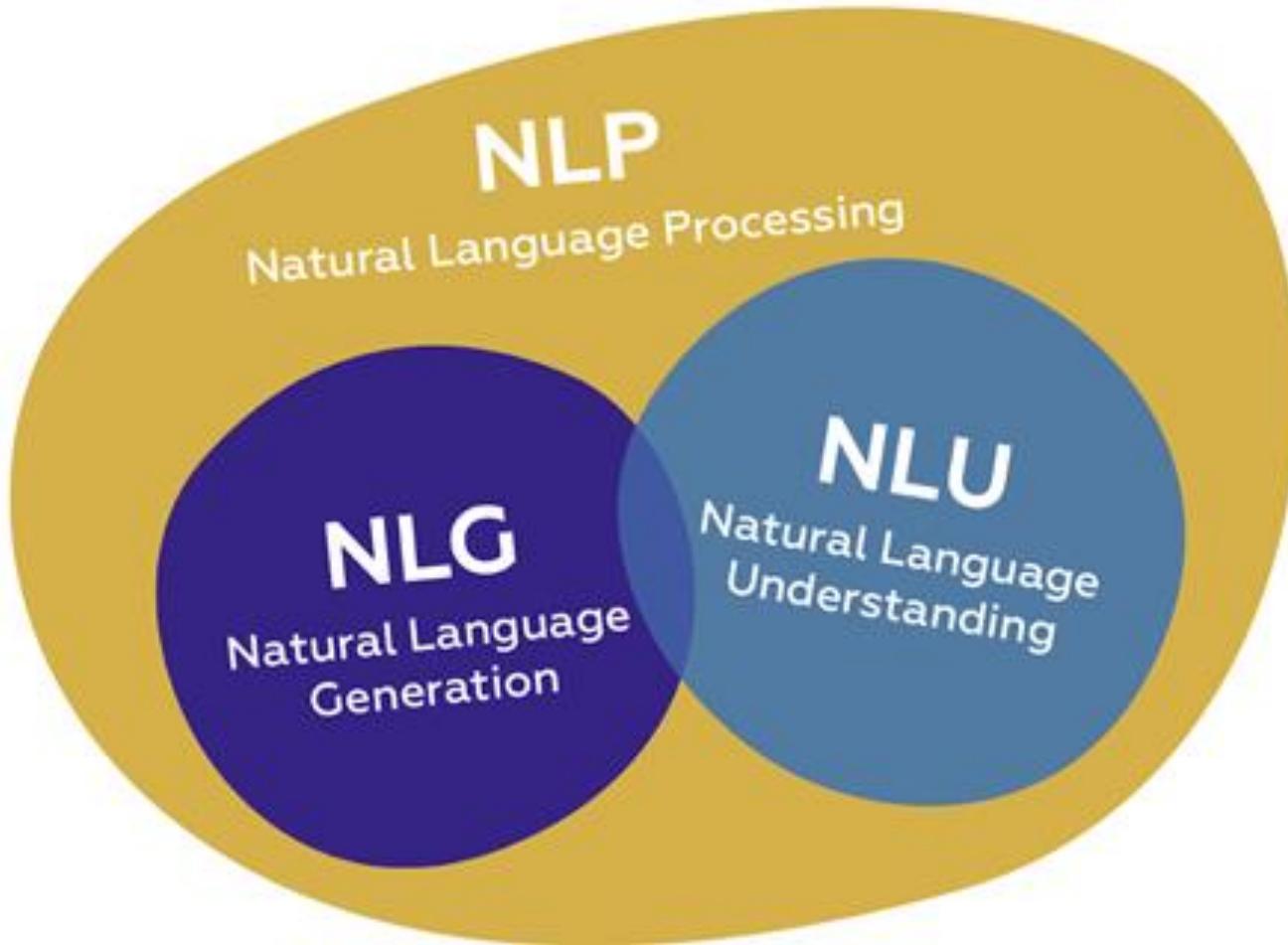
Natural Language Processing



Natural Language Processing

- Natural Language Processing (NLP) refers to AI method of communicating with an intelligent systems using a natural language such as English.
- Processing of Natural Language is required when you want an intelligent system like robot to perform as per your instructions, when you want to hear decision from a dialogue based clinical expert system, etc.
- The field of NLP involves making computers to perform useful tasks with the natural languages humans use. The input and output of an NLP system can be –
 - Speech
 - Written Text

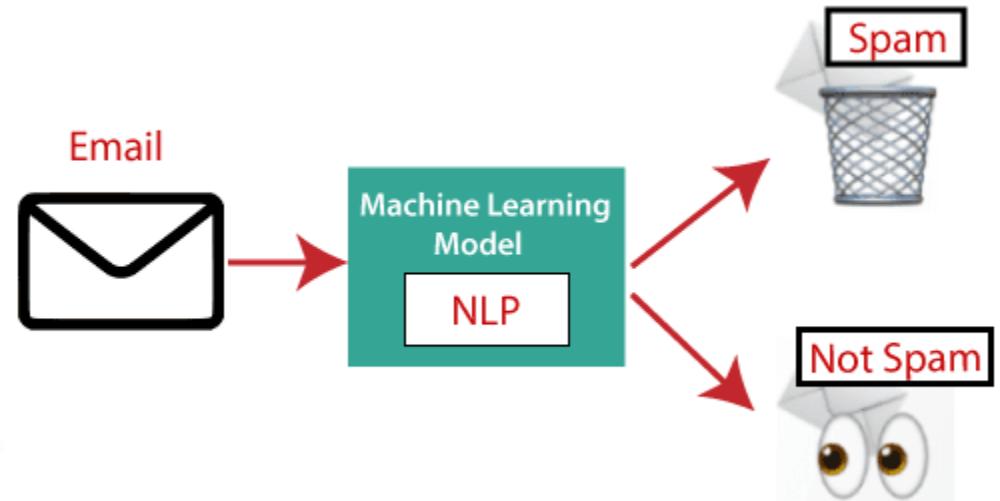
Components of NLP



Components of NLP

- Natural Language Understanding (NLU)
 - Understanding involves the following tasks –
 - Mapping the given input in natural language into useful representations.
 - Analyzing different aspects of the language.
- Natural Language Generation (NLG)
 - It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation.

Applications of NLP



Applications of NLP

- Translation
- Spelling Correction
- Speech Recognition
- Chatbot
- Information extraction

Components of NLP

- It involves –
 - **Text planning** – It includes retrieving the relevant content from knowledge base.
 - **Sentence planning** – It includes choosing required words, forming meaningful phrases, setting tone of the sentence.
 - **Text Realization** – It is mapping sentence plan into sentence structure.

Difficulties in NLU

- **Lexical ambiguity** – It is at very primitive level such as word-level.
 - For example, treating the word “board” as noun or verb?
- **Syntax Level ambiguity** – A sentence can be parsed in different ways.
 - For example, “He lifted the beetle with red cap.” – Did he use cap to lift the beetle or he lifted a beetle that had red cap?
- **Referential ambiguity** – Referring to something using pronouns. For example, Rima went to Gauri. She said, “I am tired.” – Exactly who is tired?

How to build an NLP pipeline

- Step1: Sentence Segmentation
- Sentence Segment is the first step for building the NLP pipeline. It breaks the paragraph into separate sentences.
- Example: Consider the following paragraph -
 - Independence Day is one of the important festivals for every Indian citizen. It is celebrated on the 15th of August each year ever since India got independence from the British rule. The day celebrates independence in the true sense.



How to build an NLP pipeline

- Sentence Segment produces the following result:
- "Independence Day is one of the important festivals for every Indian citizen."
- "It is celebrated on the 15th of August each year ever since India got independence from the British rule."
- "This day celebrates independence in the true sense."



How to build an NLP pipeline

- Step2: Word Tokenization
- Word Tokenizer is used to break the sentence into separate words or tokens.
- Example:
- Veb offers Corporate Training, Summer Training, Online Training, and Winter Training.
- Word Tokenizer generates the following result:
- "Veb", "offers", "Corporate", "Training", "Summer", "Training", "Online", "Training", "and", "Winter", "Training", ":"

How to build an NLP pipeline

- Step3: Stemming
- Stemming is used to normalize words into its base form or root form.
- For example, celebrates, celebrated and celebrating, all these words are originated with a single root word "celebrate."
- The big problem with stemming is that sometimes it produces the root word which may not have any meaning.
- For Example, intelligence, intelligent, and intelligently, all these words are originated with a single root word "intelligen." In English, the word "intelligen" do not have any meaning.

How to build an NLP pipeline

- Step 4: Lemmatization
- Lemmatization is quite similar to the Stemming. It is used to group different inflected forms of the word, called Lemma.
- The main difference between Stemming and lemmatization is that it produces the root word, which has a meaning.
- For example: In lemmatization, the words intelligence, intelligent, and intelligently has a root word intelligent, which has a meaning.

How to build an NLP pipeline

- Step 5: Identifying Stop Words
- In English, there are a lot of words that appear very frequently like "is", "and", "the", and "a". NLP pipelines will flag these words as stop words. Stop words might be filtered out before doing any statistical analysis.
- Example: He is a good boy.

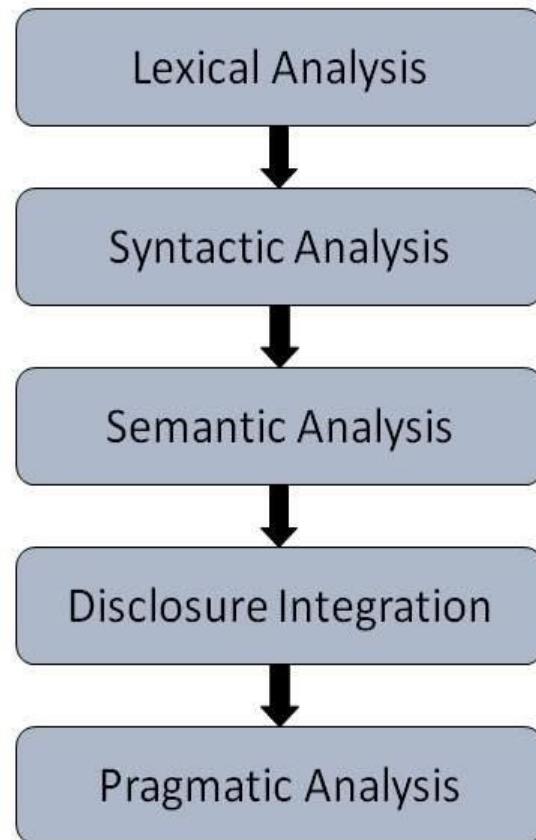
How to build an NLP pipeline

- Step 6: Dependency Parsing
- Dependency Parsing is used to find that how all the words in the sentence are related to each other.
- Step 7: POS tags
- POS stands for parts of speech, which includes Noun, verb, adverb, and Adjective. It indicates that how a word functions with its meaning as well as grammatically within the sentences. A word has one or more parts of speech based on the context in which it is used.
- Example: "Google" something on the Internet.

How to build an NLP pipeline

- Step 8: Named Entity Recognition (NER)
- Named Entity Recognition (NER) is the process of detecting the named entity such as person name, movie name, organization name, or location.
- Example: Steve Jobs introduced iPhone at the Macworld Conference in San Francisco, California.
- Step 9: Chunking
- Chunking is used to collect the individual piece of information and grouping them into bigger pieces of sentences.

Steps in NLP



Steps in NLP

- Lexical Analysis and Morphological
- The first phase of NLP is the Lexical Analysis. This phase scans the source code as a stream of characters and converts it into meaningful lexemes.
- It divides the whole text into paragraphs, sentences, and words.

Steps in NLP

- 2. Syntactic Analysis (Parsing)
- Syntactic Analysis is used to check grammar, word arrangements, and shows the relationship among the words.
- Example: Agra goes to the Poonam
- In the real world, Agra goes to the Poonam, does not make any sense, so this sentence is rejected by the Syntactic analyzer.
- 3. Semantic Analysis
- Semantic analysis is concerned with the meaning representation. It mainly focuses on the literal meaning of words, phrases, and sentences.

Steps in NLP

- 4. Discourse Integration
- Discourse Integration depends upon the sentences that proceeds it and also invokes the meaning of the sentences that follow it.
- 5. Pragmatic Analysis
- Pragmatic is the fifth and last phase of NLP. It helps you to discover the intended effect by applying a set of rules that characterize cooperative dialogues.
- For Example: "Open the door" is interpreted as a request instead of an order.

NLP APIs

- IBM Watson API
- Chatbot API
- Speech to text API
- Sentiment Analysis API
- Translation API by SYSTRAN (free)
- Text Analysis API by AYLIEN
- Cloud NLP API (free)
- Google Cloud Natural Language API

NLP Libraries

- Scikit-learn: It provides a wide range of algorithms for building machine learning models in Python.
- Natural language Toolkit (NLTK): NLTK is a complete toolkit for all NLP techniques.
- Pattern: It is a web mining module for NLP and machine learning.
- TextBlob: It provides an easy interface to learn basic NLP tasks like sentiment analysis, noun phrase extraction, or pos-tagging.
- Quepy: Quepy is used to transform natural language questions into queries in a database query language.
- SpaCy: SpaCy is an open-source NLP library which is used for Data Extraction, Data Analysis, Sentiment Analysis, and Text Summarization.
- Gensim: Gensim works with large datasets and processes data streams.

Where Can Developers Use NLP Algorithms For?



- **Summarize blocks of text** (Summarizer tool)
- <https://algorithmia.com/>
- Create a **chat bot**
- **Parsey McParseface** --- Parse sentences with ease.
- **Automatically generate keyword tags**
- **Identify the type of entity extracted,**
- Use Sentiment Analysis to **identify the sentiment of a string of text**,
- **Reduce words to their root**, or stem, using PorterStemmer, or **break up text into tokens** using Tokenizer.

Open Source NLP Libraries



- Apache OpenNLP: a machine learning toolkit that provides tokenizers, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, coreference resolution, and more.
- Natural Language Toolkit (NLTK): a Python library that provides modules for processing text, classifying, tokenizing, stemming, tagging, parsing, and more.
- Standford NLP: a suite of NLP tools that provide part-of-speech tagging, the named entity recognizer, coreference resolutionsystem, sentiment analysis, and more.
- MALLET: a Java package that provides Latent Dirichlet Allocation, document classification, clustering, topic modeling, information extraction, and more.

- Install NLTK 3.0, downloadable for free from <http://nltk.org/>
- The book module contains the data
- Any time we want to find out about these texts, we just have to enter their names
- A concordance view shows us every occurrence of a given word, together with some context.
- A concordance permits us to see words in context.
- It is one thing to automatically detect that a particular word occurs in a text, and to display some words that appear in the same context.
- However, we can also determine the *location* of a word in the text: how many words from the beginning it appears.
- This positional information can be displayed using a **dispersion plot**.
- **Counting Vocabulary**

- Frequency Distributions
- Fine-grained Selection of Words(Long Words)
- Collocations and Bigrams
- A **collocation** is a sequence of words that occur together unusually often.
- Thus *red wine* is a collocation, whereas *the wine* is not. , *maroon wine* sounds definitely odd.

Automatic Natural Language Understanding

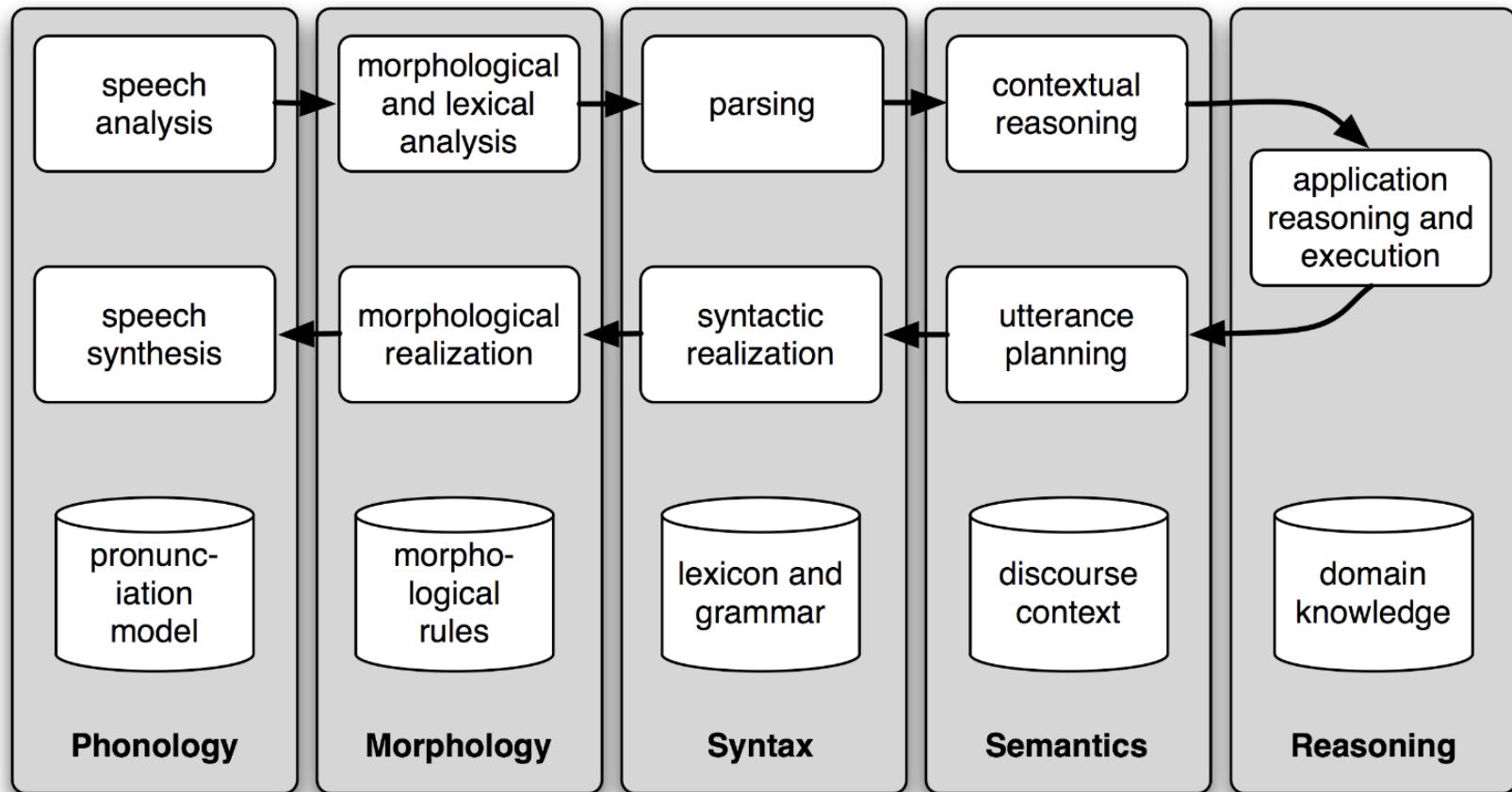


- **Word Sense Disambiguation**
 - serve: help with food or drink; hold an office; put ball into play
 - dish: plate; course of a meal; communications device
- **Pronoun Resolution**
 - A deeper kind of language understanding is to work out "who did what to whom"
- **Generating Language Output**
 - a. **Text:** ... The thieves stole the paintings. They were subsequently sold. ...
 - b. **Human:** Who or what was sold? (**Question to machine**)
 - c. **Machine:** The paintings. (**Machine Answers**)

Automatic Natural Language Understanding



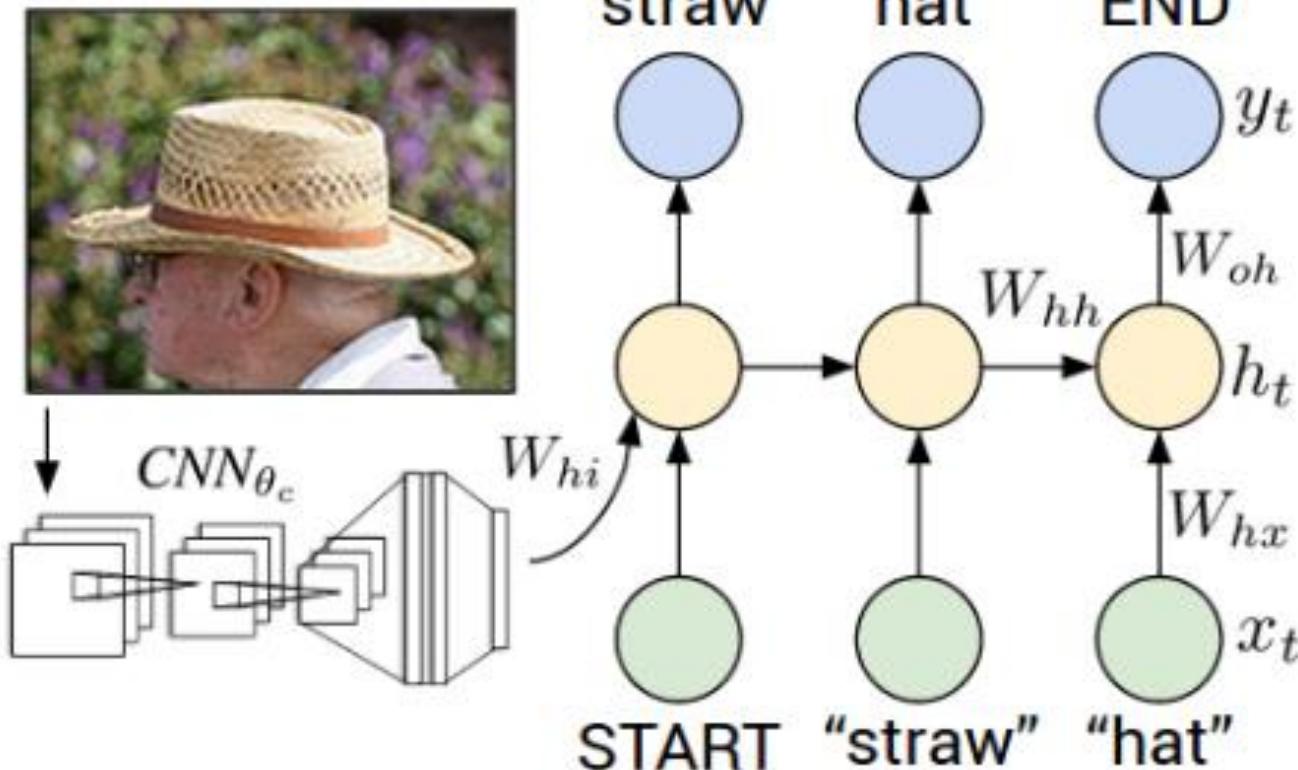
- Spoken Dialog Systems



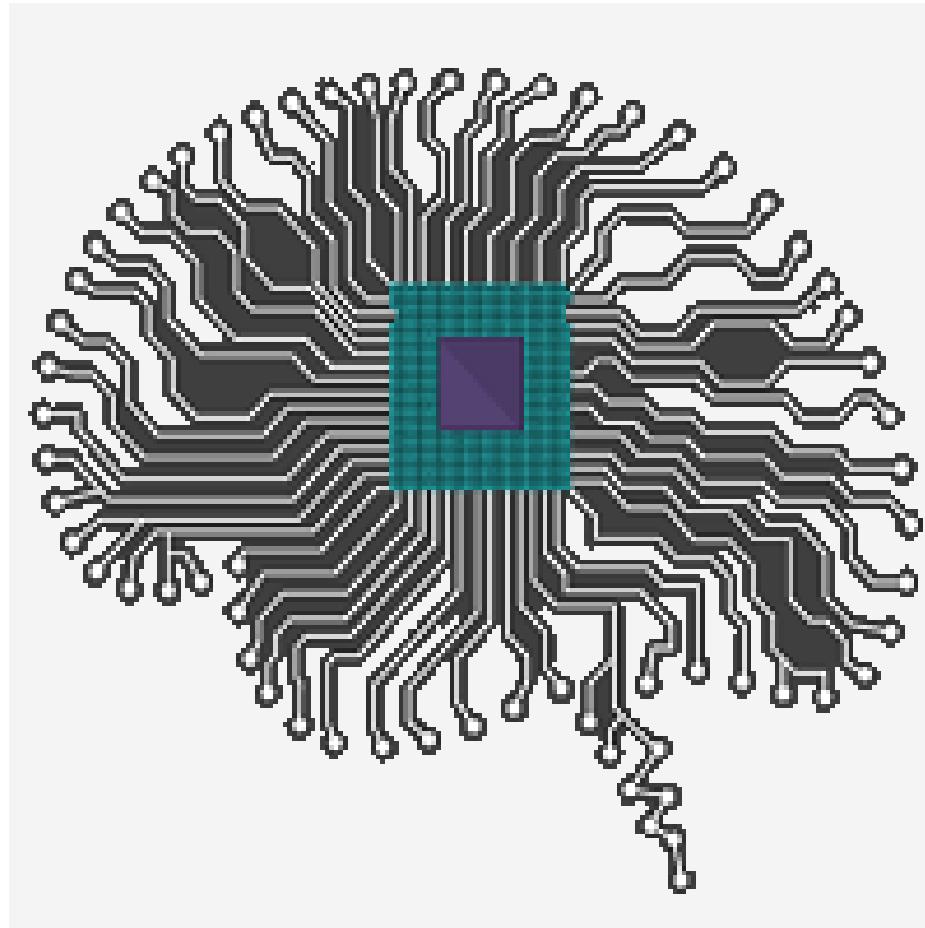
Deep Learning

- conda install theano
- Tensor flow
- Keras

Image Captioning in Deep Learning



Artificial Neural Network



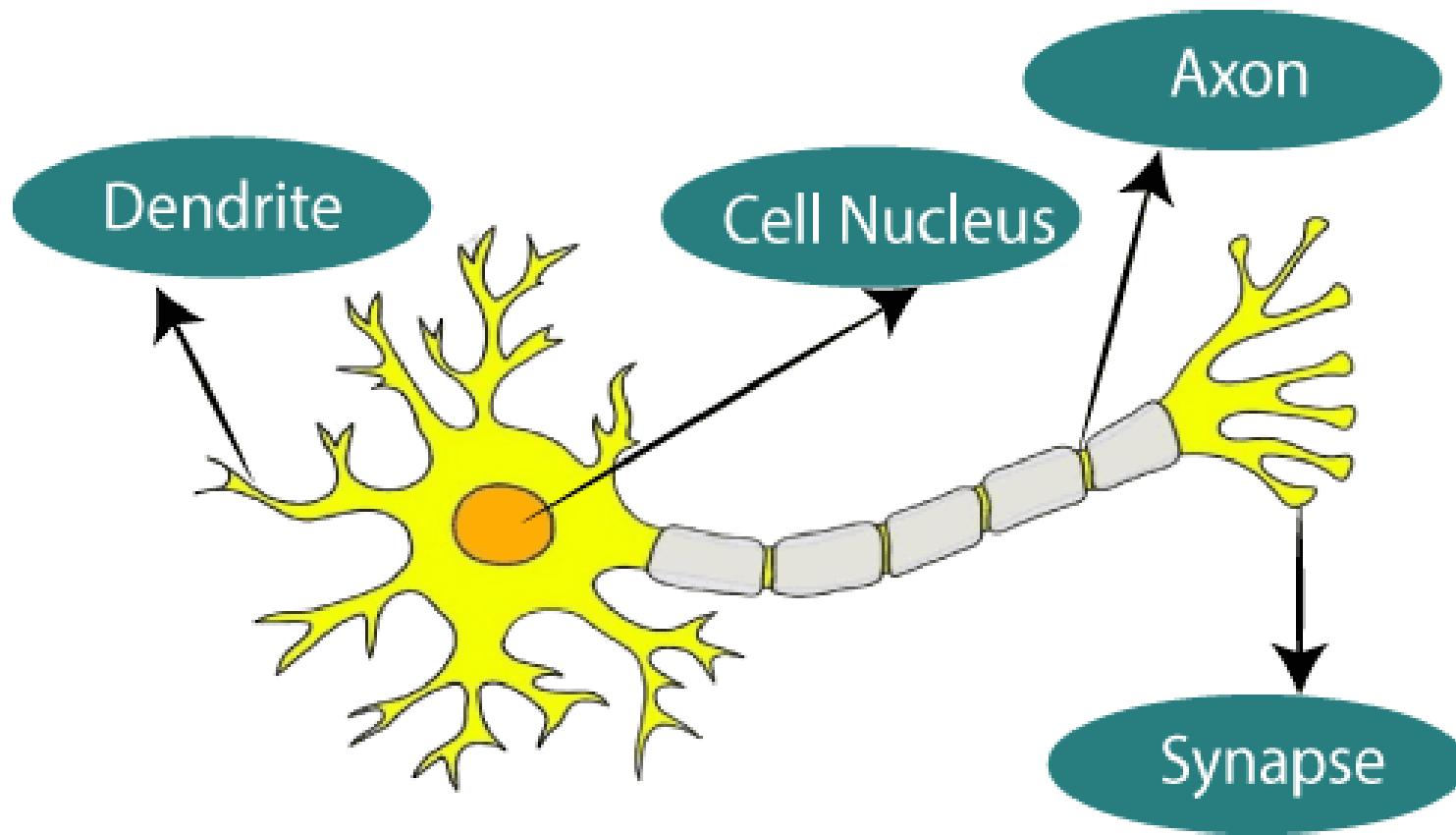
Artificial Neural Network

- The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain.
- An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain.
- Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks.
- These neurons are known as nodes.

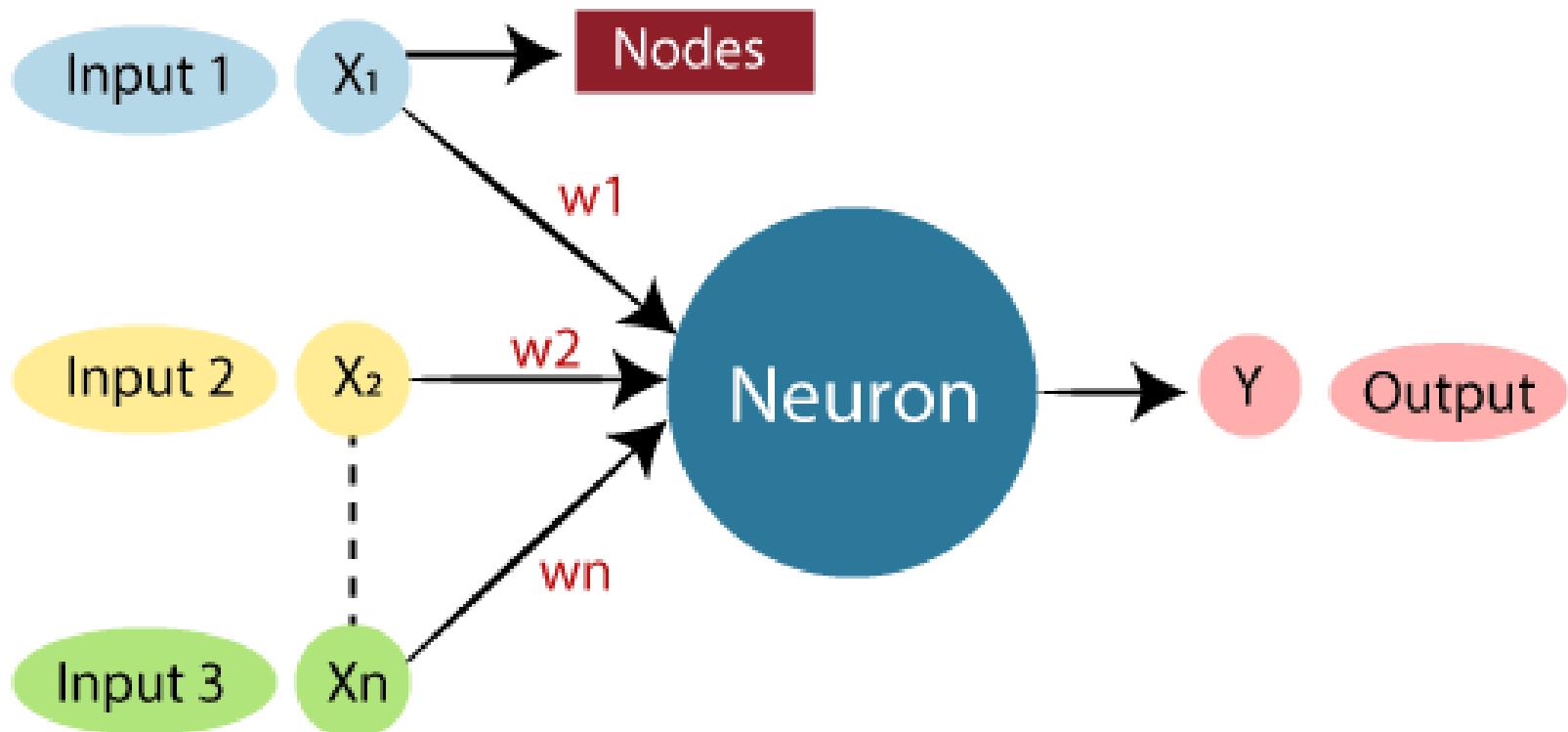
What is Artificial Neural Network?

- The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain.
- Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks.
- These neurons are known as nodes.

What is Artificial Neural Network?



What is Artificial Neural Network?



What is Artificial Neural Network?

- Dendrites from Biological Neural Network represent inputs in Artificial Neural Networks, cell nucleus represents Nodes, synapse represents Weights, and Axon represents Output.

Relationship between Biological neural network and artificial neural network:

| Biological Neural Network | Artificial Neural Network |
|---------------------------|---------------------------|
| Dendrites | Inputs |
| Cell nucleus | Nodes |
| Synapse | Weights |
| Axon | Output |

What is Artificial Neural Network?

- An Artificial Neural Network in the field of Artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner.
- The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.

What is Artificial Neural Network?

- There are around 1000 billion neurons in the human brain.
- Each neuron has an association point somewhere in the range of 1,000 and 100,000.
- In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data when necessary from our memory parallelly.
- We can say that the human brain is made up of incredibly amazing parallel processors.



What is Artificial Neural Network?

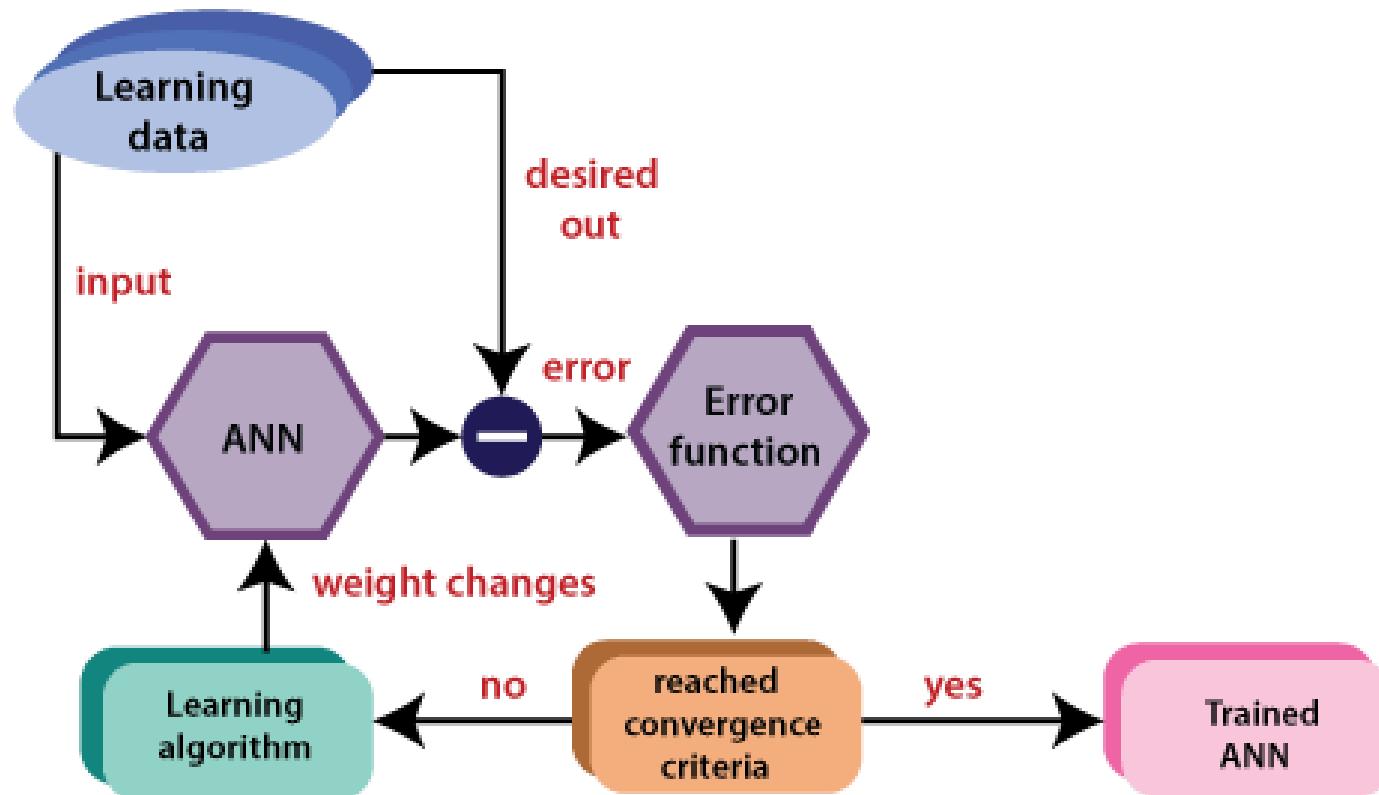
- We can understand the artificial neural network with an example, consider an example of a digital logic gate that takes an input and gives an output.
- "OR" gate, which takes two inputs.
- If one or both the inputs are "On," then we get "On" in output.
- If both the inputs are "Off," then we get "Off" in output.
- Here the output depends upon input.
- Our brain does not perform the same task.
- The outputs to inputs relationship keep changing because of the neurons in our brain, which are "learning."

The architecture of an artificial neural network



- To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of.
- In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers.

ANN Supervised Learning



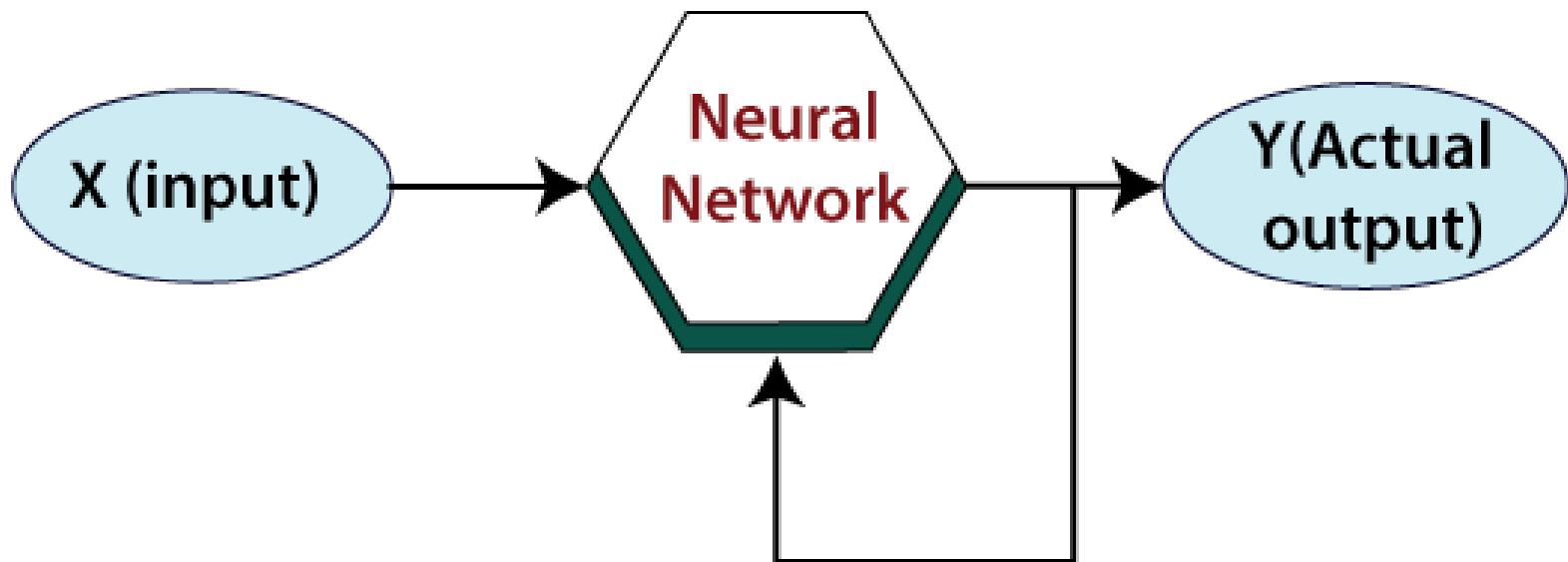
ANN Supervised Learning

- Supervised learning consists of two words supervised and learning. Supervise intends to guide.
- We have supervisors whose duty is to guide and show the way.
- We can see a similar case in the case of learning. Here the machine or program is learning with the help of the existing data set.
- We have a data set, and we assume the results of new data relying upon the behavior of the existing data sets.
- It implies the existing data sets acts as a supervisor or boss to find the new data.
- A basic example being electronic gadgets price prediction.
- The price of electronic gadgets is predicted depending on what is observed with the prices of other digital gadgets.

ANN Supervised Learning

- During the training of artificial neural networks under supervised learning, the input vector is given to the network, which offers an output vector.
- Afterward, the output vector is compared with the desired output vector.
- An error signal is produced if there is a difference between the actual output and the desired output vector.
- Based on this error signal, the weight is adjusted until the actual output is matched with the desired output.

ANN UnSupervised Learning



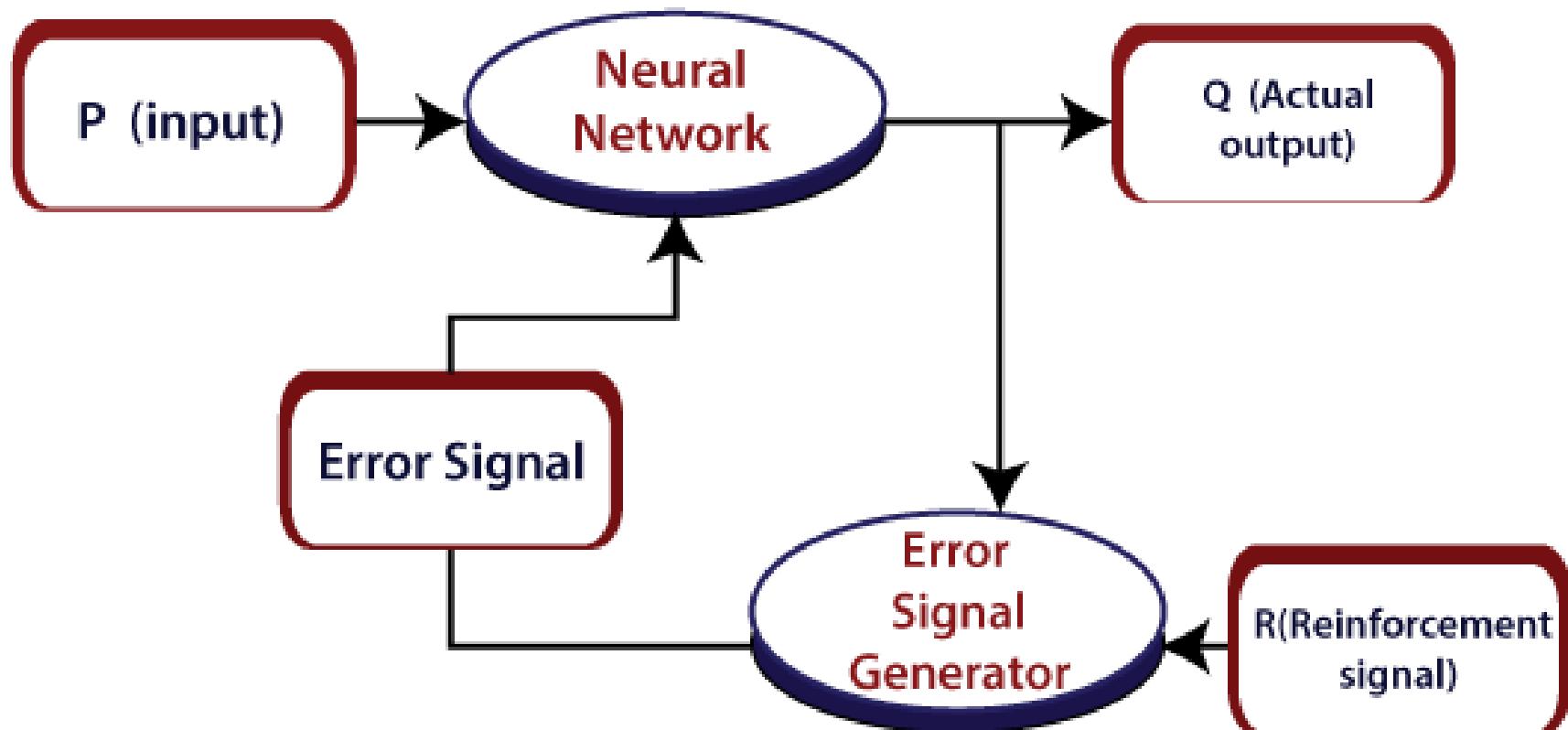
ANN UnSupervised Learning

- In this learning, the program learns by dividing the data with similar characteristics into similar groups. In supervised learning, the data are grouped, relying upon similar characteristics.
- In this situation, there are no existing data to look for direction.
- In other words, there is no supervisor.
- During the training of the artificial neural network under unsupervised learning, the input vectors of a comparative type are joined to form clusters.
- At the point when a new input pattern is implemented, then the neural network gives an output response showing the class to which the input pattern belongs.
- There is no feedback from the environment about what should be the ideal output and if it is either correct or incorrect.
- Consequently, in this type of learning, the network itself must find the patterns and features from the input data and the connection for the input data over the output.

Reinforcement learning:

- Reinforcement Learning (RL) is a technique that helps to solve control optimization issues. By using control optimization, we can recognize the best action in each state visited by the system in order to optimize some objective function.
- Typically, reinforcement learning comes into existence when the system has a huge number of states and has a complex stochastic structure, which is not responsible to closed-form analysis.
- If issues have a relatively small number of states, then the random structure is relatively simple, so that one can utilize dynamic programming.

Reinforcement learning:



Reinforcement learning:

- As the name suggests, this kind of learning is used to strengthen the network over some analyst data.
- This learning procedure is like supervised learning.
- In reinforcement learning, during the training of the network, the network gets some feedback from the system.
- This makes it fairly like supervised learning.
- The feedback acquired here is evaluative, not instructive, which implies there is no instructor as in supervised learning.
- After getting the feedback, the networks perform modifications of the weights to get better Analyst data in the future.

Activation Function:

- Activation functions refer to the functions used in neural networks to compute the weighted sum of input and biases, which is used to choose the neuron that can be fire or not.
- It controls the presented information through some gradient processing, normally gradient descent.
- It produces an output for the neural network that includes the parameters in the data.
- Activation function can either be linear or non-linear, relying on the function it shows.
- It is used to control the output of outer neural networks across various areas, such as speech recognition, segmentation, fingerprint detection, cancer detection system, etc.
- In the artificial neural network, we can use activation functions over the input to get the precise output.
- These are some activation functions that are used in ANN.

Linear Activation Function:

- The equation of the linear activation function is the same as the equation of a straight line i.e.
- $Y = MX + C$
- If we have many layers and all the layers are linear in nature, then the final activation function of the last layer is the same as the linear function of the first layer. The range of a linear function is –infinitive to + infinitive.
- Linear activation function can be used at only one place that is the output layer.

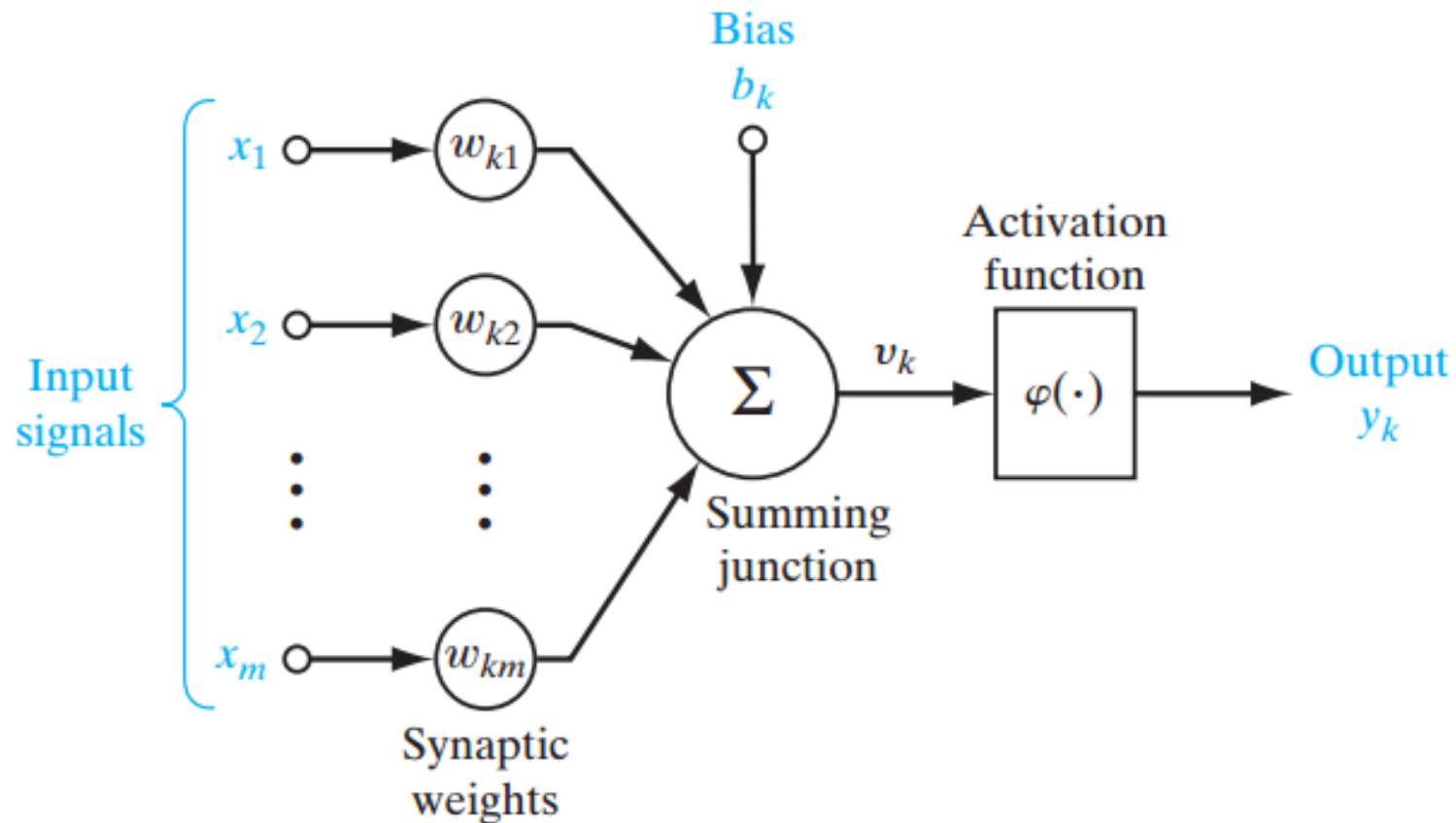
Sigmoid Activation function

- Sigmoid function refers to a function that is projected as S - shaped graph.
- $A = 1/(1+e^{-x})$
- This function is non-linear, and the values of x lie between -2 to +2. So that the value of X is directly proportional to the values of Y. It means a slight change in the value of x would also bring about changes in the value of y.

Tanh Activation function

- The activation function, which is more efficient than the sigmoid function is Tanh function.
- Tanh function is also known as Tangent Hyperbolic Function.
- It is a mathematical updated version of the sigmoid function. Sigmoid and Tanh function are similar to each other and can be derived from each other.
- $F(x) = \tanh(x) = 2/(1+e^{-2x}) - 1$
- OR
- $\tanh(x) = 2 * \text{sigmoid}(2x) - 1$
- This function is non-linear, and the value range lies between -1 to +1

Single Layer Network



Single Layer Network

- The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias.
- This computation is represented in the form of a transfer function.

$$\sum_{i=1}^n w_i * x_i + b$$

- It determines weighted total is passed as an input to an activation function to produce the output.
- Activation functions choose whether a node should fire or not.
- Only those who are fired make it to the output layer.
- There are distinctive activation functions available that can be applied upon the sort of task we are performing.

Advantages of Artificial Neural Network (ANN)

- Parallel processing capability:
- Artificial neural networks have a numerical value that can perform more than one task simultaneously.
- Storing data on the entire network:
- Data that is used in traditional programming is stored on the whole network, not on a database.
- The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.
- Capability to work with incomplete knowledge:
- After ANN training, the information may produce output even with inadequate data.
- The loss of performance here relies upon the significance of missing data.

Advantages of Artificial Neural Network (ANN)

- Having a memory distribution:
- For ANN to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these examples to the network.
- The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.
- Having fault tolerance:
- Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

Disadvantages of Artificial Neural Network:

- Assurance of proper network structure:
- There is no particular guideline for determining the structure of artificial neural networks. The appropriate network structure is accomplished through experience, trial, and error.
- Unrecognized behavior of the network:
- It is the most significant issue of ANN. When ANN produces a testing solution, it does not provide insight concerning why and how. It decreases trust in the network.
- Hardware dependence:
- Artificial neural networks need processors with parallel processing power, as per their structure. Therefore, the realization of the equipment is dependent.

Disadvantages of Artificial Neural Network:

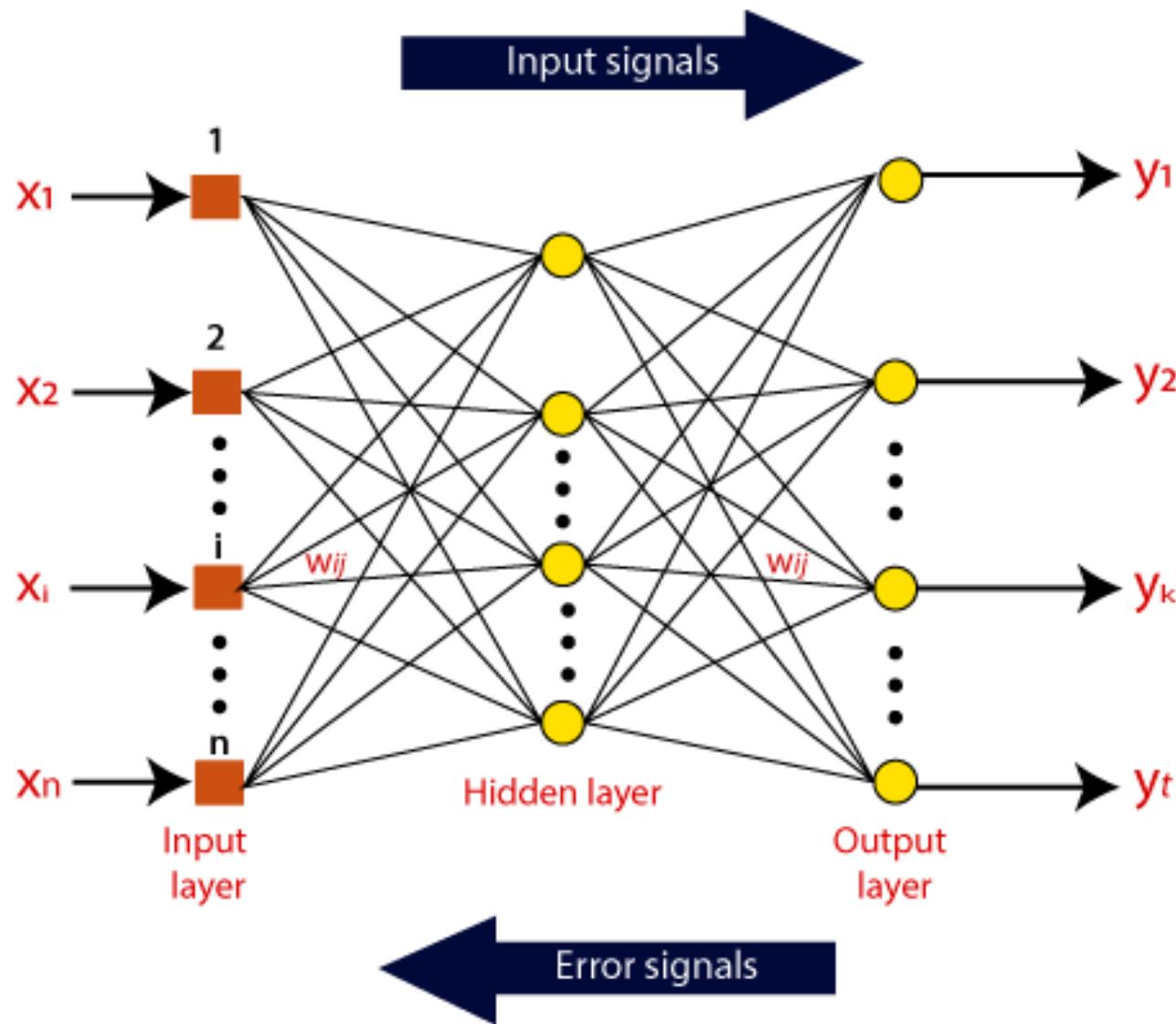
- Difficulty of showing the issue to the network:
- ANNs can work with numerical data. Problems must be converted into numerical values before being introduced to ANN.
- The presentation mechanism to be resolved here will directly impact the performance of the network. It relies on the user's abilities.
- The duration of the network is unknown:
- The network is reduced to a specific value of the error, and this value does not give us optimum results.



How do artificial neural networks work?

- Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes.
- The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights.
- The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector.
- These inputs are then mathematically assigned by the notations $x(n)$ for every n number of inputs.

How do artificial neural networks work?



How do artificial neural networks work?

- Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem).
- In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network.
- All the weighted inputs are summarized inside the computing unit.

How do artificial neural networks work?

- If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response.
- Bias has the same input, and weight equals to 1.
- Here the total of weighted inputs can be in the range of 0 to positive infinity.
- Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.

How do artificial neural networks work?

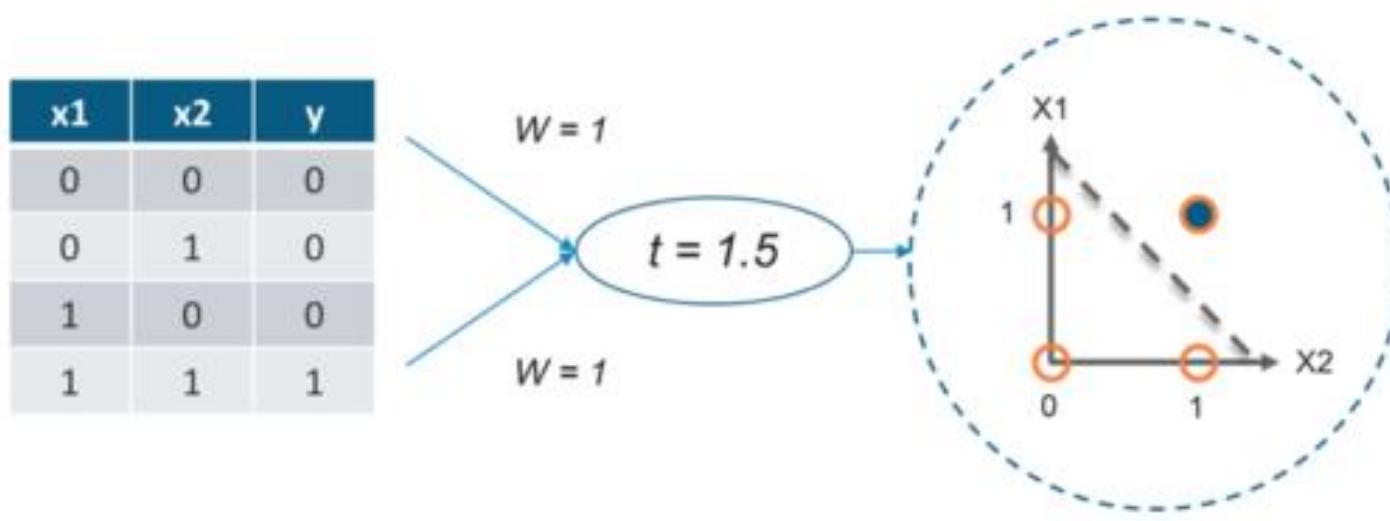
- The activation function refers to the set of transfer functions used to achieve the desired output.
- There is a different kind of the activation function, but primarily either linear or non-linear sets of functions.
- Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions.

How do artificial neural networks work?

- Binary:
- In binary activation function, the output is either a one or a 0. Here, to accomplish this, there is a threshold value set up. If the net weighted input of neurons is more than 1, then the final output of the activation function is returned as one or else the output is returned as 0.
- Sigmoidal Hyperbolic:
- The Sigmoidal Hyperbola function is generally seen as an "S" shaped curve. Here the tan hyperbolic function is used to approximate output from the actual net input. The function is defined as:
$$F(x) = \frac{1}{1 + \exp(-\text{????}x)}$$
- Where ???? is considered the Steepness parameter.

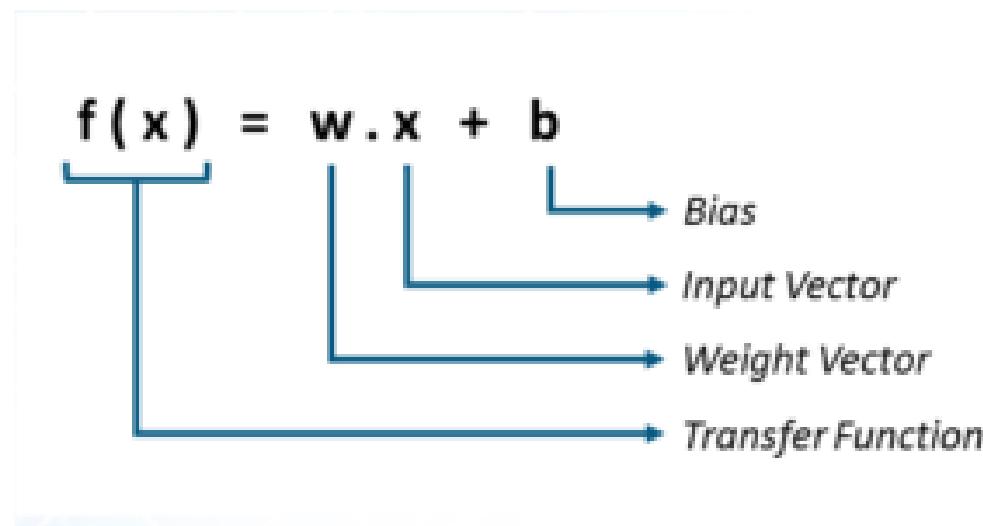
How do artificial neural networks work?

- The below diagram shows the above idea of classifying the inputs of AND Gate using a perceptron:



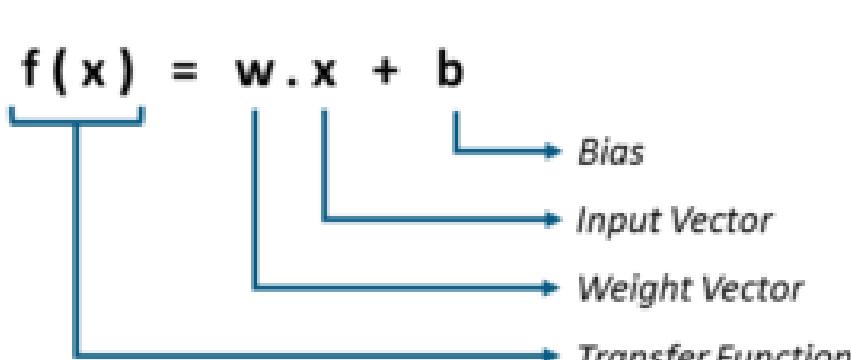
How do artificial neural networks work?

- Mathematically, one can represent a perceptron as a function of weights, inputs and bias (vertical offset):



How do artificial neural networks work?

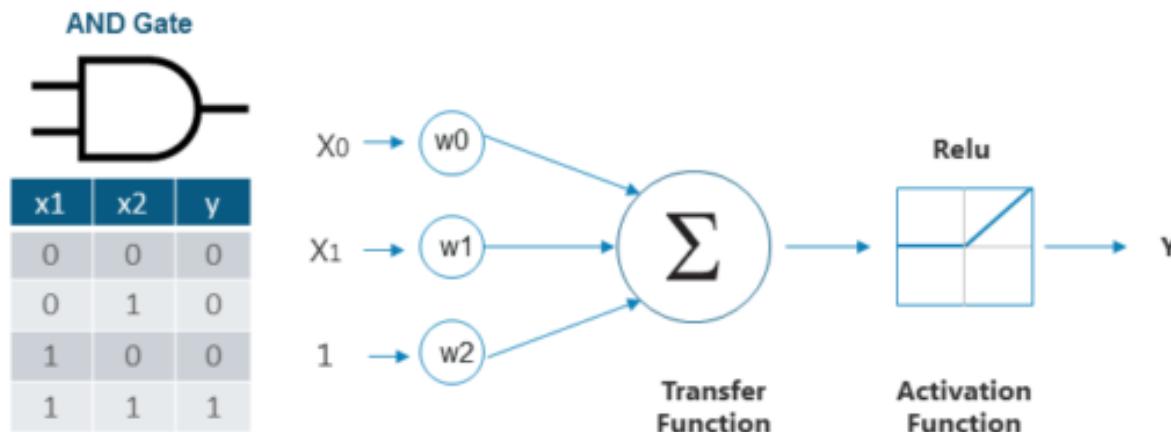
- Each of the input received by the perceptron has been weighted based on the amount of its contribution for obtaining the final output.
- Bias allows us to shift the decision line so that it can best separate the inputs into two classes.

$$f(x) = w \cdot x + b$$


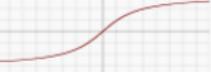
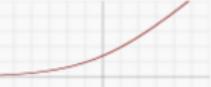
The diagram illustrates the perceptron equation $f(x) = w \cdot x + b$. It features a bracket under the term $w \cdot x$ pointing to the text "Input Vector". Another bracket under the term b points to the text "Bias". A long arrow pointing to the right is labeled "Transfer Function" and spans the entire equation from the left side to the plus sign.

Activation Function

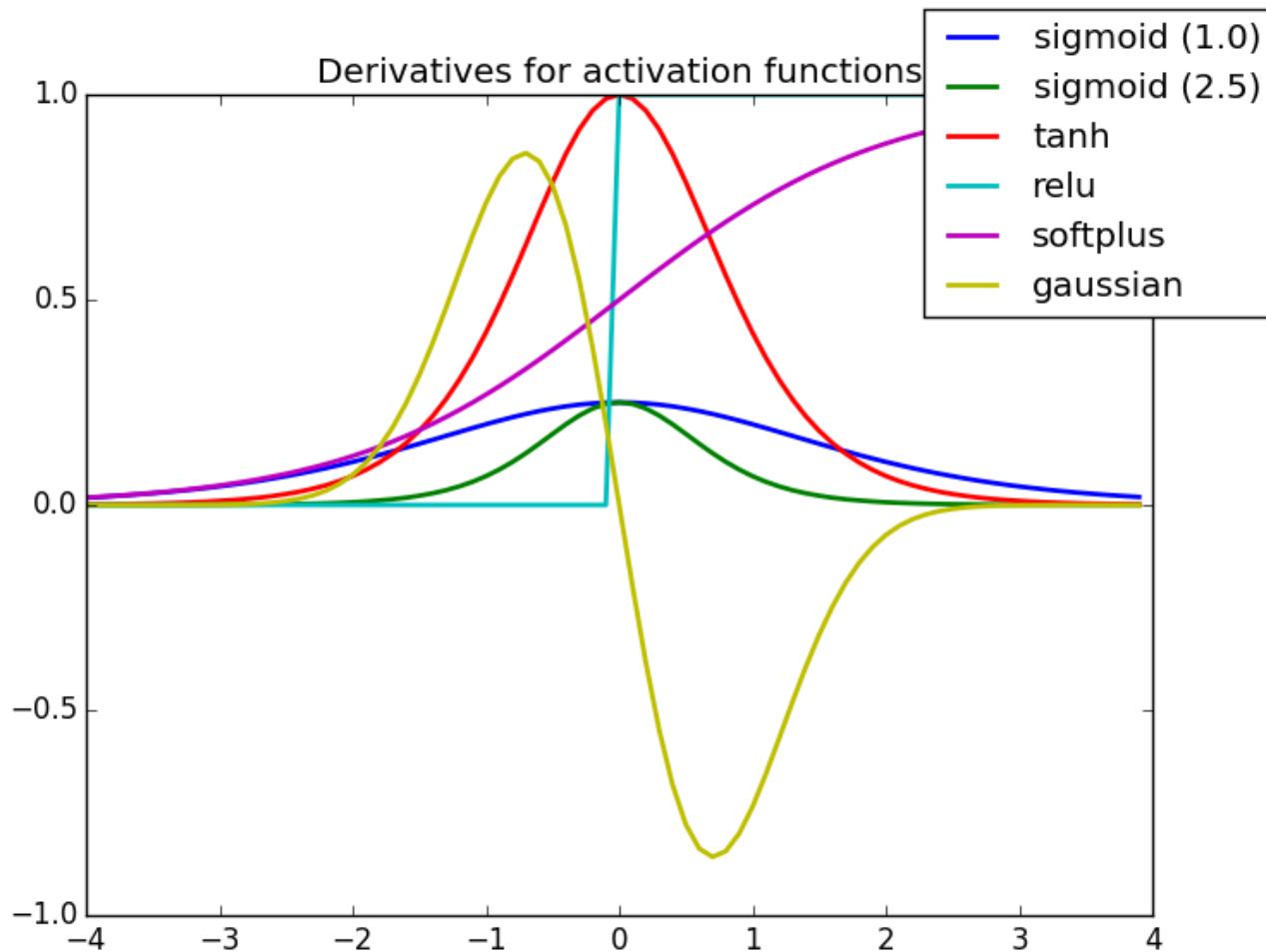
- The input received by a perceptron is first multiplied by the respective weights and then, all these weighted inputs are summed together.
- This summed value is then fed to activation for obtaining the final result as shown in the image below followed by the code:



Activation Functions

| Name | Plot | Equation | Derivative |
|---|---|--|--|
| Identity |  | $f(x) = x$ | $f'(x) = 1$ |
| Binary step |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $f'(x) \stackrel{\text{?}}{=} \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$ |
| Logistic (a.k.a Soft step) |  | $f(x) = \frac{1}{1 + e^{-x}}$ | $f'(x) = f(x)(1 - f(x))$ |
| TanH |  | $f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ |
| ArcTan |  | $f(x) = \tan^{-1}(x)$ | $f'(x) = \frac{1}{x^2 + 1}$ |
| Rectified Linear Unit (ReLU) |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Parameteric Rectified Linear Unit (PReLU) [2] |  | $f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Exponential Linear Unit (ELU) [3] |  | $f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| SoftPlus |  | $f(x) = \log_e(1 + e^x)$ | $f'(x) = \frac{1}{1 + e^{-x}}$ |

Activation Functions



Activation Function

The sequence of exemplars presented may go like this:

| input x | network fires or not (y) | told correct answer (O) | what to do with w's | what to do with t |
|-----------|-----------------------------|----------------------------|---------------------|-------------------|
| (0,1,0,0) | 0 | 1 | increase | reduce |
| (1,0,0,0) | 0 | 0 | no change | no change |
| (0,1,1,1) | 1 | 0 | reduce | increase |
| (1,0,1,0) | 1 | 0 | reduce | increase |
| (1,1,1,1) | 0 | 1 | increase | reduce |
| (0,1,0,0) | 1 | 1 | no change | no change |
| | | | | |

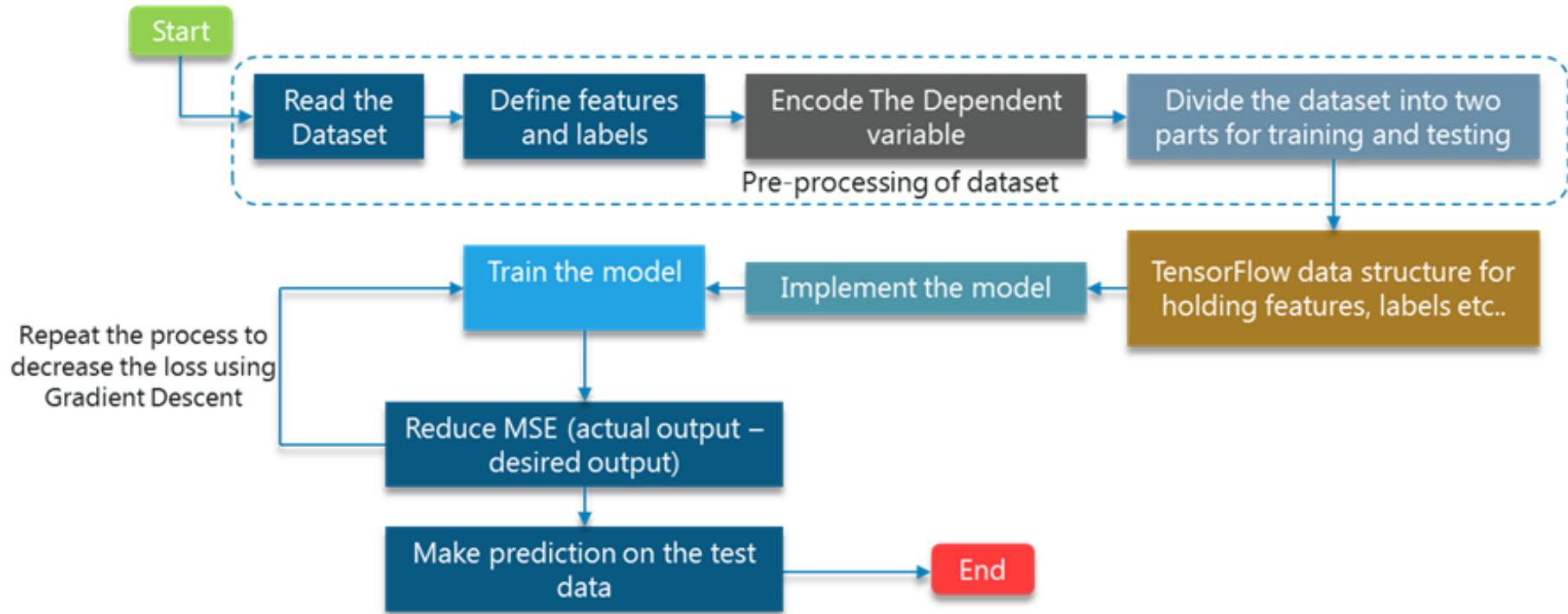
Limitations of Single-Layer Perceptron:

- Well, there are two major problems:
- Single-Layer Perceptrons cannot classify non-linearly separable data points.
- Complex problems, that involve a lot of parameters cannot be solved by Single-Layer Perceptrons.

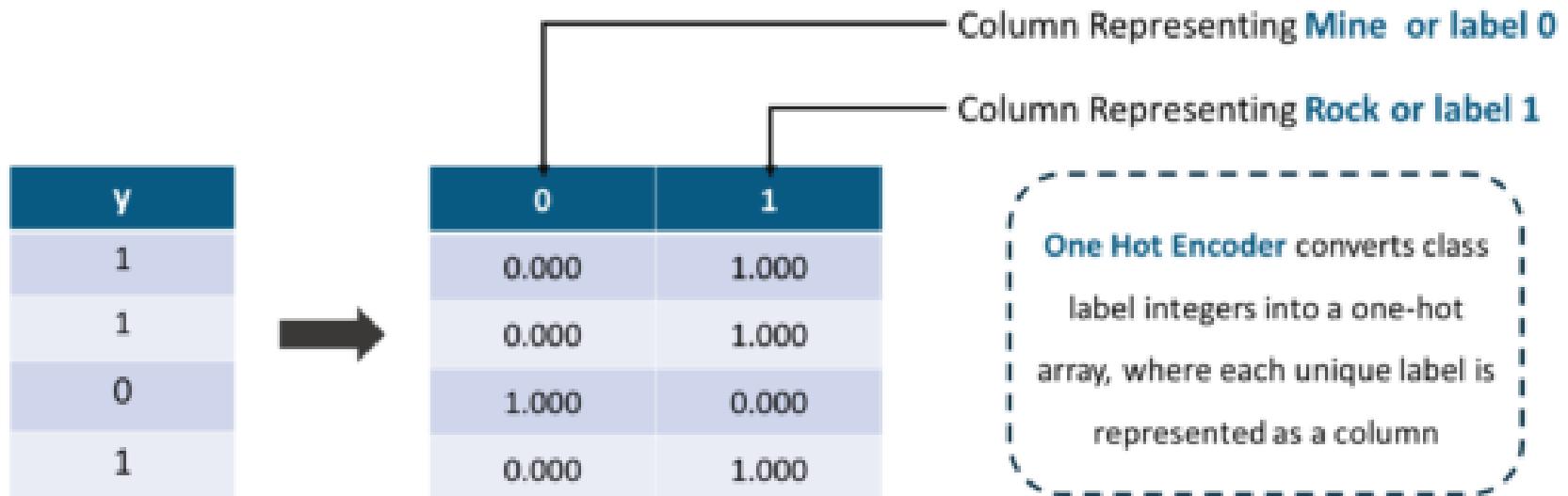
SONAR Data Classification Using Single Layer Perceptrons



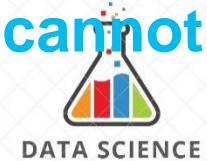
SONAR Data Classification Using Single Layer Perceptrons



SONAR Data Classification Using Single Layer Perceptrons

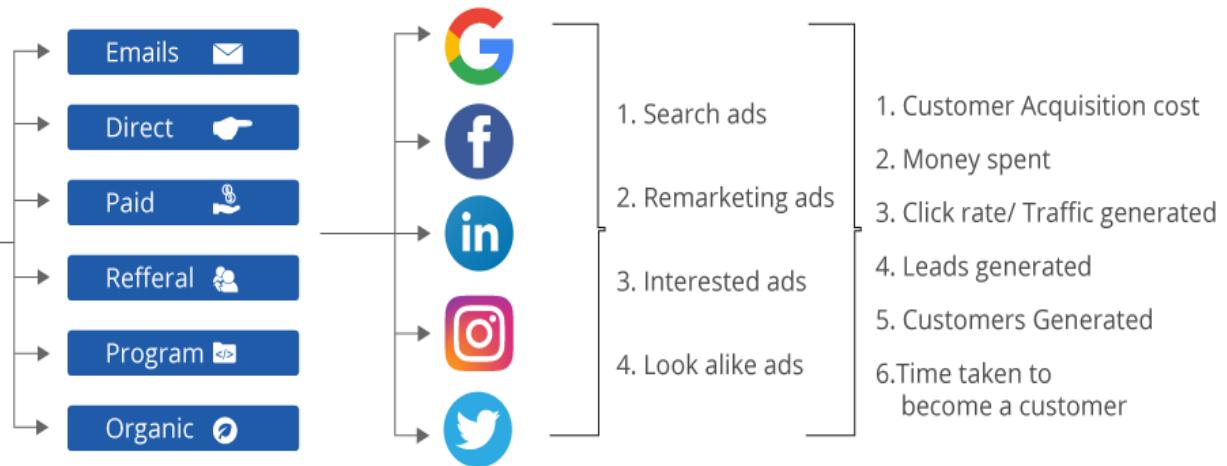


Complex problems, that involve a lot of parameters cannot be solved by Single-Layer Perceptrons:



- The marketing team can market your product through various ways, such as:
 - Google Ads
 - Personal emails
 - Sale advertisement on relevant sites
 - Reference program
 - Blogs and so on . . .
- Considering all the factors and options available, marketing team has to decide a strategy to do optimal and efficient marketing, but this task is too complex for a human to analyse, because number of parameters are quite high.
- This problem will have to be solved using Deep Learning.

Complex problems, that involve a lot of parameters cannot be solved by Single-Layer Perceptrons:



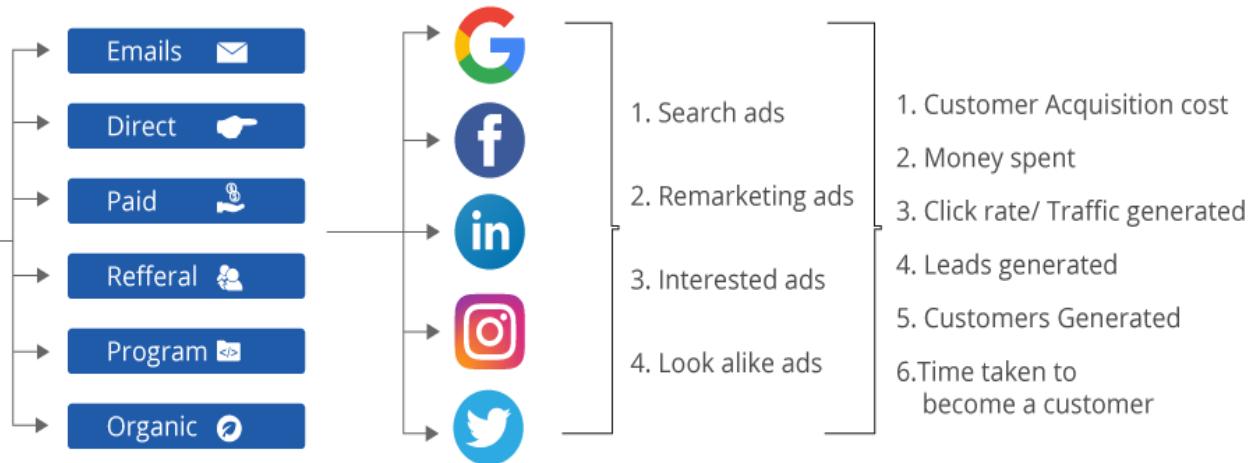
Complex problems, that involve a lot of parameters cannot be solved by Single-Layer Perceptrons:



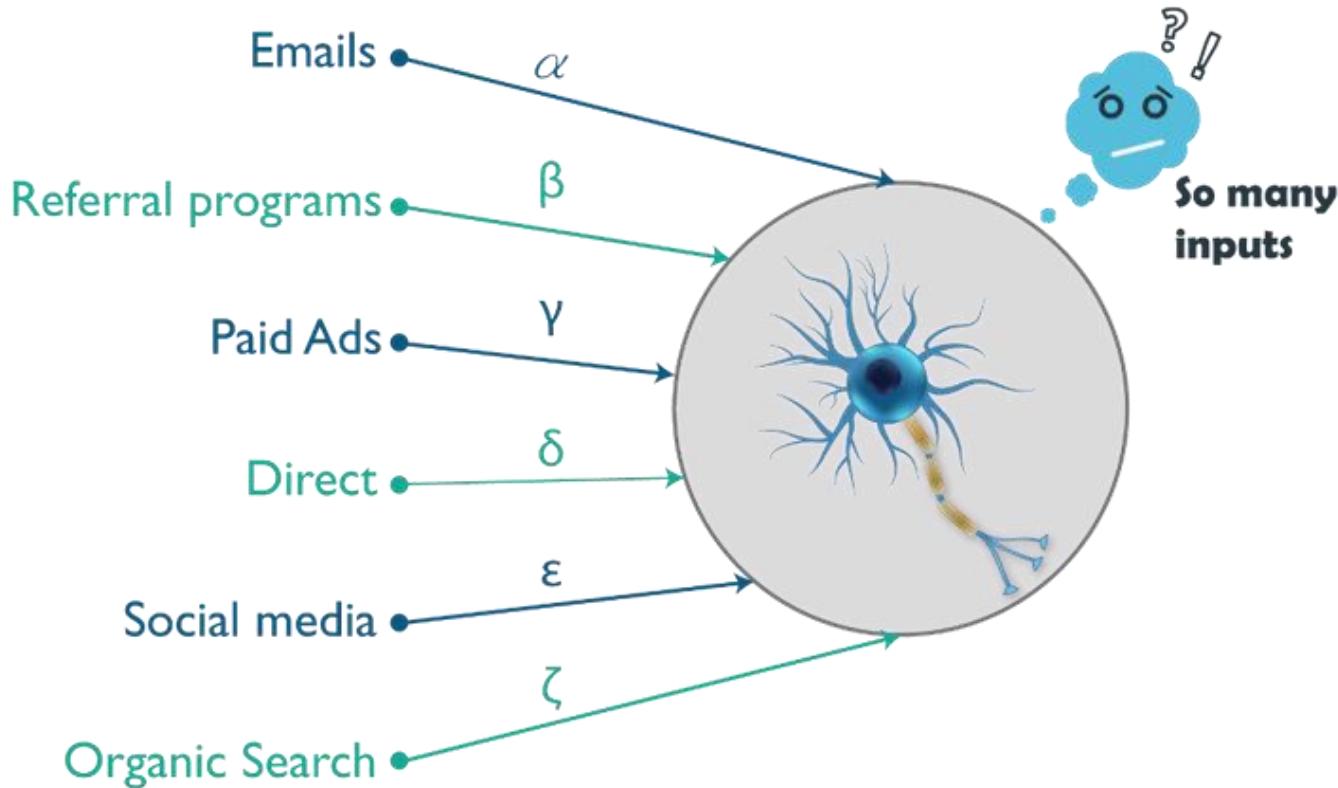
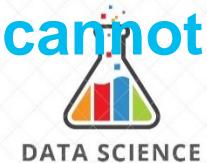
DATA SCIENCE



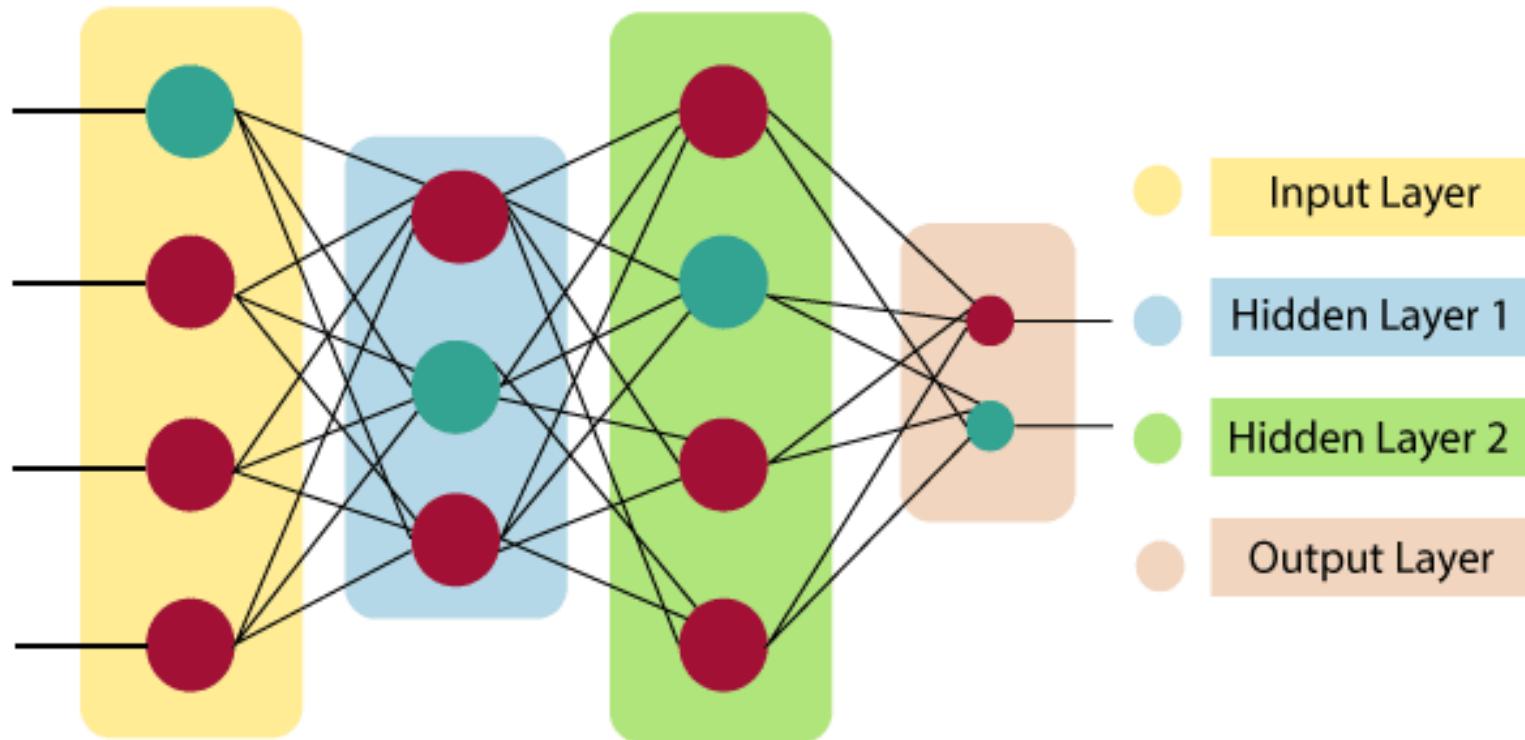
Marketing



Complex problems, that involve a lot of parameters cannot be solved by Single-Layer Perceptrons:



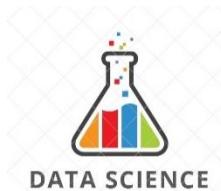
Multi Layer Network



Artificial Neural Network

- Input Layer:
- As the name suggests, it accepts inputs in several different formats provided by the programmer.
- Hidden Layer:
- The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.
- Output Layer:
- The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

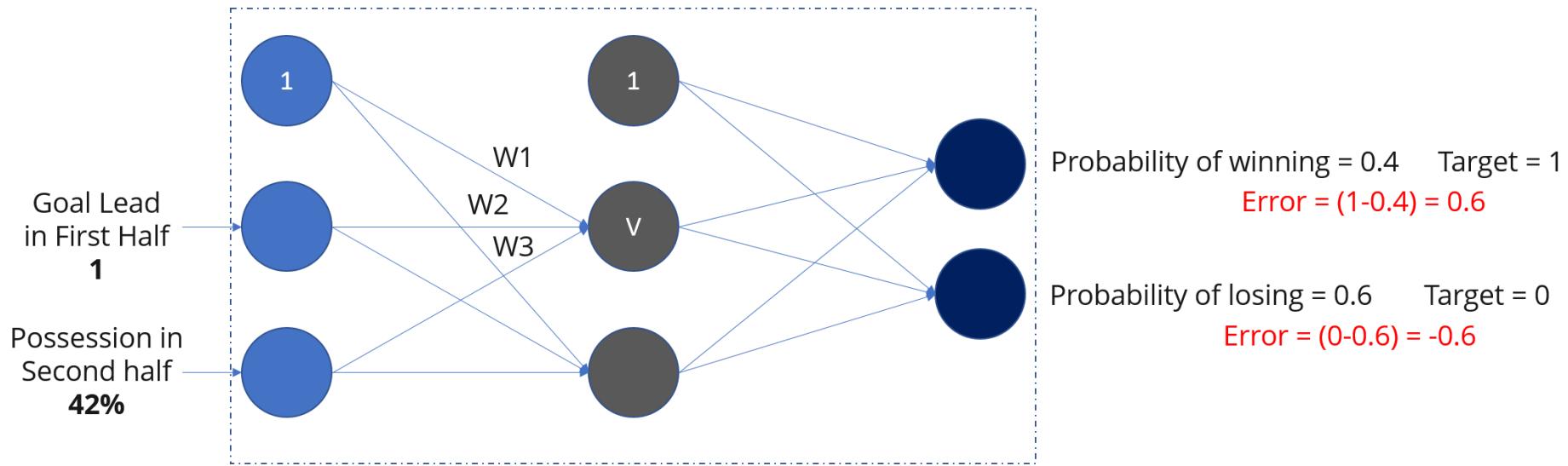
What is Multi-Layer Perceptron?



| Goal Lead in First Half | Possession in Second Half | Won or Lost (1,0)? |
|-------------------------|---------------------------|--------------------|
| 0 | 80% | 1 |
| 0 | 35% | 0 |
| 1 | 42% | 1 |
| 2 | 20% | 0 |
| -1 | 75% | 1 |

- Suppose we have data of a football team, **Chelsea**. The data contains three columns. The last column tells whether Chelsea won the match or they lost it. The other two columns are about, goal lead in the first half and possession in the second half. Possession is the amount of time for which the team has the ball in percentage. So, if I say that a team has 50% possession in one half (45 minutes), it means that, the team had ball for 22.5 minutes out of 45 minutes.

What is Multi-Layer Perceptron?

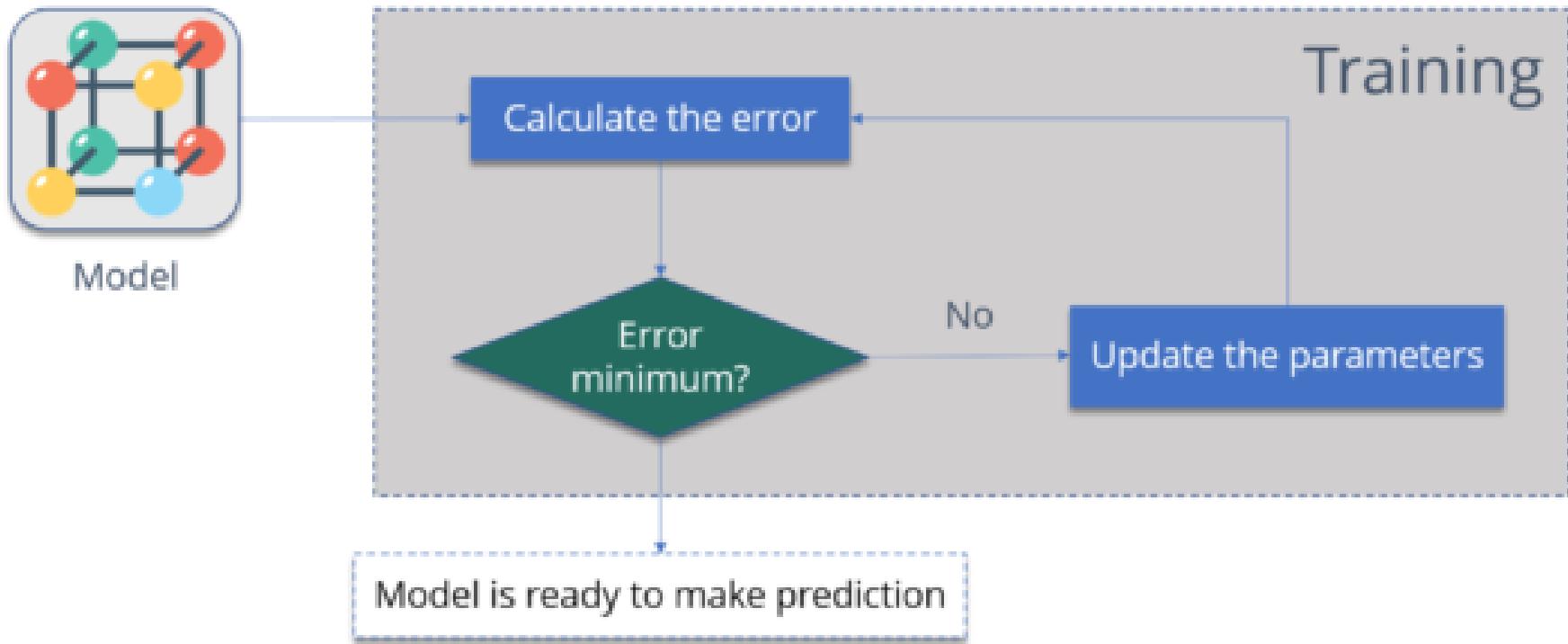


Backpropagation:



- Backpropagation is a supervised learning algorithm, for training Multi-layer Perceptrons (Artificial Neural Networks).
- **Why We Need Backpropagation?**
- While designing a Neural Network, in the beginning, we initialize weights with some random values or any variable for that fact.
- Now obviously, we are not *superhuman*. So, it's not necessary that whatever weight values we have selected will be correct, or it fits our model the best.
- Okay, fine, we have selected some weight values in the beginning, but our model output is way different than our actual output i.e. the error value is huge.
- Now, how will you reduce the error?
- Basically, what we need to do, we need to somehow explain the model to change the parameters (weights), such that error becomes minimum.

Backpropagation:



Backpropagation:

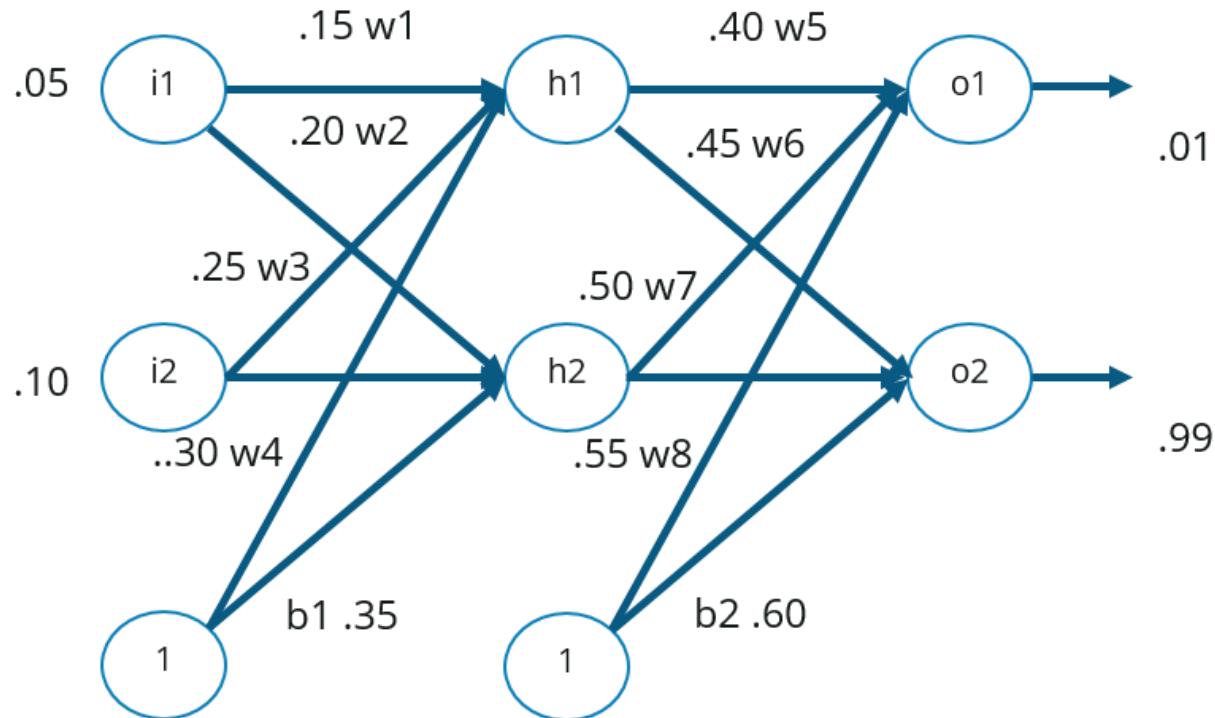


- Let me summarize the steps for you:
- Calculate the error – How far is your model output from the actual output.
- Error minimum? – Check whether the error is minimized or not.
- Update the parameters – If the error is huge then, update the parameters (weights and biases). After that again check the error. Repeat the process until the error becomes minimum.
- Model is ready to make a prediction – Once the error becomes minimum, you can feed some inputs to your model and it will produce the output.

Backpropagation Algorithm

- initialize network weights (often small random values)
- do
 - forEach training example named ex
 - prediction = neural-net-output(network, ex) // forward pass
 - actual = teacher-output(ex)
 - compute error (prediction - actual) at the output units
 - compute $\Delta w_{\{h\}}$ for all weights from hidden layer to output layer // backward pass
 - compute $\Delta w_{\{i\}}$ for all weights from input layer to hidden layer // backward pass continued
 - update network weights // input layer not modified by error estimate
 - until all examples classified correctly or another stopping criterion satisfied
 - return the network

How Backpropagation Works?



How does a machine look at an image?

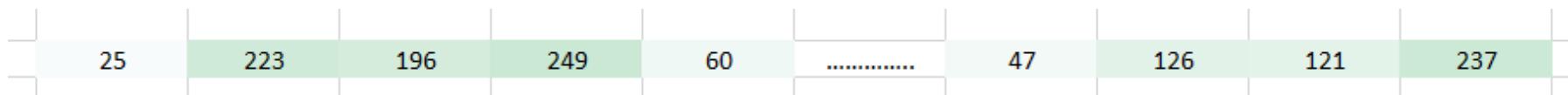


| | | | |
|-----|-----|-----|-----|
| 25 | 2 | 1 | 44 |
| 223 | 7 | 6 | 60 |
| 196 | 8 | 2 | 148 |
| 249 | 1 | 3 | 40 |
| 60 | 7 | 1 | 154 |
| 59 | 1 | 7 | 213 |
| 214 | 7 | 3 | 163 |
| 89 | 182 | 219 | 13 |
| 74 | 146 | 113 | 72 |
| 89 | 18 | 244 | 85 |
| 1 | 4 | 8 | 97 |
| 3 | 4 | 2 | 121 |
| 2 | 1 | 2 | 131 |
| 7 | 6 | 8 | 47 |
| 3 | 5 | 5 | 126 |
| 7 | 6 | 8 | 121 |
| 5 | 3 | 1 | 237 |

How do we help a neural network to identify images ?



Flattened Array



How do we help a neural network to identify images ?



Using Artificial Neural Network, we need to figure out, if the bank notes are real or fake?



How do we help a neural network to identify images ?



| | | | | |
|----------|----------|----------|---------|---|
| 4.0127 | 10.1477 | -3.9366 | -4.0728 | 0 |
| 2.6606 | 3.1681 | 1.9619 | 0.18662 | 0 |
| 3.931 | 1.8541 | -0.02343 | 1.2314 | 0 |
| 0.01727 | 8.693 | 1.3989 | -3.9668 | 0 |
| 3.2414 | 0.40971 | 1.4015 | 1.1952 | 0 |
| 2.2504 | 3.5757 | 0.35273 | 0.2836 | 0 |
| -1.3971 | 3.3191 | -1.3927 | -1.9948 | 1 |
| 0.39012 | -0.14279 | -0.03199 | 0.35084 | 1 |
| -1.6677 | -7.1535 | 7.8929 | 0.96765 | 1 |
| -3.8483 | -12.8047 | 15.6824 | -1.281 | 1 |
| -3.5681 | -8.213 | 10.083 | 0.96765 | 1 |
| -2.2804 | -0.30626 | 1.3347 | 1.3763 | 1 |
| -1.7582 | 2.7397 | -2.5323 | -2.234 | 1 |
| -0.89409 | 3.1991 | -1.8219 | -2.9452 | 1 |
| 0.3434 | 0.12415 | -0.28733 | 0.14654 | 1 |

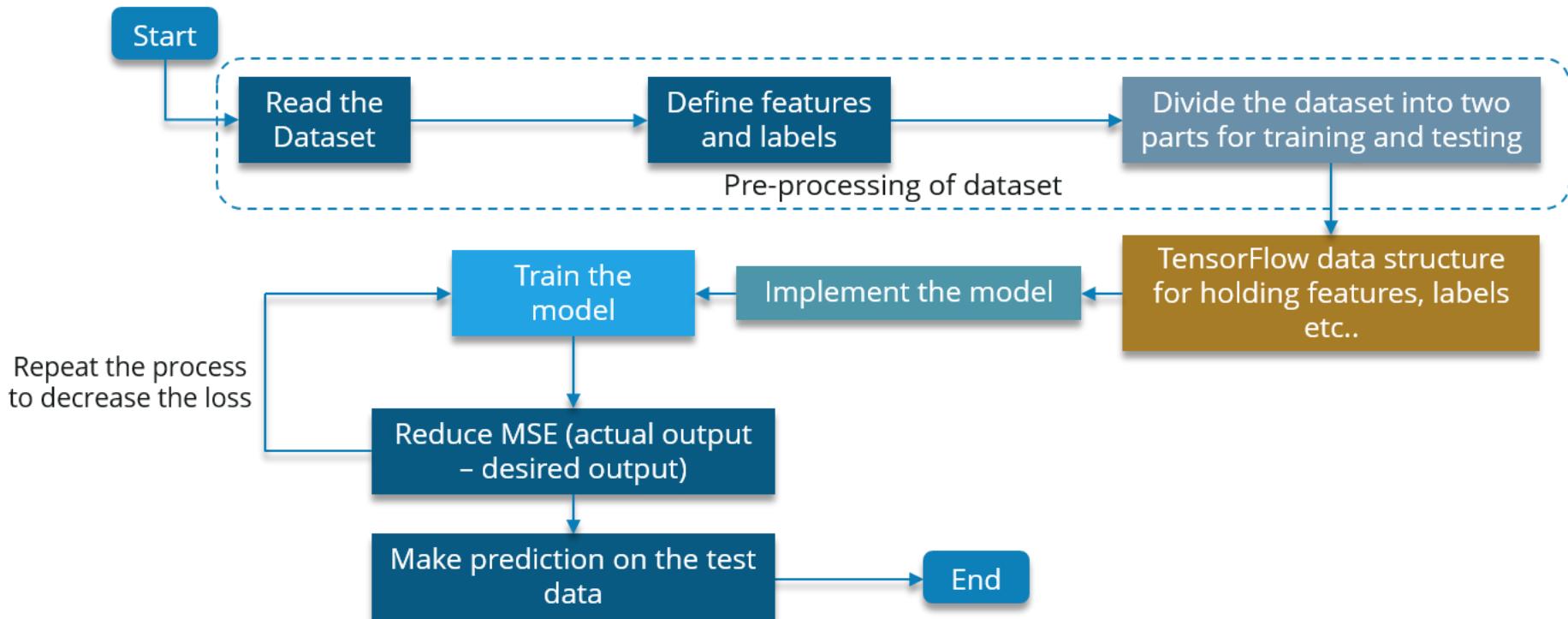
Features

- Variance of Wavelet Transformed image
- Skewness of Wavelet Transformed image
- Kurtosis of Wavelet Transformed image
- Entropy of image

Label

1 – Real, 0 - Fake

Backpropagation



Convolutional Neural Networks



- CNNs, like neural networks, are made up of neurons with learnable weights and biases.
- Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output.
- The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply on CNNs.

Defining a Convolutional Neural Network



- We need three basic components to define a basic convolutional network.
- The convolutional layer
- The Pooling layer[optional]
- The output layer

The Convolution Layer



- Suppose we have an image of size 6*6. We define a weight matrix which extracts certain features from the images

| INPUT IMAGE | | | | | |
|-------------|-----|-----|-----|-----|-----|
| 18 | 54 | 51 | 239 | 244 | 188 |
| 55 | 121 | 75 | 78 | 95 | 88 |
| 35 | 24 | 204 | 113 | 109 | 221 |
| 3 | 154 | 104 | 235 | 25 | 130 |
| 15 | 253 | 225 | 159 | 78 | 233 |
| 68 | 85 | 180 | 214 | 245 | 0 |

| WEIGHT | | |
|--------|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

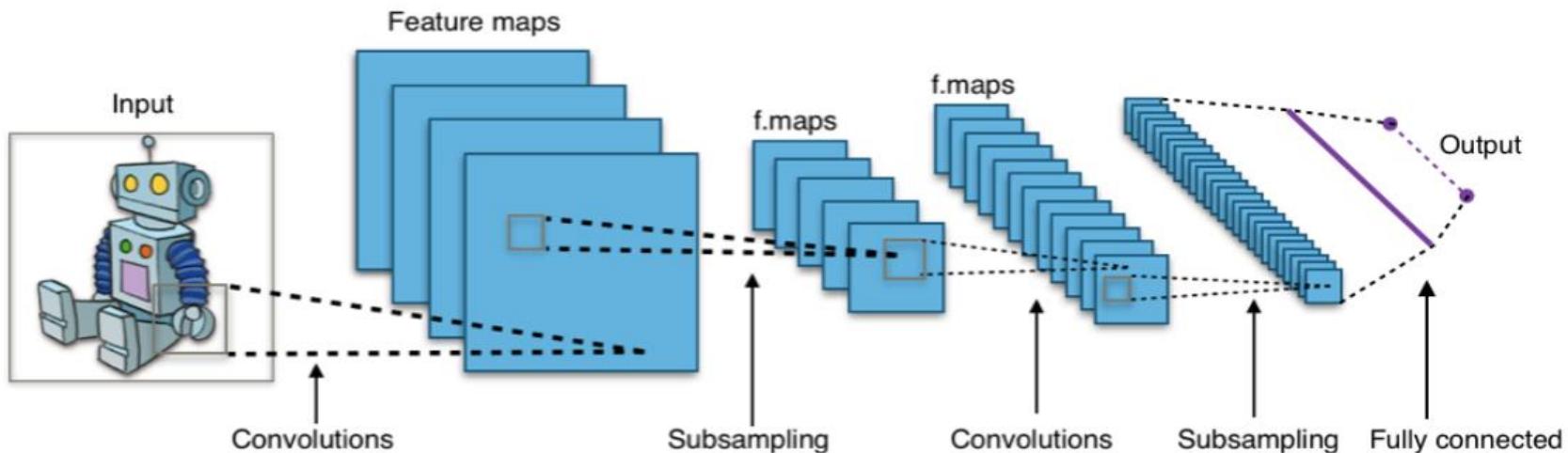
429

| INPUT IMAGE | | | | | |
|-------------|-----|-----|-----|-----|-----|
| 18 | 54 | 51 | 239 | 244 | 188 |
| 55 | 121 | 75 | 78 | 95 | 88 |
| 35 | 24 | 204 | 113 | 109 | 221 |
| 3 | 154 | 104 | 235 | 25 | 130 |
| 15 | 253 | 225 | 159 | 78 | 233 |
| 68 | 85 | 180 | 214 | 245 | 0 |

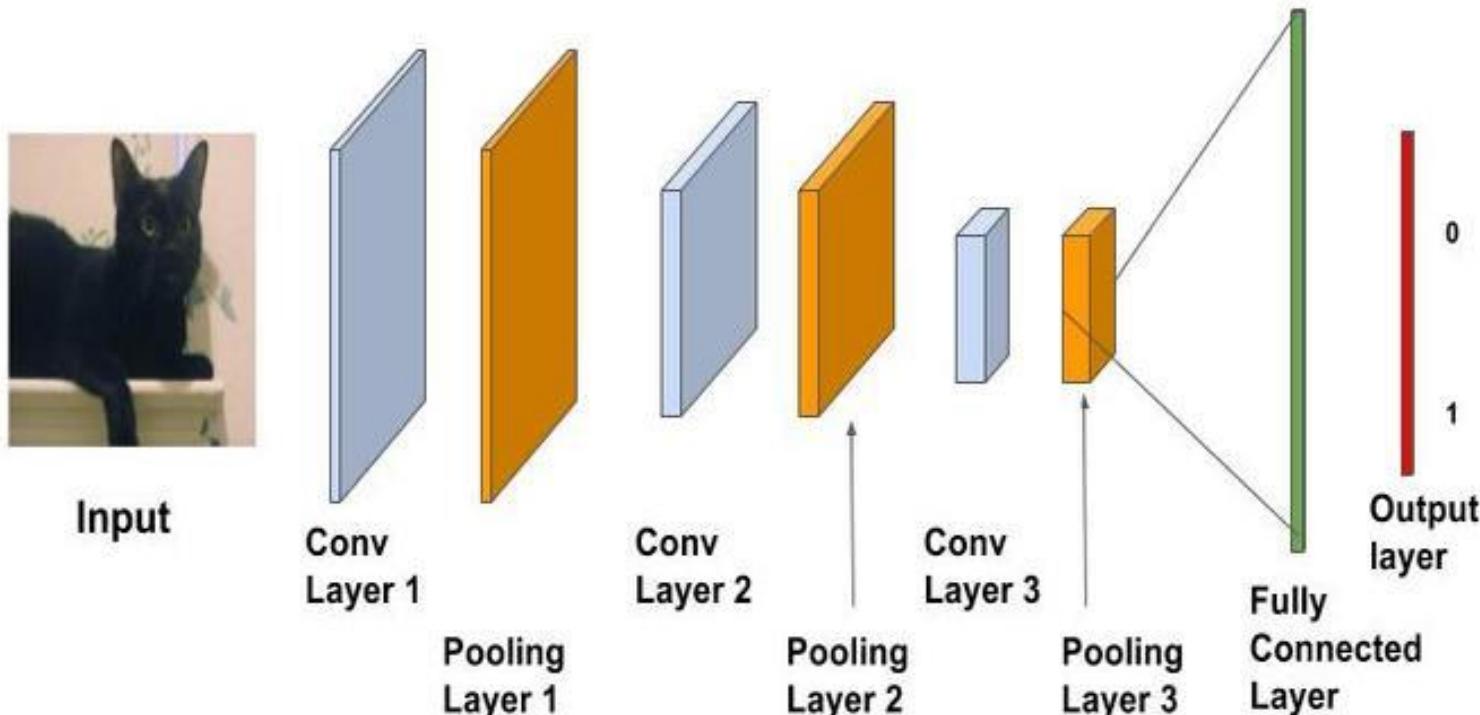
| WEIGHT | | |
|--------|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

429

The Convolution Layer



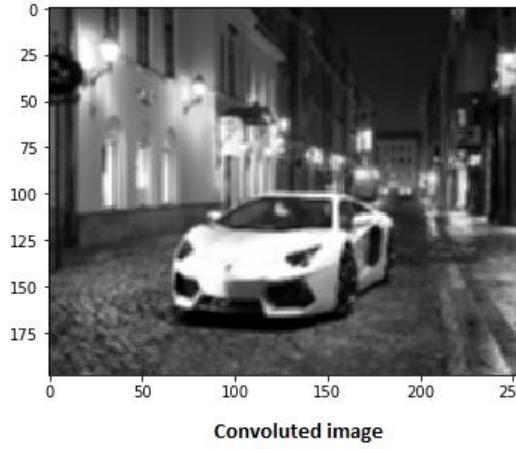
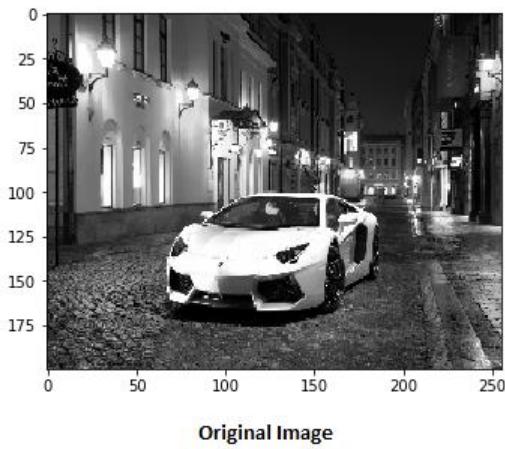
The Convolution Layer



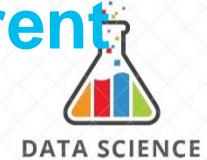
The Convolution Layer



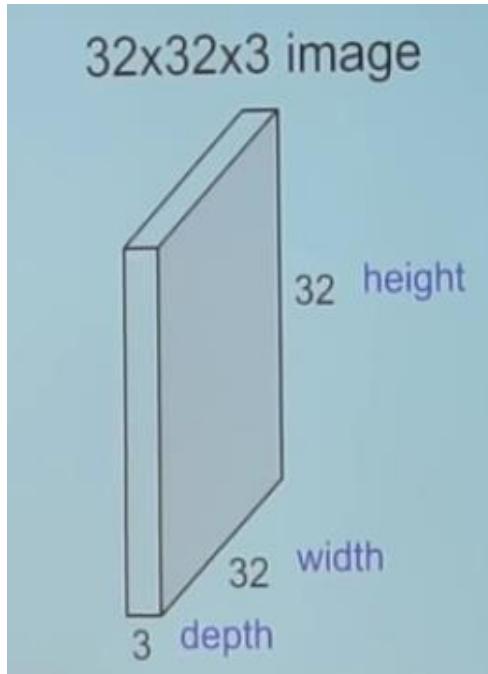
- Suppose we have an image of size $6*6$. We define a weight matrix which extracts certain features from the images



How are Convolutional Neural Networks different than Neural Networks?



CNNs operate over Volumes !

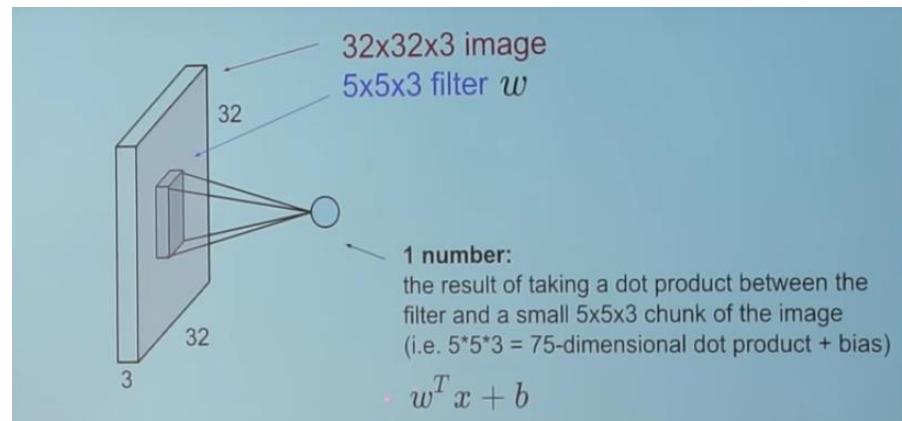
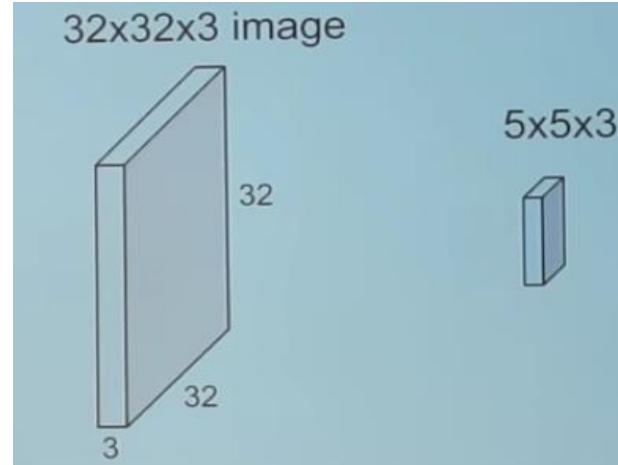


Unlike neural networks, where the input is a vector, here the input is a multi-channelled image (3 channelled in this case).

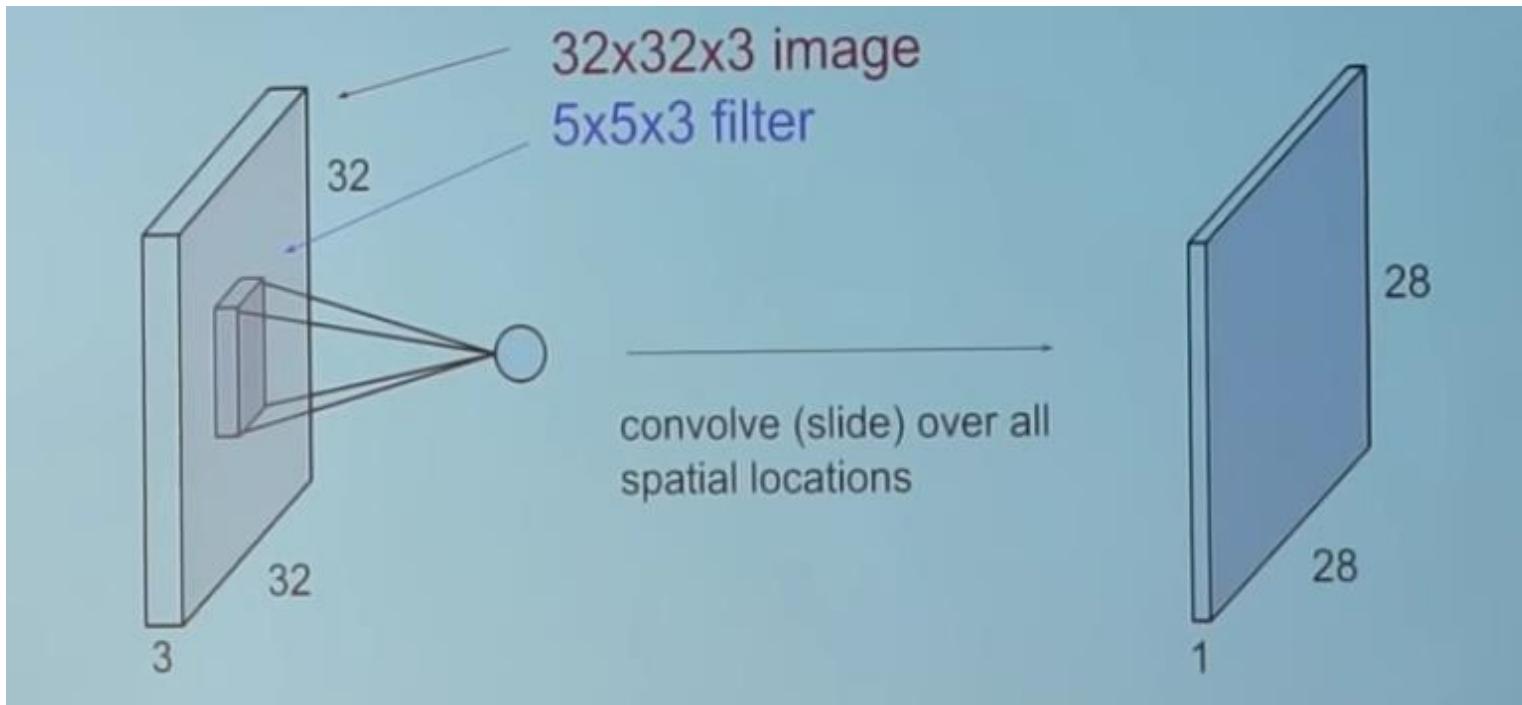
Convolution



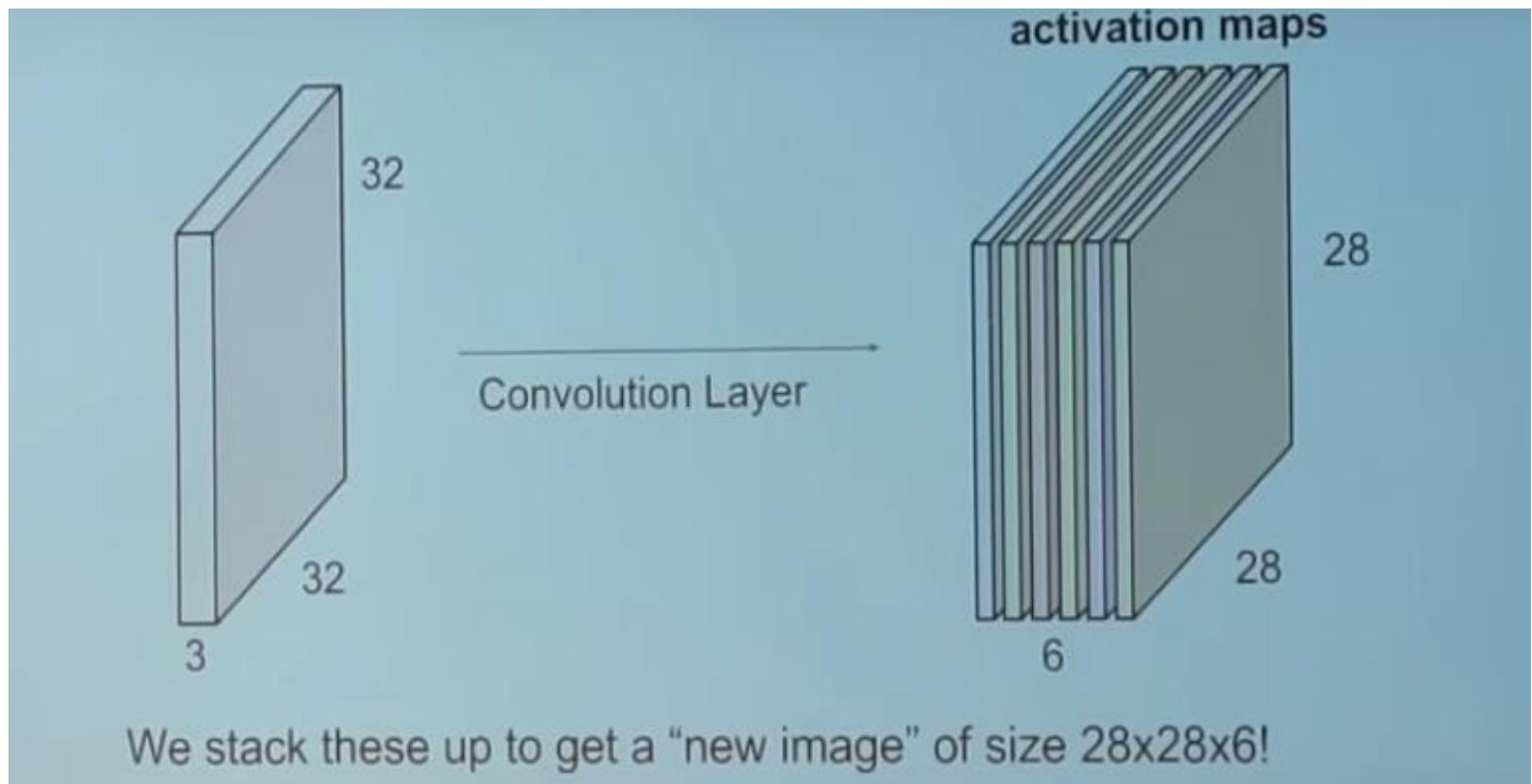
- Convolution is a mathematical way of combining two signals to form a third signal.
- It is the single most important technique in Digital Signal Processing.
- Using the strategy of impulse decomposition, systems are described by a signal called the *impulse response*.
- Convolution is important because it relates the three signals of interest: the input signal, the output signal, and the impulse response.



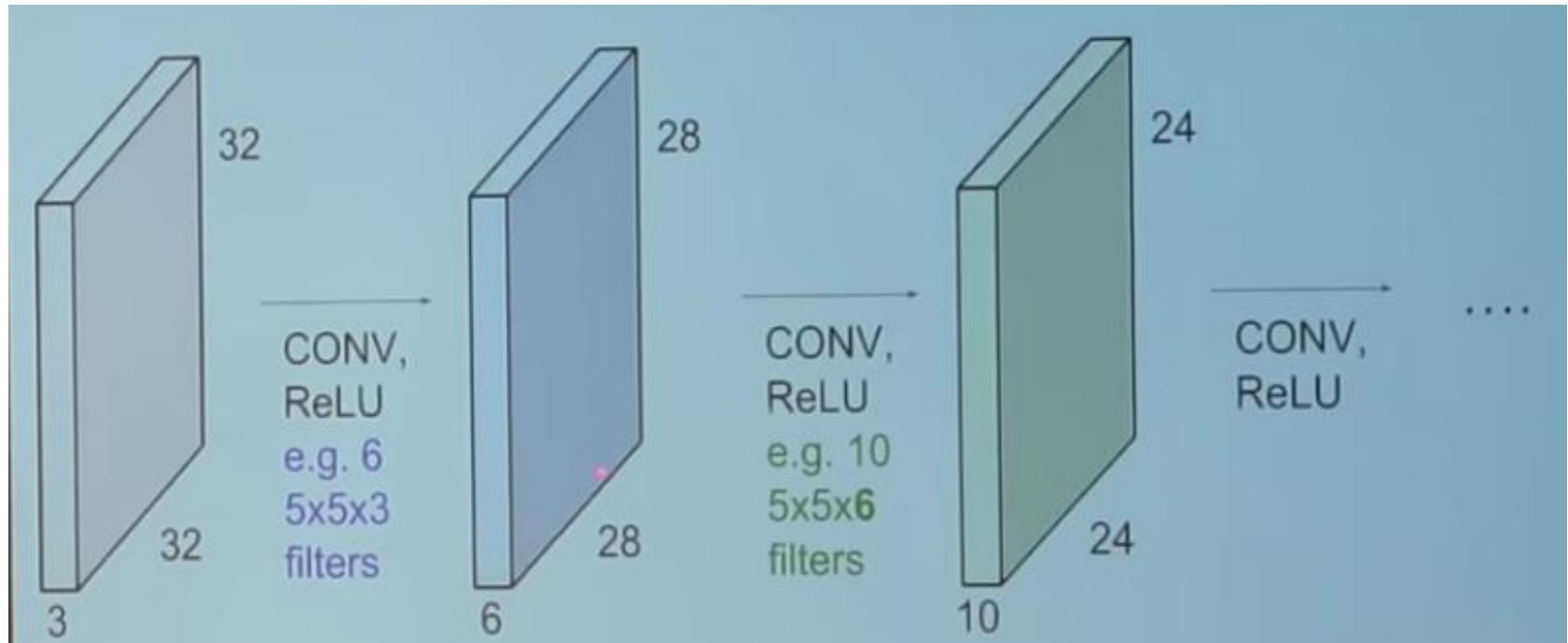
Convolution



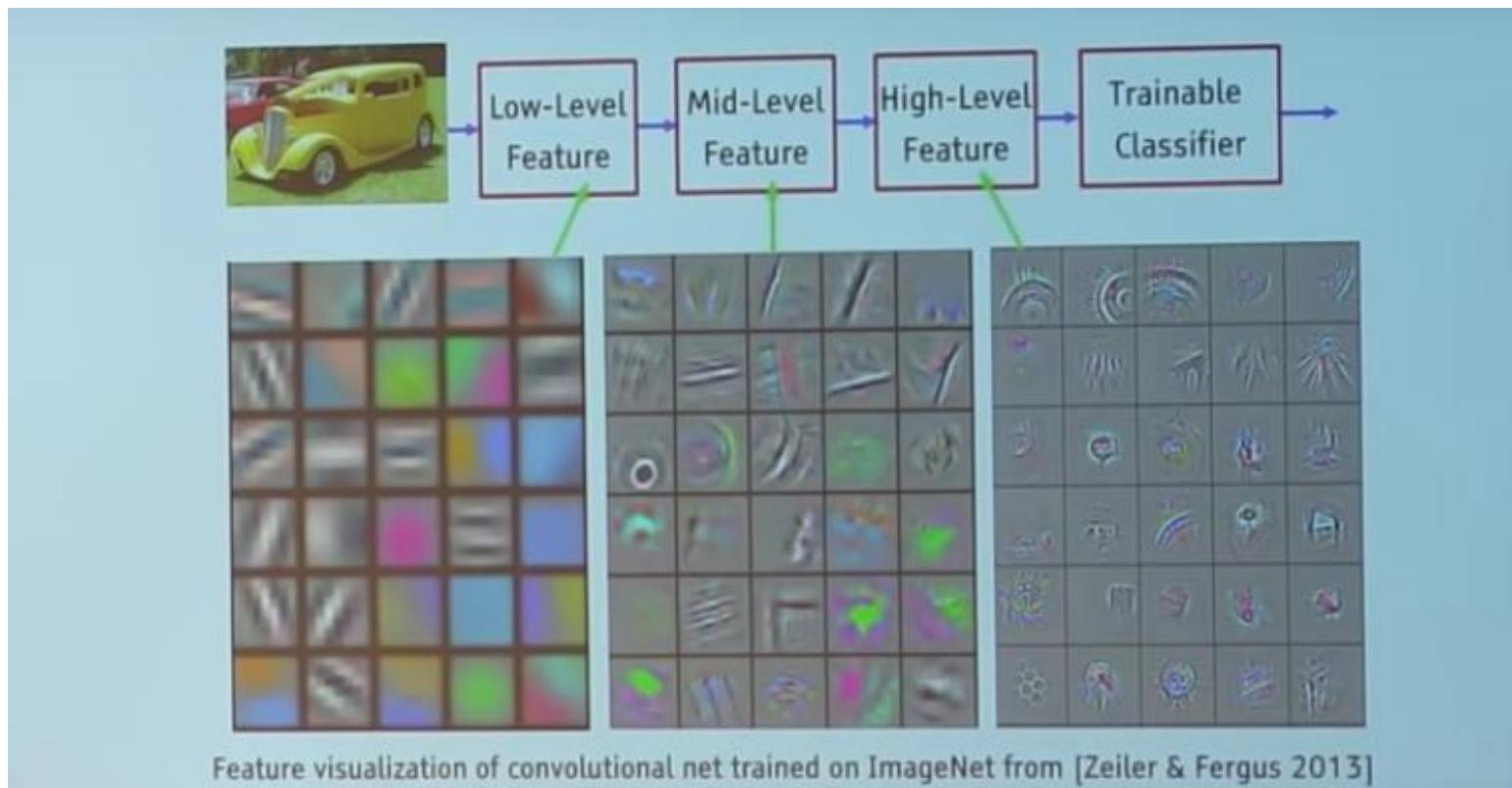
Convolution



Convolution

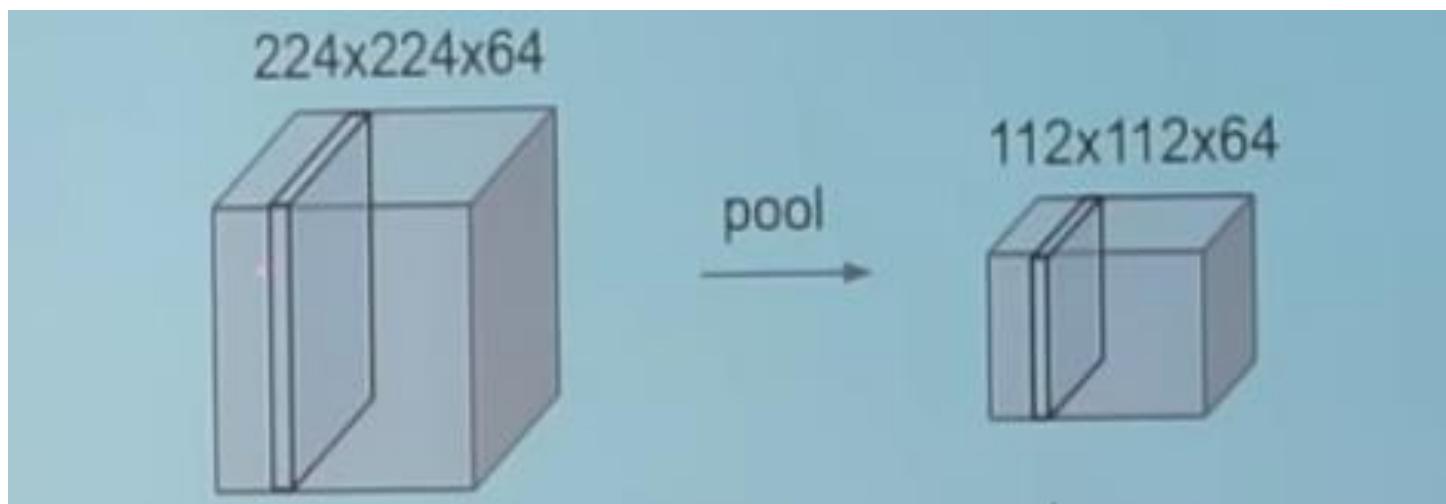


Convolution

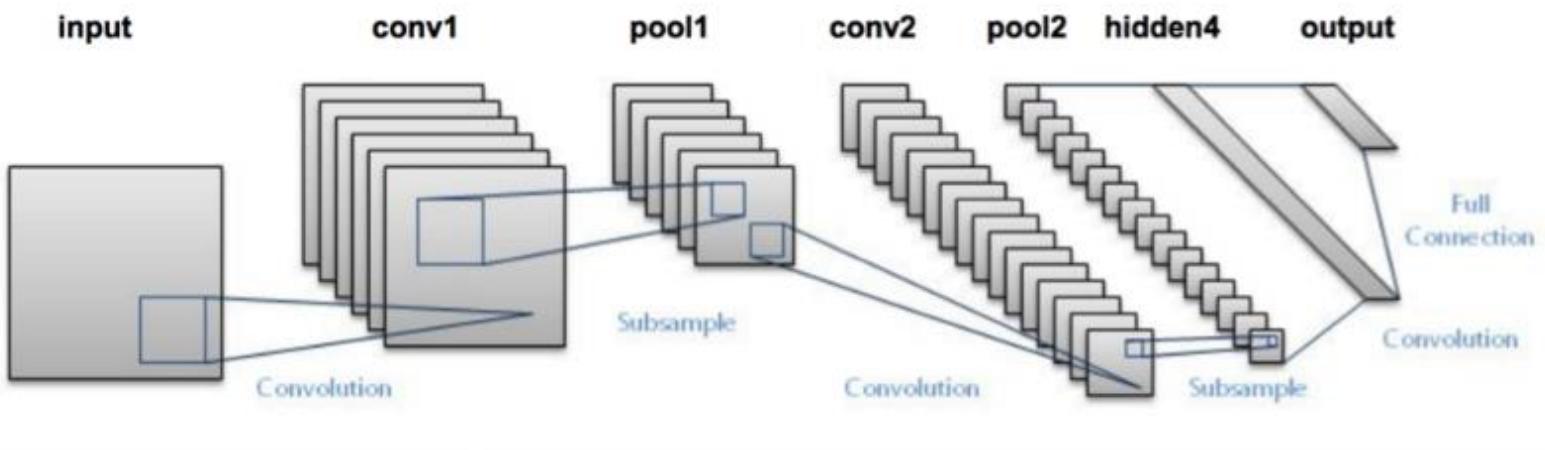
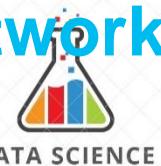


Convolution

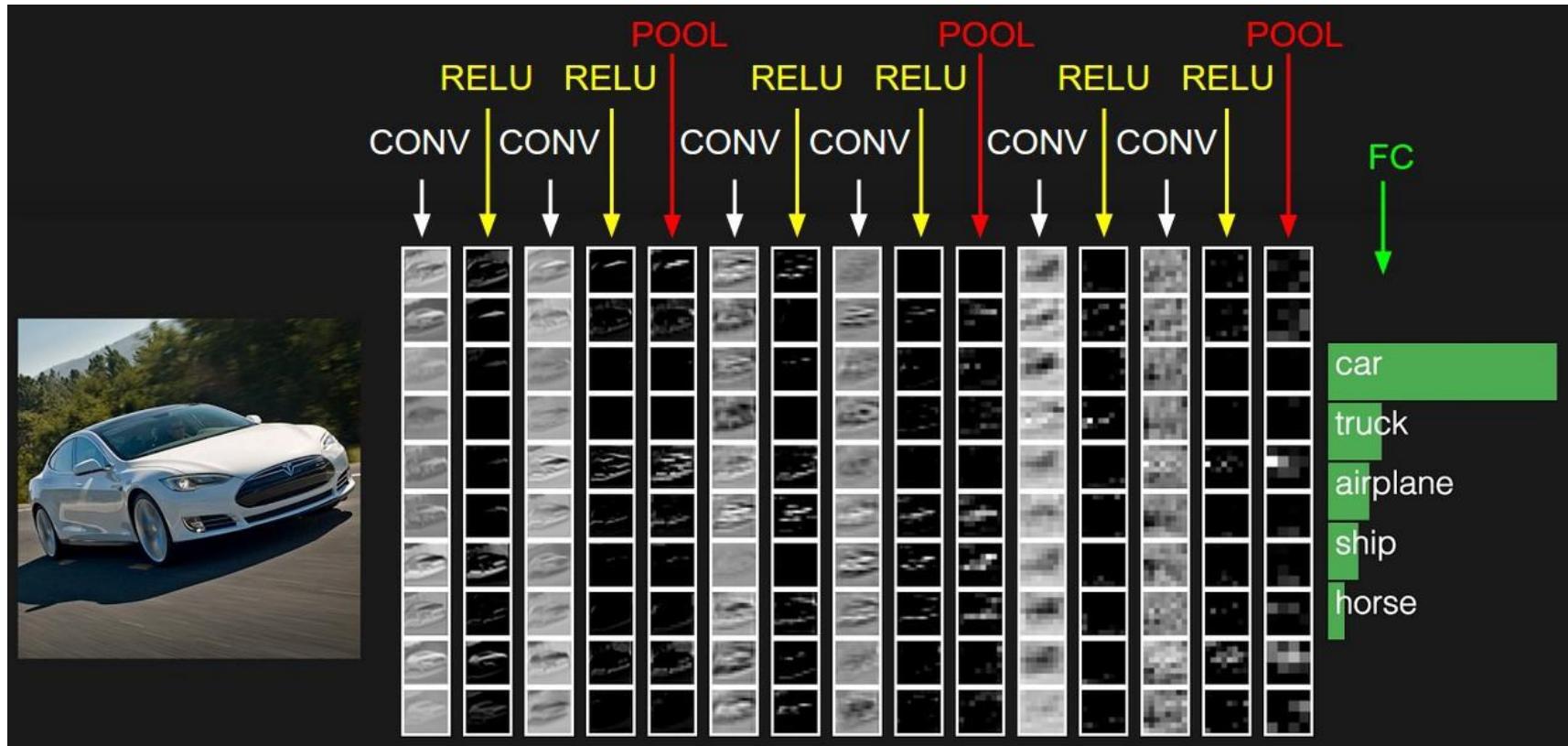
Pooling Layers



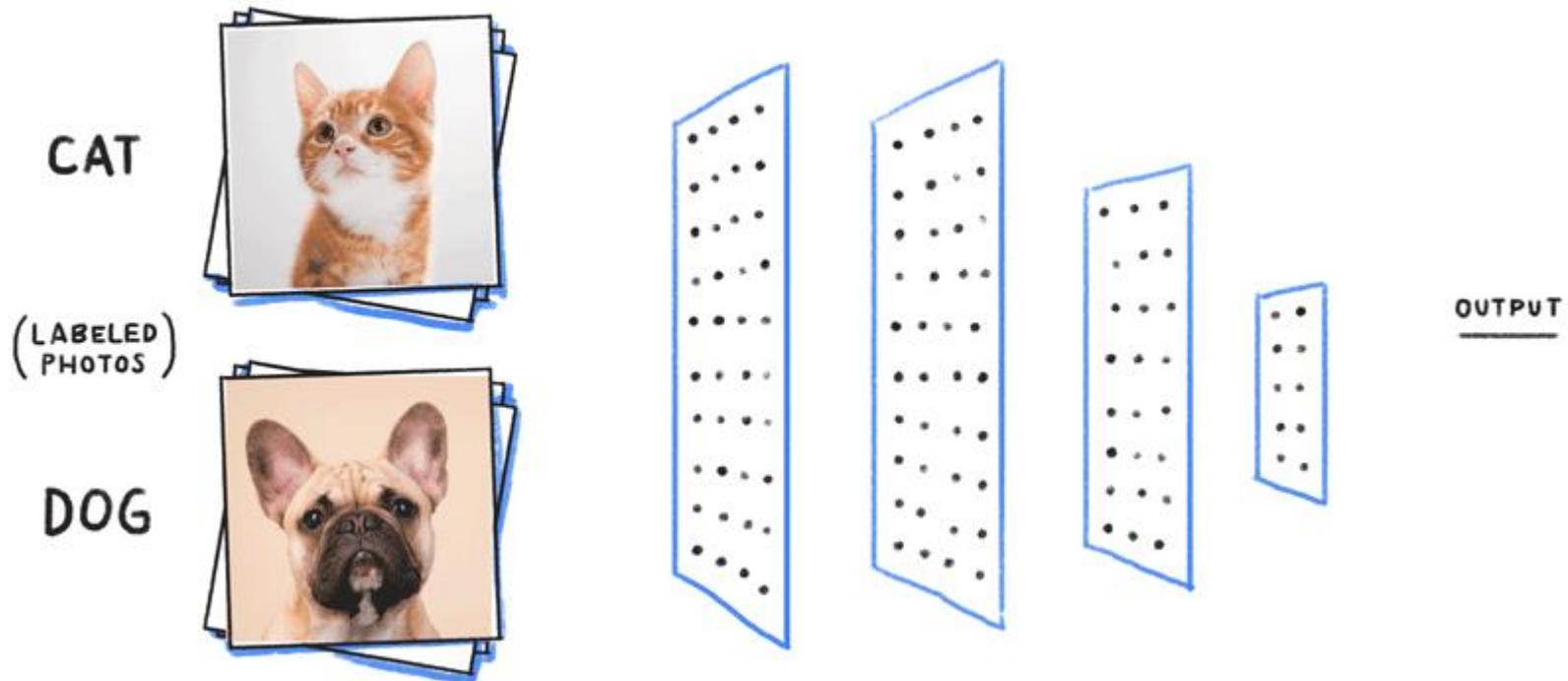
Putting it all together – How does the entire network look like ?



Typical Architecture of a CNN



Typical Architecture of a CNN



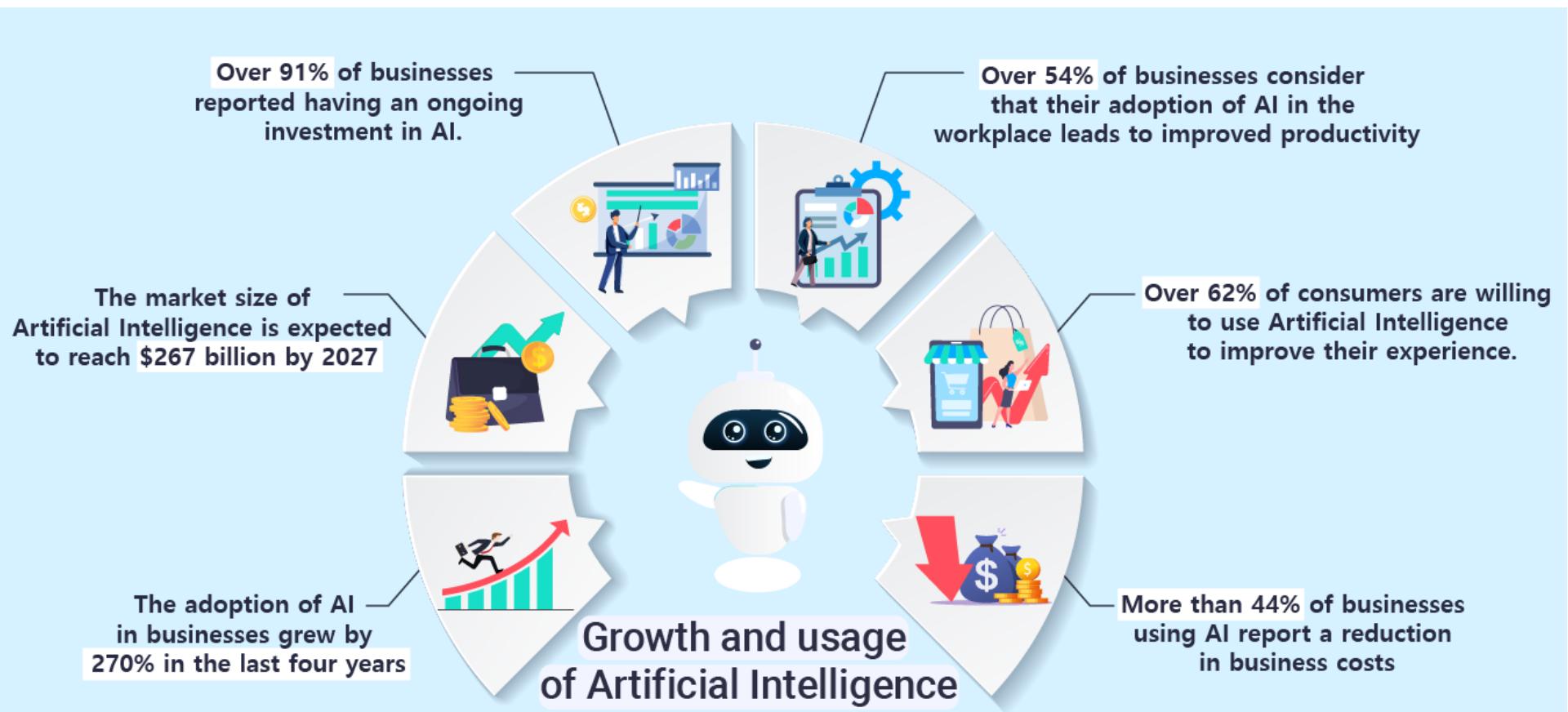
Artificial Intelligence

- “A system, built through coding, business rules, and increasingly self-learning capabilities, that is able to supplement human cognition and activities and interacts with humans natural, but also understands the environment, solves human problems and performs human tasks. “

AI in Software Testing

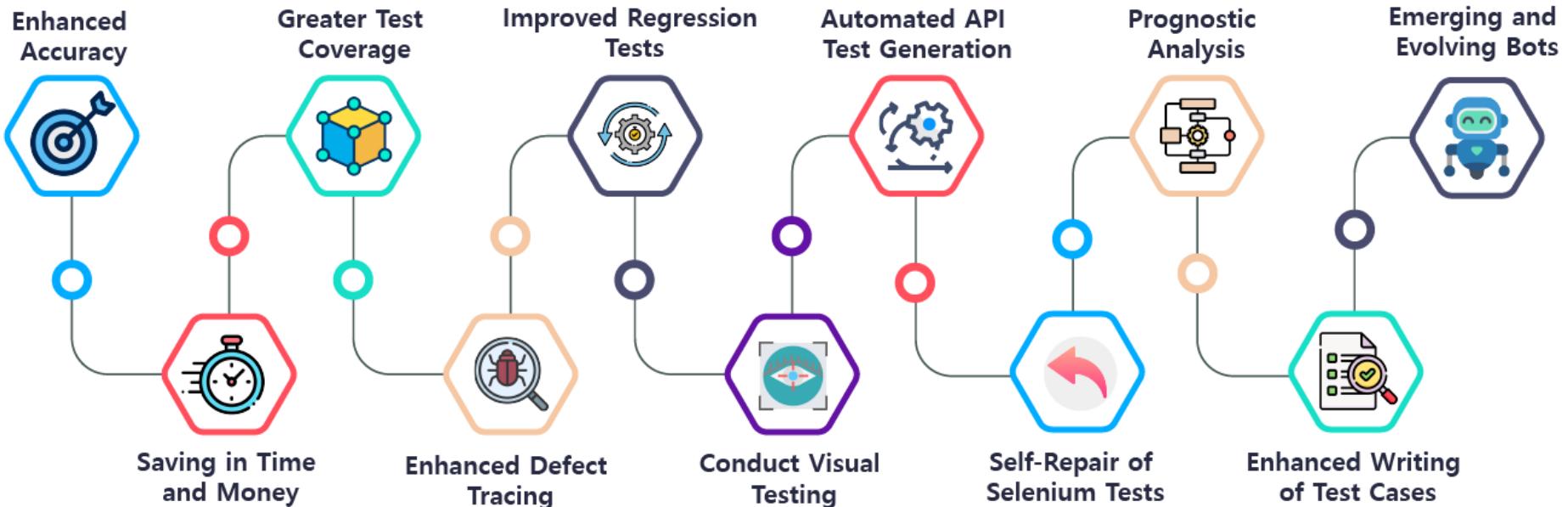
- Software testing is an important process that ensures customer satisfaction in the application.
- It is the planned way in test automation where an application observed under specific conditions where the testers understand the threshold and the risks involved in the software implementation.
- AI in Software Testing helps to safeguard an application against potential application fail-overs which may turn out being harmful to the application and the organization later on.
- As more and more Artificial Intelligence comes into our lives, the need for testing with Artificial intelligence is increasing.
- Taking the self-driving cars as an example: if the car's intelligence does not work properly and it makes a wrong decision, or the response time is slow, it could easily result in a car crash and puts human life in danger.

AI in Software Testing

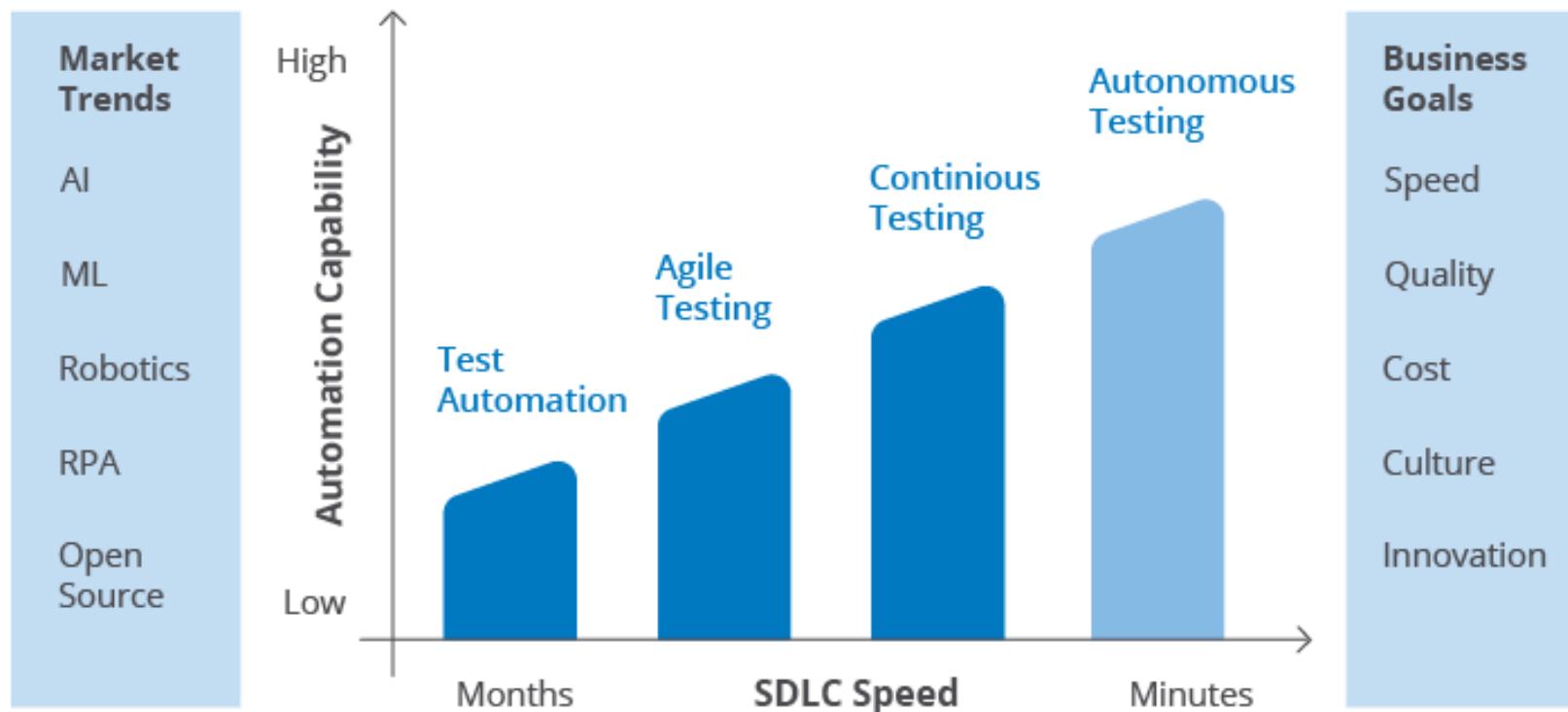


Why AI Testing

Why use AI Automation Testing?



How AI Will Impact Software Testing



How AI is changing the Dynamics of Software Testing?



- In an attempt to make the application safeguard, we are turning increasingly towards Artificial Intelligence (AI).
- As testing progressively moving towards greater automation, we may be turning over most of it to AI.
- This implies that instead of manual testing done by humans, we are slowly moving towards a scenario where machines will take over the execution of test codes.
- With minimal human input, however, will be required to help machines 'learn' and enhance themselves.

How AI is changing the Dynamics of Software Testing?



- It has, therefore, become essential to create an association directly pursuing the Grand Dream of Testing, where all the things are genuinely automated without human intervention and systems convey better testing over current application test teams.
- Make this thought a step further and imagine a world where software can test, diagnose, and heal itself.

What are the benefits of AI in Test Automation?

- The benefits of Artificial Intelligence in test automation are listed below:
 - Finding the right set of people
- Businesses may overcome the difficulty of finding a suitable team and skill set by leveraging AI-based test automation technologies that give testers with a semi- or completely scriptless scripting environment.



What are the benefits of AI in Test Automation?

- The amount of time spent on repeated jobs
- Every time a new test automation project arises, regardless of how reusable the components are, teams wind up creating a lot of comparable code again, which takes a long time.
- AI may be used to swiftly and automatically develop test scripts. AI tools may be taught based on past project inputs and results to automatically generate test scripts for comparable projects.

What are the benefits of AI in Test Automation?

- Flaky Test
- Teams of testers spend hours assessing whether a failed test was caused by application bugs or poorly prepared test cases.
- These kinds of test failures are known as flaky tests, and they cause a release to be held up needlessly, resulting in software delivery delays.
- AI can assist teams in overcoming the difficulty of flaky tests by developing more resilient test cases and detecting trends in random test failures to accelerate the process.

What are the benefits of AI in Test Automation?

- UI changes necessitate frequent script modifications.
- Businesses frequently adjust the app User Interface to deliver a consistent User Experience (UX) (UI).
- Even if the change is modest or invisible, it may cause the test scripts to fail while executing various operations on the page.
- AI and ML algorithm-based technologies may be trained to detect tiny changes in code or application issues.
- These technologies can then take appropriate steps, reducing the need for human intervention in script updates for such modest modifications.

What are the benefits of AI in Test Automation?

- Maintaining the test suites and test scripts.
- Maintaining a large number of test scripts gets difficult as an application grows.
- AI tools may be used to maintain and extract the appropriate test scripts based on testing needs, allowing AI to be utilized to tackle this difficulty.
- As a result, AI is supposed to assist overcome the problems of traditional test automation and usher in a revolution in test automation.

Need of Artificial Intelligence in Software Testing

- Software testing is a procedure that constitutes a very fundamental aspect in the area of development.
- However, many times developers can't do exhaustive testing (test approach in which all possible data combinations are used for testing) of an application because of the lack of resources and time.

Need of Artificial Intelligence in Software Testing

- We need a requirement for a system that could intelligently recognize regions that will be elaborated and more focused on the aspects that could be taken through automation based on repetitive patterns.
- Software testing takes up the most amount of time, human resource, and capital.
- And with the developers seeking faster deployments with inadequate infrastructure, Artificial Intelligence is a suitable path forward.
- Since 80% of testing is only a repetition of the checks that the software already possess, Artificial Intelligence will be helpful to automate the processes in an efficient way instead of a human tester which unnecessarily inflates costs and effort.

Need of Artificial Intelligence in Software Testing

- It would be a good practice if human intelligence as well as automation through AI to recognize the application issues by making exceptional and innovative test environments.
- It is ideal to leave the repetitive work to the Artificial Intelligence-powered automation which leaves just 20% of the testing operations to human creativity and reasoning ability.
- Artificial Intelligence algorithms can be tremendously helpful in the testing industry in making a smarter and more productive software for the end-user. It is, however, essential to interpret how to use Artificial Intelligence brilliantly.

Need of Artificial Intelligence in Software Testing

- Algorithms that operate like a genuine user accessing automation.
- From that point onward, one must identify the areas within the process that can be optimized with Artificial Intelligence and apply the Machine learning and deep learning algorithm.
- Having a smart algorithm can encourage the process, help testers to find the maximum number of bugs in less time and it will make the application more reliable and accurate.
- The outcomes after that can be used by the developers to refine the product and learn from trial and error.

How AI can be used in software testing?

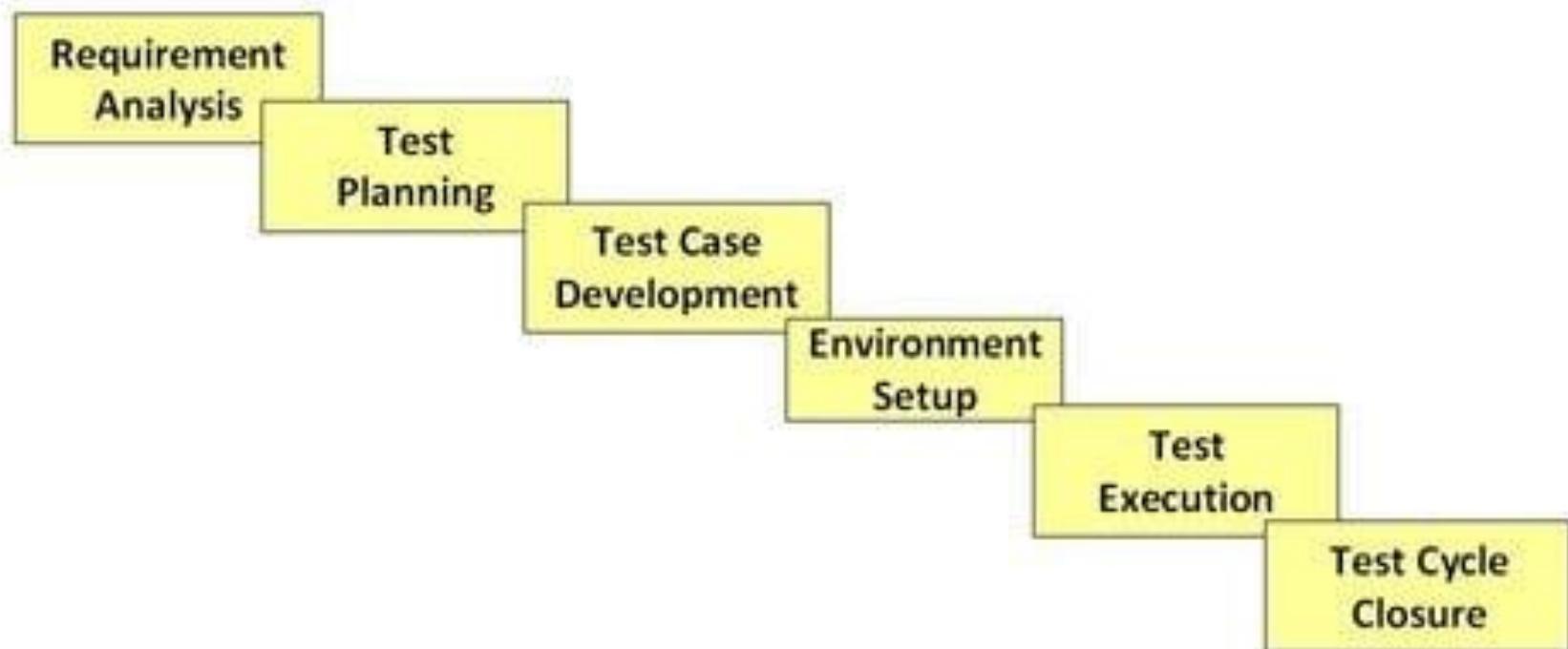
- a) Unit tests – Unit testing is very important to make sure that the build is stable and testable. With AI-powered unit test tools like RPA, a developer can get reduce the flaky test cases and maintenance of unit tests.
- b) API testing- API testing saves time and effort by getting into the root cause of the issue. The problem with UI tests is that they are not reliable anymore as UI keeps changing in agile, while API tests give a deeper insight into the application and directly hit the root cause of an issue eventually making the application more robust.
- There are many tools which are using artificial intelligence to help take the complexity out of API testing by converting manual UI tests into automated API tests, lowering the technical skills required to adopt API testing and helping organizations build a comprehensive API testing strategy that scales.
- c) UI testing- The first step in automation is to convert manual UI tests into automated tests. There are tools which leverage AI to run the test cases on multiple platforms and browsers and also learn from the functional flow, reducing the maintenance effort and making testing more reliable.

AI Powered Testing Tools

- 1. Applitools- It is an AI-powered visual testing and monitoring tool that can run tests on different browsers and platforms. It uses AI to identify the meaningful changes in UI and also identify them as bugs/ desired changes.
- It also leverages ML/AI-based for automated maintenance (being able to group together similar groups of changes from different pages/browsers/devices)
- 2. Testim- It leverages machine learning into the most critical part of automation which is execution and maintenance of tests.
- 3. Sealights- Sealights uses AI and machine learning to analyze the code and run tests which cover the impacted area. It can be any kind of test- unit, functional, performance, manual, etc.
- It provides a useful insight ‘Quality Risks’ which focuses user efforts on the things that matter by letting him or she knows exactly which files/methods/lines have changed in the last build that wasn’t tested by a specific test type (or any test type).

AI Powered Testing Tools

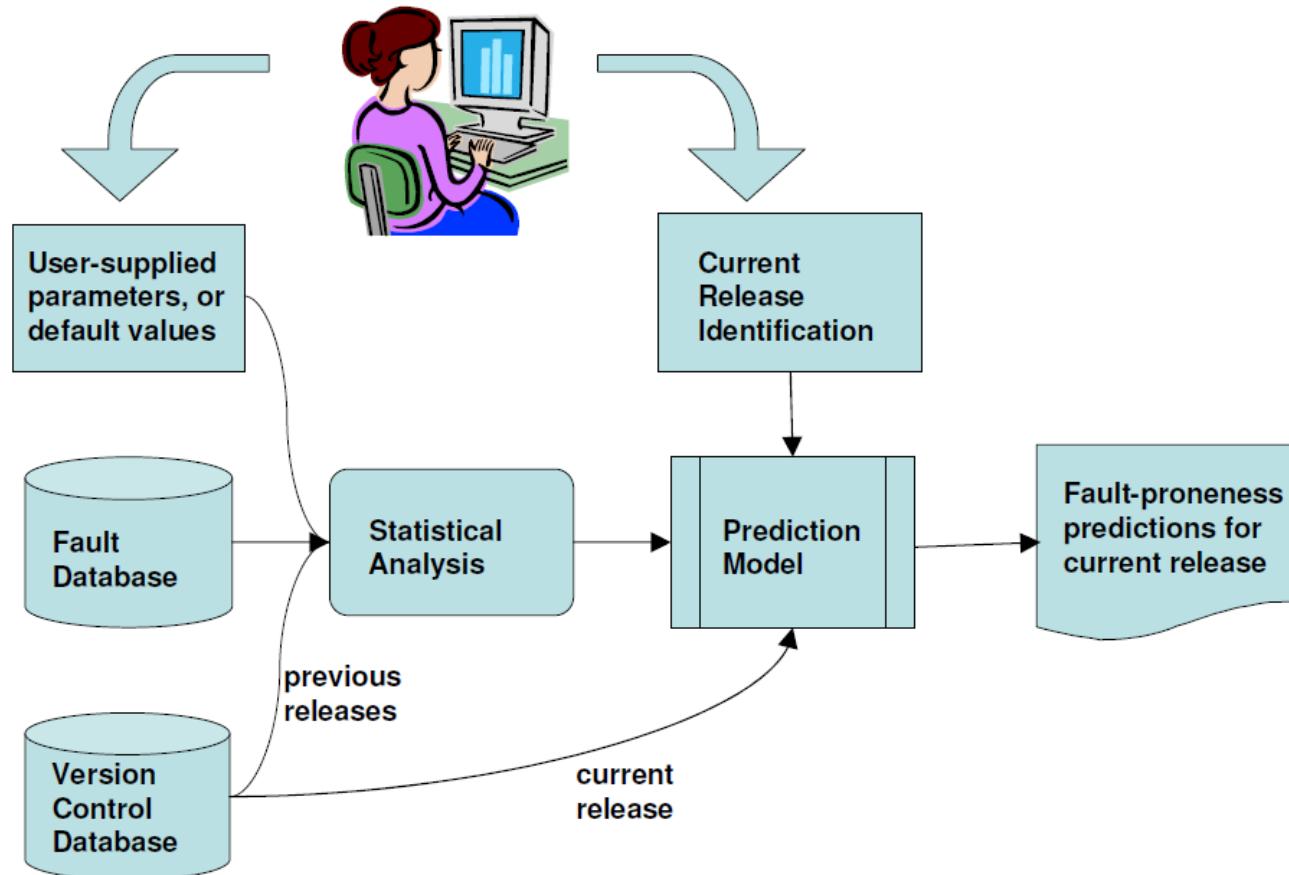
- 4. Test.AI- Test.AI is building as a tool that will add an AI brain to Selenium and Appium. It was created by Jason Arbon, co-author of How Google Tests Software and the founder of Appdiff. Tests are defined in a simple format similar to the BDD syntax of Cucumber, so it requires no code and no need to mess with element identifiers.
- AI just like a real person, IDENTIFIES the screens and elements in your app.
- AI EXECUTES user scenarios—test on-demand whenever you're ready
- AI RECOGNIZES elements so that even if things change, your test doesn't break.
- 5. MABL- Like the other AI-based test automation tools, MABL can automatically detect whether elements of your application have changed, and dynamically updates the tests to compensate for those changes. You just need to show the workflow that has to be tested and MABL does the rest.
- 6. Retest- Retest propagates an innovative testing approach, which is a combination of “intelligent” monkey testing and “difference testing” and works actually more like a GUI version management than conventional testing.
- This tool does Monkey testing whereby the monkey(is called Surili) is artificially intelligent and can be trained by users by capturing user actions.



Defect management and Automated Defect Prediction

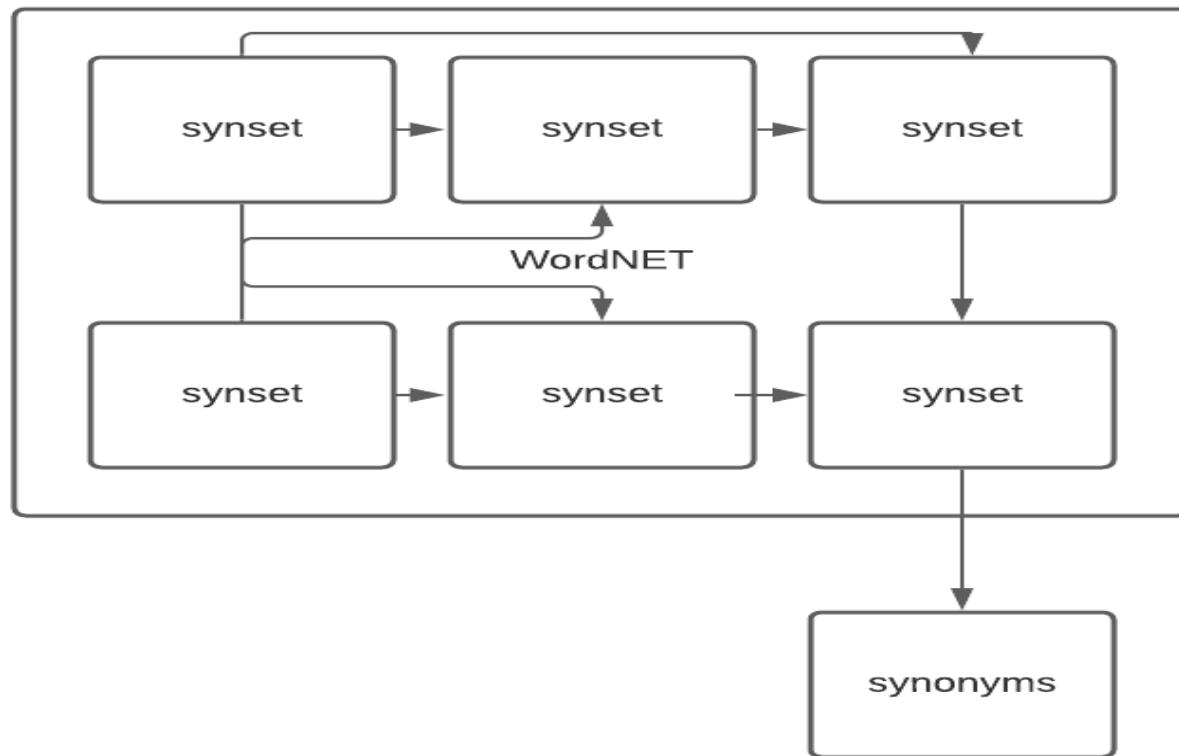


Prediction Tool Overview



WordNet

- <https://analyticsindiamag.com/a-complete-guide-to-using-wordnet-in-nlp-applications/>



WordNet

- <https://analyticsindiamag.com/a-complete-guide-to-using-wordnet-in-nlp-applications/>

WordNet Search - 3.1
- [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **journey**, [journeying](#) (the act of traveling from one place to another)

Verb

- S: (v) [travel](#), **journey** (undertake a journey or trip)
- S: (v) [travel](#), **journey** (travel upon or across) "*travel the oceans*"

AI Powered Testing Tools

- 7. ReportPortal- ReportPortal, as the name suggests, is an AI-powered automation tool which focuses more on report analysis and management. As per its website it-
 - Manage all your automation results and reports in one place
 - Make automation results analysis actionable & collaborative
 - Establish fast traceability with defect management
 - Accelerate routine results analysis
 - Visualize metrics and analytics
 - Make smarter decisions together
- 8. Functionlize- Functionlize provides an overall solution for seamless automation with less/no efforts in maintenance all with the help of AI. Its AEA tool finds and fixes the broken test scripts thus eliminating the manual maintenance.
- Functionlize uses machine learning for functional testing and is very similar to other tools in the market regarding its capabilities such as being able to create tests quickly (without scripts), execute multiple tests in minutes, and carry out in-depth analyses.

Questions



Module Summary

- Spring Integration Framework.
- Message, Channel and Adapter
- Understood the different Component Integration
- Understood the Event-Driven Architecture

