Machine Learning
BEGINNER → EXPERT

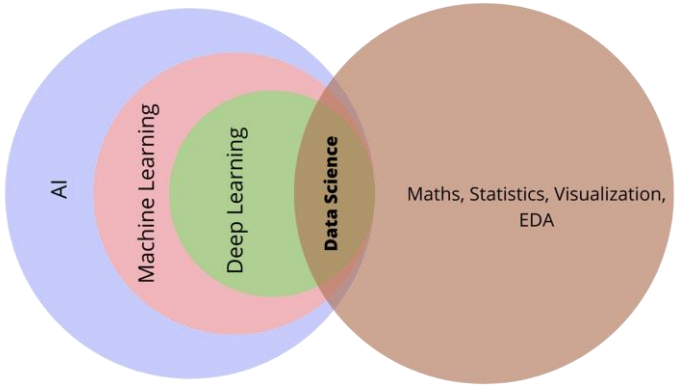
Agenda

1. Introduction
2. Use Cases
3. Types of Learning
4. Essential Libraries
5. Feature Scaling
6. Regression Algo's
7. Classification Algo's
8. Clustering Algo's
9. Association Rule Learning
10. Ensemble Techniques
11. Time Series Analysis
12. Dimensionality Reduction - Feature Engineering
13. Hyperparameter Optimization

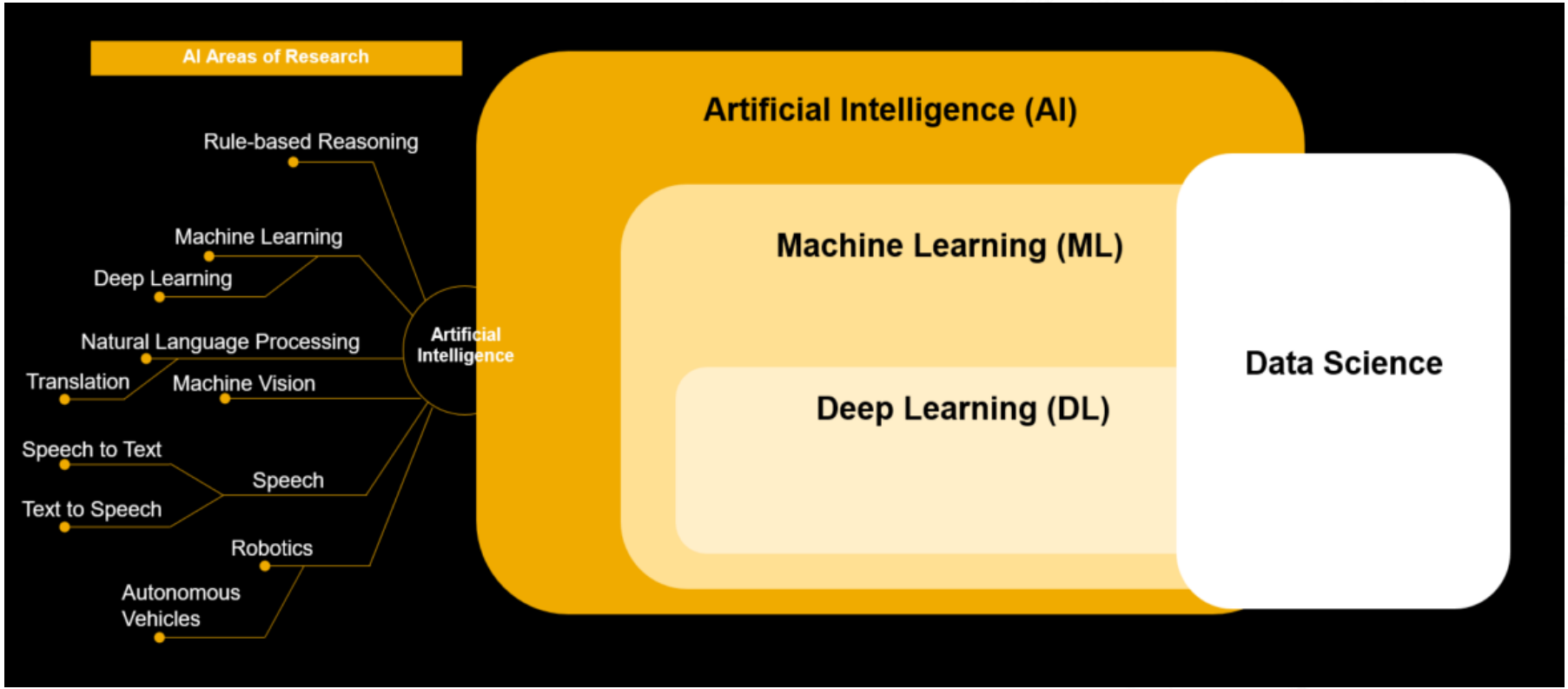
Introduction

Introduction

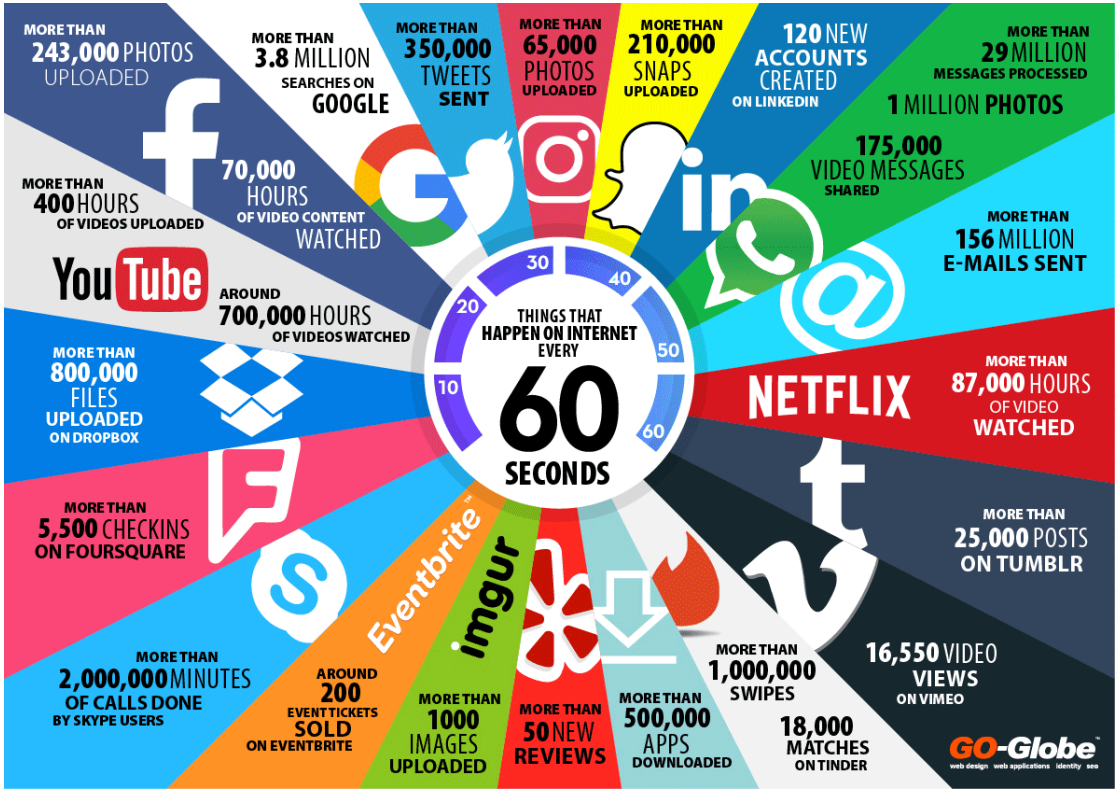
- ML is a subset of AI.
- Name derived from the concept that it deals with “construction & study of systems that can learn from data”



Biggest Confusion: AI vs ML vs DS vs DL



What is Machine Learning?

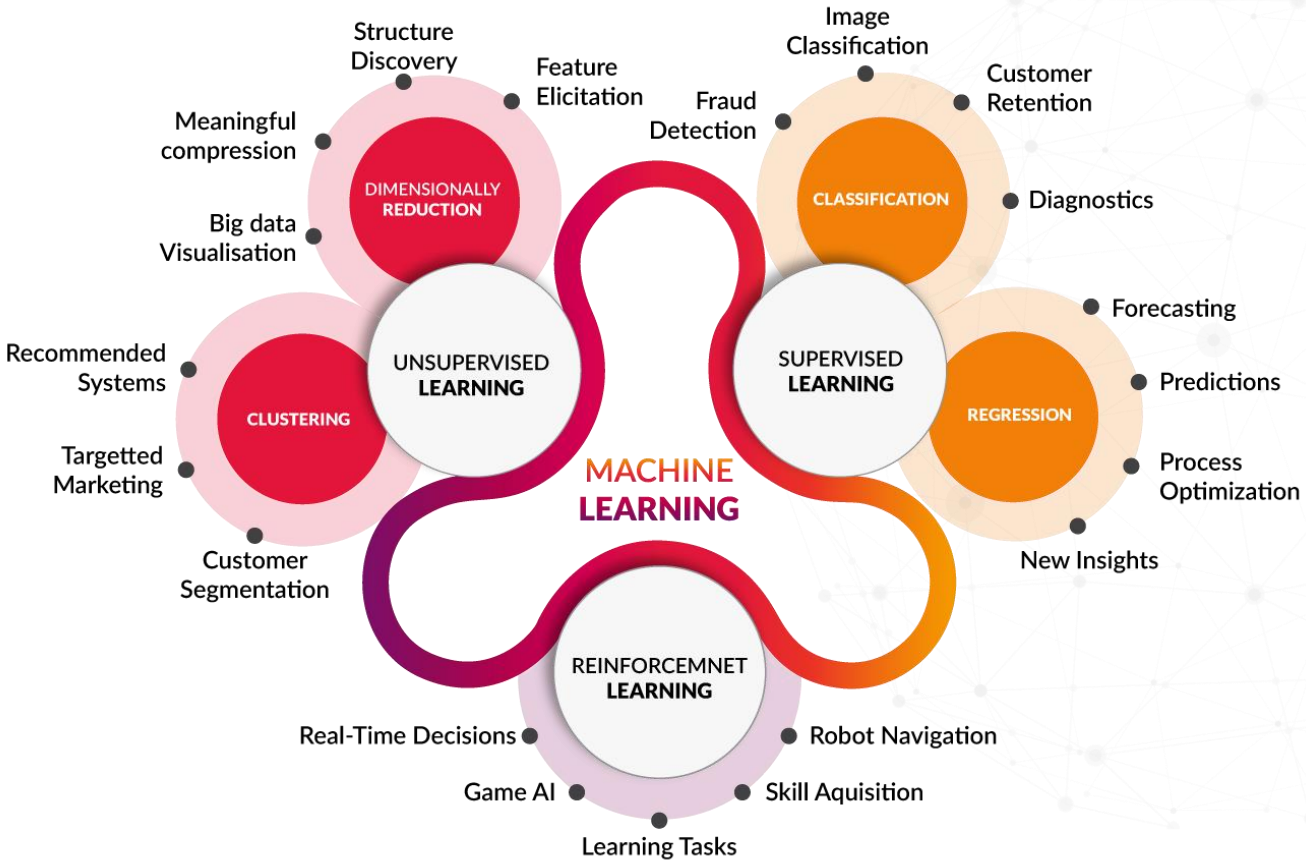


- Machine learning is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed.
- Machine learning focuses on the development of computer programs that can teach themselves to grow and change when exposed to new data.

Why we need Machine Learning?

- Volume of data collected growing day by day.
- Data production is 5000% greater in 2021 than in 2010.
- Every day, 2.5 quintillion bytes of data are created
- Data is nearly doubling in size every two years.
- Knowledge Discovery is needed to make sense and use of data.
- Machine Learning is a technique in which computers learn from data to obtain insight and help in knowledge discovery

Types of Learning

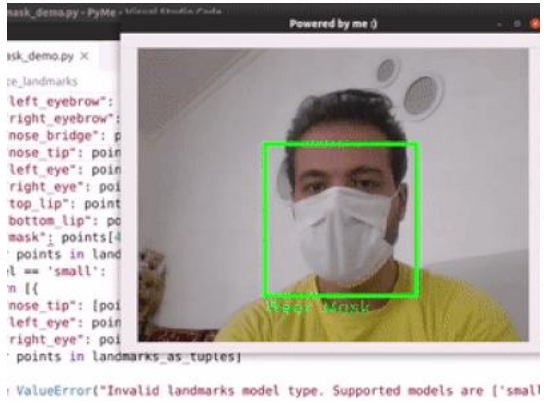


Use Cases

Applications of ML

amazon.com

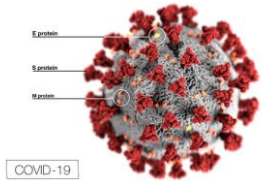
NETFLIX



Google Maps

Health Care Use Cases

Better Imaging & Diagnostic Techniques

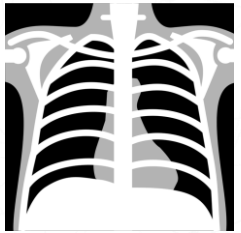


Disease Detection using Machine Learning

Providing Personalized Treatment

Preventing Medical Insurance Frauds

Drug Discovery & Research

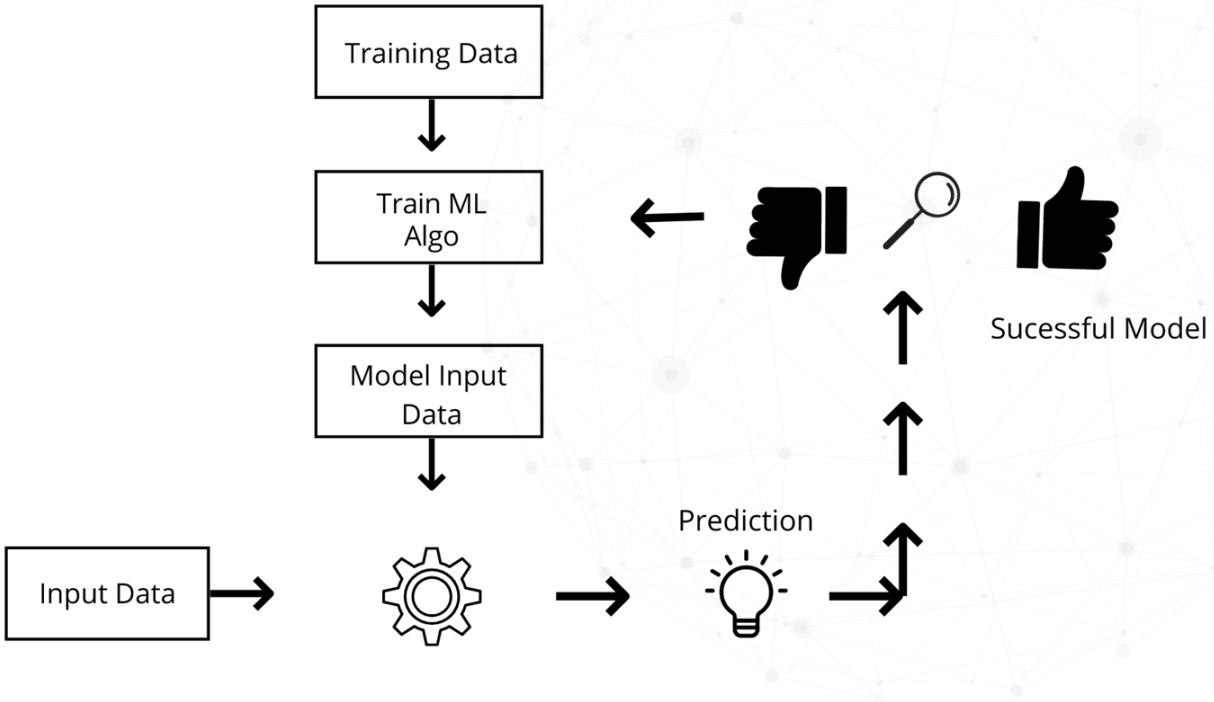


Disease detection using Deep Learning

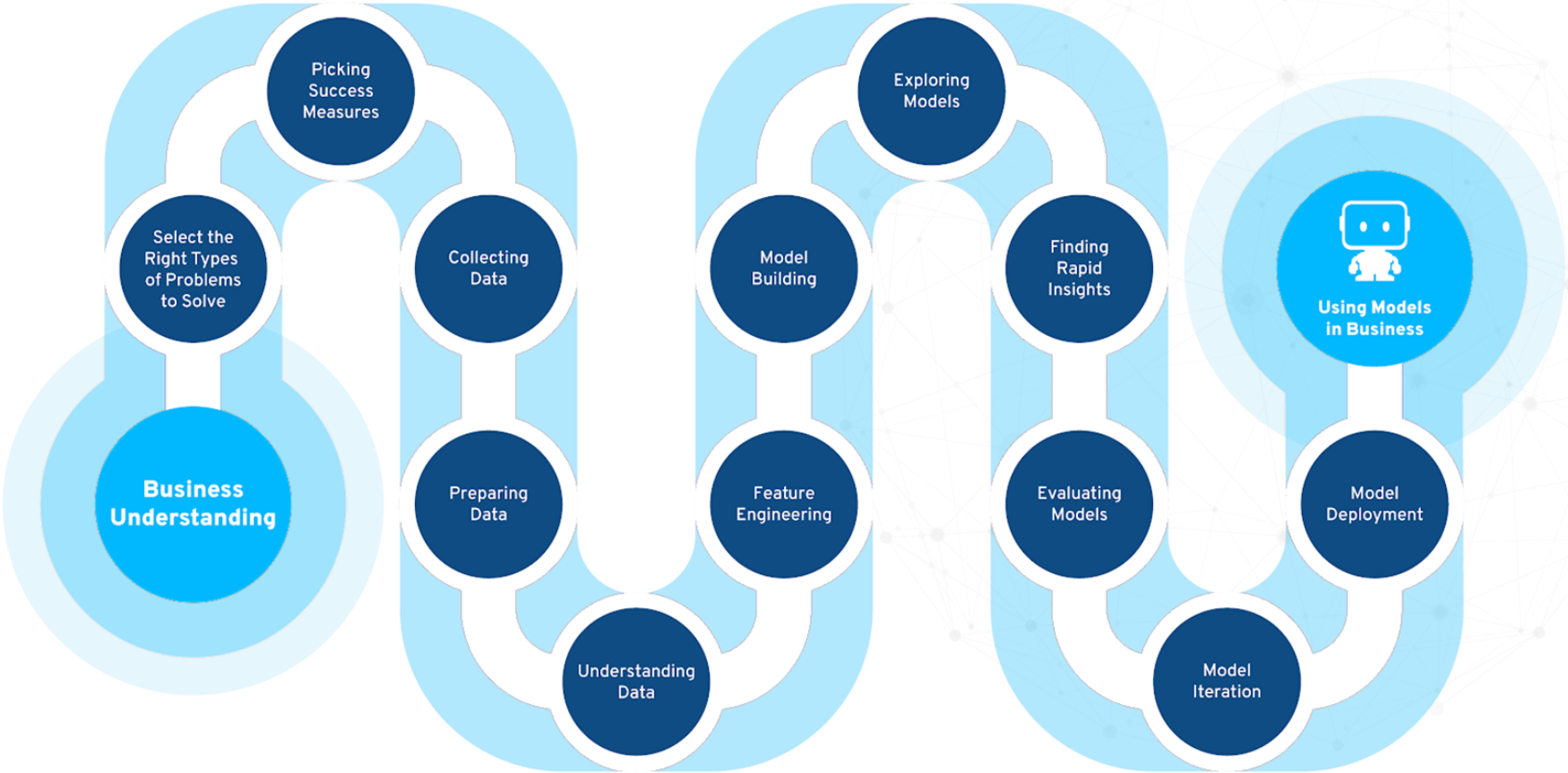
Types of Learning

Types of Learning

1. Supervised
2. Unsupervised
3. Reinforcement



Learning Path



Getting your dream job

1. **Internet presence** – Create profiles on Naukri, Monster, LinkedIn, Hirist, Glassdoor, Indeed, StackOverFlow etc.
2. Push reusable code/ apps to **Github**. Link it to your profile.
3. **Write Blogs**, link it on your profiles.
4. **Differentiator** - Focus both on depth and breadth of Data Science.
5. Profiles should contain all the **tech stacks keywords** – e.g. Deep Learning, Machine Learning, NLP, Spark, Flask, Kafka, NoSQL, Python etc.
6. **Headlines are important** – Data Scientist, Machine Learning Engineer, etc.
7. **Prefix-Suffix** with lead/ architect/ Head based on the years of experience and current role.
8. **Project are a must** showing good amount of relevant project experience solving industry level use cases.
9. Participate in **online Hackathons** and **Code challenges**.

Essential Libraries

Essential Libraries

1. numpy: The matrix / numerical analysis layer at the bottom
2. scipy: Scientific computing utilities (linalg, FFT, signal/image processing...)
3. sklearn: Machine learning (our focus here)
4. matplotlib: Plotting and visualization
5. opencv: Computer vision
6. pandas: Data analysis
7. caffe, theano, minerva, tensorflow, keras: Deep neural networks
8. spyder: The front end (Scientific Python Development Environment)

Feature Scaling

Train Test Split

The train_test_split() method is used to split our data into train and test sets.

First, we need to divide our data into features (X) and labels (y). The dataframe gets divided into X_train,X_test , y_train and y_test. X_train and y_train sets are used for training and fitting the model. The X_test and y_test sets are used for testing the model if it's predicting the right outputs/labels. we can explicitly test the size of the train and test sets. It is suggested to keep our train sets larger than the test sets.

The train_test_split() method is used to split our data into train and test sets.

First, we need to divide our data into features (X) and labels (y). The dataframe gets divided into X_train,X_test , y_train and y_test. X_train and y_train sets are used for training and fitting the model. The X_test and y_test sets are used for testing the model if it's predicting the right outputs/labels. we can explicitly test the size of the train and test sets. It is suggested to keep our train sets larger than the test sets.

Train set: *The training dataset is a set of data that was utilized to fit the model. The dataset on which the model is trained. This data is seen and learned by the model.*

Test set: *The test dataset is a subset of the training dataset that is utilized to give an accurate evaluation of a final model fit.*

validation set: *A validation dataset is a sample of data from your model's training set that is used to estimate model performance while tuning the model's hyperparameters.*

Feature Scaling

Sepal Length, Sepal Width → What is the magnitude & Units.

Most ML's use Euclidean distance, hence without feature scaling, most of the algo's neglect the Units and focus on Magnitude, in that case the Euclidean distance may vary significantly → Hence, o/p will be impacted.

1. Standardisation
2. Mean Normalization
3. Min-Max Scaling
4. Vector Scaling

Regression Algorithms

Regression

Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables.

Let's go through various regression algorithms:

1. Linear Regression
2. Lasso Regression
3. Polynomial Regression
4. Stepwise Regression
5. Ridge Regression

Linear Regression

Simple linear regression is a statistical method that enables users to summarise and study the relationships between two continuous (quantitative) variables. Linear regression is a linear model wherein a model that assumes a linear relationship between the input variables (x) and the single output variable (y). Here, y can be calculated from a linear combination of the input variables (x). When there is a single input variable (x), the method is called a simple linear regression. When there are multiple input variables, the procedure is referred to as multiple linear regression.

Given our simple linear equation

$$y=mx+b$$

we can calculate MSE as:

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - (mx_i + b))^2$$

Note

- N is the total number of observations (data points)
- $\frac{1}{N}\sum_{i=1}^{n}$ is the mean
- y_i is the actual value of an observation and $mx_i + b$ is our prediction

Multiple Linear Regression

The difference between simple linear regression and multiple linear regression, multiple linear regression has (>1) independent variables, whereas simple linear regression has only 1 independent variable.

Simple Linear Regression → y = bo + b1*x1

Multiple Linear Regression → y = bo + b1*x1 + b2*x2 + ... + bn*xn

Where, y → Dependent variable

x1, x2, ...xn → Independent variables

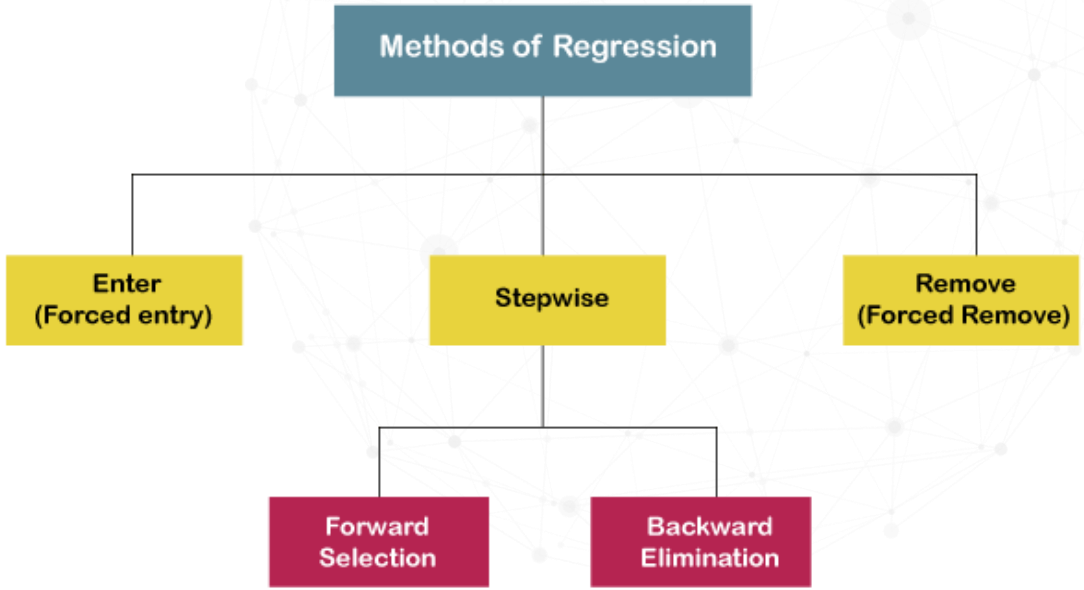
dataset - DataFrame

Index	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349	136898	471784	New York	192262
1	162598	151378	443899	California	191792
2	153442	101146	407935	Florida	191050
3	144372	118672	383200	New York	182902
4	142107	91391.8	366168	Florida	166188
5	131877	99814.7	362861	New York	156991
6	134615	147199	127717	California	156123

Multiple Linear Regression

5 methods of building models:

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



Multiple Linear Regression

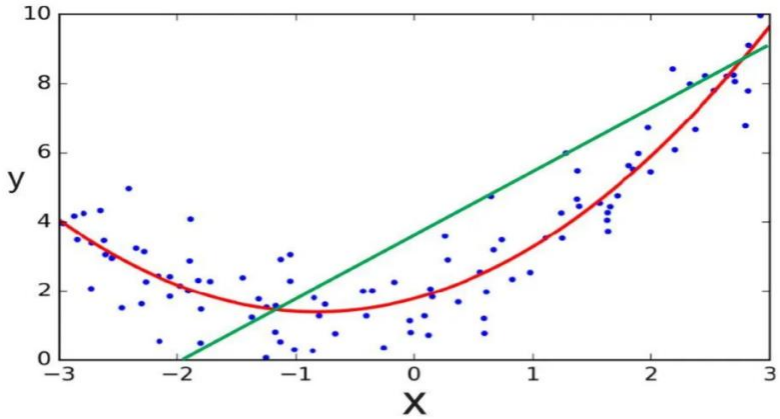
Backward Elimination → Stepwise

1. Select a significance level to stay in the model (Let say 0.05)
2. Fit the full model with all possible predictors
3. Consider the predictor with highest p-value and if p-value > SL, go to step 4, else FIN
4. Remove that predictor
5. Fit the model without this variable

Forward Pass → Stepwise

1. Select a significance level to stay in the model (Let say 0.05)
2. Fit all simple regression models. Select one with lowest p-value
3. Keep the variable & fit all the possible models with one extra predictor added to one you already have.
4. Consider the predictor with lowest p-value. If p>SL, go to Step 3 else go to FIN

Polynomial Regression



In this type of data, linear equation might not be a good fit as seen in the green line, hence we use Polynomial equation i.e. red line, which completely fits in with the data points and comes with an equation of

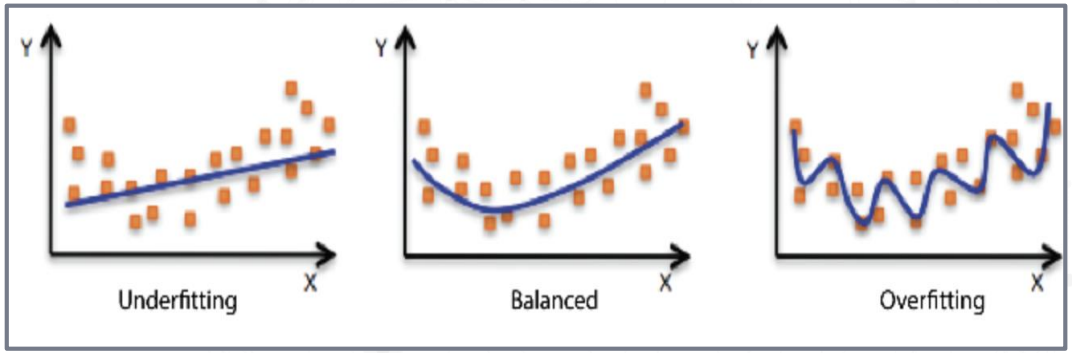
$$y = b_0 + b_1 x_1 + b_2 x_1^2$$

The equation of Polynomial Regression is:

Simple Linear Regression	$y = b_0 + b_1 x_1$
Multiple Linear Regression	$y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$
Polynomial Linear Regression	$y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n$

LASSO and RIDGE Regression

- Regularisation helps avoid overfitting data at the cost of some added error.
- LASSO (Least Absolute Shrinkage Selector Operator) and Ridge regression are regularisation techniques used for improving the robustness of the model.
- LASSO or L1 penalty can be used for dimensionality reduction also



$$\sum_{i=1}^{n}(y_i - \sum_{j} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

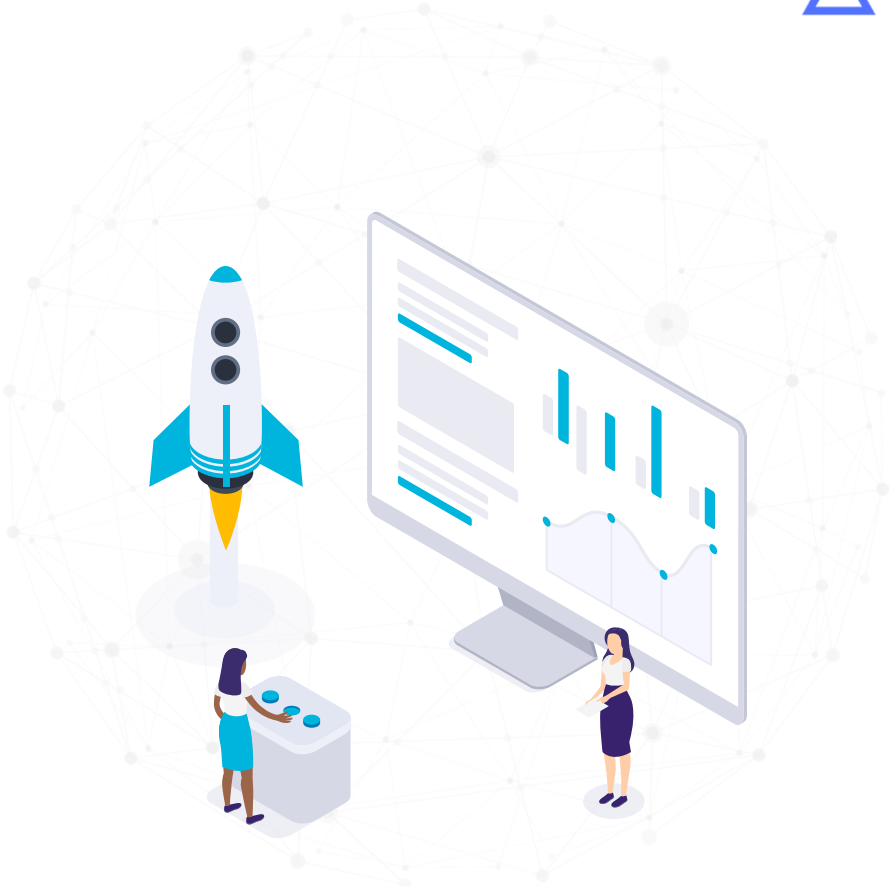
LASSO and RIDGE Regression

- Regularisation helps avoid overfitting data at the cost of some added error.
- LASSO (Least Absolute Shrinkage Selector Operator) and Ridge regression are regularisation techniques used for improving the robustness of the model.
- LASSO or L1 penalty can be used for dimensionality reduction also

Regression Error Metrics

MAE, MSE, RMSE, MAPE etc.

- MSE (Mean Squared Error)
- RMSE (Root Mean Squared Error)
- MAE (Mean Absolute Error)
- MAPE (Mean Absolute Percentage Error)



Mean Absolute Error (MAE)

MAE: We just look at the absolute difference between data and model's predictions

Lower MAE → Better model

$$MAE = \frac{1}{n}\sum \left| y - \hat{y} \right|$$

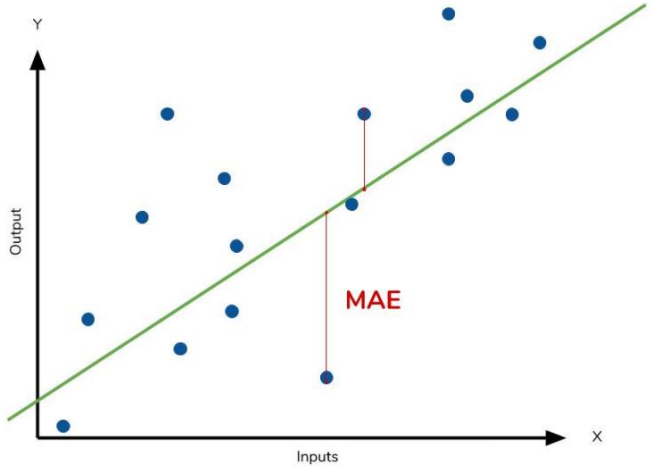
Divide by the total number of data points

Predicted output value

Actual output value

Sum of

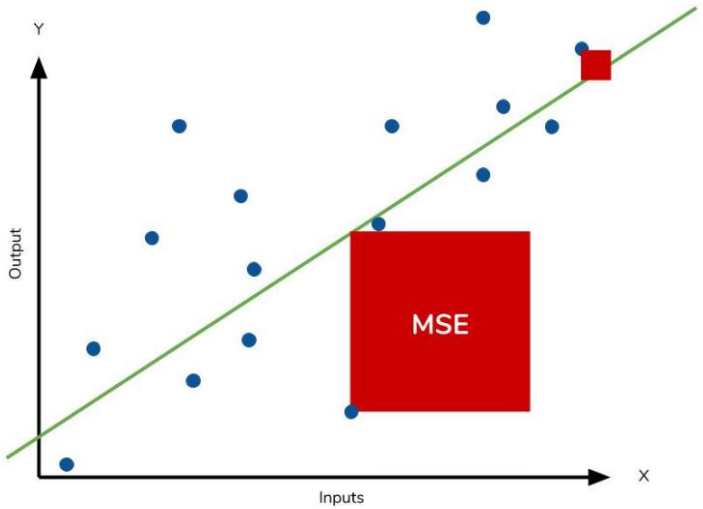
The absolute value of the residual



Mean Squared Error (MSE)

MSE is going to be a huge number because of squaring, hence, we can't compare it with MAE. This ultimately means that outliers in our data will contribute to much higher total error in the MSE than they would the MAE. Similarly, our model will be penalized more for making predictions that differ greatly from the corresponding actual value. This is to say that large differences between actual and predicted are punished more in MSE than in MAE.

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$



Root Mean Squared Error (RMSE)

RMSE = Square root(MSE)

As the name suggests, it is the square root of the MSE. Because the MSE is squared, its units do not match that of the original output. Researchers will often use RMSE to convert the error metric back into similar units, making interpretation easier. Since the MSE and RMSE both square the residual, they are similarly affected by outliers.

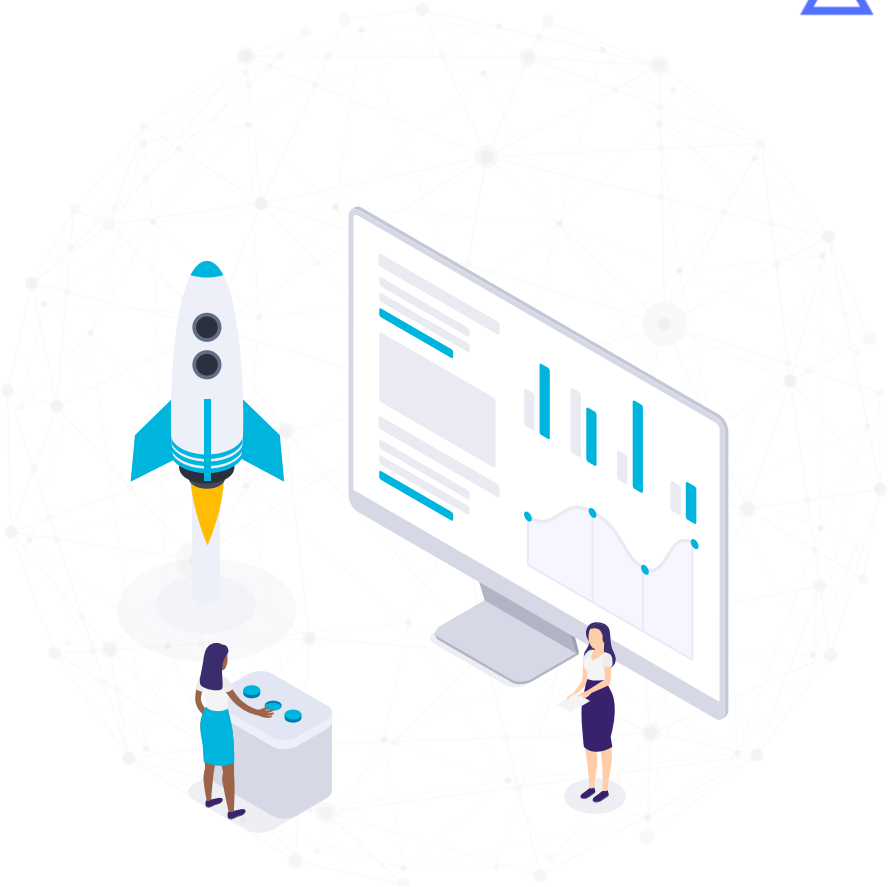
Classification Algorithms

Classification

Predicting category of new observations

Let's go through various classification algorithms:

1. k-nn (K- nearest neighbours)
2. Decision Trees
3. Random Forest
4. Naive Bayes
5. Support Vector Machines
6. Logistic Regression

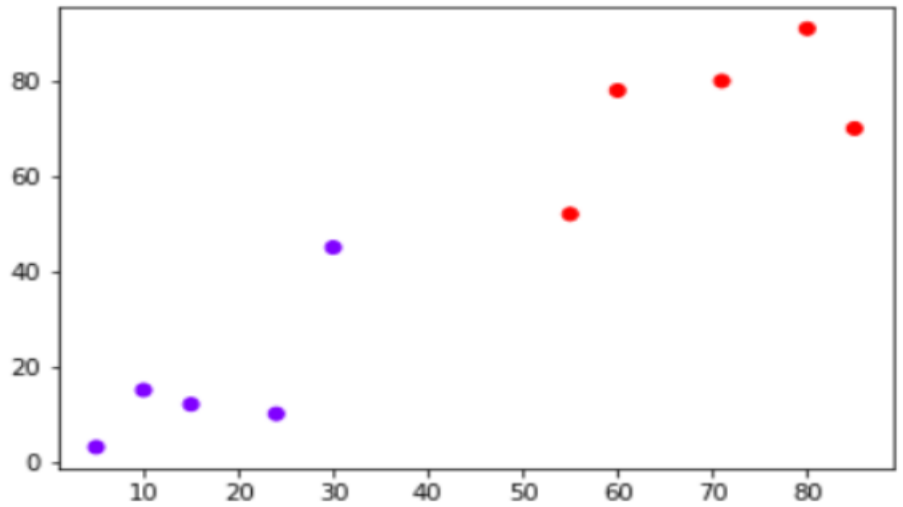


KNN (K-Nearest Neighbors)

Let say, we have a point at (40,60)

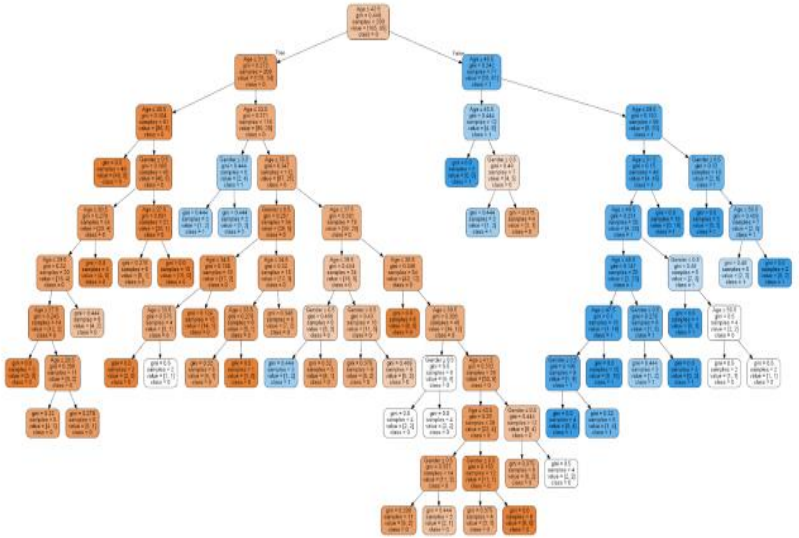
As we have 2 classes here, let's assume k = 3 (k can be any value as multiple of number of classes + 1)

As, we see 2 reds and 1 blue nearby the (40,60) mark, we can assume the new datapoint as red class.



Decision Tree

It works on the principle of identifying the root node, and creates a tree to traverse from top-bottom approach to identify the right classes.

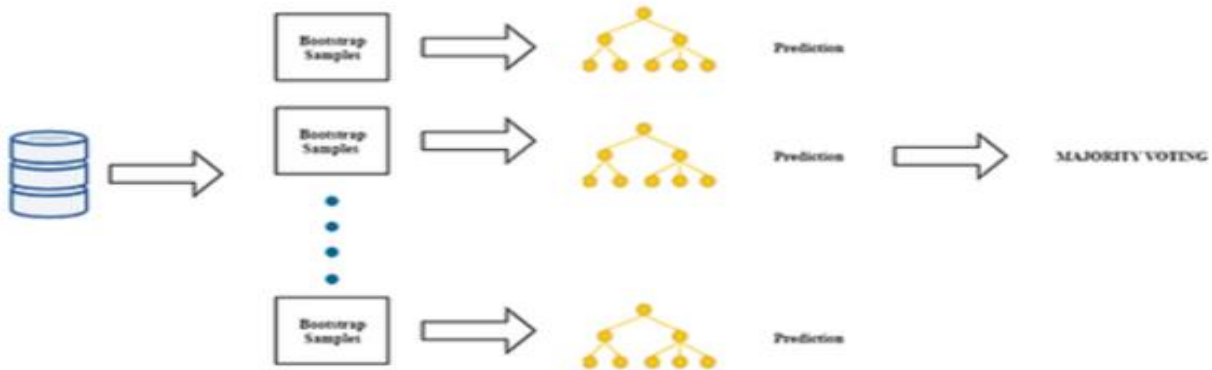


Decision Tree

Age	Competition	Type	Profit
Old	Yes	Software	Down
Old	No	Software	Down
Old	No	Hardware	Down
Mid	Yes	Software	Down
Mid	Yes	Hardware	Down
Mid	No	Hardware	Up
Mid	No	Software	Up
New	Yes	Software	Up
New	No	Hardware	Up
New	No	Software	Up

Random Forest

Random Forest uses multiple learning algorithms to obtain a better predictive performance. It uses decision trees as base learners.



Naive Bayes

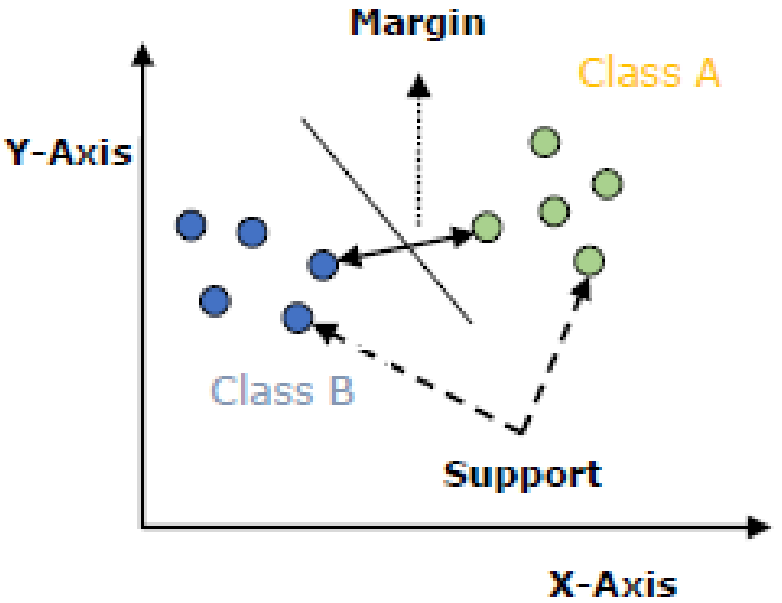
Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

To start with, let us consider a dataset.

Day	Temp	Humidity	Wind	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Overcast	Cool	Normal	Strong	No

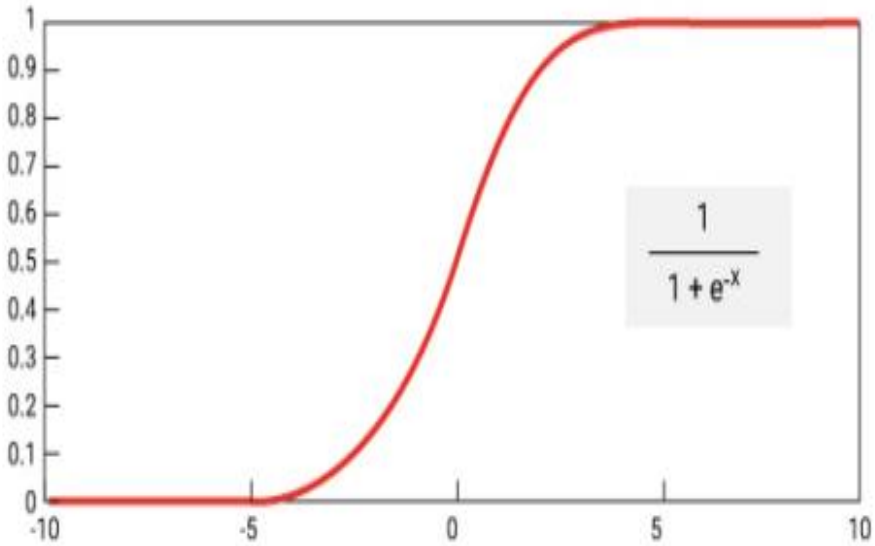
Support Vector Machines

An SVM model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).



Logistic Regression

Logistic Regression is a supervised Machine Learning algorithm and despite the word 'Regression', it is used in binary classification. By binary classification, it meant that it can only categorize data as 1 (yes/success) or a 0 (no/failure). In other words, we can say that the Logistic Regression model predicts P(Y=1) as a function of X.

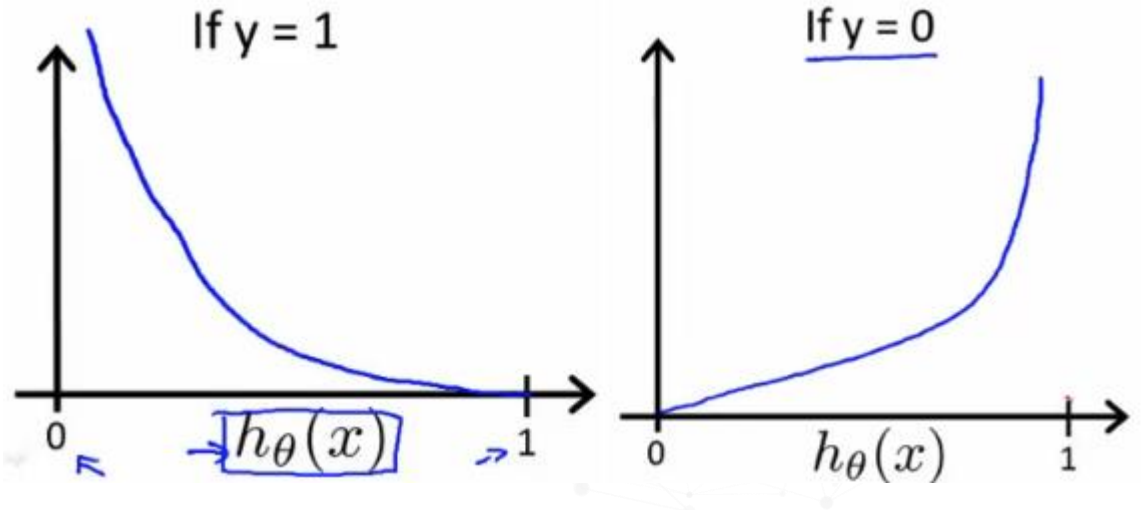


Classification Metrics – Log Loss

Cost = -(yact) ln (ypred) - (1-yact) ln (1 - ypred)

Cost = -ln ypred, where yact = 1

Cost = -ln (1-ypred), where yact = 0



Confusion Matrix

Useful for measuring recall, precision, specificity, accuracy & AUC-ROC Curve.

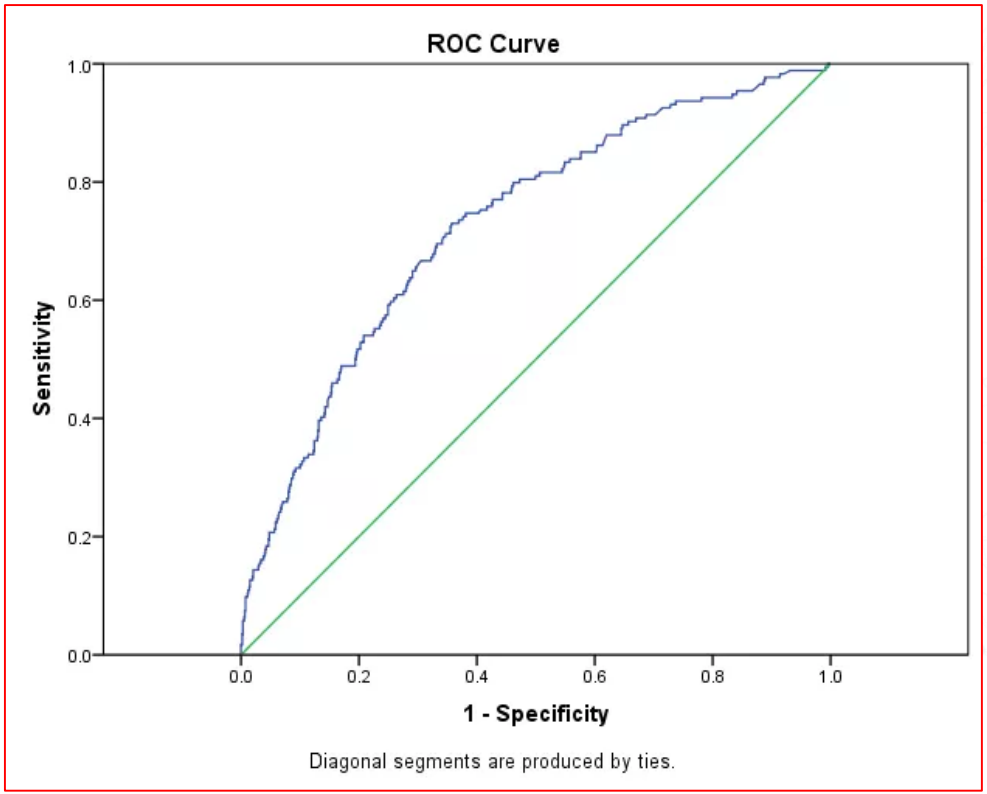
False Positive: Type I Error
False Negative: Type II Error

	Actual : 0	Actual : 1
Predicted : 0	#TN	#FN
Predicted : 1	#FP	#TP

Confusion Matrix

y	ypred	output for threshold 0.6	TP	TN	FP	FN	Recall	Precision	Accuracy
0	0.5	0							
1	0.9	1							
0	0.7	1							
1	0.7	1	2	2	1	2	1/2	2/3	4/7
1	0.3	0							
0	0.4	0							
1	0.5	0							

Classification Metrics – Area under ROC



$$SN = \frac{TP}{TP + FN} = \frac{TP}{P}$$

$$SP = \frac{TN}{TN + FP} = \frac{TN}{N}$$

Issues in Classification

There are many issues that occur while you solve a Classification problem, such as

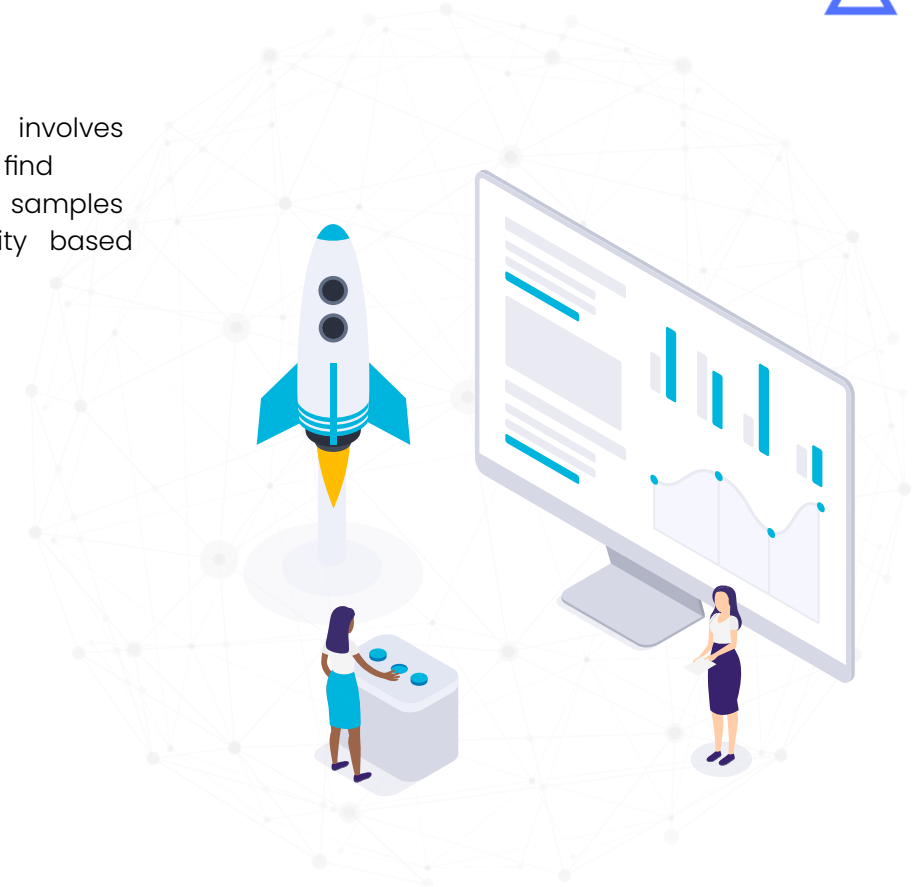
1. Overfitting
2. Class Imbalance Problems

Clustering Algorithms

Clustering

Clustering methods are a Machine Learning technique that involves the grouping of data points. These methods are used to find similarity as well as the relationship patterns among data samples and then cluster those samples into groups having similarity based on features.

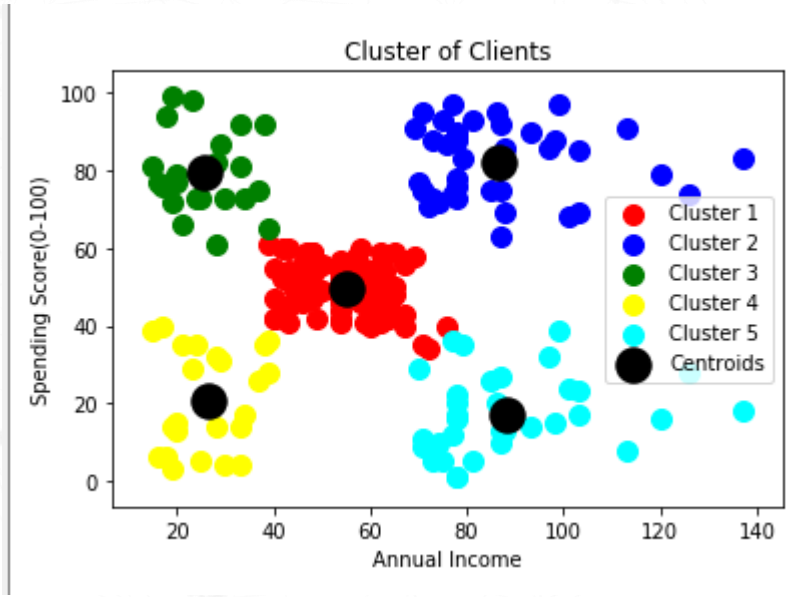
Types of Clustering Algorithms:
1. K-Means Clustering
2. Hierarchical Clustering
3. Mean-Shift Algorithm



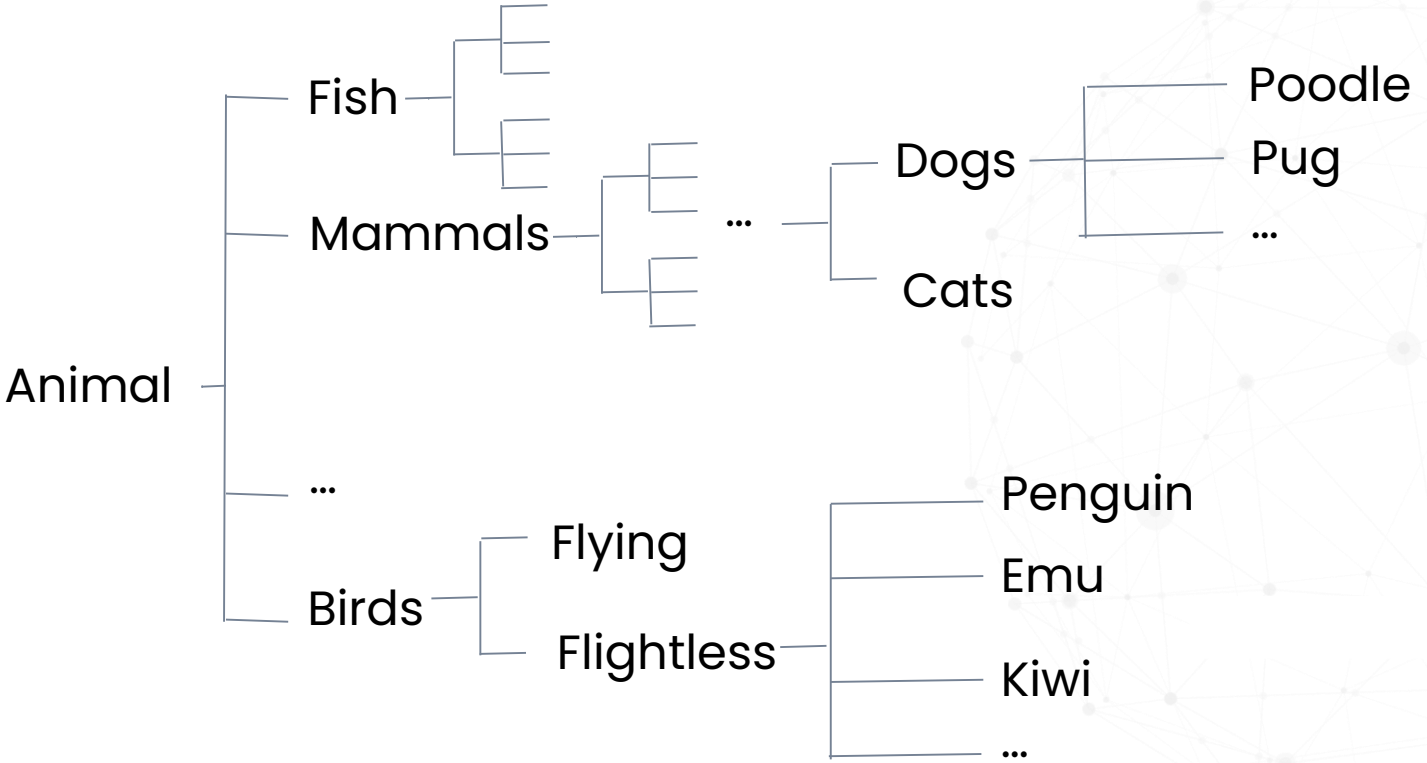
K-means Clustering

Pseudo Code:

1. Choose number of K Clusters
2. Select a random K points, the centroids (not necessarily from the dataset)
3. Assign each data set point to the nearest centroid → That forms K clusters
4. Compute & place new centroid for each cluster
5. Reassign each data point to the closest centroid.
6. If any reassignment took place, go to Step 4, else FINISH.



Hierarchical Clustering

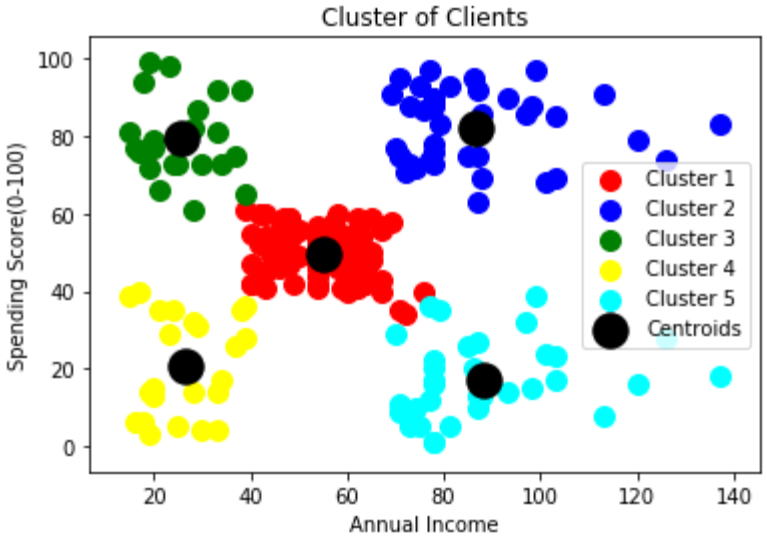


Hierarchical Clustering

Pseudo Code:

1. Make each data point a single point cluster → N Clusters
2. Take two closest data points & make 1 cluster → N -1 Clusters
3. Take two closest clusters & make 1 → N - 2 Clusters
4. Repeat Step 3 until there is one cluster.

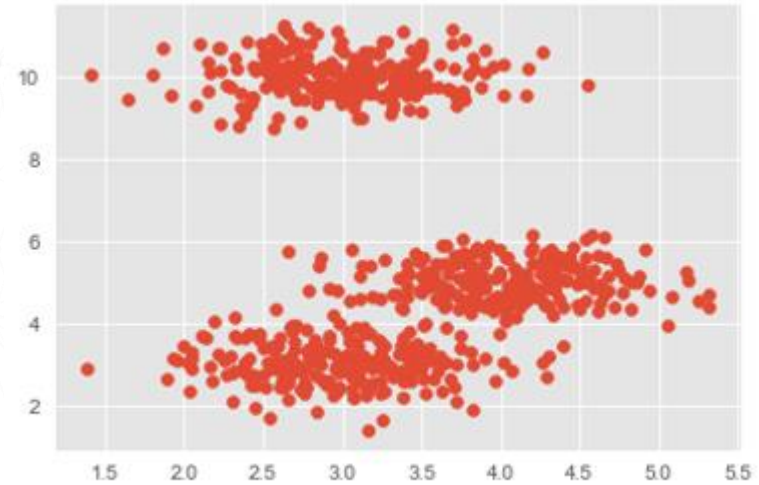
Pros:
HC shows all the possible linkages between clusters, & we understand the data better.
Cons: Can't handle big data



Mean Shift Clustering

Mean-shift algorithm basically assigns the data points to the clusters iteratively by shifting points towards the highest density of data points i.e. cluster centroid.

The difference between K-Means algorithm and Mean-Shift is that later one does not need to specify the number of clusters in advance because the number of clusters will be determined by the algorithm w.r.t. data.



Association Rule Learning

Association Rule Learning

From Wikipedia:

Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered databases using some measures of interest.

Association Rule Learning - Apriori

Support:
This explains how important an itemset is, as measured by the proportion of transactions in which an itemset appears.
Support(Apple) = 4/8

Confidence:
This says how likely item Y is purchased when item X is purchased, expressed as [X→Y]
Confidence(Apple→Beer) =
Support(Apple,Beer)/Support(Apple)
= (⅜)/(4/8) = ⅜*2 = ¾

Lift: This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is.
Lift(Apple→Beer) =
Support(Apple,Beer)/(Support(Apple)*Support(Beer))
= (⅜) / (4/8*6/8) = ⅜ * 64/24 = 1

Suppose a dataset exists such as the one below:

Transactions	Items
1	Apple → Beer → Chips → Chicken
2	Apple → Beer → Chips
3	Apple → Beer
4	Apple → Mango → Banana
5	Milk → Beer → Chips → Chicken
6	Milk → Beer → Chips
7	Milk → Beer
8	Milk → Mango

Association Rule Learning - Eclat

ECLAT: Equivalence Class Clustering and bottom-up Lattice Traversal

How the algorithm work?

- The basic idea is to use Transaction Id Sets(tidsets) intersections to compute the support value of a candidate and avoiding the generation of subsets which do not exist in the prefix tree.
- Uses breadth first search and hence does not use a lot of memory
- Computationally Faster than apriori
- In Apriori we need to add Support, Confidence But in Eclat only we need to give Support.

Transaction Id	Bread	Butter	Milk	Coke	Jam
T1	1	1	0	0	1
T2	0	1	0	1	0
T3	0	1	1	0	0
T4	1	1	0	1	0
T5	1	0	1	0	0
T6	0	1	1	0	0
T7	1	0	1	0	0
T8	1	1	1	0	1
T9	1	1	1	0	0

Association Rule Learning – Eclat

k = 1, minimum support = 2

ITEM	TIDSET
Bread	{T1, T4, T5, T7, T8, T9}
Butter	{T1, T2, T3, T4, T6, T8, T9}
Milk	{T3, T5, T6, T7, T8, T9}
Coke	{T2, T4}
Jam	{T1, T8}

k = 2

ITEM	TIDSET
{Bread, Butter}	{T1, T4, T8, T9}
{Bread, Milk}	{T5, T7, T8, T9}
{Bread, Coke}	{T4}
{Bread, Jam}	{T1, T8}
{Butter, Milk}	{T3, T6, T8, T9}
{Butter, Coke}	{T2, T4}
{Butter, Jam}	{T1, T8}
{Milk, Jam}	{T8}

Association Rule Learning – Eclat

k = 3

ITEM	TIDSET
{Bread, Butter, Milk}	{T8, T9}
{Bread, Butter, Jam}	{T1, T8}

k = 4

ITEM	TIDSET
{Bread, Butter, Milk, Jam}	{T8}

ITEMS BOUGHT	RECOMMENDED PRODUCTS
Bread	Butter
Bread	Milk
Bread	Jam
Butter	Milk
Butter	Coke
Butter	Jam
Bread and Butter	Milk
Bread and Butter	Jam

Ensemble Techniques

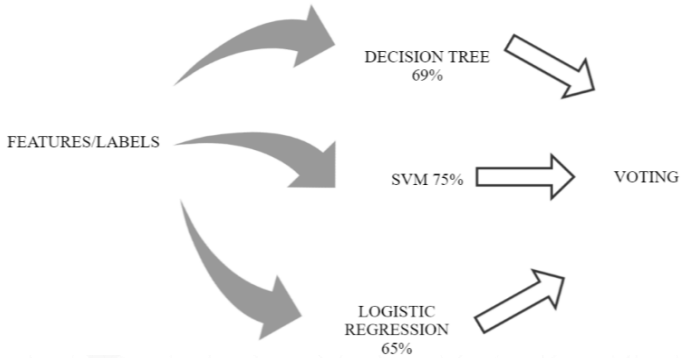
Ensemble Techniques

Ensemble Learning is the mechanism to use multiple algorithms together to have a better prediction than the individual models.

Let say, we extract the features from a use case, and try to check the model's accuracy or behaviour individually.

Features → Decision Trees → 69%
Features → SVM → 75%
Features → Logistic Regression → 65%

So why use Ensemble Learning?
1. Better Accuracy (Low Error)
2. Higher Consistency (Avoids Overfitting)
3. Reduces Bias & Variance Errors



Ensemble Techniques – Cont.

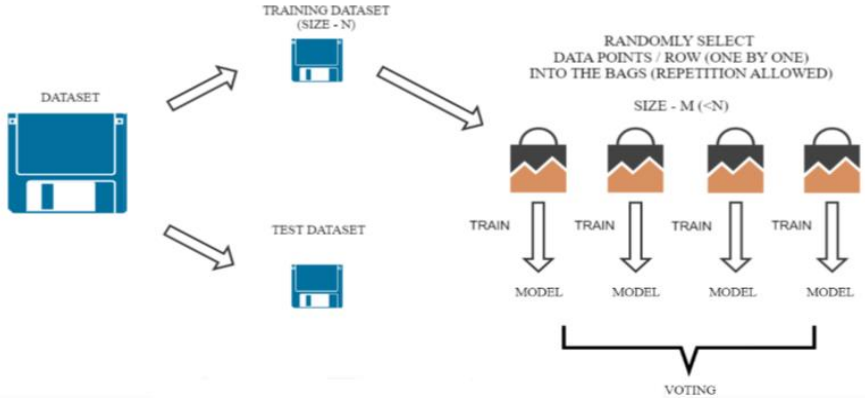
When and Where do we use Ensemble Learning?

1. Single model overfits
2. Results worth the extra training
3. Can be used for classification & regression both.

Popular Ensemble Methods

1. Bootstrap Aggregation (Bagging)

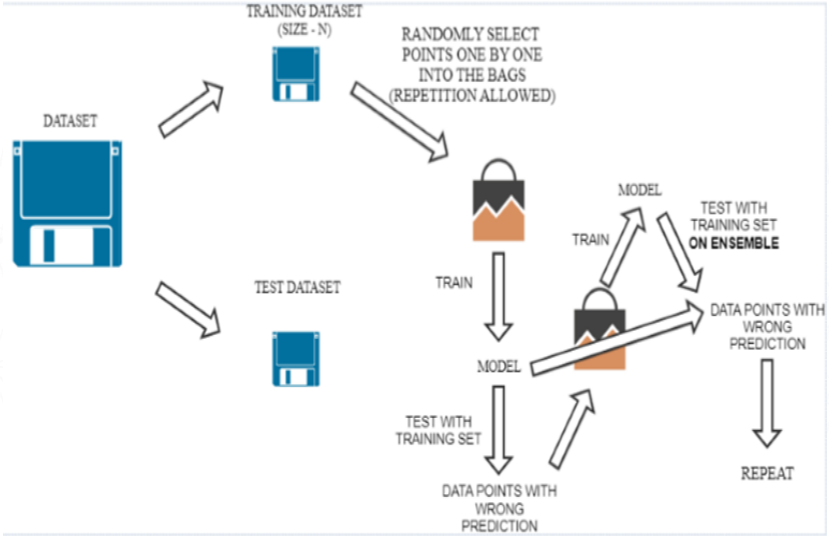
Bagging Algorithms:
a. Bagged Decision Trees
b. Random Forest
c. Extra Trees



Ensemble Techniques – Cont.

2. Boosting

Boosting is used to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors. Consecutive trees (random sample) are fit and at every step, the goal is to improve the accuracy from the prior tree. When an input is misclassified by a hypothesis, its weight is increased so that the next hypothesis is more likely to classify it correctly. This process converts weak learners into better performing model.



Ensemble Techniques – Cont.

Bagging:

1. Bagged Decision Trees
   All features are used for splitting a node.
2. Random Forest
   Subset of features are used at random out of the total and the best split feature from the subset is used to split each node in a tree,
3. Extra Trees

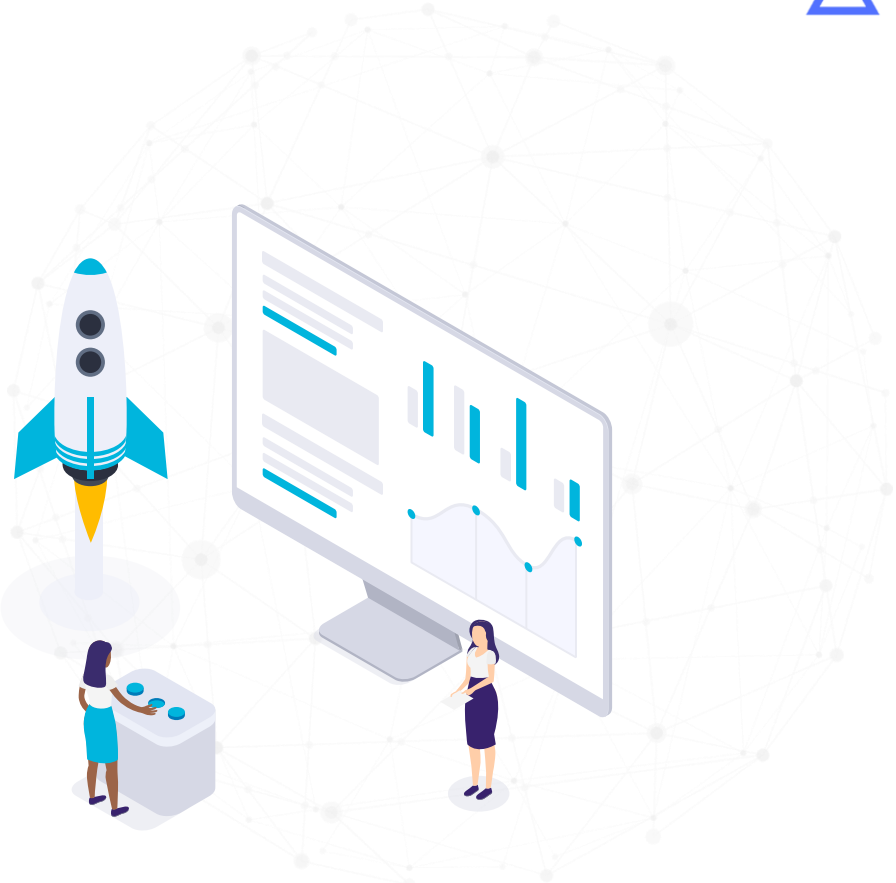
Boosting

1. AdaBoost
2. Stochastic Gradient Boosting

Time Series Analysis

Time Series Analysis

- Time series forecasting is an important area of machine learning that is often neglected.
- It is important because there are so many prediction problems that involve a time component and these problems are often neglected because it is this time component that makes time series problems more difficult to handle.
- Before getting started with Time Series Analysis, let's get our basics clear on Anomaly Detection



What is Time Series?

It is a series of observations taken at specified times basically at equal intervals. It is used to predict future values based on past observed values.

Components of Time Series

Trend

Increasing or Decreasing value in the series

Seasonality

A general systematic linear or (most often) non-linear component that changes over time and does repeat.

Irregularity

The data in the time series follows a temporal sequence, but the measurements might not happen at a regular time interval.

Cyclic

Pattern exists when data exhibit rises & falls that are not of fixed period.

Testing TS Stationarity

There are many methods to check whether a time series is stationary or non-stationary.

1. **Look at Plots:** You can review a time series plot of your data and visually check if there are any obvious trends or seasonality.
2. **Summary Statistics:** You can review the summary statistics for your data for seasons or random partitions and check for obvious or significant differences.
3. **Statistical Tests:** You can use statistical tests to check if the expectations of stationarity are met or have been violated.

You can split your time series into two (or more) partitions and compare the mean and variance of each group. If they differ and the difference is statistically significant, the time series is likely non-stationary.

Testing TS Stationarity

Statistical tests make strong assumptions about your data. They can only be used to inform the degree to which a null hypothesis can be rejected or fail to be reject. The result must be interpreted for a given problem to be meaningful.

Nevertheless, they can provide a quick check and confirmatory evidence that your time series is stationary or non-stationary.

ADF Test is otherwise known as **unit root test.**

Testing TS Stationarity

The null hypothesis of the test is that the time series can be represented by a unit root, that it is not stationary (has some time-dependent structure). The alternate hypothesis (rejecting the null hypothesis) is that the time series is stationary.

- **Null Hypothesis (H0):** If failed to be rejected, it suggests the time series has a unit root, meaning it is non-stationary. It has some time dependent structure.
- **Alternate Hypothesis (H1):** The null hypothesis is rejected; it suggests the time series does not have a unit root, meaning it is stationary. It does not have time-dependent structure.

We interpret this result using the p-value from the test. A p-value below a threshold (such as 5% or 1%) suggests we reject the null hypothesis (stationary), otherwise a p-value above the threshold suggests we fail to reject the null hypothesis (non-stationary).

- **p-value > 0.05:** Fail to reject the null hypothesis (H0), the data has a unit root and is non-stationary.
- **p-value <= 0.05:** Reject the null hypothesis (H0), the data does not have a unit root and is stationary.

Algorithms

ARIMA (AR, MA, ARMA, ARIMA)
Facebook Prophet
LSTMs
Holt's Winter Exponential Smoothing
GARCH
SARIMA/SARIMAX
VAR
VARMA etc. etc.

Quick Link: https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/

ARIMA

Auto regressive

Moving Average

Autoregressive Moving Average (No differencing)

Autoregressive Integrated Moving Average

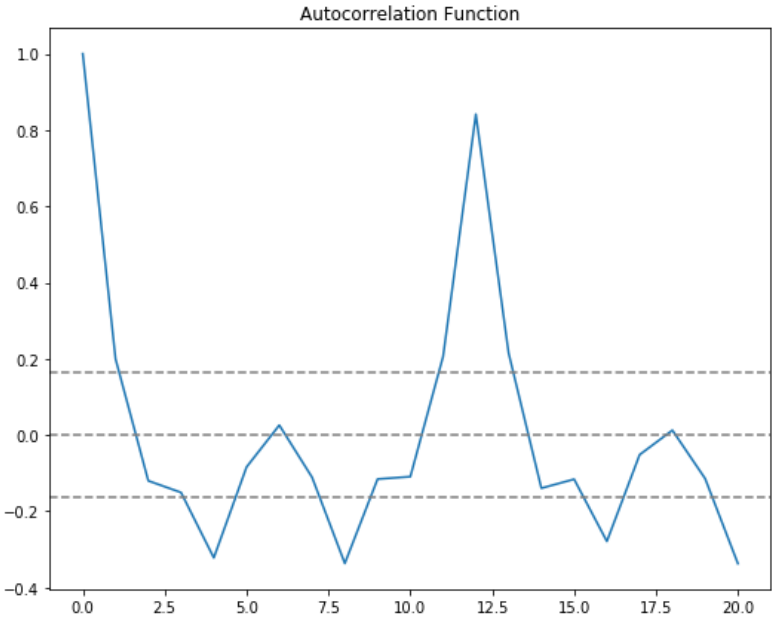
ARIMA

There are different techniques to find the right parameters for ARIMA(p,d,q)

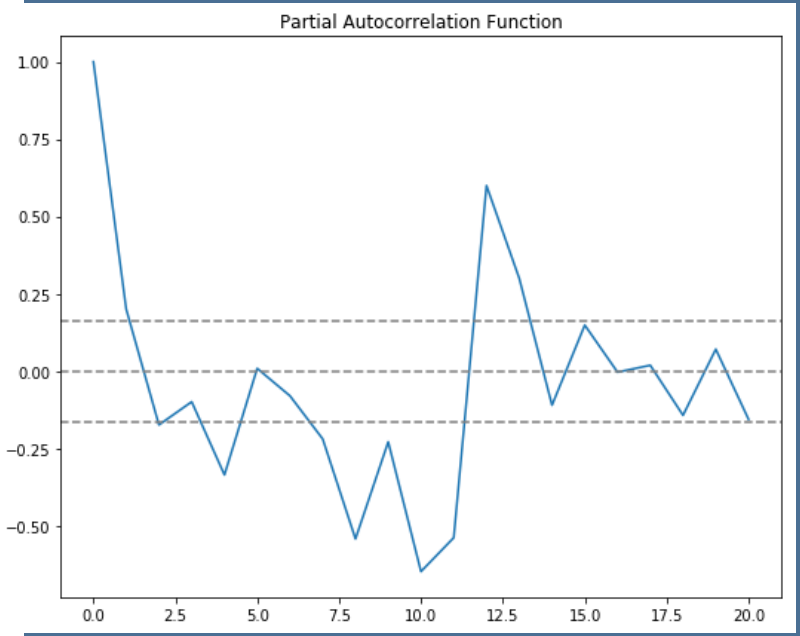
- ACF/PACF Plots
- Grid Search
- Auto Arima

Let's learn about these techniques in the next slides.

Auto Correlation



Partial Auto Correlation



1. **p** – The lag value where the **PACF** chart crosses the upper confidence interval for the first time. If you notice closely, in this case p=2.
2. **q** – The lag value where the **ACF** chart crosses the upper confidence interval for the first time. If you notice closely, in this case q=2.

Grid Search

ACF/PACF plots are some traditional methods of obtaining p & q values, and are sometimes misleading, hence we need to perform a hyper parameter optimization step in Time Series Analysis to get the optimum p,d & q values

Auto Arima

Grid Search techniques are manual ways, the same task can be achieved in few lines of coding and with a better efficiency using Auto Arima

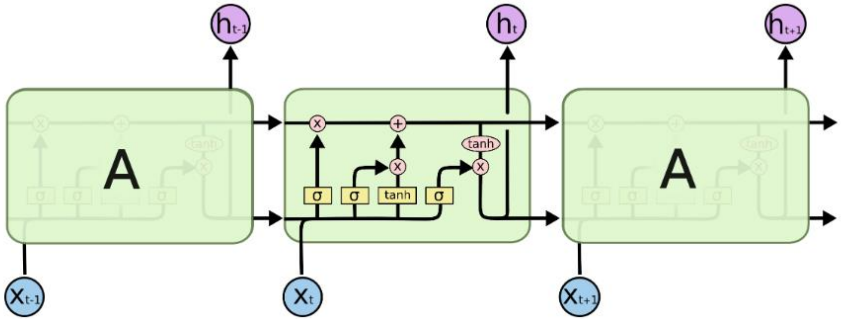
Facebook Prophet

Quick Link:
https://facebook.github.io/prophet/docs/quick_start.html#python-api

Features:

- Very fast
- An additive regression model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects
- Robust to missing data & shifts in trend, and handles outliers automatically.
- Easy procedure to tweak & adjust forecast while adding domain knowledge or business insights.

LSTMs



Quick Link:
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Features:

LSTM cell in place of standard neural network layers:

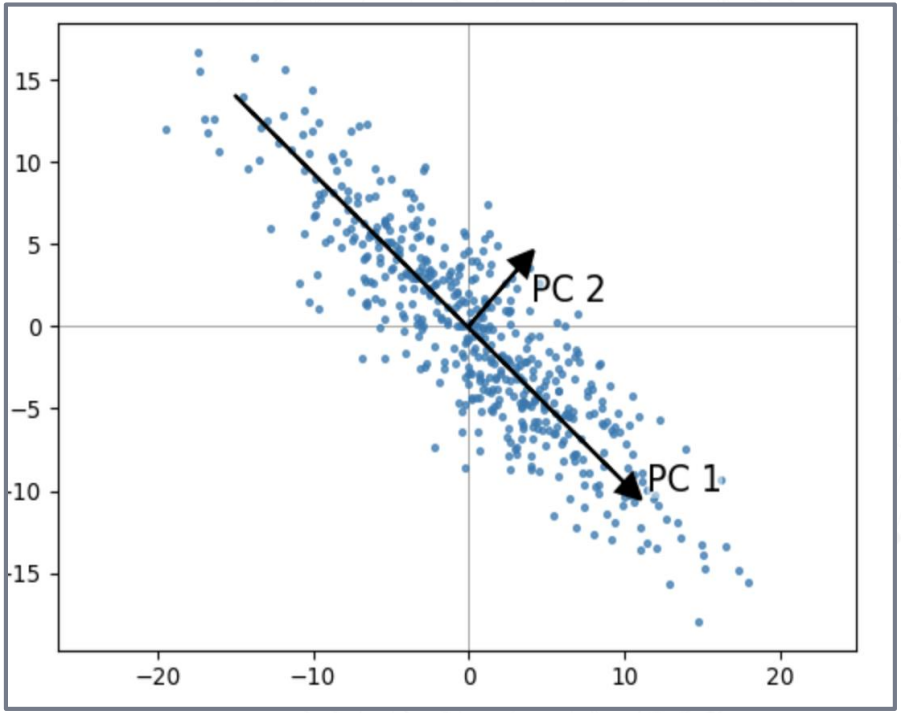
1. Input gate
2. Forget gate
3. Output gate

Dimensionality Reduction – Feature Engineering

Dimensionality Reduction

Two Techniques:

1. Feature Selection
    a. Wrapper methods
        i. Recursive feature elimination
        ii. Successive feature selection
    b. Filtering methods: IG, Chi Square, Correlation Coefficient
    c. Embedded methods: Decision Trees
2. Feature Extraction
    a. Principal Component Analysis
    b. Linear Discriminant Analysis
    c. Quadrant Discriminant Analysis
    d. Kernel PCA

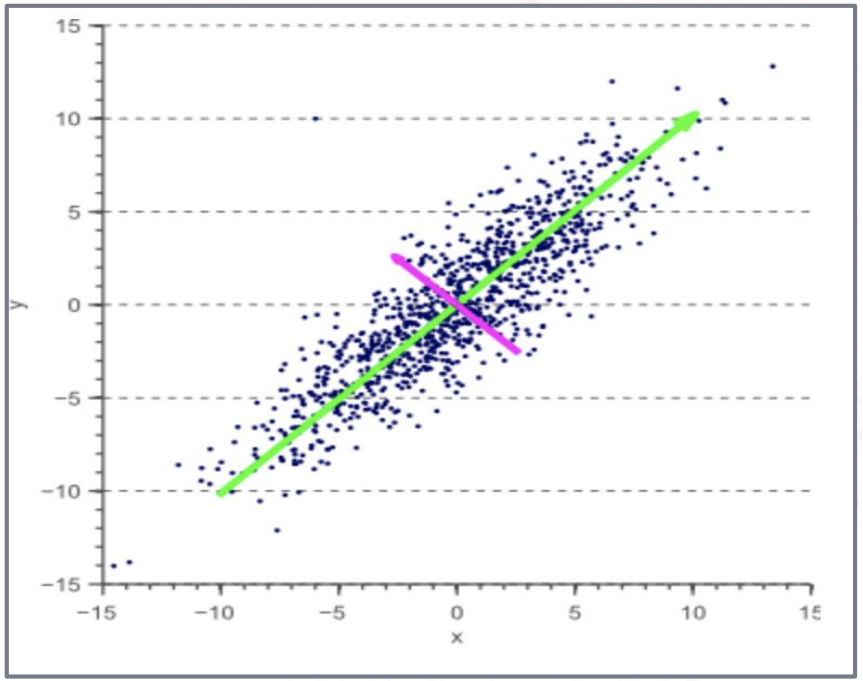


Dimensionality Reduction

Principal Component Analysis

- Principal Component Analysis (PCA) is basically a dimensionality reduction algorithm, but it can also be useful as a tool for visualization, noise filtering, feature extraction & engineering, and much more.

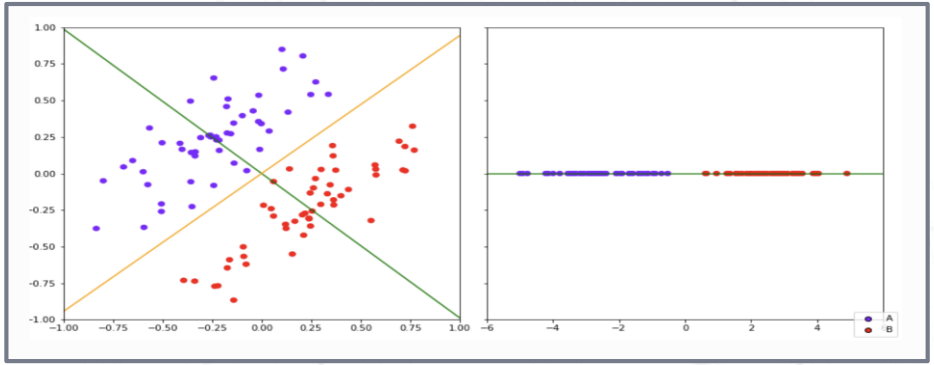
- PCA creates a visualization of data that minimizes residual variance in the least squares sense and maximizes the variance of the projection coordinates

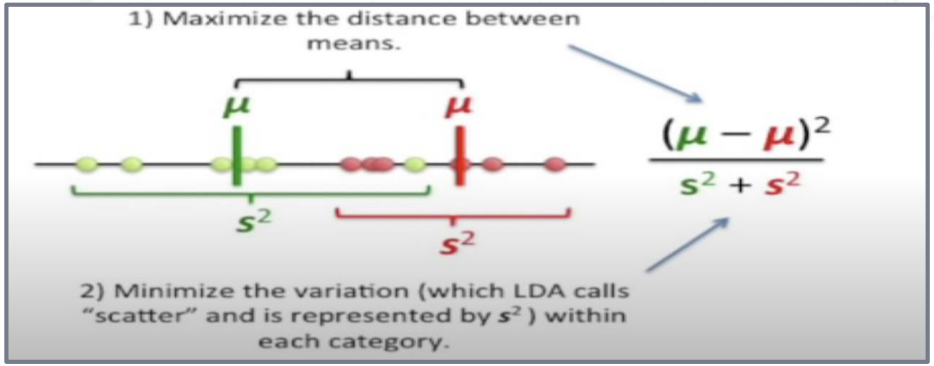


Dimensionality Reduction

Linear Discriminant Analysis

- LDA is like PCA but it focuses on maximizing separatibility among known categories
- LDA is a supervised algorithm
- Maximising distance between classes and minimising spread/scatter within each category.
- LDA has substantially lower variance. This can potentially lead to improved prediction performance.

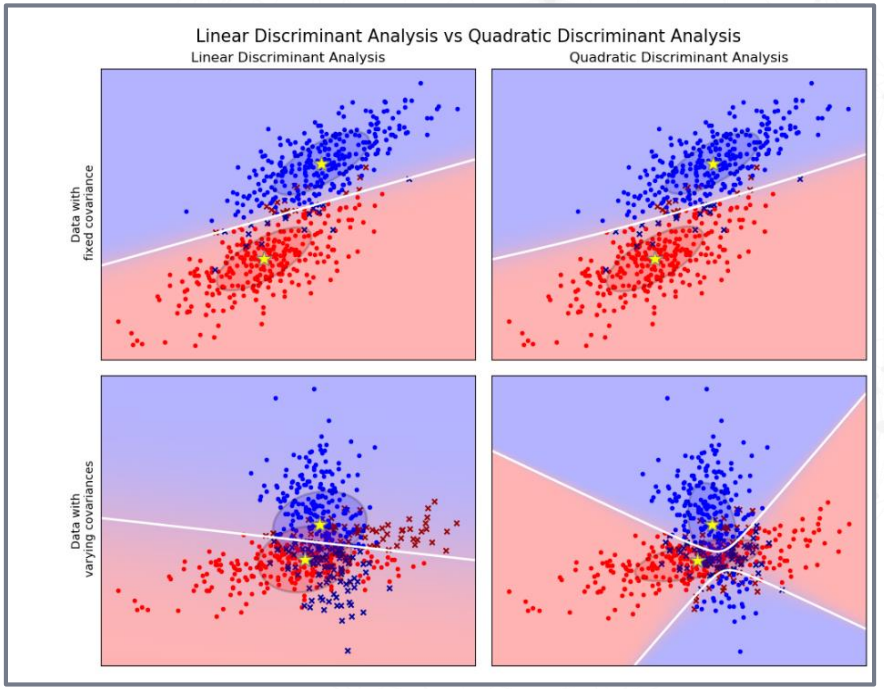




Dimensionality Reduction

Quadratic Discriminant Analysis

- QDA (Quadratic Discriminant Analysis) is used to find a non-linear boundary between classifiers.
- The more the classes are separable and the more the distribution is normal, the better will be the classification result for LDA and QDA.



Hyperparameter Optimization

Hyper Parameter Optimization

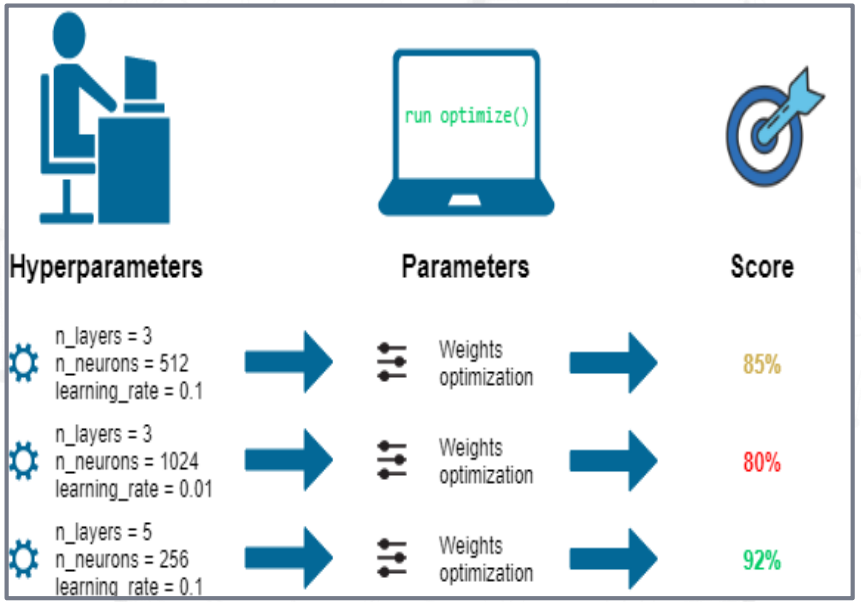
Machine Learning models are composed of two different types of parameters:

- **Hyperparameters** = are all the parameters which can be arbitrarily set by the user before starting training (eg. number of estimators in Random Forest).
- **Model parameters** = are instead learned during the model training (eg. weights in Neural Networks, Linear Regression).

The model parameters define how to use input data to get the desired output and are learned at training time. Instead, Hyperparameters determine how our model is structured in the first place.

Hyper Parameter Optimization

Machine Learning models tuning is a type of optimization problem. We have a set of hyperparameters and we aim to find the right combination of their values which can help us to find either the minimum (eg. loss) or the maximum (eg. accuracy) of a function
This can be particularly important when comparing how different Machine Learning models performs on a dataset. In fact, it would be unfair for example to compare an SVM model with the best Hyperparameters against a Random Forest model which has not been optimized.



Hyper Parameter Optimization

The aim of hyperparameter optimization in machine learning is to find the hyperparameters of a given machine learning algorithm that return the best performance as measured on a validation set.

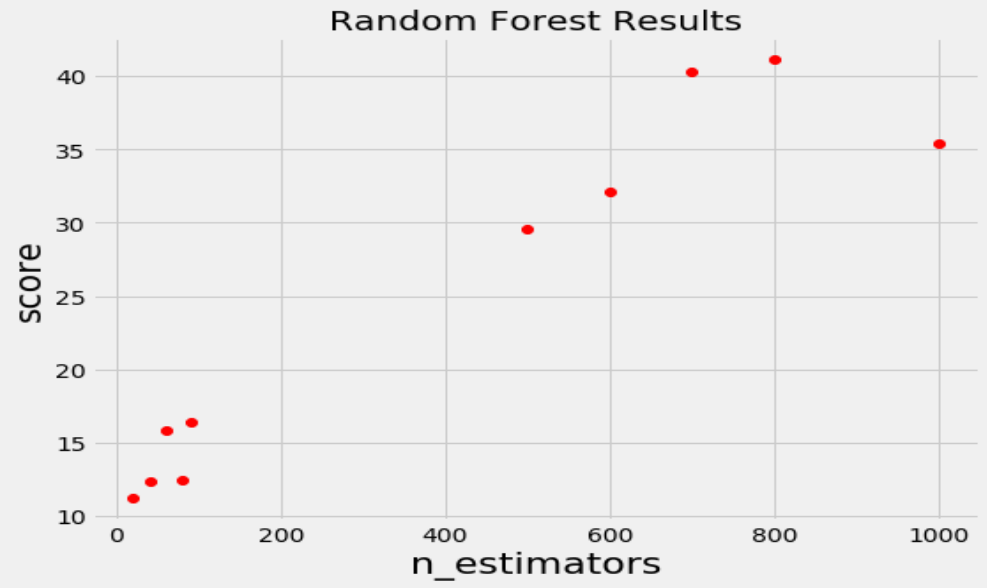
f(x) - an objective score to minimize— such as RMSE or error rate— evaluated on the validation set;
x* - is the set of hyperparameters that yields the lowest value of the score
x - can take on any value in the domain X.
In simple terms, we want to **find the model hyperparameters that yield the best score on the validation set metric.**

$$x^\star = \arg\min_{x \in \mathcal{X}} f(x)$$

Hyper Parameter Optimization

Following are four common methods of hyperparameter optimization for machine learning in order of increasing efficiency:

- Grid search
- Manual
- Random search
- Bayesian model-based optimization



.

Hyper Parameter Optimization

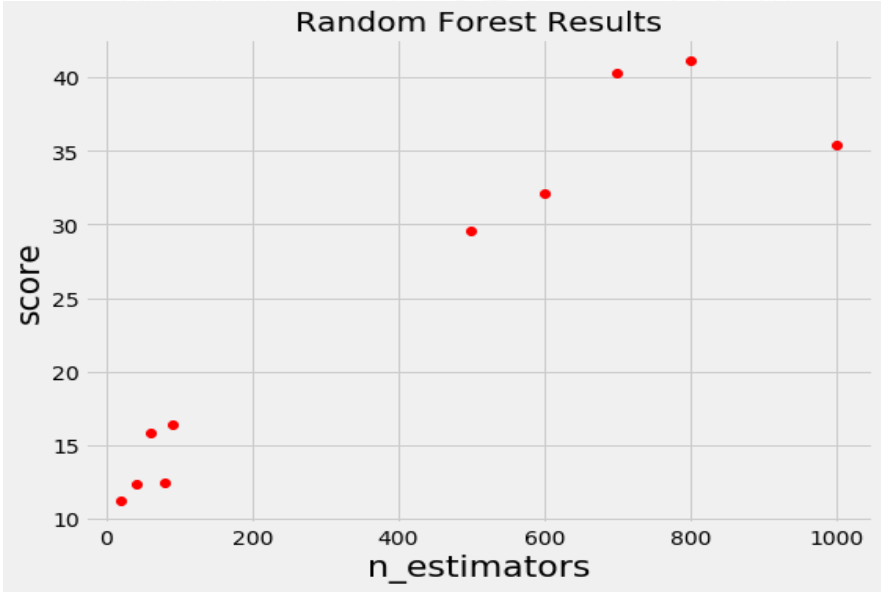
Each time we try different hyperparameters, we have to train a model on the training data, make predictions on the validation data, and then calculate the validation metric. With a large number of hyperparameters and complex models such as ensembles or deep neural networks that can take days to train, this process quickly becomes intractable to do by hand!Grid search and random search are slightly better than manual tuning because we set up a grid of model hyperparameters and run the train-predict -evaluate cycle automatically in a loop while we do more productive things .

Grid and random search are completely ***uninformed*** **by past evaluations,** and as a result, often spend a significant amount of time evaluating “bad” hyperparameters.

Hyper Parameter Optimization

For example, if we have the following graph with a lower score being better, where does it make sense to concentrate our search?
If you said below 200 estimators, then you already have the idea of Bayesian optimization! We want to focus on the most promising hyperparameters, and if we have a record of evaluations, then it makes sense to use this information for our next choice

Random and grid search pay no attention to past results at all and would keep searching across the entire range of the number of estimators even though it's clear the optimal answer (probably) lies in a small region!



Random Forest Hyper Parameter

max_depth → Longest path between root node and leaf node

min_sample_split → parameter that tells the decision tree in a random forest the minimum required no. of observations in any given node in order to split it. Default value is 2.

max_terminal_nodes → Condition on the splitting the nodes in tree, and restricts growth of tree

n_estimators → Number of trees should we consider

max_samples → what fraction of original dataset is given to any individual tree. (Bootstrap sample fraction)

max_features → How many features to use in each tree in a RF.

Thank you