SASURIE INSTITUTIONS

COIMBATORE | TIRUPPUR | VIJAYAMANGALAM

SASURIE COLLEGE OF ENGINEERING

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

SUBJECT: MACHINE LEARNING

AL 3451 - Machine Learning

UNIT - I

I. Introduction to Machine Learning.

1. What is Machine Learning ?

Machine Learning is a part of Artificial Intelligence which combine data with statistical tools to predict an output which can be used to make actionable insights.

It is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer.

Def: Machine learning is the study of algorithm that

- improve their performance P
- at some task T
- with experience E,

A well defined learning task is given by <P, T, E>

A typical machine learning tasks are to provide a recommendation. for example, for those who have a Netflix account, all recommendations of movies / series are based on the user's historical data.

2. Machine Learning Vs Traditional Programming:

In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. when the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

$$\begin{tikzcd}
& \text{Rules} \arrow[d, Rightarrow] & \\
\text{Data} \arrow[r, Rightarrow] & \boxed{\begin{array}{c}\text{Computer}\\ \text{(Traditional}\\ \text{programming)}\end{array}} \arrow[r, Rightarrow] & \text{output}
\end{tikzcd}$$

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experience to improve efficiency over time.

$$\begin{tikzcd}
\text{Data} \arrow[dr, Rightarrow] & & \\
& \boxed{\begin{array}{c}\text{Computer}\\ \text{(Machine Learning)}\end{array}} \arrow[r, Rightarrow] & \text{Rules} \\
\text{Output} \arrow[ur, Rightarrow] & &
\end{tikzcd}$$

3. How does Machine Learning work?

The machine learning process starts with inputting training data into the selected Algorithm. Training data being known or unknown data to develop the final Machine learning algorithm. The type of training data input does impact the algorithm, and that concept will be covered further momentarily.

New input data is fed into the machine learning algorithm to test whether the algorithm works correctly. The prediction and results are then checked against each other.

If the prediction and results don't match, the algorithm is re-trained multiple times until the data scientist gets the desired outcome.

This enables the machine learning algorithm to continuously learn on its own and produce the optimal answer, gradually increasing in accuracy over time.

The life of machine learning program can be summarized in the following:

1. Define a question
2. Collect data
3. Visualize data
4. Training algorithm
5. Test the Algorithm
6. Collect feedback
7. Re-train the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction.

4. When do we use Machine Learning?

ML is used when,

- Human expertise doesnot exist (Navigating on Mars)
- Human can't explain their expertise (Speech recognition, vision)
- Models must be customized. (personalized medicine)
- Models are based on huge amounts of data (Genomics)

Learning is not always useful,

- There is no need to learn to calculate payroll.

5. Application of Machine Learning:

Augmentation: Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as virtual Assistant, Data analysis, Software solution.

Automation: Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Health Industry: Healthcare was one of the first industry to use machine learning with image detection.

Marketing: With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

Supply chain: Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.

6. Why is machine learning Important?

* The machine learning can take decision with minimal human intervention.
* It gives enterprises a view of trends in customer behavior and business operational pattern, as well as suppots the development of new products.

7. Machine Learning Algorithms:

The three machine learning types are

(i) Supervised learning

(ii) Unsupervised learning

(iii) Reinforcement learning

The below diagram illustrates the different ML algorithm with the categories:

Machine Learning

- Supervised
  - Regression
    - Linear
    - Polynomial
  - Decision Tree
  - Random forest
  - classification
    - KNN
    - Trees
    - Logistic Regression
    - Naive - Bayes
    - SVM
- Un Supervised
  - clustering
    - SVD
    - PCA
    - k-means
  - Association analysis
    - Apriori
    - FP-growth
  - Hidden Marker Model
- Reinforcement
  - Game AI
  - Robot navigation
  - Skill acquisition

II. Supervised Learning Vs Unsupervised Learning Vs Reinforcement Learning.

1. Supervised learning :

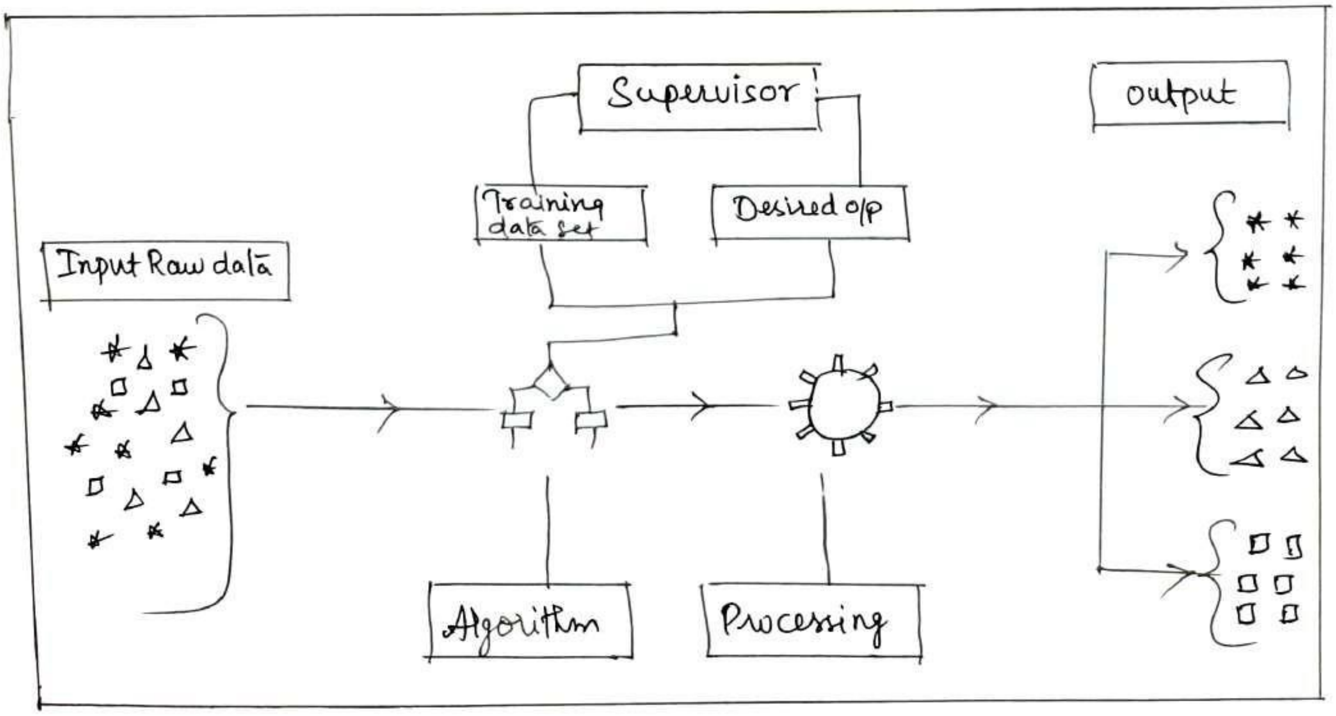
what is it ?

with input provided as a labeled dataset, a model can learn from it. Labeled dataset means, for each data set given, an answer or solution to it is given as well. This would help the model in learning and hence provide the result of the problem easily

Example :

A labeled dataset of animal images would tell the model whether an image is of a dog, a cat, etc. Using which, a model gets training, and so, whenever a new image comes up to the model, it can compare that image with the labeled dataset for predicting the correct label.

Supervised Learning.



Types of Problems:

There are two types of problems.

(i) Classification Problems:

Ask the algorithm to predict a discrete value that can identify the input data as a member of a particular class or group. Taking up the animal photos dataset, each photo has been labeled as a dog, a cat, etc., and then algorithm has to classify the new image into any of these labeled categories.

(ii) Regression Problems:

These are responsible for continuous data, e.g., for predicting the price of a piece of land in a city, given area, location etc., Here the input is sent to the machine for predicting the price according to previous instances. And the machine determines a function that would map the pairs. If it is unable to provide accurate results, backward propagation is used to repeat the whole function until it receives satisfactory results.

2. Unsupervised Learning:

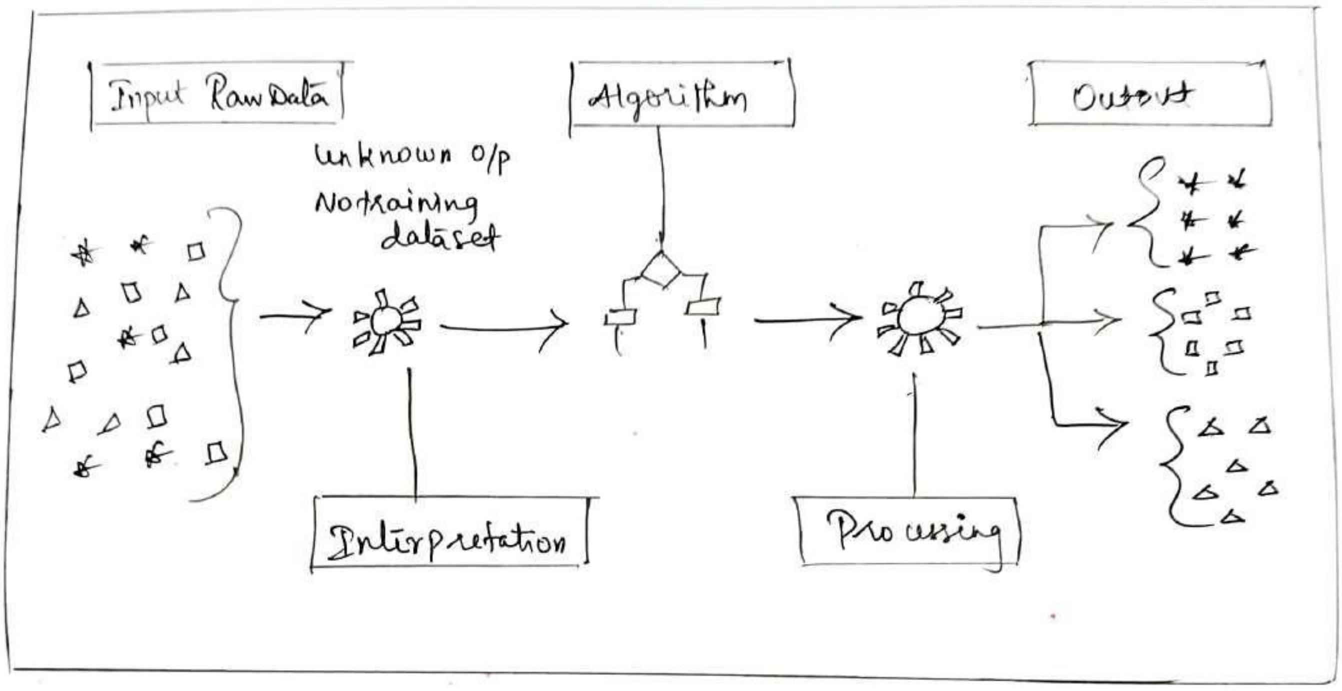
What is it?

Unsupervised is a type of self-organized learning that helps find previously unknown pattern in data sets without pre-existing labels.

The major difference between supervised and unsupervised learning is that there is no complete and clean labeled data set in unsupervised learning.

Here, a model receives a data set without providing any instructions. Also, we don't know what you need to get from the model as an output yet.

Unsupervised Learning



Example!

Consider the animal photo example used in supervised learning. Suppose there is no dabeled data set provided. Then how can the model find out if an animal is a cat or a dog or a bird?

If the model has been provided some information such as if an animal has feathers, a beak, wings, etc, it is a bird. In the same way, If an animal has fluffy fur, floppy ears, a curly tail, and maybe some spots, it is a dog and so on.

Hence, according to the information, the model can distinuguish the animals successfully.

Difference between Supervised and Unsupervised Learning:

Criteria	Supervised Learning	Unsupervised Learning
Method	Input and output variables are given	Only the input data is given
Analysis of Data	Uses offline analysis	Uses real-time Analysis
Goal	The o/p is predicted using the labeled i/p dataset	The o/p is predicted based on the patterns in the input dataset.

3. Reinforcement Learning:

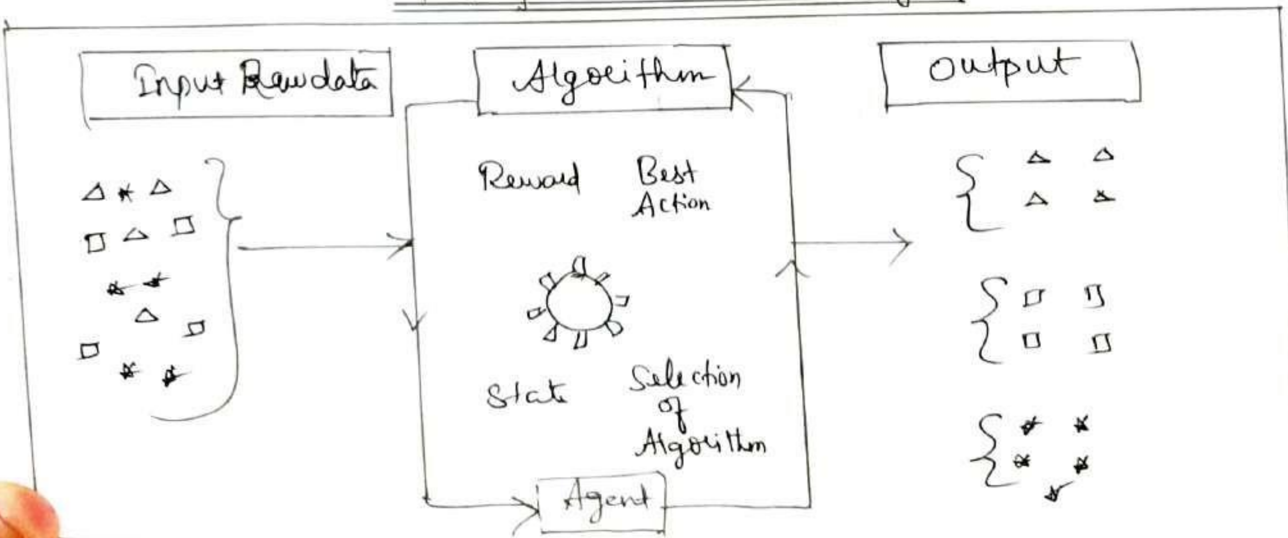
It is a type of learning that is based on interaction with the environment.

To begin with, there is always a start and an end state for an agent (the AI-driven system); however, there might be different paths for reaching the end state, like a maze. This is the scenario wherein reinforcement learning is able to find a solution for a problem.

Examples:

Self-navigating vaccum cleaners, driverless cars etc.

Reinforcement Learning



Differences between supervised, Unsupervised and reinforcement learning:

Criteria	Supervised Learning	Unsupervised Learning	Reinforcement Learning
Definition	The machine learns by using labeled data	The machine is trained on unlabeled data without any guidance	An agent interacts with its environment by performing actions & learning from errors or rewards
Type of problems	Regression & classification	Association & clustering	Reward - based
Type of data	Labeled data	Un labeled data	No predefined data
Training	External supervision	No supervision	No supervision
Approach	Maps the labeled inputs to the known outputs	Understanding patterns and discovering the output	follows the trial-and-error method.

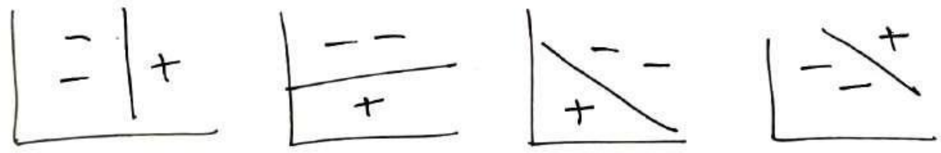
III. Vapnik - Chervonenkis (VC) dimension.

VC dimensions are used to quantify how powerful is the model. In a real-world, apply one by one model on given data set and find the accuracy of each model. The model gives the highest accuracy that is powerful model. But when comes to statistical. Machine learning VC dimensions used to find which model is powerful.

VC dimension of a model = Maximum no. of points that can be separated by a model for all possible configuration.

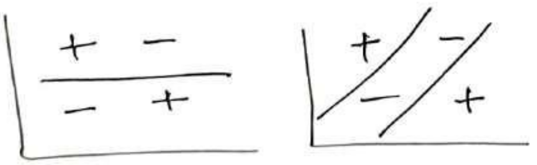
Example:-

3 points Linear models:



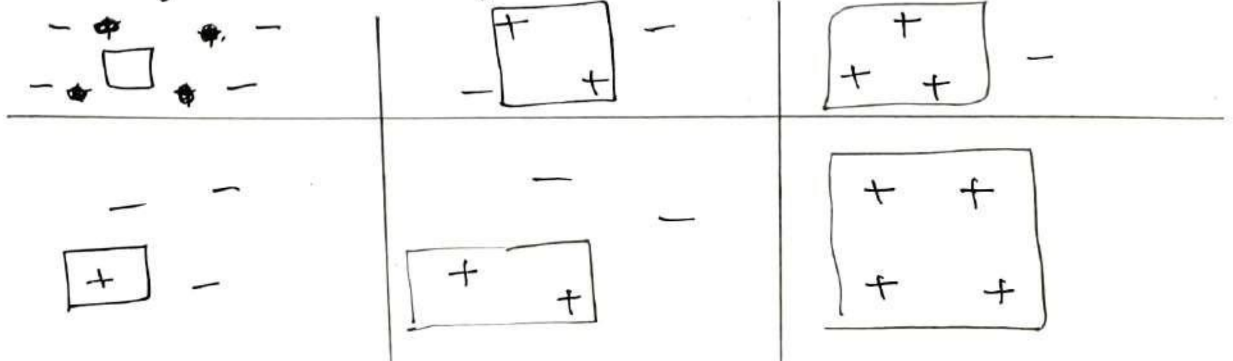
Three points are like any way we always have a possible way to separate them. (Classify them).

4 points Linear models:



} Not able to separate all possible configurations.

Proving that rectangle concept space shatters at least 4.



It can be seen that a straight line can shatter 3 points but it can not shatter 4 points. Thus VC dimension of model straight line in 2D plane is 3.

The VC dimension of a model is d if there exists some sample $|S| = d$ which can be shattered by the model. This does not mean that all samples of size d are shattered by the model.

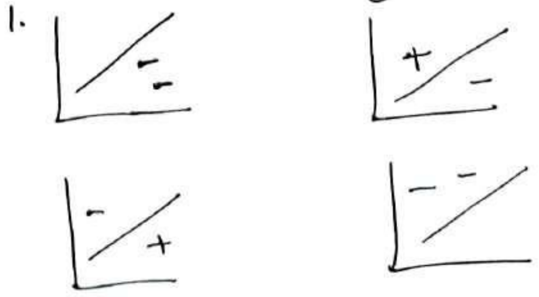
1. Let us consider a simple binary classification model, which states that for all points (a, b) such that $a < x < b$, label them as 1, otherwise label them as 0. Find VC dimension.

$h(x) = 1$, if $a < x < b$
$h(x) = 0$, otherwise.

$(a, b) \in \mathbb{R}^2$

List of 2^2 distinct labels in binary classification.

1. $h(m) = 0$ $h(n) = 0$
2. $h(m) = 0$ $h(n) = 1$
3. $h(m) = 1$ $h(n) = 0$
4. $h(m) = 1$ $h(n) = 1$



Our model successfully shattered with 2 points in the data set.

IV. Probably Approximatily Correct (PAC) learning

Probably approximate correct (PAC) learning is a theoretical framework for analyzing the generalization error of a learning algorithm in terms of its error on a training set and measure of complexity. The goal is typically to show that an algorithm achieves low generalization with high probability.

PAC learning requires,

1. with probability at least $(1-\delta)$, where δ gives the probability of failure.

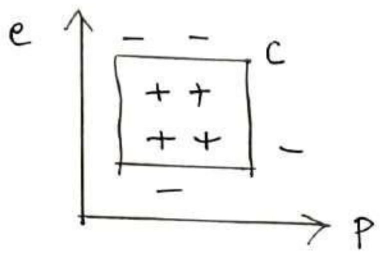
2. with accuracy at most $(1-\varepsilon)$, where ε is upper bound on the error.

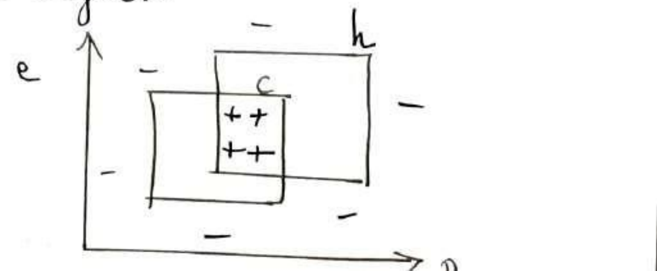
For example,

Consider the problem of N number of car having price P, and engine power e, as training set (P, e) and find the car is family car or not.

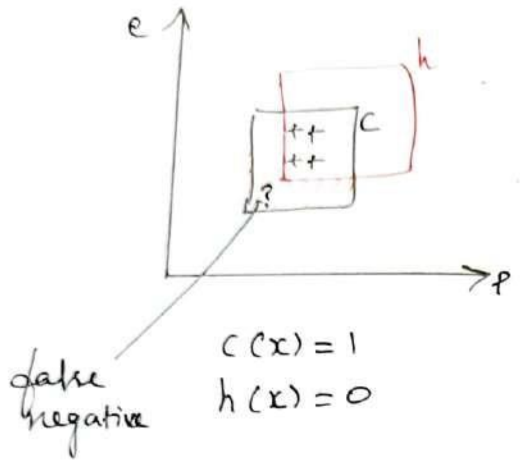
An algorithm gives answer whether the car is family car or not.

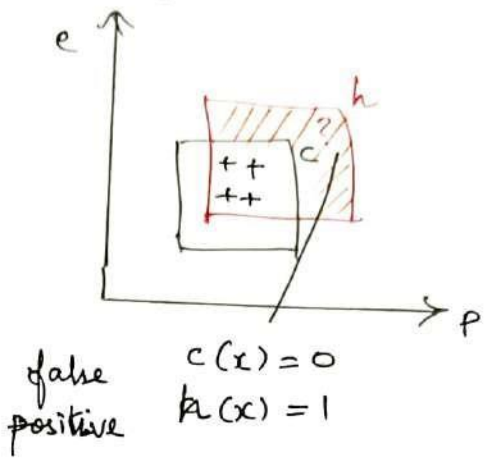
Instances within rectangle C represents family cars and outside are not family cars, and hypothesis h is closely approximate to C with error region.





false Negative and false Positive:





Instances lies on shaded region are positive/negative according to the actual function c, but those are negative/positive based on the hypothesis h. Hence it is called as false negative or false positive.

False negative — Negative example is classified as +ve.
False positive — Positive example is classified as −ve.

Error region:

The error region can be identified with P (C XOR h). Always it should be small.

$$\boxed{\text{Error region} : P(C \text{ XOR } h) <= \varepsilon .}$$

Probably Approximately correct:

The hypothesis h, that approximately correct and error is less than or equal to ε

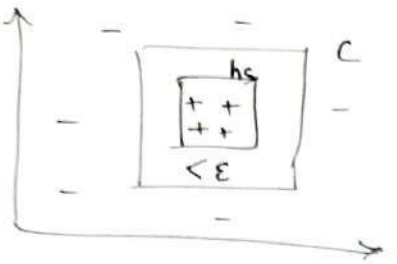
$$P(c \oplus h) <= \varepsilon$$

∴ PAC can be defined by,

$$P((\text{error}(h) <= \varepsilon)) <= 1-\delta$$

$$\text{(i)} \; P((P(c \oplus h) <= \varepsilon)) <= 1-\delta$$

PAC learnability for axis-aligned rectangle:



where h_s is the tightest possible rectangle around a set of positive training example.

$h_s \subseteq C$, Hence

Error region = C − h.

(X.) If generated hypothesis does not touch any of these error region, error region is greater than ε and not approximately correct.

Example-1: Hypothesis h_1, h_2 generated the errors with respect to the price and engine power of given 10 samples with $\varepsilon = 0.05$ and $\delta = 0.2$

Instance	Error(h_1)	Error(h_2)
1	0.001	0.001
2	0.025	0.025
3	$\boxed{0.07}$	$\boxed{0.071}$
4	0.003	$\boxed{0.063}$
5	0.035	0.035
6	0.045	0.045
7	0.027	0.027
8	$\boxed{0.065}$	$\boxed{0.086}$
9	0.012	0.012
10	0.036	0.036

Solution: ①

1. 3rd and 8th values are greater than ε.

$$P(h_1) = \frac{8}{10} = 0.8$$

2. $\delta = 0.2$

$1 - \delta = 0.8$

Hence h_1 is probably approximately correct.

Solution: ②

1. 3rd, 4th and 8th values are greater than ε.

$$P(h_2) = \frac{7}{10} = 0.7$$

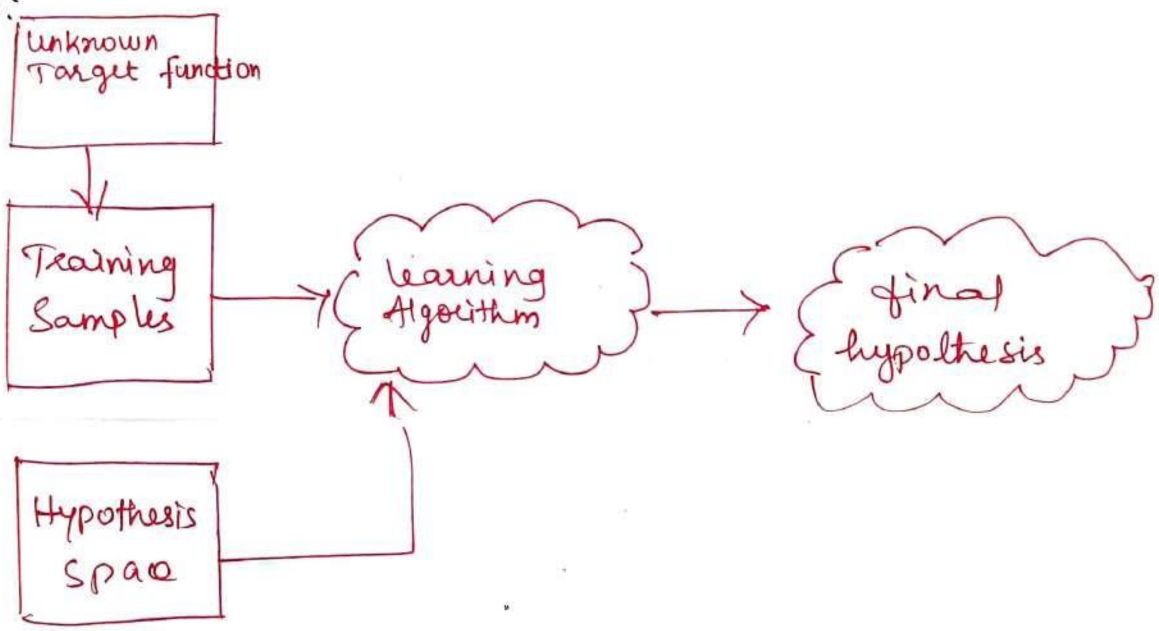
2. $\delta = 0.2$
$1 - \delta = 0.8$

Hence h_2 is not probably approximately correct.

V. Hypothesis Space

In most supervised machine learning algorithm, our main goal is to find out a possible hypothesis from the hypothesis space that could possibly map out the inputs to the proper outputs.

The following figure shows the common method to find out the possible hypothesis from the hypothesis space:



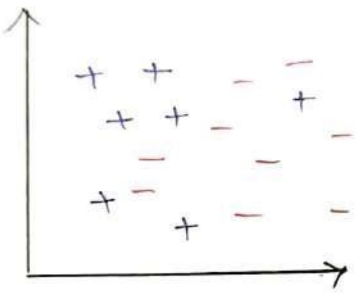
Hypothesis Space (H)

Hypothesis space is the set of all the possible legal hypothesis. This is the set from which the machine learning algorithm would determine the best possible (only one) which would best describe the target function or the outputs.

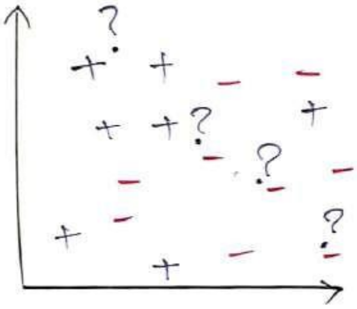
Hypothesis (h):

A hypothesis is a function that best describes the target in supervised machine learning. It would come up depends upon the data, depends upon the restrictions and bias.

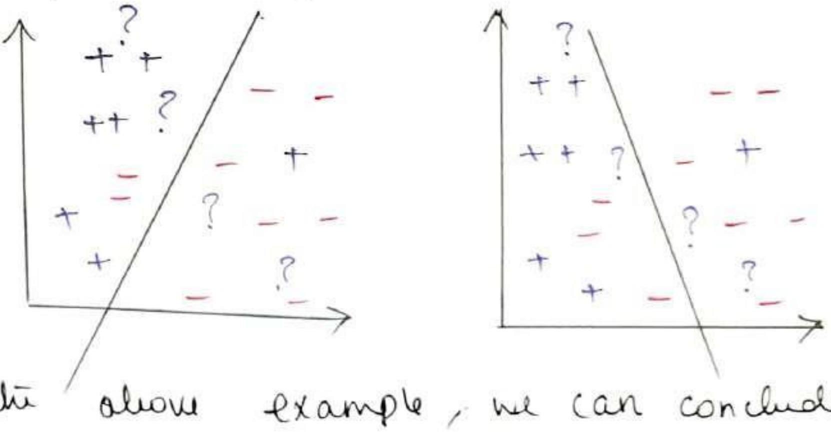
Example: Let us understand the hypothesis (h) and hypothesis space (H) with a two-dimensional coordinate plane showing the distribution of data as follows:



Now assume we have some test data by which ML algorithms predict the outputs for input as follows:



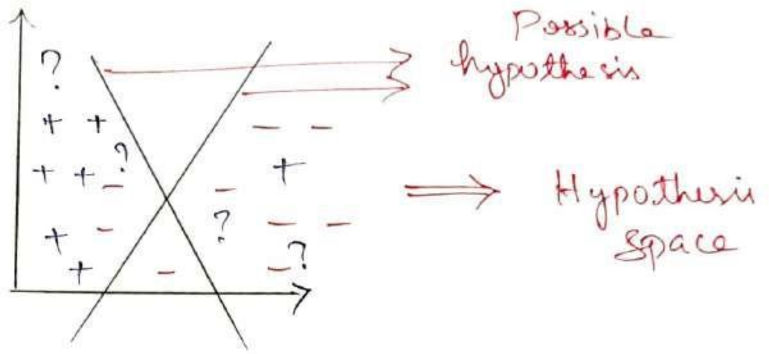
If we divide this coordinate plane in such as way that it can help you to predict output or result as follows: (Different ways)



with the above example, we can conclude that,

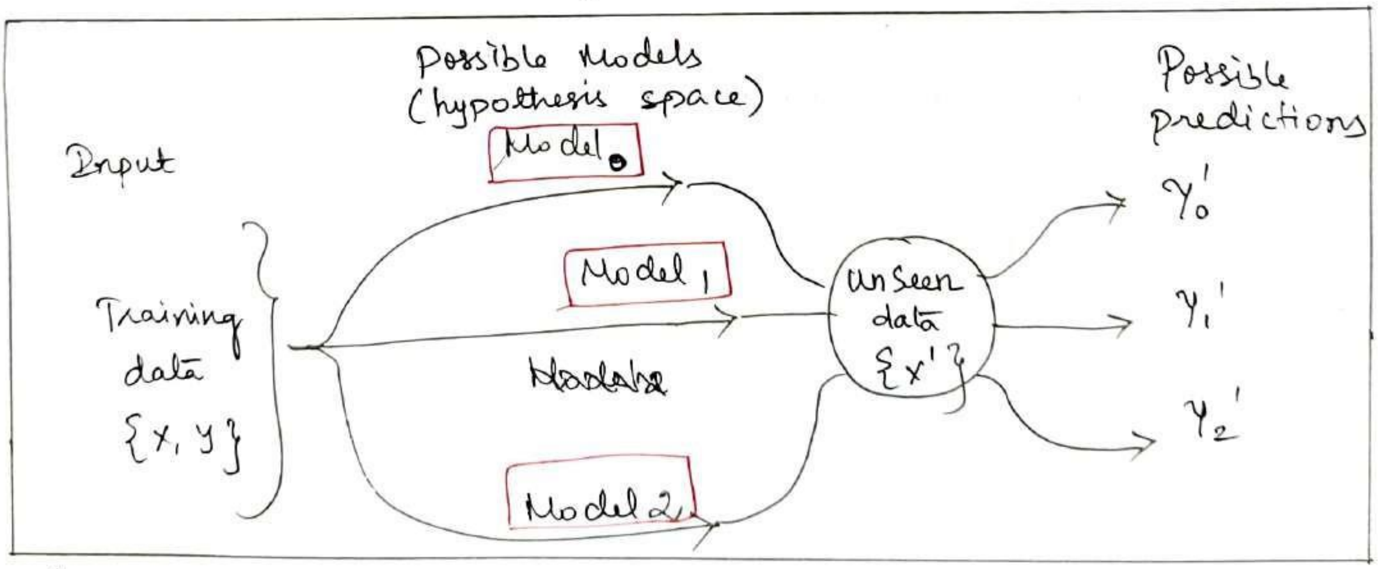
Hypothesis space (H) is the composition of all legal best possible ways to divide the coordinate plane so that it best maps input to proper output.

further, each individual best possible way is called a hypothesis (h), hence the hypothesis and hypothesis space would be like this:



VI. Inductive Bias

A learning algorithm's inductive bias, (learning bias) is a collection of preassumptions used by the learner to forecast outcomes of given inputs that it has never seen before.



Different models can be trained based on fixed training data. All those models will behave differently for new unseen data.

Examples of Inductive Bias in ML:

The phrase "Preferring one answer over another after viewing certain instances" sums up inductive bias. Every model has a bias of its own. A few examples of inductive bias are listed below:

- The linear model presupposes that each of the input characteristics and the target have a linear connection.
- Decision trees internalize in their nodes constant models.
- The layer-based structure of a convolutional neural network imposes a bias toward hierarchical processing.
- Bayesian modeling: The priors selected in this case greatly reveal the bias (which tells the model what happens when not much data is available)
- In linear regression, the model assumes that the relationship between the output, or dependent variable, and the independent variable is linear. The model has an inductive bias in this regard.

Importance of Bias in ML:

We know that unknown circumstances of input might results in any output value. This issue cannot be resolved without any further pre assumptions.

Occam's razor, which holds that the best hypothesis about the target function.

Occam's razor:

First razor: Given two models with the same generalization error, the simpler one should be preferred because simplicity is desirable in it self.

Second razor: Given two models with the same training - set error, the simpler on should be preferred because it is likely to have lower generalization error.

(∵) If two models have the same performance on the validation/ testing dataset select the simpler model because it is more likely to generalize well.

Types of Inductive Bias in ML:

Types of Inductive Bias in ML

1. Maximum Conditional independence
It aims to maximize conditional independence if the hypothesis can be framed within a Bayesian framework. The Naive Bayes classifier employs this

2. Minimum cross-validation error.
It picks the hypothesis with the lowest cross-validation error when trying to decide between them. Despite the fact that cross-validation may appear to be bias-free.

3. Maximum margin:
When dividing group of students, try to make the boundary as wide as possible

4. Minimum Description length
when formulating a hypothesis, make an effort to keep the description as brief as possible.

5. Minimum features:
Unless a feature is supported by a solid evidence, it should be removed.

6. Nearest neighbours:
In a small neighborhood in feature space, it is reasonable to assume that are close to one another typically belong to the same class.

VII. Generalization.

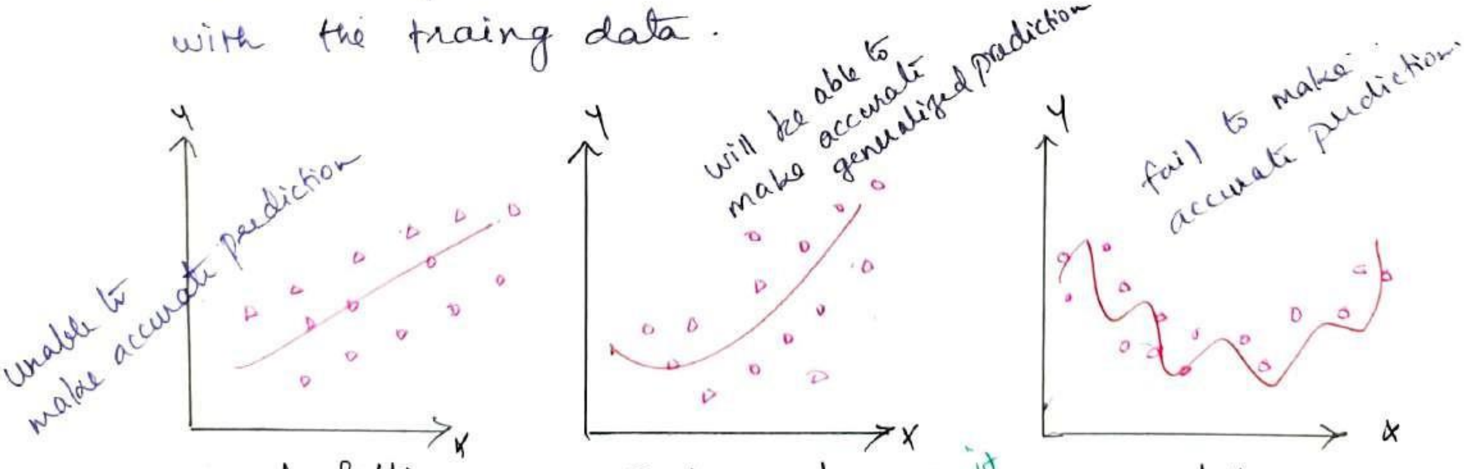
Generalization is a term used to describe a model's ability to react to new data.

After being trained on a training set, a model can digest new data and make accurate predictions.

A model's ability to generalize is central to the success of a model.

If model has been trained too well on training data, it will be unable to generalize. It will make inaccurate predictions when given new data, making the model useless even though it is able to make accurate predictions for the training data. This is called overfitting.

Underfitting happens when a model has not been trained enough on the data. In this case, it makes the model just as useless and it is not capable of making accurate predictions, even with the traing data.



underfitting

Some trend in the data, but it is not specific enough to capture relevant info.

Balanced

Accurately models it

Overfitting

learned well. It will fail to make accurate prediction.

VIII. Bias - Variance Tradeoff

There is a tradeoff between a model's ability to minimize bias and variance which is referred to as the best solution for selecting a value of regularization constant.

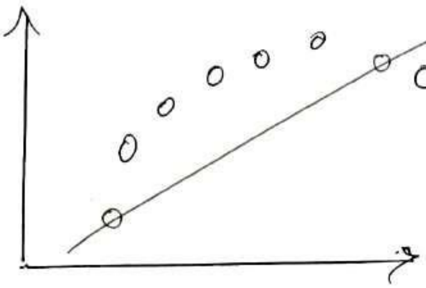
Proper understanding of these errors would help to avoid the overfitting and underfitting of a dataset while training the algorithm.

Bias:

The bias is known as the difference between the prediction of the values by the ML model and the correct value.

Being high in bias gives a large error in training as well as testing data. It is recommended that an algorithm should always be low biased to avoid the problem of underfitting.

By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data set. Such fitting is known as underfitting of Data. This happens when the hypothesis is too simple or linear in nature. Refer to the graph given below for an example of such a situation.



In such a problem, a hypothesis looks like,

$$\boxed{h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 ...)}$$

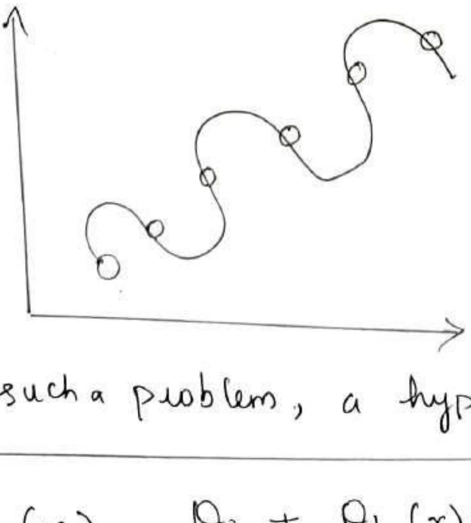
Variance:

The variability of model prediction for a given data point which tells us spread of our data is called the variance of the model.

The model with high variance:
- has a very complex fit to the training data
- is not able to fit accurately on the data which it hasn't seen before
- Perform very well on training data
- But has high error rate on test data.

When a model is high on variance, it is said to be overfitting of Data.

The high variance Data looks like,



In such a problem, a hypothesis looks like,

$$\boxed{h_\theta(x) = \theta_0 + \theta_1(x) + \theta_2(x^2) + \theta_3 x^3 + \cdots}$$

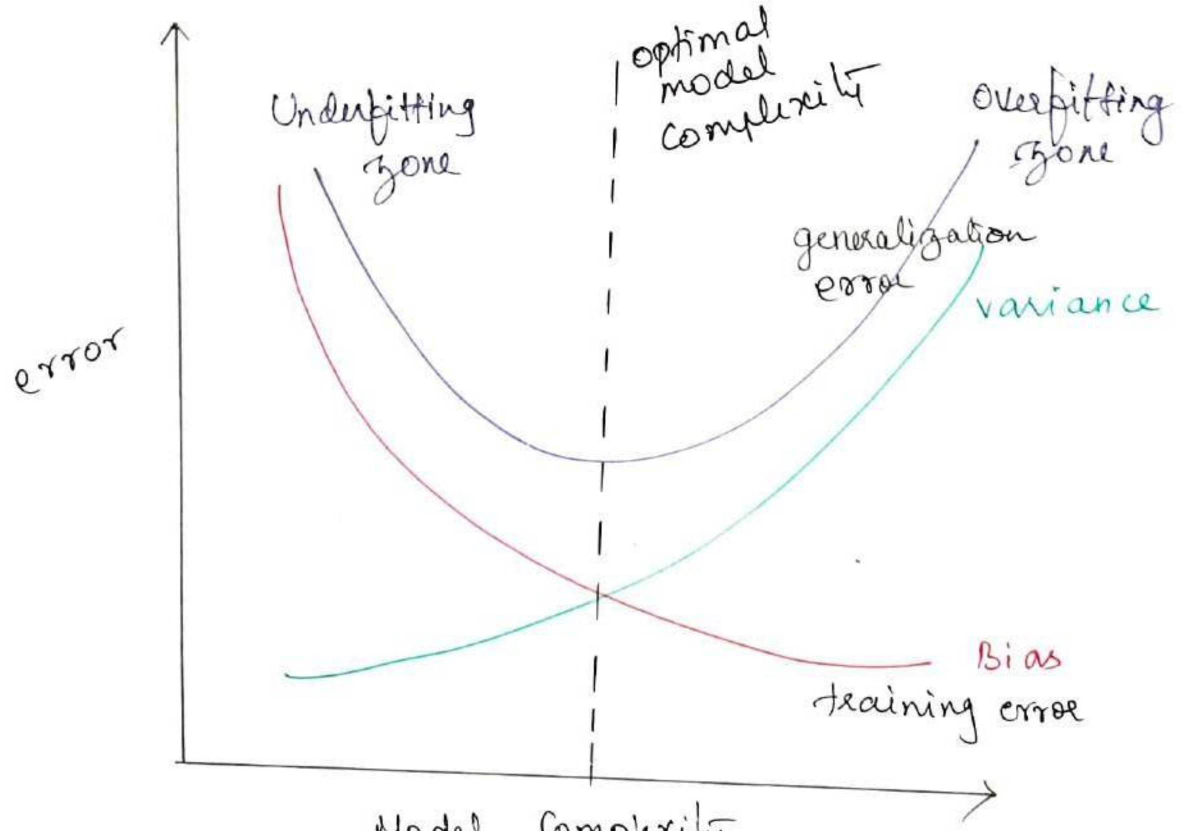
Bias Variance Tradeoff:

If the algorithm is too simple (hypothesis with linear equation) then it may be on high bias and low variance condition and thus is error-prone.

If the algorithm fit too Complex (hypothesis with high degree equation) then it may be on high variance and low bias and new entries will not perform well.

So, there is something between both of these conditions, known as Trade-off or Bias Varian Trade-off.

The perfect tradeoff will be like,



Ill - posed problems ?

A problem is ill - posed if it does not satisfy the 3 conditions of a well - posed problem:

1. Existence - There exists a solution
2. Uniqueness - Solution must be unique
3. Stability - solution depends continuously on initial conditions.

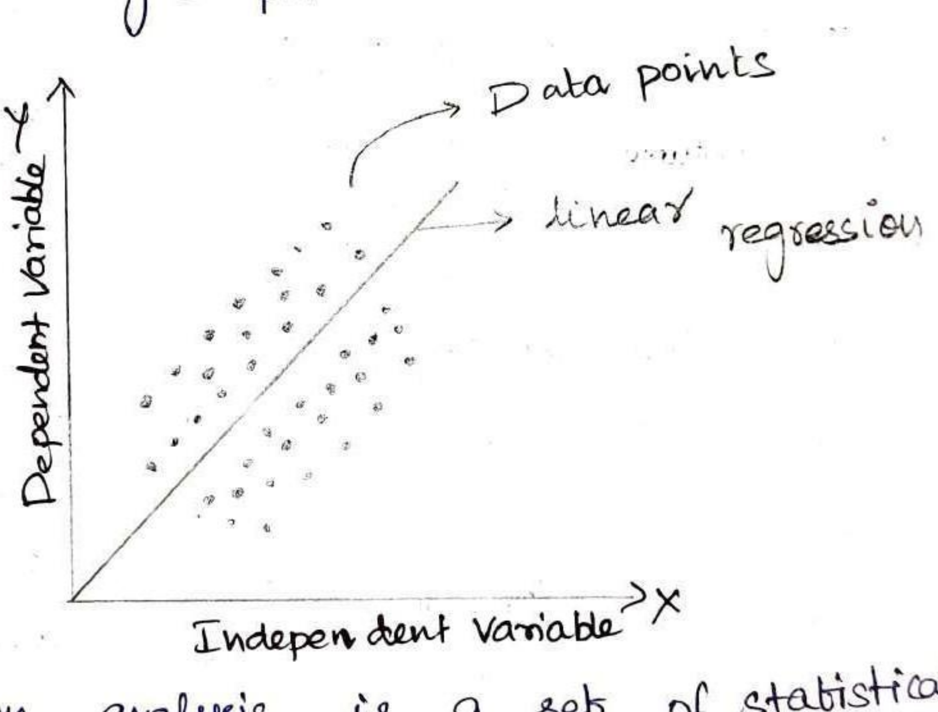
UNIT- II

SUPERVISED LEARNING

REGRESSION

Regression find correlations between dependent and independent Variables. If the desired output consists of one or more continous variable then the task is called as regression

Regression algorithm help predict continous variable such as house prices, market trends, weather patterns oil and gas prices ect.



Regression analysis is a set of statistical model or methods used for the estimation relationship between variables and for modelling the future relationship between them

LINEAR REGRESSION MODELS :

* Linear regression is a statistical method that allows us to summarize and study relationship between two continuous quantitative variables.

* The objectives of a linear regression model is to find a relationship between the input variables and a target variable.

* One variable, denoted y is regared as the response, outcome or dependent Variable

* The other, denoted x, is regared as the response, predictor explanatory or independent variable.

Regression models predict a continuous variable such as the sales made on the day or predict temparature of a city. Let's imagine that we fit a line with the training point that we have. If we want to add another data point but to fit it, we need to change existing model.

classification predicts categoriel lables (classes), prediction model continuous-valued functions. classification is considered to be supervised learning.

REGRESSION LINE :

It gives the average relationship between the two variables in mathematical form.

For two variables x and y, there are always two lines of regression.

Regression line of x on y gives the best estimate for the value of x for any $X = a + by$

Where

$a \rightarrow x$ - intercept

b = slope of the line

x = Dependent Variable

y = Independent Variable.

Regression line y on X

It gives the best estimate for the value of y for any specific give values of x

$$y = a + bx$$

Where

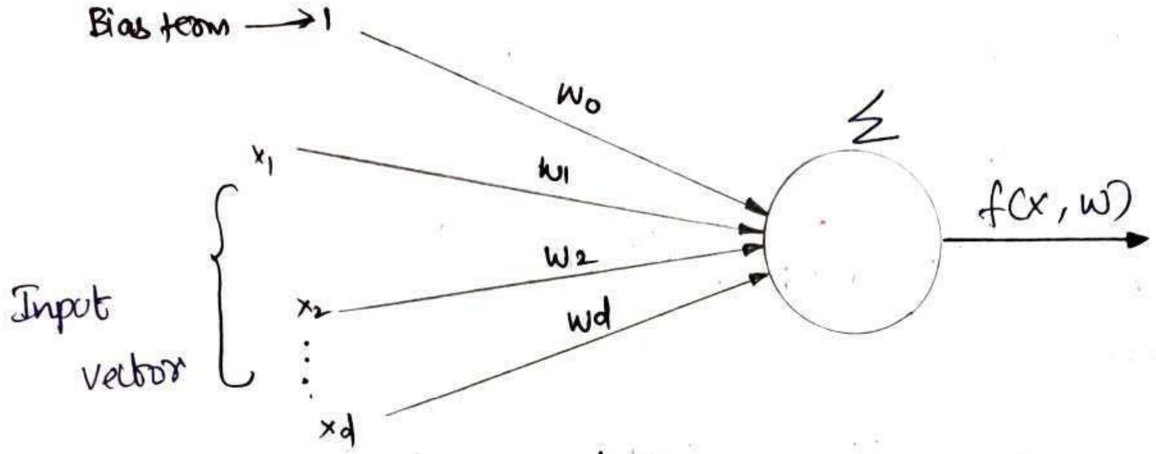
$a = y$ - intercept

b = slope of the line

y = Dependent Variable

x = independen Variable

By using least square method we are able to construct a best fitting to Scatter diagram points and then formulate a regression equation in the form of:



$$\hat{y} = a + bx$$
$$\hat{y} = \bar{y} + b(x - \bar{x})$$

Regression analysis is the art and science of fitting straight lines to patterns of data. In linear regression model the variable of interest (dependent variable) is predicted from k other variables (independent Variables using linear equation. If y denotes the dependent variable and $X_1 \dots X_k$ are the independent variables then the assumption is that the value of Y at time t in the data sample is determined by the linear equation.

$$Y_1 = \beta_0 + \beta_1 X_{1t} + \beta_2 X_{2t} + \dots + \beta_k X_{kt} + \epsilon_t$$

Where the betas are constants and the epsilons are independent and identically distributed noormal random variables with mean zero.

At each split point the "error" between the predicted Value and the actual values is Squared to get a "Sum of Squared Errors (SSE)". The Split point errors across the variables are compared and the Variable point yielding the lowest SSE is Chosen as the root node slipt point. This process is recurseivily continued.

Advantages:

Training a linear regression model is usually much faster than method such as neural networks.

Linear regression models are Simple and require mi$\overset{ni}{\hat{}}$mum memory to implement.

LEAST SQUARE:

The method of least Squares is about estimating parameters by minimizing the Squared discrepancies between observed data on the one hand and expected values on the other

The least Square Criterion states that the Sum of Square of error is minimum

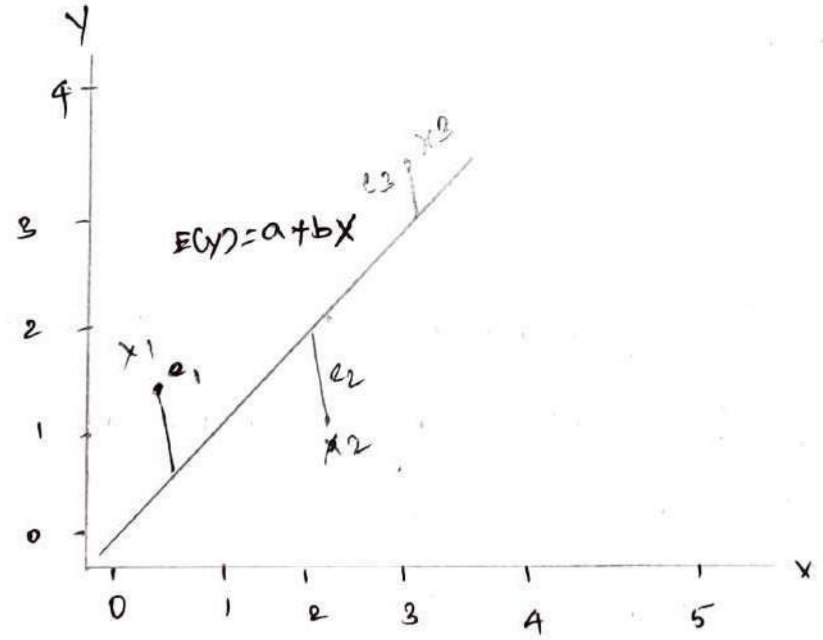
The least square solutions yields $y(x)$ whose sum to 1 but do not ensure outputs to be in the range [0, 1]

How to draw such a line based on the data points observed? Suppose a imaginary line of $y = a + bx$

Imagine a vertical distance between the line and the data point $E = Y - E(Y)$

The error is the deviation of the data point from the imaginary line, regression line. Then what is the best values of a and b?

a & b that minimize sum of such errors.



Deviation does not have good properties for computation
Then why do we use square of deviation?

Let us get a & b that can minimize the sum of squared deviations. This method is called least squares.

Least square method minimizes the sum of squares of errors. Such as a & b are called least square estimators. i.e estimators of parameters α & B

The process of getting parameter estimators (eg. a&b) is called estimation. Least square method is the estimation method of ordinary least squares (OLS).

Disadvantages of least square

* Lack robustness to outliers
* Certain datasets unsuitable for least square classification.
* Decision boundary corresponds to Machine learning Solution

MULTIPLE REGRESSION :

Regression analysis is used to predict the value of one or more responses from a set of predictors. It can also be used for estimate the linear association between the predictors and responses. predictors can be continuous or categorical or a mixture of both.

If the multiple independent variables affect the response Variable, Then the analysis call for a model different from that used for the Single predictor valuable. In a situation where more than one independent factor (Variable) affects the outcome of a process, a multiple regression model is used This is referred to as multiple linear regression model or multivariate least Square fitting.

$$Y_1 = \beta_0 + \beta_1 z_{j1} + \beta_2 z_{j2} + \cdots + \beta_r z_{jr} + \varepsilon_j$$

Where ε is the random error

$\beta_i, i = 0, 1 \ldots r$ are un-known ~~re~~ regression

Co-efficient

Difference between Simple Regression and Multiple Regression

Simple Regression	Multiple Regression
One dependent Variable Y predicted from one independent Variable X	One dependent Variable Y predicted from a set of independent Variables $(X_1, X_2 \ldots X_k)$
One regression coefficient	One regression coefficient for each independent Variable

BAYESIAN LINEAR REGRESSION:

* Bayesian linear regression allows a useful mechanism to deal with insufficient data or poor distributed data
* It allows user to put a prior on the coefficients and on the noise so that in the absence of data the priors can take over. A prior is a distribution on a parameter.

* If we could flip the coin an infinite number of times, inferring its bias would be easy by the law of larage numbers.
* However What if we could only filp the coin a handful of times? Would we guess that a coin is baised if we saw three heads in three flips, an event that happens one out of eight times with unbiased coins? They overfit these data, inferring a coin bias of $p=1$

* Bayesian methods allows us to estimate model parameters to construct model forecasts and to conduct model comparisons. Bayesian learning algorithms can calculate explicit probabilities for hypotheses.

Bayesian classifiers use a Simple idea that the training data are utilized to calculate an observed probablity of each class based on feature values.

When the Bayesian classifier is used for unclassified data, it uses the observed probabilities to predict the most likely class for the new features.

Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct.

Bayesian methods can accommodate hypotheses that make probabilistic predictions. New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.

Even in cases where Bayesian methods prove computationally intractable, they can provide a standard of optimal decision making against which other practical methods can be measured.

Uses of Bayesian classifiers:

*Used in text-based classification for finding spam or junk mail filtering

*Medical diagnosis

*Network security such as detecting illegal instruction.

Basic procedure for implementing Bayesian linear Regression :

Specify priors for the model parameters

Create a model mapping the training inputs to the training outputs.

Have a Markov chain Monte Carlo (MCMC) algorithm draw Samples from the posterior distributions for the parameters.

Gradient Descent :

Gradient descent is a first-order optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient of the function at the current point.

Gradiant descent is popular for very larage scale optimization problems because it is easy to Implement, can handle black box functions and each iteration is cheap

The gradient will give the slope of the curve at that x and its direction will point to an increase in the function. so we can change x in the opposite direction to lower the function value.

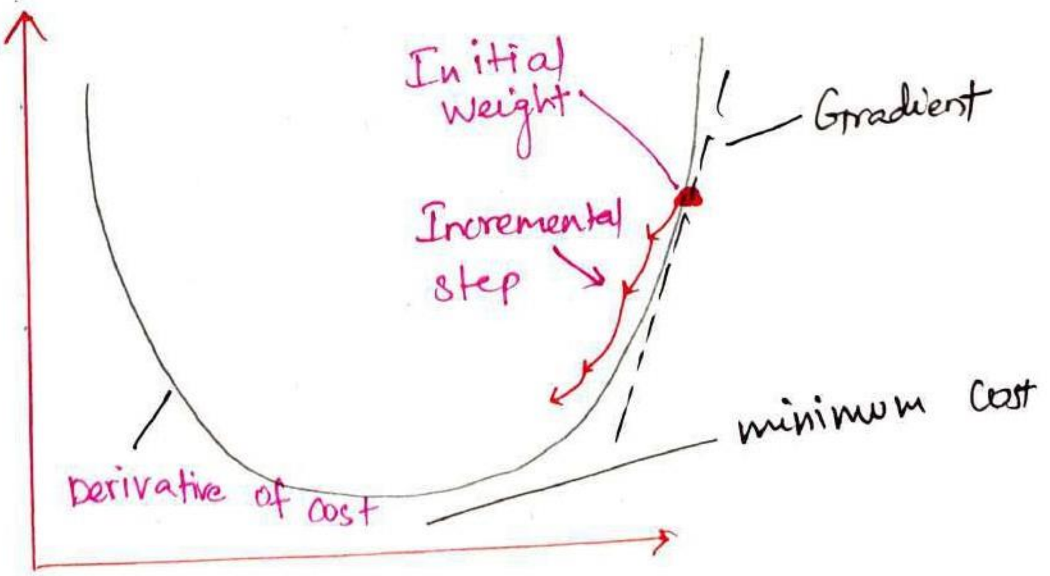
$$x_{k+1} = x_k - \lambda \nabla f(x_k)$$

The $\lambda > 0$ is a small number that forces the algorithm to make small jumps.

Limitation of Gradient Descent

Gradient descent is relatively slow close to the minimum: technically its asymptotic rate of convergence is inferior to many other methods.

For poorly conditioned convex problems gradient descent increasingly 'zigzags' as the gradient points nearly orthogonally to the shortest direction to a minimum point.



If we move towards a negative gradient or away from the gradient of the function at the current point it will give the local minimum of the function.

Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the local maximum of the function.

This entire procedure is known as Gradient ascent which also known as steepest descent. The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.

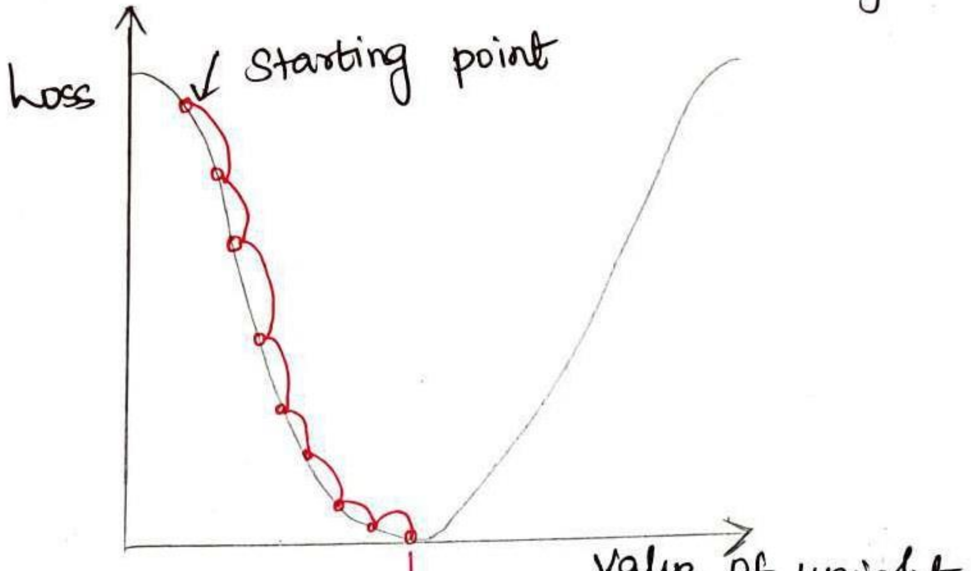
calculate the first order derivative of the function to compute the gradient or slope of the function.

Move away from the direction of the gradient which means slope increased from the current point by alpha times, where alpha is defined as learning Rate. It is a tuning parameter in the optimization process which helps to decide the length of the steps.

Working of Gradient Descent

$$y = mx + c$$

Where m represents the slope of the line and c represents the intercepts on the y-axis



point of convergence
i.e where the cost function is at its minimum

The slope becomes steeper at the starting point or arbitary point but whenever new parameters are generated then steepness gradually reduces and at the lowest point, it approaches the lowest point which is called a point of convergence.

Learning Rate: It is defined as the step size taken to reach the minimum or lowest point. This is typically a small value that is evaluated and updated based on the behavior of the cost function. If the learning rate is high it results in larger steps but it also leads to risks of overshooting the minimum.

At the same time a low learning rate shows the small step sizes which compromises overall efficiency but gives the advanage of more precision.

Types of Gradient Descent

1. Batch Gradient Descent
2. Stochastic gradient Descent
3. Mini Batch Gradient Descent

Batch Gradient Descent:

It is used to find the error for each point in the training set and update the model after evalvating all training examples. It is known as training epoch.

Stochastic gradient Descent

Stochastic gradient Descent is a type of gradient Descent that runs one training example per iteration. It is more efficient for large datasets.

Mini batch Gradient Descent

Mini batch gradient Descent is the combinati of both batch gradient descent and stochastic gradient descent. It divides the training datasets into Small batch sizes then performs the updates

LINEAR CLASSIFICATION MODELS :

A classification algorithm that makes its classification based on linear predictor function combining a set of weights with the feature vector.

It does classification decision based on the value of a linear combination of the characteristic. Imagine that the linear classifier will merge into its weights all the characteristics that define a particular class.

Discriminative functions :

Linear Discriminant Analysis (LDA) is one of the commonly used dimensionality reduction techniques in machine learning to slove more than two - class classification problems. It is known as Normal Discriminant Analysis (NDA) or Discriminant Functio Analysis(DFA)

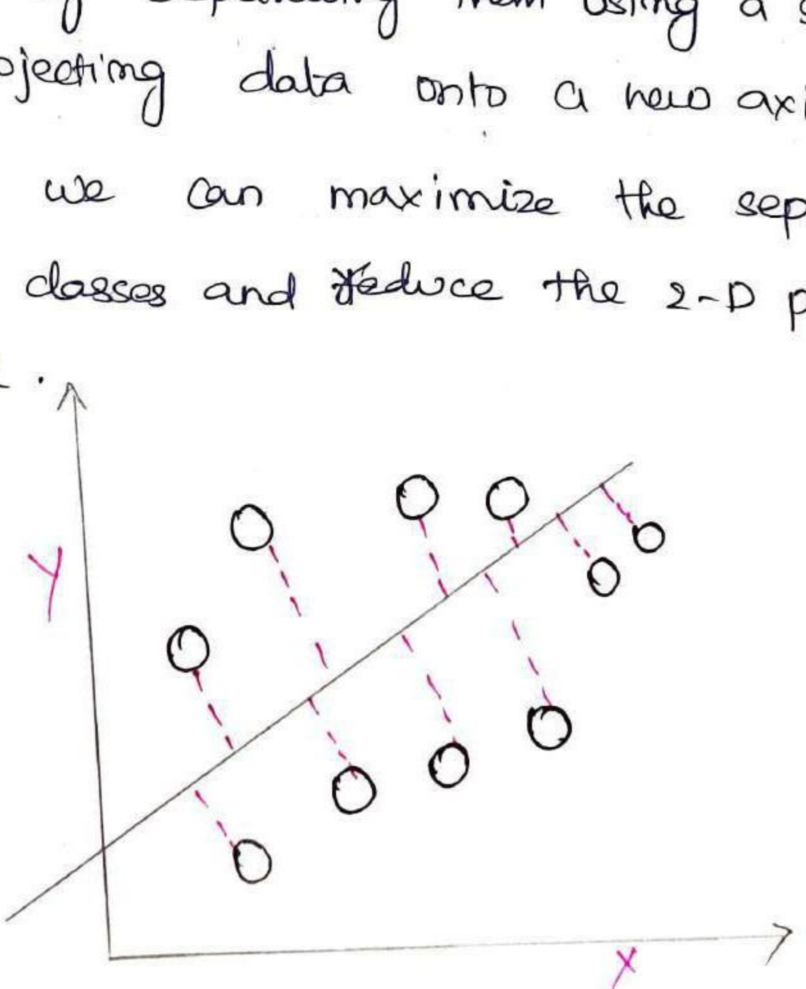
Linear Discriminant analysis is one of the most popular dimensionality reduction techniques used for Supervised classification problems in machine learning. It is also considered a pre-processing step for modeling differences in ML and application of

Linear Discriminant analysis is used as a dimensionality reduction technique in machine learning using which we can easily transform a 2-D and 3-D graph into a one dimensional plane

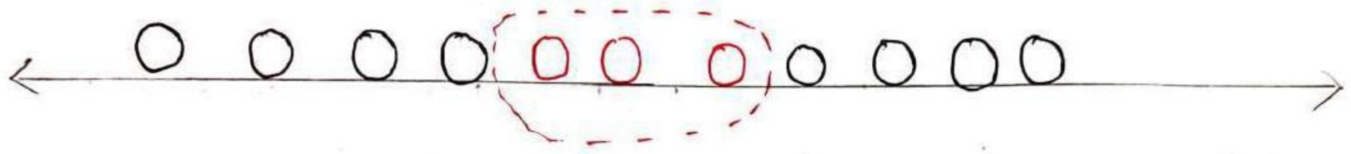
Let's consider an example where we have two classes in a 2-D plane having an x-y axis and we need to classify them efficiently. As we have already seen in the above example that LDA enable us to draw a straight line that can completely separate the two classes of data points.

Here LDA uses an x-y axis to create a new axis by separating them using a straight line and projecting data onto a new axis.

Hence we can maximize the separation between these classes and reduce the 2-D plane into one dimensional.



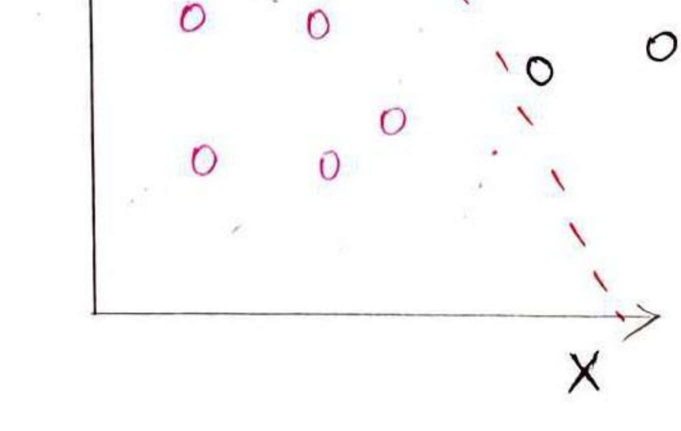
pattern classification. For eg, If we have two classes with multiple features and need to separate them efficiently. When we classify them using a single feature, Then it may show overlapping.



overlapping.

To overcome the overlapping issue in the classification process, we must increase the number of features regularly.

Eg:
Let's assume we have to classify two different classes having two sets of data points" in a 2-dimensional plane



To create a new axis, Linear Discriminant Analysis uses the following criteria:

⇒ It maximizes the distance between means of two classes.

⇒ It minimizes the variance within the individual class.

In other words we can say that the new axis will increase the separation between the data points of two classes and plot them onto the new axis.

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.

* Therefore the outcome must be a categorical or discrete value. It can be either yes or No, 0 or 1, true or flase ect. but Instead of giving the exact value as 0 & 1 it gives the probabilistic values which lies between 0 & 1

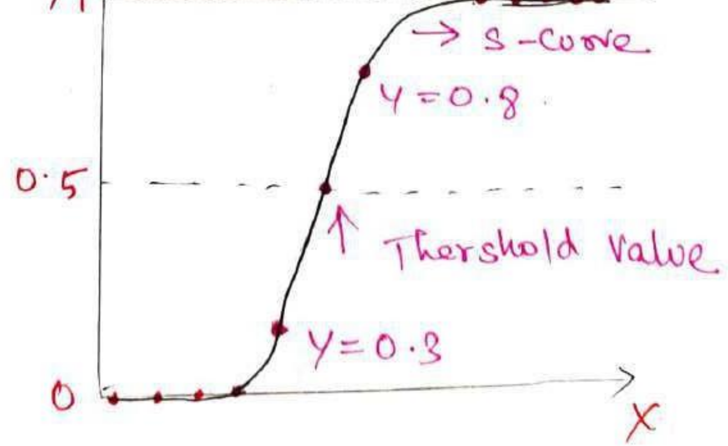
* Logistic regression is used for sloving the classification problems.

* In logistic regression, instead of fitting a regression line, we fit an 'S' shaped logistic function, Which predicts two maximum values(0 & 1)

* In logistic regression instead of fitting a regression line, we fit an "S" shaped logistic function Which predicts ~~two~~ two maximum values (0 and 1)

It is a Significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets

It can be used to classify the observation using different types of data and can easily determine the most effective variables used for the classification.



LOGISTIC FUNCTION :

* The sigmoid function or logistic 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 dependent variable should be categorical in nature

* The independent variable should not have multi-collinearity.

Logistic Regression Equation:

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

In logistic Regression y can be between 0 and 1 only so for this lets 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 -[infinity] to +[infinity] then take logarithm of the equation it will become.

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

Type of Logistic Regression

Binomial :

In this regression there can be only two possible types of the dependent Variables such as 0 or 1 pass or fail ect.

Multinomial :

In this regression there can be 3 or more possible unordered types of the dependent Variable such as "cat" or "dog" or "sheep".

Ordinal : In this regression there can be 3 or more possible Ordered types of dependent variables such as "low", "medium", or "high".

GENARATIVE MODEL :

* Generative models are a class of statistical models that generate new data instances.

* These models are used in unsupervised machine learning to perform tasks such as probability and likelihood estimation, modelling data points and distinguishing between classes using these probabilities.

* Generative models rely on the Bayes theorem to find the joint probability.

* Generative model describe how data is generated using probabilistic models.

* They predict $P(y/x)$, the probability of y given x, calculating the $P(x,y)$, the probability of x and y.

NAVIE BAYES CLASSIFIER:

* Navie 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 high dimensional training dataset.

It helps in building the fast machine learning algorithms or 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 Naive Bayes Algoorithm are spam filtration, Sentimental analysis, and Classifying articles.

* It is called Naive Because of it assumes that The ocuerence of a certain feature is independent of the ocuerence 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.

Baye's Theorem:

It is also know as Bayes Role or Bayes law which is used to determine the probability of a hypothesis with prior knowledge. It depends on The Conditional probability.

The formula for Bayes theorem is given as

$$P(A/B) = \frac{P(B/A)\,P(A)}{P(B)}$$

Where

P(A/B) is a posterior probability, probability of hypothesis A on the observed event B

P(B/A) is a likelihood probability, probability of the evidence given that the probability of the hypothesis is true

P(A) is a prior probability, probability of hypothesis before evidence.

P(B) is Marginal probability, probability of evidence.

Difference between generative and Discriminative Model

Generative model	Discriminative model
It generates new data	This discriminate between differents kind of data instances
Generative model revolves around the distribution of a dataset to return a probability for a given example.	It makes predictions based on conditional probability and is either used for classification or regression

* Generative model capture the joint probability $P(x, y)$ or just $p(x)$ if there are no labels.	* Discriminative models capture the conditional probability $p(y/x)$
* A generative model includes the distribution of the data itself, and tells you how likely a given example is	* A discriminative model ignores the question of whether a given instance is likely and just tells you how likely a lable is apply to the instance.
* Generative models are used in unsupervised machine learning to perform task such as probability a likelihood estimation.	* This model particularly used for supervised learning.
* Eg: Gussains, Nawie Bayes	* Eg: logistic Regression SVM

SUPPORT VECTOR MACHINE:

* Support Vector Machines (SVMs) are a set of supervised learning methods which learn from the dataset and used for classification.

* SVM is a classifier derived from statistical learning Theory by Vapnik and Chervonenkis.

Simply speaking we can think of an SVM model as representing the examples as points in space mapped so that each of examples of the separate classes are dividied by a gap That is wide as possible.

Example of Bad Decision Boundaries

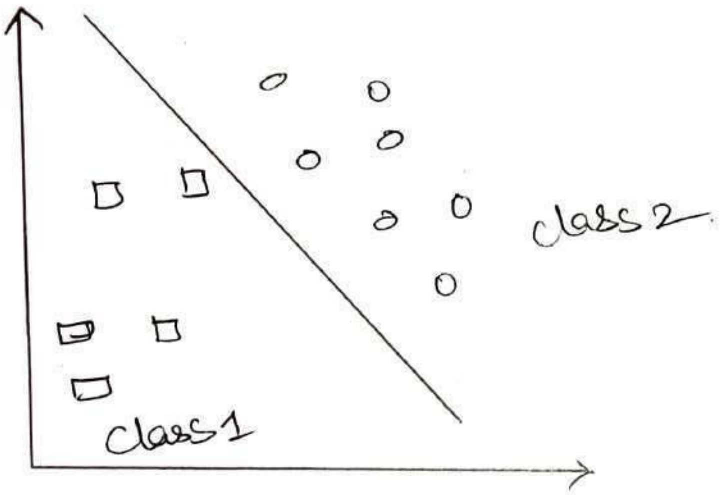
SVM are primarily two-class classifiers with The distinct characteristic that they aim to find the optimal hyperplane such that the expected generalization error is minimized.

Instead of directly minimizing the empirical risk calculated from the training data SVMs perform structural risk minimization to achieve good generalization.

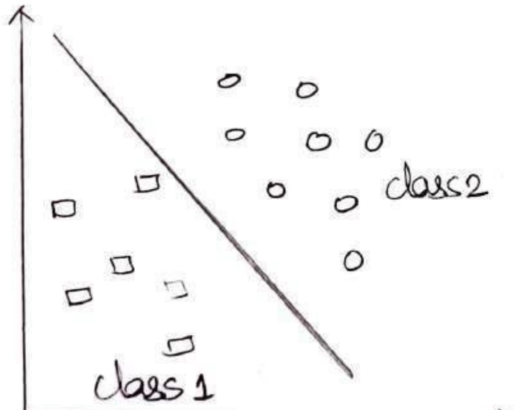
The empirical risk is the average loss of an estimator for a finite set of data drawn from p.

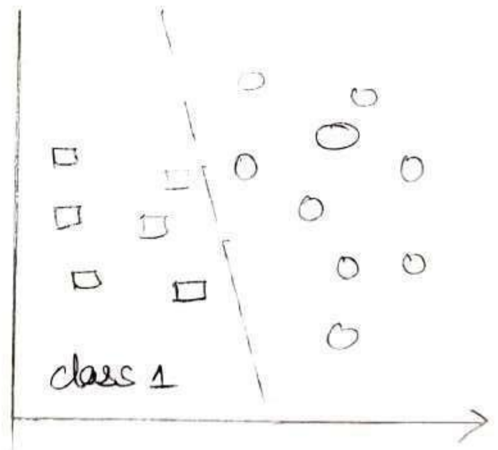
The idea of risk mimization is not only measure the performance of an estimator by its risk, but to actually search for The estimator That minimizes is ~~over~~ over distribution p.

* It is a kind of large Margin classifier.
* It is a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far away any point in training data.



Two class problem





Bad decision boundary of SVM

Given a set of training examples, each marked as belonging to one of two classes an SVM algorithm builds a model that predicts whether a new example falls into one class to the other

Because we dont know distribution p we instead minimize empirical risk over a training dataset drawn from p. This general learning technique is called empirical risk minimization.

Good Decision Boundary :

The decision boundary should be as far away from the data of both classes as possible.

If the data points lie very close to the boundary, the classifier may be consistent but is more likely to make errors on new instances from the distribution.

Hence we prefere classifiers that maximize the minimal distance of data points to the separator.

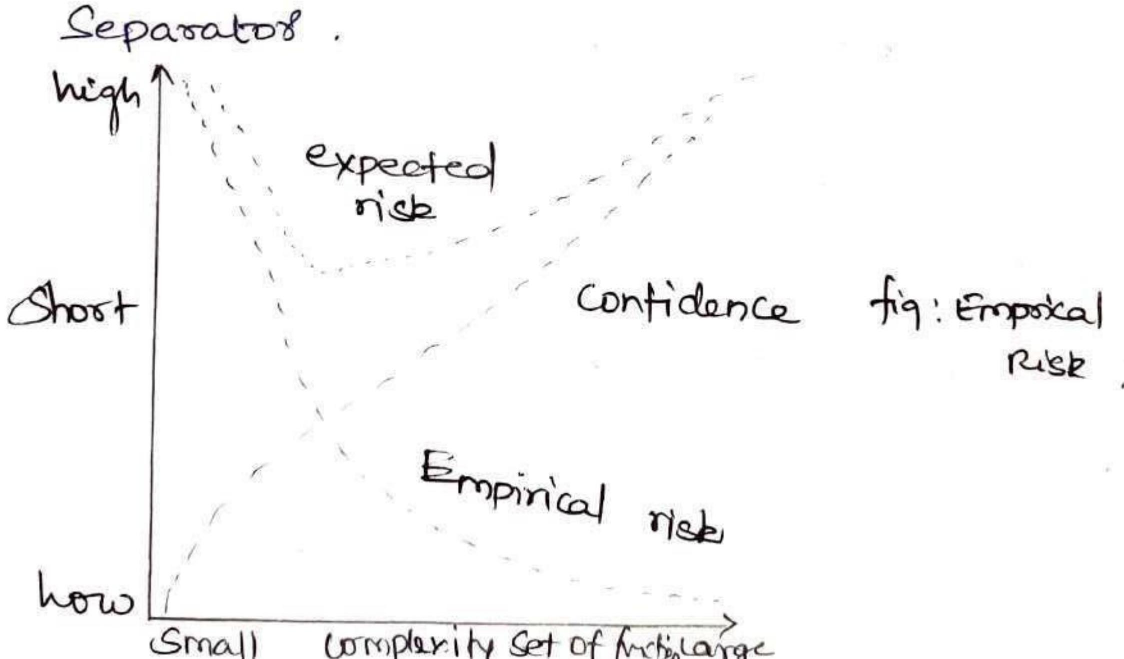
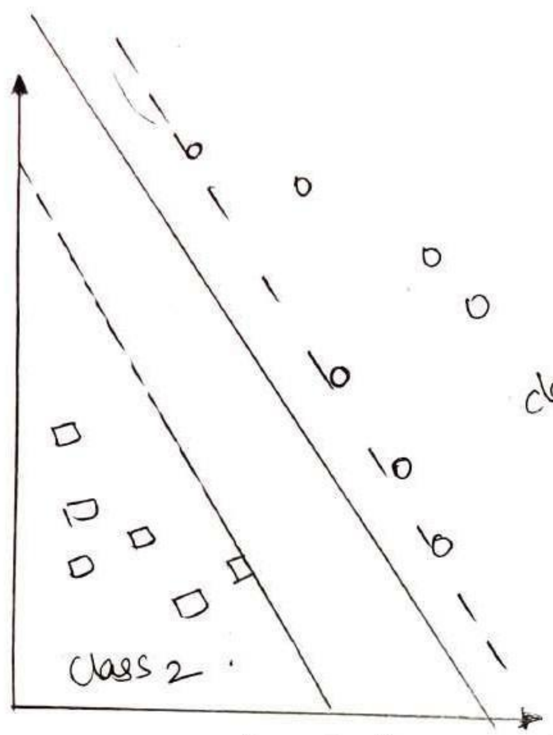


fig: Empirical Risk



Good decision bondary.

* The gap between data points and the Classifier bondary.

* The margin is the minimom distance of any Sample to the decision bondary.

* Margin of the Separator is distance between Support Vectors.

$$\text{Margin } (m) = \frac{2}{\|w\|}$$

* Maximal margin classifier is a classifier in The family F that maximizes the margin. Maximizing the margin is good according to intoition and PAC Theory.

* Implies That only Support vectors matter other training examples are ignorable.

Key properties of SVM :

* Use a single hyperplane which subdivides the space into two half spaces one which is occupied by class 1 and the other by Class 2 .

* They maximize the margin of the decision boundary using quadratic optimization techniques, which find the optimal hyperplane

* Ability to handle large feature spaces
* Overfitting can be controlled by soft margin apporach

SVM Applications

SVM has been used successfully in many real word problems .

* Text (and hypertext) categorization .
* Image classification
* Bioinformatics (protein classification, cancer classification)
* Hand written character recognition
* Determination of SPAM email

Limitations of SVM:

* It is sensitive to noise.
* The biggest limitation of SVM lies in the choice of kernel.
* Another limitations is speed and size
* The optimal design for multiclass SVM Classifier is also a research area.

SOFT MARGIN

For the very high dimensional problems common in text classification, sometimes the data are linearly separable.

But in the general case they are not and even if they are, we might prefer a solution that better separates the bulk of the data while ignoring a few weird noise documents.

What if the training set is not linearly separable? slack variables can be added to allow misclassification of difficult or noisy examples resulting margin called soft.

A Soft margin allows a few variables to cross into the margin or over the hyperplane.

DECISION TREE

* A decision tree is a simple representation for classifying examples. Decision tree learning is one of the most successful techniques for supervised classification learning

* In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions

* Each leaf node has a class label, determined by majority vote of training examples reaching that leaf.

* Each terminal node is a question on features. It branches out according to the answers.

* Decision tree learning is a method for approximating discrete valued target functions The learned function is represented by decision tree.

* A learned decision tree can also be re-represented as a set of if - then rules

* Decision tree learning is one of the most widely used and practical methods for inductive inference.

* It is robust to noisy data and capable of learning disjunctive expression.

* Decision tree learning method searches a Completely expressive hypothesis

DECISION TREE REPRESENTATION

* Build a decision tree for classifying examples as positive & negative instance of concept.

* Each non leaf node has associated with it an attribute.

* Each leaf node has associated with it a classification (+ or −)

* Each arc node has associated with it one of the possible values of the attribute at the node from which the arc is directed

* Internal node denotes a test on an attribute. Branch represents an outcome of the test. Leaf node represents class labels or class distribution

A decision tree is a flow chart like structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test and tree leaves represent classes or class distributions.

DECISION TREE ALGORITHM

To generate decision tree from the training tuple of data partition D

Input

Data partition (D)
Attribute List
Attribute Selection method

ALGORITHM :

⇒ Create a node (N)

⇒ If tuples in D are all same class then

⇒ Return node (N) as a leaf node labeled with the class C

⇒ If attribute list is empty then return N as a leaf node labeled with the majority class in D

⇒ Apply attribute selection method (D, attribute list) to find the 'best' splitting criterion.

⇒ Label node N with splitting attribute

⇒ If splitting attribute is discrete valued and multiway splits allowed.

⇒ Then attribute list -> attribute list -> splitting attribute

⇒ For (each outcome j of splitting criterion)

⇒ Let D'j be the set of data tuples in D statisfying outcome j

⇒ If D'j is empty then attach a leaf labeled with majority class in D to Node N;

⇒ Else attach the node returned by Generate

· Decision tree (D'j, attribute list) to Node N;

⇒ End of for loop

⇒ Return N

Decision tree generation consists of two phases one is tree construction & pruning.

In tree construction phase, all the training examples are at the root. partition examples recursively based on selected attributes

In tree purning phase, the identification & removal of branches that reflect noise or outliers

Advantages:

* Roles are simple and easy to understand
* Decision tree can handle both nominal & numerical attributes
* Decision tree are capable of handling datasets that may have errors.
* It has capable of handling datasets that may have missing values.
* Decision trees are considered to be a nonparametric method.
* Decision tree are self - explantory.

Disadvantages:

* Most of the algorithms require that the target attribute will have only discrete values
* Some problem are difficult to solve like XOR.
* Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute

RANDOM FORESTS :

Random forests is a famous system learning set of rules that belongs to the Supervised getting to known method.

It may be used for both classification and regression issues in ML

It is based totally on the concept of ensemble studying that's a process of combining multiple classifiers to solve a complex problem and to enhance the overall performance of the model.

Random forest is a classifier that incorporates some of choice timber on diverse subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

Random Forests Algorithm Working

Random forest works into two section first is to create the random woodland by combining N selection trees and Second is to make predictions for each tree created inside the first Segment.

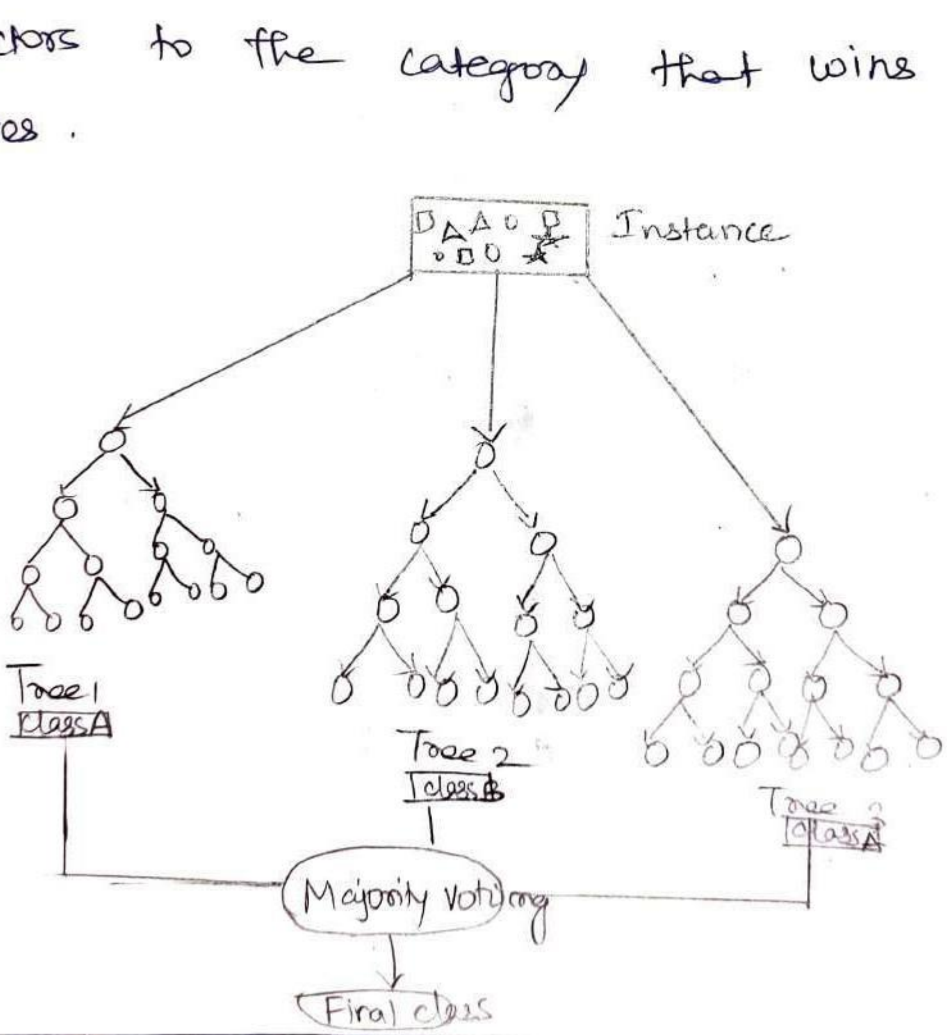
1 ⇒ Select random K statistics points from the Schooling set

2 ⇒ Build the Selection trees associated with the selected information points.

3 ⇒ choose wide variety of N for Selection trees associated with the selected information which we want to build.

4 ⇒ Repeat step 1 and 2

5 ⇒ For new factors locate the predictions of each choice tree and assign the new record factors to the category that wins most people votes.



Application of Random forest :

* Banking ⇒ Banking zone in general uses This algorithm for the identification of loan clangese

* Medicine : With the assistance of this set of roles, disorder traits and risks of the disorder may be recognized.

* Land use : We can perceive the areas of comparable land use with the aid of This algorithm.

* Marketing : Marketing tendencies can be recognized by the usage of this algorithm.

Advantages of Random Forest :

* It is capable of managing large database with high dimensionality.

* It enhances the accuracy of the version and forestalls the overfitting trouble.

Disadvantage :

* Although random forest can be used for both class + regression ~~respon~~ responsibilities it isn't extra appropriate for regression obligations.

UNIT - III

Ensemble Techniques and Unsupervised Learning.

Combining Multiple learners

* When designing a learning machine, we generally make some choices like parameters of machine, training data and representation This implies some of sort of variance in performance

* For example in a classification setting we can use a parametric classifier or in a multilayer perception, we should also decide on the number of hidden units.

* Each learning algorithm dicates a certain model that comes with a set of assumptions.

* This inductive bias leads to error if the assumptions do not hold for the data

* Different learning algorithm have different acuracies. The no free lanch theorem asserts that no single learning algorithm always acheive the best performance in any domain

* They can be combined to attain high acuracy.

* Data fusion is the process of fusing multiple records representing the same real world object into a single, consistent and clean representation.

* Fusion of data for improving prediction accuracy and reliability is an important problem in machine learning.

* Combining different models is done to improve the performance of deep learning models.

* Building a new model by combination requires less time, data and computational resources

* The most common method to combine models is by averaging multiple models, where taking a weighted average improves the accuracy.

Generating Diverse Learners:

Different Algorithms: We can use different learning algorithms to train different base-learners. It make different assumptions about the data and lead to different classifi

Different Hyper-parameters :

We can use the same learning algorithm but use it with different hyper-parameters.

Different Input Representations :

Different representations make different characteristics explicit allowing better identification.

Different Training sets :

Another possibility is to train different base-learners by different subsets of the training set.

Model Combination Schemes

Different methods are used for generating final output for multiple base learners are multiexpert and multistage combination.

Multiexpert Combination :

* It is a methods have base-learners that work in parallel.

* Global approach : given an input all base learners generate an output and those outputs

are used such as voting and stacking.

Local approach: in Mixture of experts there is a gating model, Which looks at the input and choose one (or very few) of the learners as responsible for generating the output.

Multistage Combination:

* It is a methods use a serial approach where the next multistage Combination base-learners are not accurate enough.

* Lets assume that we want to construct a function that maps inputs to outputs from a set of known N train input-output pairs

$$D_{train} = \{(x_i, y_i)\}_{i=1}^{N_{train}}.$$

Where $x_i \in X$ is a D dimensional feature input vector, $y_i \in Y$ is the output.

Classification: When the output take values in the discrete set class labels $Y = \{c_1, c_2 \cdots c_K\}$ Where K is the number of different classes.

Regression consists in predicting continuous ordered outputs $Y = R$

voting :

* The Simplest way to combine multiple classifiers is by voting which corresponds to taking a linear combination of the learners.

* Voting is an ensemble machine learning algorithm.

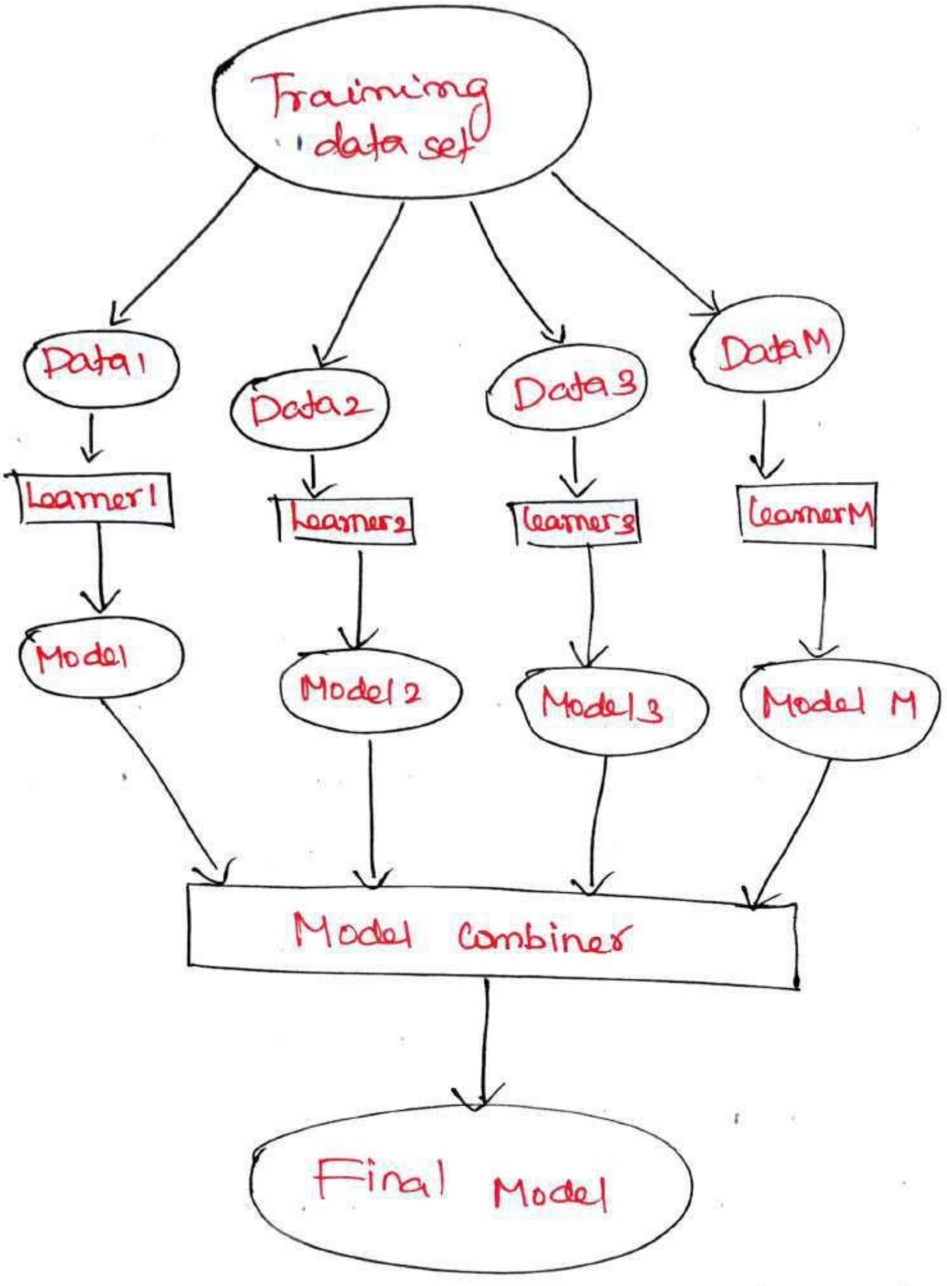
* For regression, a voting ensemble involves making a prediction that is the average of multiple other regression models.

* In classification a hard voting ensembles involves summing the votes for crisp class labels from other models and predicting the class with the most votes..

* A soft voting ensemble involves summing the predicted probabilities for class stables and predicting the class label with the largest sum probability

In this methods, the first step is to create multiple classification / regression models using some training dataset.

Each base model can be created using different splits of the same training dataset and same algorithm or using the same dataset with different algorithms or any other method.



When combining multiple independent and diverse decisions each of which is at least more accurate than random guessing, random errors cancel each other out and correct decisions are reinforced.
Human ensembles are demonstrably better

Use a single, arbitrary learning algorithm but manipulate training data to make it learn multiple models.

Error - correcting Output Codes

In error correcting Output codes main classification task is defined in terms of a number of subtasks that are implemented by all base learners

The idea is that the original task of separating one class from all other classes may be a difficult problem.

So we want to define a set of simpler classification problems, each specializing in one aspect of the task and combining these simpler classification we get final classifier.

Base learners are binary classifiers having output -1 +1 and there is a code matrix W of K x L whose

K rows are the binary codes of classes in terms of the L base-learners dj.

Ensemble Learning

* The idea of ensemble learning is to employ multiple learners and combine their predictions.

* If we have a committee of M models with uncorrelated errors, simply by averaging them the average error of a model can be reduced by a factor of M

* Unfortunately, the key assumption that the errors due to the individual models are uncorrelated is unrealistic, in practice, the errors are typically high correlated, so the reduction in overall error is generally small

* Ensemble modelling is the process of running two or more related but different analytical models and then Synthesizing the results into a single score or spread in order to improve the accuracy of predictive analytics and data mining applications

* Ensembles of classifiers is a set of Classifiers whose individual decisions combined in some way to classify new examples.

* Ensemble methods combine several decision tree classifiers to produce better predictive performance than a single decision tree classifier.

* The main principle behind the ensemble model is that a group of weak learners come together to from a strong learner thus increasing the accuracy of the model

ENSEMBLE METHODS WORKING

VARIANCE REDUCTION:

If the training sets are completely independent, it will always help to average an ensemble because this will reduce variance without affecting bias (eq, bagging) and reduce sensitivity to individual data points.

BIAS REDUCTION:

For simple methods, average of models has much greater capacity than single model Averaging models can reduce bias substantially by increasing capacity and control variance by citting one component at a time

BAGGING :

* Bagging is also called Bootstrap aggregating. Bagging and boosting are meta - algorithms that pool decision from multiple classifiers.

* It creates ensembles by repeatedly randomly resampling The training data

* Bagging was the first effective method of ensemble learning and is one of The Simplest method of arching.

* The meta - algorithm which is a Special case of the model averaging, was originally designed for classification and usually applied todecision tree models, but it can be used with any type of model for classification or regression.

* Ensemble classifiers such as bagging, boostingand model averaging are know to have improved accuracy and robustness over a single model

* AlThough unsuperised model, such as clustering do not directly generate label prediction for each individual They provide useful constraints for the joint prediction of a set of related objects

* For a given training set of size n Create m samples of size n by drawing n examples from the original data with replacement

* Each bootstrap sample will on average Contain 63.2% of the unique training examples, the rest are replicates.

* It combines The m resulting models using Simple majority vote.

* In particular on each round, The base learner is trained on What is often called a bootstrap replicate of the original data set.

* Suppose training set consists of n examples.

* Then a bootstrap replicate is a new training set That also consists of n examples and Which is formed by repeatedly selecting uniformly at random and with replacement n examples from the original training set.

* This means that The same example may appear multiple times in The bootstrap replicat or it may appear not at all.

* It also decreases error by decreasing the variances in the result due to unstable learner algorithms (like decision trees) whose output can change dramatically when the training data is slightly changed.

Bagging Steps:

* Suppose there are N observations and M features in training dataset
* A sample from training data set is taken randomly with replacement.
* A Subset of M features is Selected randomly and whichever feature gives the best split is used to split the node iteratively.
* The tree is grown to the largest

Above steps are repeated n times and prediction is given based on the aggregation of predictions from n number of trees.

Advantages of Bagging:

* Reduces over fitting of the model.
* Handles higher dimensionality data very well
* Maintains accuracy of missing data

Disadvantages of Bagging

* Since final prediction is based on the mean predictions from subset

BOOSTING

* Boosting is a very different method to generate multiple predictions (function and estimates) combine Them linearly.

* Boosting refers to a general and provably effective method of producing a very accurate classifier by combining rough and moderately inaccurate rules of thumb

* Orginally developed by Computational learning theorists to guarantee performance improvements on fitting training data for a weak learner That only needs to generate a hypothesis with a training accuracy greater than 0.5

* Final result is the weighted sum of the results of weak classifier.

* A learner is weak if it produces a classifier that is only slightly better than random guessing, While a learner

is said to be strong if produces a classifier that acheives a low error with high confidence for a given concept.

* Revised to be a practical Algorithm Adaboost for building ensembles that empirically improves generalization performance. Examples are given weight at each iteration a new hypothesis is learned and the examples are reweighted to focus the System

* Boosting is a bias reduction technique It typically improves the performance of a single tree model.

* A reason for This is that we often cannot construct trees which are Sufficiently large due to thining out of observations in the terminal nodes.

* Boosting is then a device to come up with a more complex solution by taking linear combination of trees.

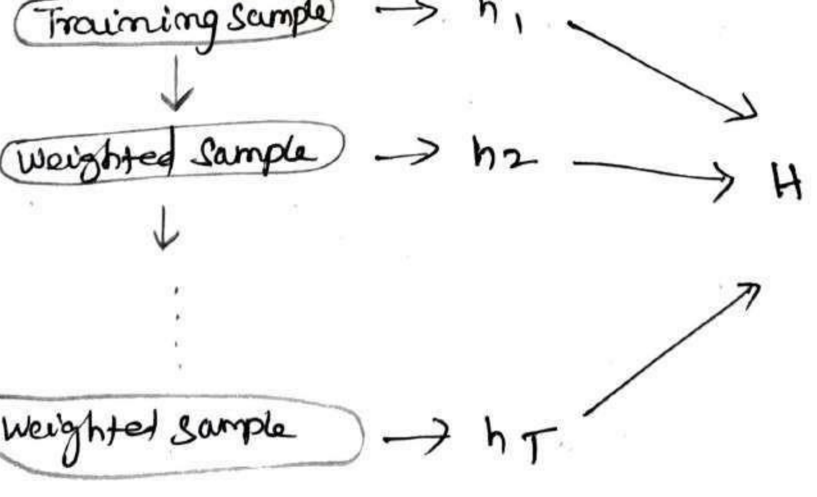
* In presence of high dimensional predictors boosting is also very useful as a regularization technique for additive or interaction modeling

* To begin we define an algorithm for finding the rules of thumb, which we call a weak learner.

* The boosting algorithm repeatedly calls this weak learner, each time feeding it a different distribution over the training data.

* Each call generates a weak classifier and we must combine all of these into a single classifier than hopefully is much more accurate than any one of the rules.

* Training a set of weak hypothesis $h_1 \cdots h_T$. The combined hypothesis H is a weighted majority vote of the T weak hypotheses. During the training focus on the examples that are misclassified



Ada Boost:

* Ada Boost short for "Adaptive Boosting", is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire who won the prestigious "Godel prize" in 2003 for their work.

* It can be used in Conjonction with many other types of learning algorithms to improve their performance

* It can be used to learn weak Classifiers and final classification based on weighted vote of weak Classifiers.

* It is linear classifier with all its desirable properties. It has good generalization properties

* To use the weak learner to form a highly accurate prediction rule by calling the weak learner repeatedly on different distributions over the training examples

Initially all weights are set equally but each round the weights of incorrectly classified examples are increased so that those observations that the previously classifier poorly eta predicts ~~re~~ receive greater weight on the next Iteration.

Advantages of AdaBoost

* Very Simple to implement
* Fairy good generalization
* The prior error need not be known ahead of time.

Disadventages of AdaBoost

* Suboptimal Solution
* Can over fit in presence of noise

Boosting Steps:

* Draw a random subset of training samples d_1 without replacement from the training set D to train a weak learner C_1
* Draw second random training subset d_2 without replacement from the training set and add 50 percent of the samples that were previously falsely classified /misclassified to train a weak learner C_2
* Find the training samples d_3 in the training set D on which C_1 and C_2 disagree to train a third weak learner C_3

* Combine all the weak learners via majority voting

Advantage of Boosting:

* Supports different loss function
* Works well with interactions

Disadvantages of Boosting:

* prone to over fitting
* Requires careful tuning of different hyper-parameters

STACKING:

* Stacking Sometimes called stacked generalization is an ensemble machine learning method that combines multiple heterogeneous base or component models via a meta model

* The base model is trained on the complete training data and then the meta-model is trained on the predictions of the base models.

* The advantages of stacking is the ability to explore the solution space with different models in the same problem

* The stacking based model can be visualized in levels and has at least two levels of the models.

* The first level typically trains the two or more base learners (can be heterogenous) and the Second level might be single meta learner that utilizes the base models predictions as input and gives the final result as output

A stacked model can have more than more than two such levels but increasing the levels does not always guarantee better performance.

Stacking is concerned with multiple Classifiers generated by different learning algorithms $L_1, \ldots L_N$ on a single dataset S, Which is composed by a feature vector

$S_i = (x_i, t_i)$

The stacking process can be broken into two phases:

Generate a set of base - level classifiers $C_1 \ldots C_N$ Where $C_i = L_i(S)$

Train a meta - level Classifier to Combine the output of the base-level classifier

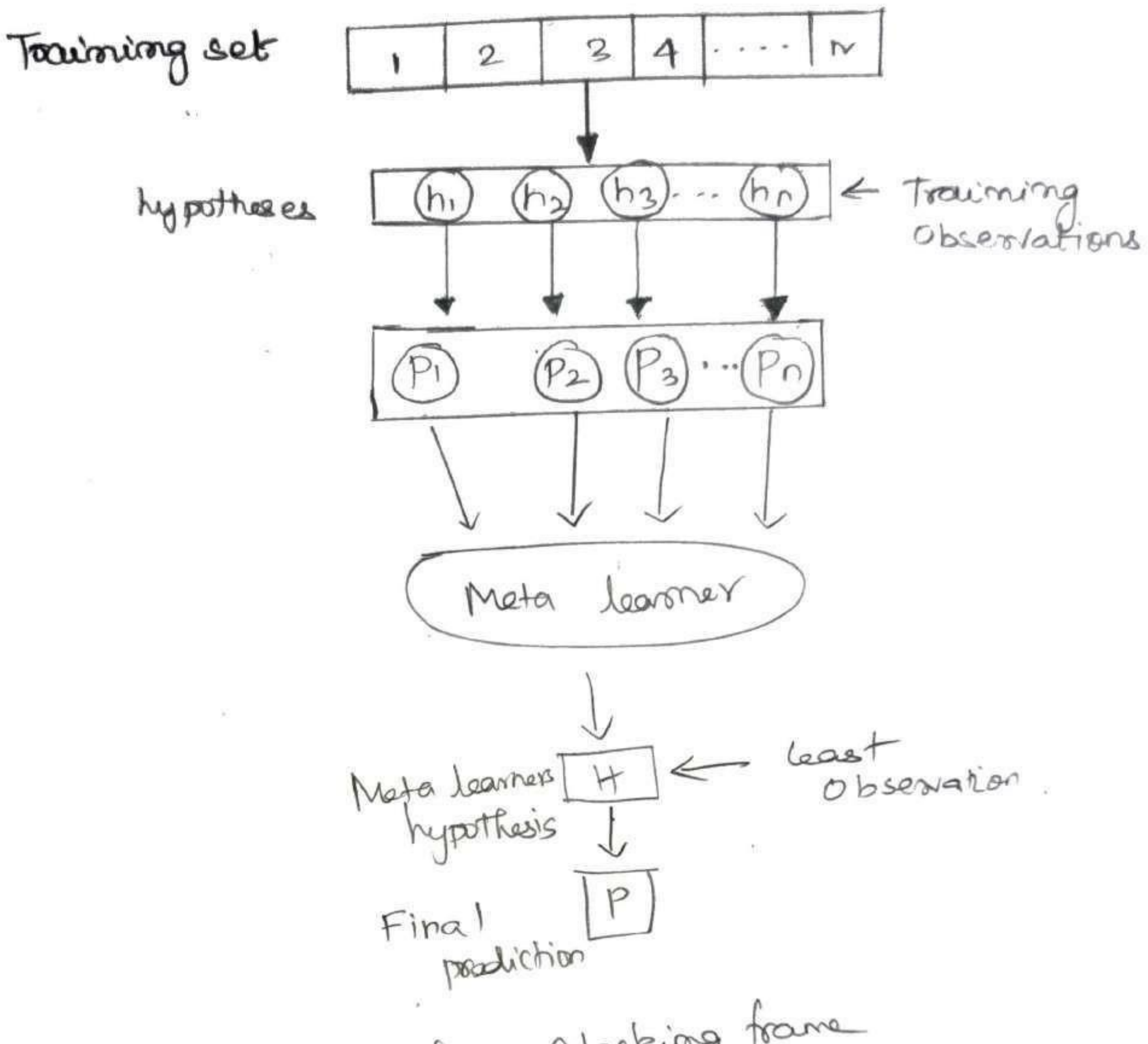


fig: Stacking frame

Based on two basic Observations.

Variance reduction:

If the training sets are Completely independent it will always help to average an ensemble because This will reduction

ADABOOST ALGORITHM:

* Ada Boost algorithm short for adaptive algorithm. It is a Boosting technique used as an ensemble method in machine learning.

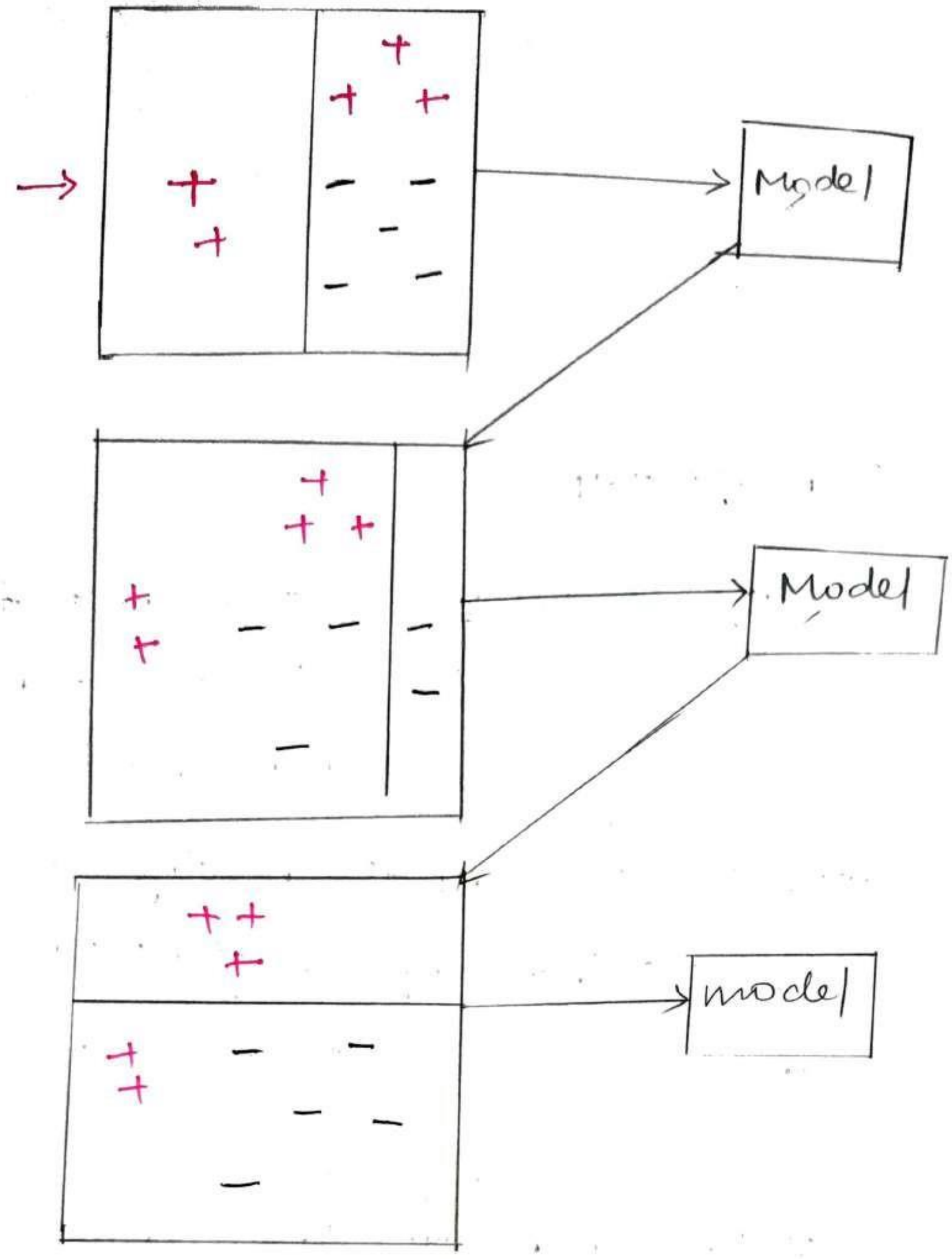
* It is called adpative boosting as the weights are reassigned to incorrectly classified instances

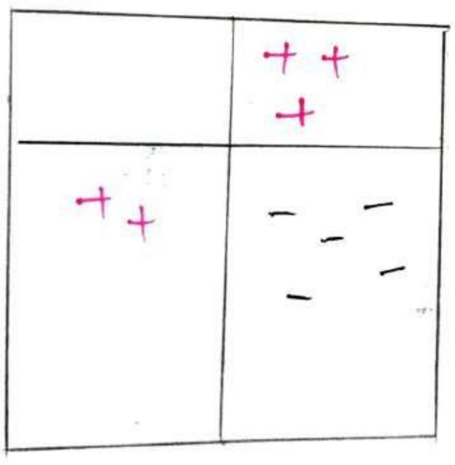
* Boosting is used to reduce bias as well as variance for supervised learning.

* It works on the principle of learners growing sequentially

Except for the first each subsequent learner is grown previously grown learners.

In Simple words, weak learners are converted into strong ones. The AdaBoost algorithm works on the same principle as boosting with a slight difference in detail


Model
Model
model



*The maximul not unsual algorithm used with AdaBoost is selection trees with one stage meaning with decision trees with most effective one split.

*These trees are also referred to as decision stumps

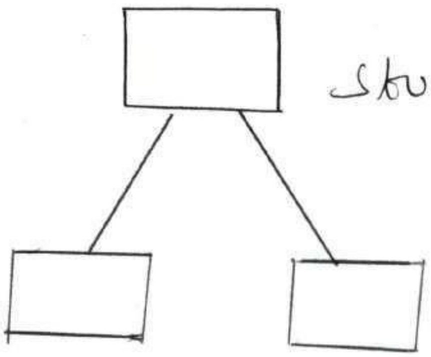
The working of Ada boost version follows the beneath referred to as decision stumps or path :

* Creation of the base learner
* Calculation of the total error via the beneath formulation
* Calculation of performance of the decision stumps
* Updating the weights in line with the misclassified factors.

Creation of new database :

AdaBoost ensemble :

In the ensemble approach we upload the susceptible fashion sequentially and then teach them the use of weighted schooling records



We hold to iterate the process till we gain the advent of a pre-set range of vulnerable learners or we can not look at further improvement at the data set

At the end of Algorithm we are left with some vulnerable learners with a stage fee.

Difference between bagging and boosting

Bagging	Boosting
It is a technique that Builds multiple homogeneous models from different Subsamples of the Same training dataset to obtain more accurate predictions	It refers to a group of algorithms that Utilize weighted averages to make weak learning algorithms stronger learning algorithms

* It helps in reducing Variance	* It helps in reducing bias and variance
* Every model receives an equal weight	* Models are weighted by their performance

CLUSTERING

* Give an set of objects, place them in group such that the objects in a group are similar (or related) to one another and different form (or unrelated to) the objects in other group

* Closter analysis can be a powerful data mining tool for any organization that needs to identify discrete groups of customers, sale transactions, or other types of behaviours and things.

* For example, insurance providers use cluster analysis to detect fraudulent claims and banks used it for credit scoring

* Closter analysis uses mathematical tool models to discover group of similar customers based on the smallest variation among customers within

each group.

* cluster is a group of objects that belong to the same class. In another words the similar object are grouped in one cluster and disimilar grouped in other cluster.

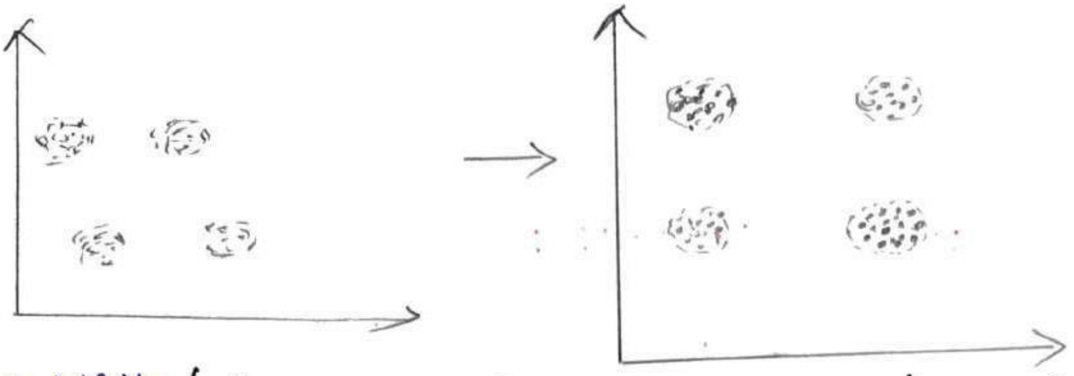
* clustering is a process of partitioning a set of data in a set of meaningful subclasses.

* Every data in the sub class shares a common trait. It helps a user understand the natural grouping or structure in the data set.

* Various types of clustering methods are partitioning methods, hierarchical, clustering, fuzzy clustering, density based clustering and model based clustering.

* Cluster analysis is process of grouping a set of data objects into clusters.

Desirable properties of a clustering algorithm are as follows



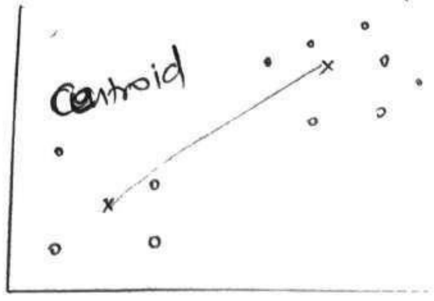
* Scalability (in terms of both time and space)
* Ability to deal with different data types
Minimal requirements for domain knowledge to determine input parameters
* Interpretability and usability

Clustering of data is a method by which large sets of data are grouped into clusters of smaller sets of similar data.

* clustering can be considered the most important unsupervised learning problem.

$$\begin{tikzcd}\boxed{\text{Raw data}} \arrow[r, Rightarrow] & \boxed{\text{clustering algorithm}} \arrow[r, Rightarrow] & \boxed{\text{clusters of data}}\end{tikzcd}$$

* clustering means grouping of data or dividing a large set into smaller data sets of some similarity



CLUSTER CENTROID:

* The centroid of a cluster is a point whose parameter values are the mean of the parameter values of all the points in the cluster.

* Each cluster has a well defined centroid

DISTANCE:

The distance between two point is taken as common metric to see as the similarity among the components of population

The commonly used distance measure is the euclidean metric which defines the distance between two points $p=(p_1, p_2, \ldots)$ and $q=(q_1, q_2 \ldots)$ is given by

$$d = \sum_{i=1}^{k} (p_i - q_i)^2$$

* The goal of clustering is to determine the intrinsic grouping in a set of unlabled data. But how to decide What constitutes a good clustering?

* It can be shown that there is no absolute best criterion which should be independent of the final aim of clustering

clustering algorithm can be classified as listed below:

⇒ Exclusive clustering
⇒ Overlapping Clustering
⇒ Hierarchical Clustering
⇒ probabilistic clustering

A good clustering method will produce high quality clusters intra-class Similarity and low intra class similarity

The Major clustering techniques are

* partitioning methods
* Hierarchical methods
* Density Method

UNSUPERVISED K-MEANS CLUSTERING

* K-Means clustering is heoristic method. Here each closter is represented by the center of the Cluster.

* K stands for nomber of clusters, It is typically a user input to the algorithm some criteria can be used automatically estimate K.

* This method initially takes the number of components of the population eqwal to the final required nomber of closters

* In this step itself the final required nomber of closter is chosen such that the points are motvaley farthest apart.

* Given k-means algorithm consists of four steps:

⇒ Select intial centroids at random

⇒ Assign each object to the closter with nearest centroid.

⇒ Compute each centroid as the mean of the objects assigned to it.

⇒ Repeats previous 2 steps until no change

* The $x_1, \ldots x_N$ are data points or vector of observations

Each Observation (vector x_i) will be assigned to one and only one cluster. The C_i denotes cluster number for the ith Observation. K-means minimizes within-cluster point scatter

$$W(C) = \frac{1}{2} \sum_{K=1}^{K} \sum_{C(i)=K} \sum_{C(j)=K} \| x_i - x_j \|^2$$

$$= \sum_{K=1}^{K} N_K \sum_{C(i)K}^{K} \| x_i - m_K \|^2$$

Where

m_K is the mean vector of the Kth Cluster

N_K is the number of observations in Kth cluster.

K-Means Algorithm properties

* There are always K cluster
* There is always at least one item in each cluster
* The clusters are non hierarchical and they do not overlap

Every member of cluster is closer to

its cluster than any other cluster
because closeness does not always involve the center of clusters

K-Means Algorithm

1) The dataset is partitioned into K clusters and the data points are randomly assigned to the clusters that have roughly the same number of data points.

2) For each data point

*calculate the distance from the data point to each cluster.

*If the data point is closest to its own cluster, leave it where it is

*If the data point is not closest to its own cluster, move it into the closest clusters

Repeat the above step untill a complete pass through all data points result in no data point moving from one cluster to another

* K Means algorithm is iterative in nature. It converages however only a local minimum is obtained. It works only for numerical data. This method is easy to implement

Advantages of K-Means Algorithm.

* Efficient in computation
* Easy to Implement

Weakness

* Applicable only when mean is defined
* Need to Specify K the number of clusters in advanced.
* Trouble with noisy data & outliers
* Not suitable to discover clusters with non convex shapes

KNN :

K- nearest Neighbour is one of the Machine learning algorithms based totally on Supervised learning approach.

K-NN algorithm assumes the Similarity between the brand new case/facts and available instances and placed the brand new case into the category that is maximum similar to the to be had classes

KNN set of roles shops all of the be had facts and classifies a new statistics point based at the Similarity.

This means when new data seems then it may be effortlessly categorized into a properly suite class by using K-NN algorithm

K-NN set of roles can be used for regression as well as for classification however normally its miles used for the classification troubles

KNN is a non parametric algorithm because of this it does no longer makes any assumption on underlying data

It is also refered to as a lazy learner set of roles because it does not longer research

research from the training set immediately as a substitute it shops the dataset and at the time of class it plays an movement at the dataset

The KNN set of roles at the Schooling Section simply stones the dataset and when it gets new data then it classifies that statistics into a class that is an awfol lot similar to the a brand new data.

Example :

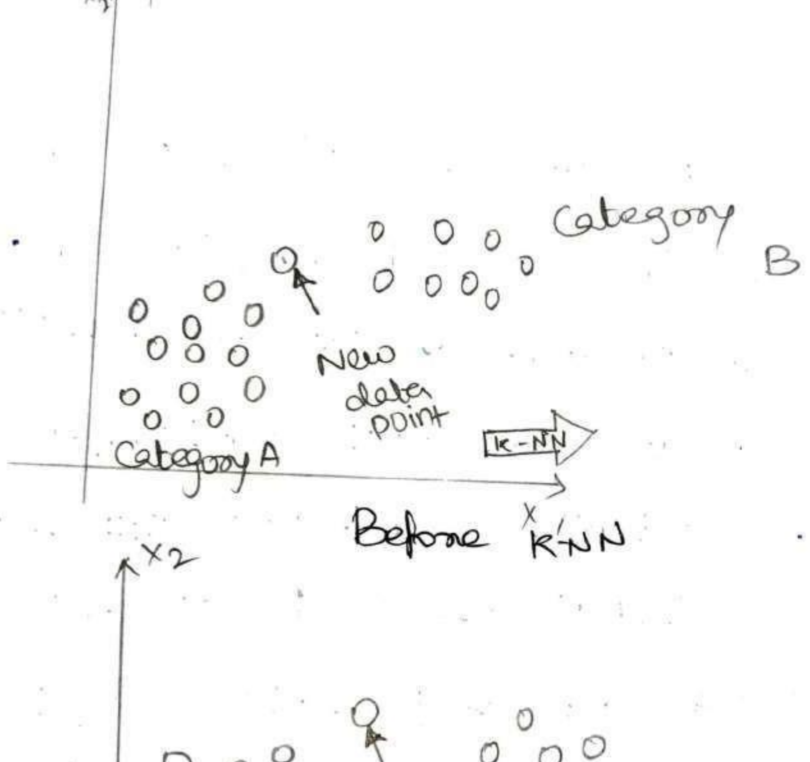
Suppose we have an picture of creature That looks much like cat and dog but we want both it is a cat or dog. So far This identity we are able to use the KNN algorithm, because it works on a similarity degree. Our KNN version will discover the similar features of the new facts set to the cats and dogs snap shots and primarily based on the most similar functions it will place it in both cat or canine class.

Why do we need KNN ?

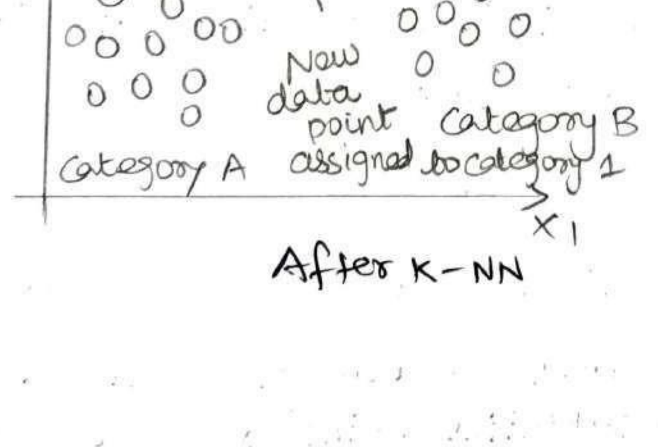
Suppose There are two categories i.e category A and category B and we have a brand new statistics point x1 and so this fact point will lie within of these classes.

To solve this sort of problem we need a K-NN set of rules.

With the help of K-NN we will without difficulty discover the category or class of a selected dataset Consider the underneath diagram.



Before K-NN



After K-NN

KNN working :

The KNN working can be explained on the basis of the below algorithm.

1 ⇒ select the wide variety K of the acquaintances

2 ⇒ Calculate the Euclidean distance of K variety of friends.

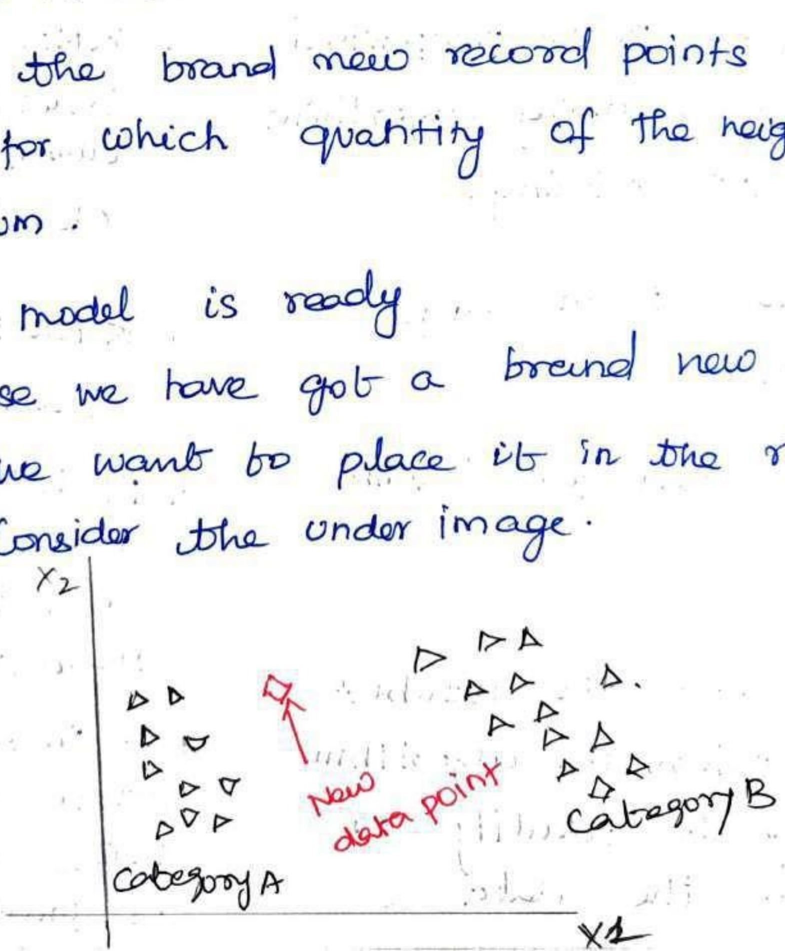
3 ⇒ Take the K nearest neighbors as according to the calculated Euclidean distance

4 ⇒ Among these ok pals, count number of the data points in each class.

5 ⇒ Assign the brand new record points to that category for which quantity of the neighbor is maximum.

6 ⇒ Our model is ready

Suppose we have got a brand new information point and we want to place it in the required category. Consider the under image.



Firstly we are able do pick the number of friends so we are able do select the OK=5.

Next we will clacukate the Euclidean distance between the fact points. The Euclidean distance is the gap between points which we have got already studied in goeometry. It may be calculated as

$$\text{Euclidean distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Difference between K means and KNN

K Means	KNN
* K Means is an unsupervised machine learning algorthim used for clustering.	* KNN is a Supervised machine learning algorithm used for Classification.
* K-means is an eager learner	* KNN is a lazy learner.
* It is used for clustering	* It is used for classification and Sometimes even for regression.
* K-means is the number of clusters the algorithm is try do identify or learn the data	* K in KNN is the number of the nearest neighbor used to classify or predict a test Sample

* K Means require unlabelled data. It gathers and groups data into K number of clusters

* KNN require labelled data and will give new data points accordingly to the K number or the closest data points.

Gaussian Mixture Models:

* Gaussian Mixture models is a soft clustering algorithm where each point probabilistically belong to all clusters. This is different than K means where each point belong to one Clusters.

* The gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mix of guassian distributions with unknown parameters.

* Gaussian mixture models consists of two parts : Mean vectors and Covariance matrices.

* A gaussian distribution is defined as a continuous probability distribution that takes a bell shaped curve. Another name of the gaussian distribution is the normal distribution

In one dimensional space the probability density function of a gaussian distribution is given by

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

Where μ is the mean and σ^2 is the variance.

* Gaussian mixture models can be used for a variety of use cases, including identifying customer segments detecting fraudulent activity and clustering images.

* GMM have variety of real world applications They are

  * Used for Signal processing
  * Used for customer Churn analysis
  * Used for language identification
  * Used in video game industry
  * Genre classification of Songs

Expectation Maximization

* In Gaussian mixture models an expectation maximization method is a powerful tool for estimating the parameters of Gaussian mixture model. This expectation is termed as E and maximization is termed M

and maximizing (M) step which computes the
maximum likelihood estimates of the parameters
by maximizing the expected likelihood found in
the E step.

* In the Expectation step, find the expected
values of the latent variables (here you
need to use the current parameter values)

* In the Maximization step first plug in the
expected values of the latent variables in the
log-likelihood of the augmented data. Then
maximize this log-likelihood to reevaluvate
the parameters.

* EM is a technique used in point estimation
Given a set of observable variables x and unknow
(latent) variables z we want to estimate parameters θ
in a model. It is a wiedly used maximization
likeli-hood estimation procedure for the statistical
models when the values of some of the
variables in the model are not observed.

* In E step, the algorithm estimates the
posterior distribution of the hidden variables Q
given the observed datas and the current parameter
settings and in the M step algorithm calculates

* Expectation is used to find the gaussian parameters which are used to represent each component of gaussian mixture models. Maximization is termed M and it is involved in determining whether new data points can be added or not.

* This algorithm used in maximum likelihood estimation where the problem involves two sets of random variables of which one x is observable and the other z is hidden.

* The goal of the algorithm is to find the parameter vector ϕ that maximizes the likelihood of the observed values of x $L(\phi|x)$

EM Algorithm :

* It is an interative method used to find maximum likelihood estimates of parameters in ~~probab~~ probabilistic models where the models depends on unobserved also called as latent variables

* EM alternate between performing an expectation (E) step which computes an expectation of the likelihood by including the latent variables as if they were observed

*The ML parameter settings with q fixed

*At the end of each iteration the lower bound on the likelihood is optimized for the given parameter setting (M-step) and the likelihood function is set to that bound E step.

* Generally EM works best when the fraction of missing information is small and the dimensionality of the data is not too large.

*EM Require many iterations and higher dimensionality can dramatically slow down The E Step.

EM is usefull for several reasons:

* conceptual simplicity
* ease of implementation

Sometimes the M-step is a constrained maximization which means that there are constraints on Valid solutions not encoded in the function itself

* Expectation maximization is an effective technique that is often used in data analysis

to manage missing data - Indeed expectation maximization overcomes some of the limitations of other techniques, such as mean Substitution or regression Substitution.

* The alternative techniques generate biased estimates and specifically underestimate the standarad errors. Expectation maximization overcome this problem.

UNIT IV NEURAL NETWORKS 9

Multilayer perceptron, activation functions, network training – gradient descent optimization – stochastic gradient descent, error backpropagation, from shallow networks to deep networks –Unit saturation (aka the vanishing gradient problem) – ReLU, hyper parameter tuning, batch normalization, regularization, dropout

4.1 Multi-layer Perceptron

Multi-Layer perceptron defines the most complex architecture of artificial neural networks. It is substantially formed from multiple layers of the perceptron.

The pictorial representation of multi-layer perceptron learning is as shown below-

MLP networks are used for supervised learning format. A typical learning algorithm for MLP networks is also called **back propagation's algorithm**.

A multilayer perceptron (MLP) is a feed forward artificial neural network that generates a set of outputs from a set of inputs. An MLP is characterized by several layers of input nodes connected as a directed graph between the input nodes connected as a directed graph between the input and output layers. MLP uses backpropagation for training the network. MLP is a deep learning method.

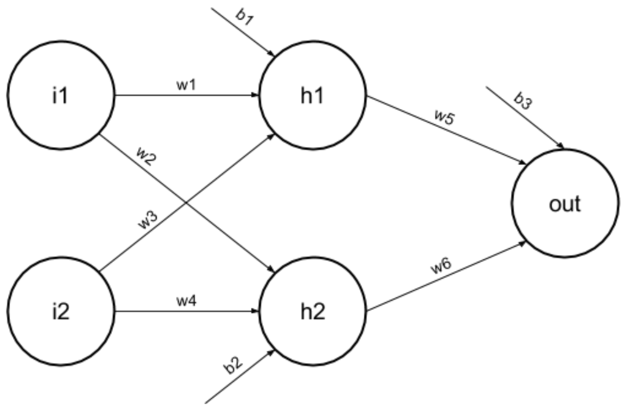
4.2 Activation Functions in Neural Networks

Elements of a Neural Network

Input Layer: This layer accepts input features. It provides information from the outside world to the network, no computation is performed at this layer, nodes here just pass on the information(features) to the hidden layer.
Hidden Layer*:* Nodes of this layer are not exposed to the outer world, they are part of the abstraction provided by any neural network. The hidden layer performs all sorts of computation on the features entered through the input layer and transfers the result to the output layer.
Output Layer: This layer bring up the information learned by the network to the outer world.
What is an activation function and why use them?
The activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it. The purpose of the activation function is to introduce non-linearity into the output of a neuron.

Explanation: We know, the neural network has neurons that work in correspondence with *weight, bias,* and their respective activation function. In a neural network, we would update the weights and biases of the neurons on the basis of the error at the output. This process is known as ***back-propagation***. Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases.
Why do we need Non-linear activation function?
A neural network without an activation function is essentially just a linear regression model. The activation function does the non-linear transformation to the input making it capable to learn and perform more complex tasks.

Mathematical proof
Suppose we have a Neural net like this :-



Elements of the diagram are as follows:
Hidden layer i.e. layer 1:
z(1) = W(1)X + b(1) a(1)

Here,

- *z(1) is the vectorized output of layer 1*
- *W(1) be the vectorized weights assigned to neurons of hidden layer i.e. w1, w2, w3 and w4*
- *X be the vectorized input features i.e. i1 and i2*
- *b is the vectorized bias assigned to neurons in hidden layer i.e. b1 and b2*
- *a(1) is the vectorized form of any linear function.*

*(**Note:** We are not considering activation function here)*

Layer 2 i.e. output layer :-
Note : *Input for layer 2 is output from layer 1*
z(2) = W(2)a(1) + b(2)

a(2) = z(2)

Calculation at Output layer
*z(2) = (W(2) * [W(1)X + b(1)]) + b(2)*

*z(2) = [W(2) * W(1)] * X + [W(2)*b(1) + b(2)]*

Let,

*[W(2) * W(1)] = W*

*[W(2)*b(1) + b(2)] = b*

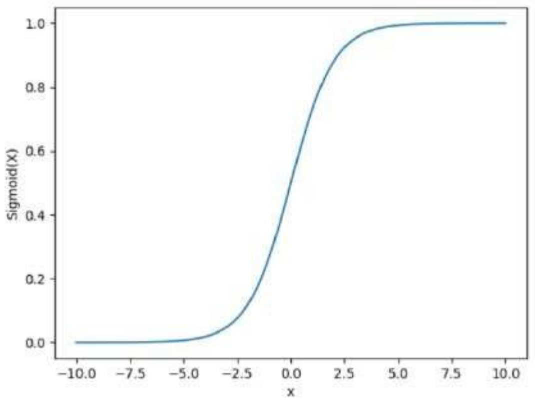
*Final output : z(2) = W*X + b*

which is again a linear function

This observation results again in a linear function even after applying a hidden layer, hence we can conclude that, doesn't matter how many hidden layer we attach in neural net, all layers will behave same way because ***the composition of two linear function is a linear function itself***. Neuron can not learn with just a linear function attached to it. A non-linear activation function will let it learn as per the difference w.r.t error. **Hence we need an activation function.**
Variants of Activation Function
Linear Function

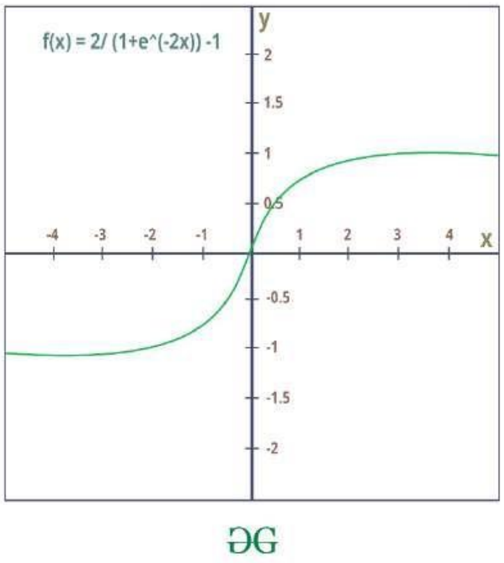
- **Equation :** Linear function has the equation similar to as of a straight line i.e. **y = x**
- No matter how many layers we have, if all are linear in nature, the final activation function of last layer is nothing but just a linear function of the input of first layer.
- **Range :** -inf to +inf
- **Uses : Linear activation function** is used at just one place i.e. output layer.
- **Issues :** If we will differentiate linear function to bring non-linearity, result will no more depend on *input "x"* and function will become constant, it won't introduce any ground-breaking behavior to our algorithm.

For example : Calculation of price of a house is a regression problem. House price may have any big/small value, so we can apply linear activation at output layer. Even in this case neural net must have any non-linear function at hidden layers.
Sigmoid Function



- It is a function which is plotted as **'S'** shaped graph.
- **Equation :** A = 1/(1 + e^{-x})
- **Nature :** Non-linear. Notice that X values lies between -2 to 2, Y values are very steep. This means, small changes in x would also bring about large changes in the value of Y.
- **Value Range :** 0 to 1
- **Uses :** Usually used in output layer of a binary classification, where result is either 0 or 1, as value for sigmoid function lies between 0 and 1 only so, result can be predicted easily to be ***1*** if value is greater than **0.5** and ***0*** otherwise.

Tanh Function

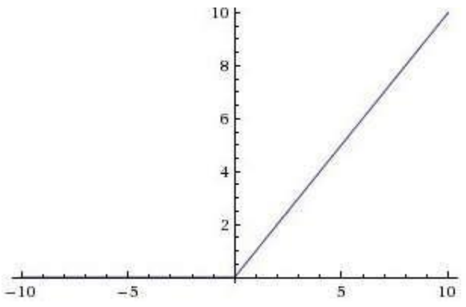


- The activation that works almost always better than sigmoid function is Tanh function also known as **Tangent Hyperbolic function**. It’s actually mathematically shifted version of the sigmoid function. Both are similar and can be derived from each other.
- **Equation :-**

$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

- **Value Range :-** -1 to +1
- **Nature :-** non-linear
- **Uses :-** Usually used in hidden layers of a neural network as it’s values lies between **-1 to 1** hence the mean for the hidden layer comes out be 0 or very close to it, hence helps in *centering the data* by bringing mean close to 0. This makes learning for the next layer much easier.

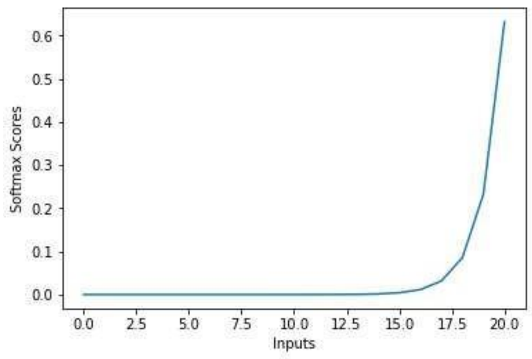
RELU Function



- It Stands for *Rectified linear unit*. It is the most widely used activation function. Chiefly implemented in *hidden layers* of Neural network.
- **Equation :-** ***A(x) = max(0,x)***. It gives an output x if x is positive and 0 otherwise.
- **Value Range :-** [0, inf)
- **Nature :-** non-linear, which means we can easily backpropagate the errors and have multiple layers of neurons being activated by the ReLU function.
- **Uses :-** ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. At a time only a few neurons are activated making the network sparse making it efficient and easy for computation.

In simple words, RELU learns *much faster* than sigmoid and Tanh function.

Softmax Function



The softmax function is also a type of sigmoid function but is handy when we are trying to handle multi- class classification problems.

- **Nature :-** non-linear
- **Uses :-** Usually used when trying to handle multiple classes. the softmax function was commonly found in the output layer of image classification problems.The softmax function would squeeze the outputs for each class between 0 and 1 and would also divide by the sum of the outputs.
- **Output:-** The softmax function is ideally used in the output layer of the classifier where we are actually trying to attain the probabilities to define the class of each input.
- The basic rule of thumb is if you really don't know what activation function to use, then simply use *RELU* as it is a general activation function in hidden layers and is used in most cases these days.
- If your output is for binary classification then, *sigmoid function* is very natural choice for output layer.
- If your output is for multi-class classification then, Softmax is very useful to predict the probabilities of each classes.

4.3. Network Training

❖ **Training: It is the process in which the network is taught to change its weight and bias.**

❖ Learning: It is the internal process of training where the artificial neural systemlearns to update/adapt the weights and biases.

Different Training /Learning procedure available in ANN are

➢ *Supervised learning*

➢ **Unsupervised learning**

➢ *Reinforced learning*

➢ **Hebbian learning**

➢ *Gradient descent learning*

➢ **Competitive learning**

➢ *Stochastic learning*

1.4.1. Requirements of Learning Laws:

- *Learning Law should lead to convergence of weights*
- **Learning or training time should be less for capturing the**

information from the trainingpairs

- *Learning should use the local information*
    - **Learning process should able to capture the complex non linear mapping availablebetween the input & output pairs**
- *Learning should able to capture as many as patterns as possible*
    - **Storage of pattern information's gathered at the time of learning should be high for thegiven network**

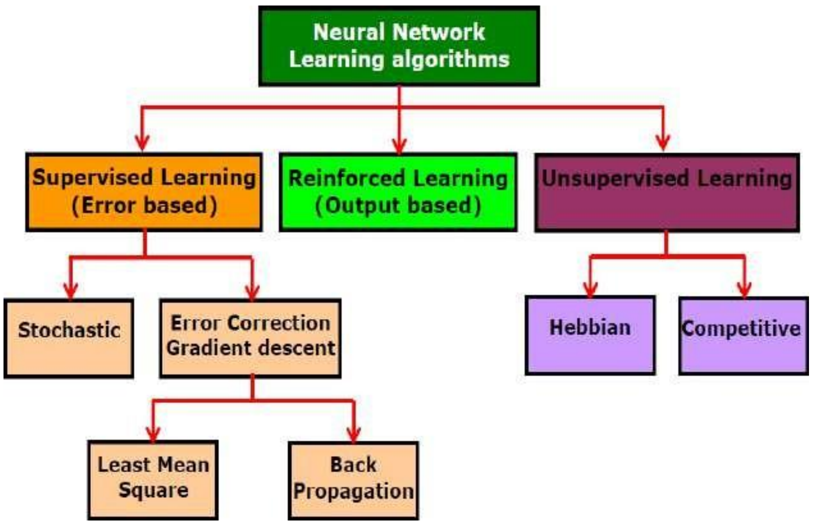

Neural Network Learning algorithms
Supervised Learning (Error based)
Reinforced Learning (Output based)
Unsupervised Learning
Stochastic
Error Correction Gradient descent
Hebbian
Competitive
Least Mean Square
Back Propagation

Figure 3: Different Training methods of ANN

Supervised learning :

Every input pattern that is used to train the network is associated with an output pattern which isthe target or the desired pattern.

A teacher is assumed to be present during the training process, when a comparison is made between the network's computed output and the correct expected output, to determine the error.The error can then be used to change network parameters, which result in an improvement in performance.

Unsupervised learning:

In this learning method the target output is not presented to the network.It is as if there is no teacher to present the desired patterns and hence the system learns of its own by discovering and adapting to structural features in the input patterns.

Reinforced learning:

In this method, a teacher though available, doesnot present the expected answer but only indicates if the computed output correct or incorrect.The information provided helps the network in the learning process.

Hebbian learning:

This rule was proposed by Hebb and is based on correlative weight adjustment.This is the oldestlearning mechanism inspired by biology.In this, the input-output pattern pairs (x_i, y_i) are associated by the weight matrix W, known as the correlation matrix.

It is computed as

$$W = \sum_{i=1}^{n} x_i y_i^T \text{ ------------------------- eq(1)}$$

Here y_i^T is the transposeof the associated output vector y_i.Numerous variants of the rule havebeen proposed.

Gradient descent learning:

This is based on the minimization of error E defined in terms of weights and activation function of the network.Also it is required that the activation function employed by the network is differentiable, as the weight update is dependent on the gradient of the error E.

Thus if Δw_{ij} is the weight update of the link connecting the i^{th} and j^{th} neuron of the two neighbouring layers, then Δw_{ij} is defined as,

$$\Delta_{ij} = \eta \frac{\partial E}{\partial wij} \text{ - ----------eq(2)}$$

Where, η is the learning rate parameter and $\frac{\partial E}{\partial wij}$ is the error gradient with reference to the weight w_{ij}.

4.4 Gradient Descent:

- Gradient Descent is a popular optimization technique in Machine Learning and Deep Learning of the learning algorithms.
- A gradient is the slope of a function.
- It measures the degree of change of a variable in response to the changes of another variable.
- Mathematically, Gradient Descent is a convex function whose output is the partial derivativeof a set of parameters of its inputs.
- The greater the gradient, the steeper the slope.Starting from an initial value, Gradient Descent is run iteratively to find the optimal values of the parameters to find the minimum possible value of the given cost function.

Types of Gradient Descent:

Typically, there are three types of Gradient Descent:

1. Batch Gradient Descent
2. Stochastic Gradient Descent
3. Mini-batch Gradient Descent

Stochastic Gradient Descent (SGD):

- The word 'stochastic' means a system or a process that is linked with a random probability.
- Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration.
- In Gradient Descent, there is a term called "batch" which denotes the totalnumber of samples from a dataset that is used for calculating the gradient for each iteration.
- In typicalGradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset.
- Although, using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, but the problem arises when our datasets gets big.
- Suppose, you have a million samples in your dataset, so if you use a typical Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima is reached. Hence, it becomes computationally very expensive to perform

4.5 Backpropagation

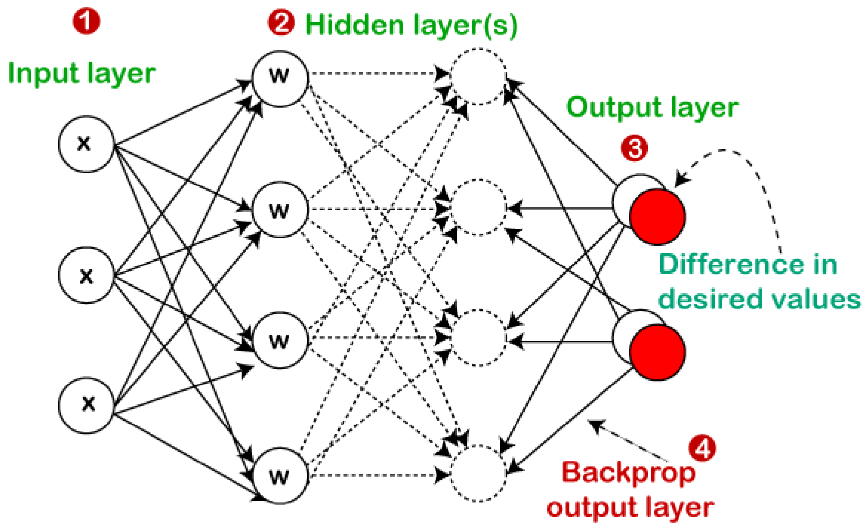
- The backpropagation consists of an input layer of neurons, an output layer, and at least one hidden layer.
- The neurons perform a weighted sum upon the input layer, which is then used by the activation function as an input, especially by the sigmoid activation function.
- It also makes use of supervised learning to teach the network.
- It constantly updates the weights of the network until the desired output is met by the network.

❖ It includes the following factors that are responsible for the training and performance of the network:

- Random (initial) values of weights.
- A number of training cycles.
- A number of hidden neurons.
- The training set.
- Teaching parameter values such as learning rate and momentum.

Working of Backpropagation

Consider the diagram given below.



1. The preconnected paths transfer the inputs **X.**
2. Then the weights **W** are randomly selected, which are used to model the input.
3. After then, the output is calculated for every individual neuron that passes from the input layer to the hidden layer and then to the output layer.
4. Lastly, the errors are evaluated in the outputs. **Error$_B$= Actual Output - Desired Output**
5. The errors are sent back to the hidden layer from the output layer for adjusting the weights to lessen the error.
6. Until the desired result is achieved, keep iterating all of the processes.

Need of Backpropagation

- Since it is fast as well as simple, it is very easy to implement.
- Apart from no of inputs, it does not encompass of any other parameter to perform tuning.

- As it does not necessitate any kind of prior knowledge, so it tends out to be more flexible.
- It is a standard method that results well.

What is a Feed Forward Network?

A feedforward neural network is an artificial neural network where the nodes never form a cycle. This kind of neural network has an input layer, hidden layers, and an output layer. It is the first and simplest type of artificial neural network.

Types of Backpropagation Networks

Two Types of Backpropagation Networks are:

- Static Back-propagation
- Recurrent Backpropagation

Static back-propagation:

It is one kind of backpropagation network which produces a mapping of a static input for static output. It is useful to solve static classification issues like optical character recognition.

Recurrent Backpropagation:

Recurrent Back propagation in data mining is fed forward until a fixed value is achieved. After that, the error is computed and propagated backward.

The main difference between both of these methods is: that the mapping is rapid in static back-propagation while it is nonstatic in recurrent backpropagation.

Best practice Backpropagation

Backpropagation in neural network can be explained with the help of "Shoe Lace" analogy

Too little tension =

- Not enough constraining and very loose

Too much tension =

- Too much constraint (overtraining)
- Taking too much time (relatively slow process)
- Higher likelihood of breaking

Pulling one lace more than other =

- Discomfort (bias)

Disadvantages of using Backpropagation

- The actual performance of backpropagation on a specific problem is dependent on the input data.
- Back propagation algorithm in data mining can be quite sensitive to noisy data
- You need to use the matrix-based approach for backpropagation instead of mini-batch.

Backpropagation Process in Deep Neural Network

Backpropagation is one of the important concepts of a neural network. Our task is to classify our data best. For this, we have to update the weights of parameter and bias, but how can we do that in a deep neural network? In the linear regression model, we use gradient descent to optimize the parameter. Similarly here we also use gradient descent algorithm using Backpropagation.

For a single training example, **Backpropagation** algorithm calculates the gradient of the **error function**. Backpropagation can be written as a function of the neural network. Backpropagation algorithms are a set of methods used to efficiently train artificial neural networks following a gradient descent approach which exploits the chain rule.

The main features of Backpropagation are the iterative, recursive and efficient method through which it calculates the updated weight to improve the network until it is not able to perform the task for which it is being trained. Derivatives of the activation function to be known at network design time is required to Backpropagation.

Now, how error function is used in Backpropagation and how Backpropagation works? Let start with an example and do it mathematically to understand how exactly updates the weight using Backpropagation.

Input values

X1=0.05
X2=0.10

Initial weight

W1=0.15 w5=0.40
W2=0.20 w6=0.45
W3=0.25 w7=0.50
W4=0.30 w8=0.55

Bias Values

b1=0.35 b2=0.60

Target Values

T1=0.01
T2=0.99

Now, we first calculate the values of H1 and H2 by a forward pass.

Forward Pass

To find the value of H1 we first multiply the input value from the weights as

$$H1=x1\times w_1+x2\times w_2+b1$$
$$H1=0.05\times0.15+0.10\times0.20+0.35$$

H1=0.3775

To calculate the final result of H1, we performed the sigmoid function as

We will calculate the value of H2 in the same way as H1

$$H2=x1\times w_3+x2\times w_4+b1$$
$$H2=0.05\times0.25+0.10\times0.30+0.35$$

H2=0.3925

To calculate the final result of H1, we performed the sigmoid function as

$$H2_{final} = \frac{1}{1+\frac{1}{e^{H2}}}$$

$$H2_{final} = \frac{1}{1+\frac{1}{e^{0.3925}}}$$

$$\mathbf{H2_{final} = 0.596884378}$$

Now, we calculate the values of y1 and y2 in the same way as we calculate the H1 and H2.

To find the value of y1, we first multiply the input value i.e., the outcome of H1 and H2 from the weights as

$$y1=H1\times w_5+H2\times w_6+b2$$
$$y1=0.593269992\times 0.40+0.596884378\times 0.45+0.60$$
$$\mathbf{y1=1.10590597}$$

To calculate the final result of y1 we performed the sigmoid function as

$$y1_{final} = \frac{1}{1+\frac{1}{e^{y1}}}$$

$$y1_{final} = \frac{1}{1+\frac{1}{e^{1.10590597}}}$$

$$\mathbf{y1_{final} = 0.75136507}$$

We will calculate the value of y2 in the same way as y1

$$y2=H1\times w_7+H2\times w_8+b2$$
$$y2=0.593269992\times 0.50+0.596884378\times 0.55+0.60$$
$$\mathbf{y2=1.2249214}$$

To calculate the final result of H1, we performed the sigmoid function as

$$y2_{final} = \frac{1}{1+\frac{1}{e^{y2}}}$$

$$y2_{final} = \frac{1}{1+\frac{1}{e^{1.2249214}}}$$

$$\mathbf{y2_{final} = 0.772928465}$$

Our target values are 0.01 and 0.99. Our y1 and y2 value is not matched with our target values T1 and T2.

Now, we will find the **total error**, which is simply the difference between the outputs from the target outputs. The total error is calculated as

$$E_{total} = \sum \frac{1}{2}(\text{target} - \text{output})^2$$

So, the total error is

$$= \frac{1}{2}(t1 - y1_{final})^2 + \frac{1}{2}(T2 - y2_{final})^2$$

$$= \frac{1}{2}(0.01 - 0.75136507)^2 + \frac{1}{2}(0.99 - 0.772928465)^2$$

$$= 0.274811084 + 0.0235600257$$

$$\mathbf{E_{total} = 0.29837111}$$

Now, we will backpropagate this error to update the weights using a backward pass.

Backward pass at the output layer

To update the weight, we calculate the error correspond to each weight with the help of a total error. The error on weight w is calculated by differentiating total error with respect to w.

$$\text{Error}_w = \frac{\partial E_{total}}{\partial w}$$

We perform backward process so first consider the last weight w5 as

$$\text{Error}_{w5} = \frac{\partial E_{total}}{\partial w5} \dots\dots\dots..(1)$$

$$E_{total} = \frac{1}{2}(T1 - y1_{final})^2 + \frac{1}{2}(T2 - y2_{final})^2 \dots\dots\dots..(2)$$

From equation two, it is clear that we cannot partially differentiate it with respect to w5 because there is no any w5. We split equation one into multiple terms so that we can easily differentiate it with respect to w5 as

$$\frac{\partial E_{total}}{\partial w5} = \frac{\partial E_{total}}{\partial y1_{final}} \times \frac{\partial y1\_final}{\partial y1} \times \frac{\partial y1}{\partial w5} \dots\dots\dots\dots\ (3)$$

Now, we calculate each term one by one to differentiate E_{total} with respect to w5 as

$$\frac{\partial E_{total}}{\partial y1_{final}} = \frac{\partial(\frac{1}{2}(T1 - y1_{final})^2 + \frac{1}{2}(T2 - y2_{final})^2)}{\partial y1_{final}}$$

$$= 2 \times \frac{1}{2} \times (T1 - y1_{final})^{2-1} \times (-1) + 0$$

$$= -(T1 - y1_{final})$$

$$= -(0.01 - 0.75136507)$$

$$\mathbf{\frac{\partial E_{total}}{\partial y1_{final}} = 0.74136507 \ldots\ldots\ldots.(4)}$$

$$y1_{final} = \frac{1}{1 + e^{-y1}} \ldots\ldots\ldots\ldots\ldots\ldots \ (5)$$

$$\frac{\partial y1_{final}}{\partial y1} = \frac{\partial(\frac{1}{1 + e^{-y1}})}{\partial y1}$$

$$= \frac{e^{-y1}}{(1 + e^{-y1})^2}$$

$$= e^{-y1} \times (y1_{final})^2 \ldots\ldots\ldots\ldots\ldots(6)$$

$$y1_{final} = \frac{1}{1 + e^{-y1}}$$

$$e^{-y1} = \frac{1 - y1_{final}}{y1_{final}} \ldots\ldots\ldots\ldots\ldots(7)$$

Putting the value of e^{-y} in equation (5)

$$= \frac{1 - y1_{final}}{y1_{final}} \times (y1_{final})^2$$

$$= y1_{final} \times (1 - y1_{final})$$

$$= 0.75136507 \times (1 - 0.75136507)$$

$$\mathbf{\frac{\partial y1_{final}}{\partial y1} = 0.186815602 \ldots\ldots\ldots.(8)}$$

$$y1 = H1_{final} \times w5 + H2_{final} \times w6 + b2 \ldots\ldots\ldots\ldots\ldots\ldots \ (9)$$

$$\frac{\partial y1}{\partial w5} = \frac{\partial(H1_{final} \times w5 + H2_{final} \times w6 + b2)}{\partial w5}$$

$$= H1_{final}$$

$$\mathbf{\frac{\partial y1}{\partial w5} = 0.596884378 \ldots\ldots\ldots(10)}$$

So, we put the values of $\frac{\partial E_{total}}{\partial y1_{final}}, \frac{\partial y1_{final}}{\partial y1}$, and $\frac{\partial y1}{\partial w5}$ in equation no (3) to find the final result.

$$\frac{\partial E_{total}}{\partial w5} = \frac{\partial E_{total}}{\partial y1_{final}} \times \frac{\partial y1_{final}}{\partial y1} \times \frac{\partial y1}{\partial w5}$$

$$= 0.74136507 \times 0.186815602 \times 0.593269992$$

$$\mathbf{Error_{w5}} = \frac{\partial \mathbf{E_{total}}}{\partial \mathbf{w5}} = \mathbf{0.0821670407} \ldots\ldots\ldots.(\mathbf{11})$$

Now, we will calculate the updated weight $w5_{new}$ with the help of the following formula

$$w5_{new} = w5 - \eta \times \frac{\partial E_{total}}{\partial w5} \quad \text{Here}, \eta = \text{learning rate} = 0.5$$

$$= 0.4 - 0.5 \times 0.0821670407$$

$$\mathbf{w5_{new}} = \mathbf{0.35891648} \ldots\ldots\ldots(\mathbf{12})$$

In the same way, we calculate $w6_{new}$,$w7_{new}$, and $w8_{new}$ and this will give us the following values

$w5_{new}$=0.35891648
$w6_{new}$=408666186
$w7_{new}$=0.511301270
$w8_{new}$=0.561370121

Backward pass at Hidden layer

Now, we will backpropagate to our hidden layer and update the weight w1, w2, w3, and w4 as we have done with w5, w6, w7, and w8 weights.

We will calculate the error at w1 as

$$\text{Error}_{w1} = \frac{\partial E_{total}}{\partial w1}$$

$$E_{total} = \frac{1}{2}(T1 - y1_{final})^2 + \frac{1}{2}(T2 - y2_{final})^2$$

From equation (2), it is clear that we cannot partially differentiate it with respect to w1 because there is no any w1. We split equation (1) into multiple terms so that we can easily differentiate it with respect to w1 as

$$\frac{\partial E_{total}}{\partial w1} = \frac{\partial E_{total}}{\partial H1_{final}} \times \frac{\partial H1\_final}{\partial H1} \times \frac{\partial H1}{\partial w1} \ldots\ldots\ldots\ldots \ (13)$$

Now, we calculate each term one by one to differentiate E_{total} with respect to w1 as

$$\frac{\partial E_{total}}{\partial H1_{final}} = \frac{\partial(\frac{1}{2}(T1 - y1_{final})^2 + \frac{1}{2}(T2 - y2_{final})^2)}{\partial H1} \ldots\ldots\ldots\ldots(14)$$

We again split this because there is no any H1^{final} term in E^{toatal} as

$$\frac{\partial E_{total}}{\partial H1_{final}} = \frac{\partial E_1}{\partial H1_{final}} + \frac{\partial E_2}{\partial H1_{final}} \ldots\ldots\ldots(15)$$

$\frac{\partial E_1}{\partial H1_{final}}$ and $\frac{\partial E_2}{\partial H1_{final}}$ will again split because in E1 and E2 there is no H1 term. Splitting is done as

$$\frac{\partial E_1}{\partial H1_{final}} = \frac{\partial E_1}{\partial y1} \times \frac{\partial y1}{\partial H1_{final}} \ldots\ldots\ldots(16)$$

$$\frac{\partial E_2}{\partial H1_{final}} = \frac{\partial E_2}{\partial y2} \times \frac{\partial y2}{\partial H1\_final} \ldots\ldots\ldots(17)$$

We again Split both $\frac{\partial E_1}{\partial y1}$ and $\frac{\partial E_2}{\partial y2}$ because there is no any y1 and y2 term in E1 and E2. We split it as

$$\frac{\partial E_1}{\partial y1} = \frac{\partial E_1}{\partial y1_{fianl}} \times \frac{\partial y1_{final}}{\partial y1} \ldots\ldots\ldots(18)$$

$$\frac{\partial E_2}{\partial y2} = \frac{\partial E_2}{\partial y2_{fianl}} \times \frac{\partial y2_{final}}{\partial y2} \ldots\ldots\ldots(19)$$

Now, we find the value of $\frac{\partial E_1}{\partial y1}$ and $\frac{\partial E_2}{\partial y2}$ by putting values in equation (18) and (19) as

From equation (18)

$$\frac{\partial E_1}{\partial y1} = \frac{\partial E_1}{\partial y1_{fianl}} \times \frac{\partial y1_{final}}{\partial y1}$$

$$= \frac{\partial(\frac{1}{2}(T1 - y1_{final})^2)}{\partial y1_{final}} \times \frac{\partial y1_{final}}{\partial y1}$$

$$= 2 \times \frac{1}{2}(T1 - y1_{final}) \times (-1) \times \frac{\partial y1_{final}}{\partial y1}$$

From equation (8)

$$= 2 \times \frac{1}{2}(0.01 - 0.75136507) \times (-1) \times 0.186815602$$

$$\mathbf{\frac{\partial E_1}{\partial y1} = 0.138498562} \ldots\ldots\ldots(20)$$

From equation (19)

$$\frac{\partial E_2}{\partial y2} = \frac{\partial E_2}{\partial y2_{\text{fianl}}} \times \frac{\partial y2_{\text{final}}}{\partial y2}$$

$$= \frac{\partial(\frac{1}{2}(T2 - y2_{\text{final}})^2)}{\partial y2_{\text{final}}} \times \frac{\partial y2_{\text{final}}}{\partial y2}$$

$$= 2 \times \frac{1}{2}(T2 - y2_{\text{final}}) \times (-1) \times \frac{\partial y2_{\text{final}}}{\partial y2} \dots\dots\dots (21)$$

$$y2_{\text{final}} = \frac{1}{1 + e^{-y2}} \dots\dots\dots\dots\dots\dots\ (22)$$

$$\frac{\partial y2_{\text{fianl}}}{\partial y2} = \frac{\partial(\frac{1}{1 + e^{-y2}})}{\partial y2}$$

$$= \frac{e^{-y2}}{(1 + e^{-y2})^2}$$

$$= e^{-y2} \times (y2_{\text{final}})^2 \dots\dots\dots\dots\dots (23)$$

$$y2_{\text{final}} = \frac{1}{1 + e^{-y2}}$$

$$e^{-y2} = \frac{1 - y2_{\text{final}}}{y2_{\text{final}}} \dots\dots\dots\dots\dots (24)$$

Putting the value of e^{-y2} in equation (23)

$$= \frac{1 - y2_{\text{final}}}{y2_{\text{final}}} \times (y2_{\text{final}})^2$$

$$= y2_{\text{final}} \times (1 - y2_{\text{final}})$$

$$= 0.772928465 \times (1 - 0.772928465)$$

$$\mathbf{\frac{\partial y2_{fianl}}{\partial y2} = 0.175510053 \dots\dots\dots. (25)}$$

From equation (21)

$$= 2 \times \frac{1}{2}(0.99 - 0.772928465) \times (-1) \times\ 0.175510053$$

$$\mathbf{\frac{\partial E_1}{\partial y1} = -0.0380982366126414 \dots\dots\dots (26)}$$

Now from equation (16) and (17)

$$\frac{\partial E_1}{\partial H1_{final}} = \frac{\partial E_1}{\partial y1} \times \frac{\partial y1}{\partial H1_{final}}$$

$$= 0.138498562 \times \frac{\partial (H1_{final} \times w_5 + H2_{final} \times w_6 + b2)}{\partial H1_{final}}$$

$$= 0.138498562 \times \frac{\partial (H1_{final} \times w_5 + H2_{final} \times w_6 + b2)}{\partial H1_{final}}$$

$$= 0.138498562 \times w5$$

$$= 0.138498562 \times 0.40$$

$$\boldsymbol{\frac{\partial E_1}{\partial H1_{final}} = 0.0553994248 \dots \dots \dots (27)}$$

$$\frac{\partial E_2}{\partial H1_{final}} = \frac{\partial E_2}{\partial y2} \times \frac{\partial y2}{\partial H1\_final}$$

$$= -0.0380982366126414 \times \frac{\partial (H1_{final} \times w_7 + H2_{final} \times w_8 + b2)}{\partial H1_{final}}$$

$$= -0.0380982366126414 \times w7$$

$$= -0.0380982366126414 \times 0.50$$

$$\boldsymbol{\frac{\partial E_2}{\partial H1_{final}} = -0.0190491183063207 \dots \dots \dots (28)}$$

Put the value of $\frac{\partial E_1}{\partial H1_{final}}$ and $\frac{\partial E_2}{\partial H1_{final}}$ in equation (15) as

$$\frac{\partial E_{total}}{\partial H1_{final}} = \frac{\partial E_1}{\partial H1_{final}} + \frac{\partial E_2}{\partial H1_{final}}$$

$$= 0.0553994248 + (-0.0190491183063207)$$

$$\boldsymbol{\frac{\partial E_{total}}{\partial H1_{final}} = 0.0364908241736793 \dots \dots \dots (29)}$$

We have $\frac{\partial E_{total}}{\partial H1_{final}}$, we need to figure out $\frac{\partial H1\_final}{\partial H1}$, $\frac{\partial H1}{\partial w1}$ as

$$\frac{\partial \mathrm{H1_{final}}}{\partial \mathrm{H1}} = \frac{\partial(\frac{1}{1+e^{-H1}})}{\partial \mathrm{H1}}$$

$$= \frac{e^{-H1}}{(1+e^{-H1})^2}$$

$$e^{-H1} \times (\mathrm{H1_{final}})^2 \ldots\ldots\ldots(30)$$

$$\mathrm{H1_{final}} = \frac{1}{1+e^{-H1}}$$

$$e^{-H1} = \frac{1-\mathrm{H1_{final}}}{\mathrm{H1_{final}}} \ldots\ldots\ldots\ldots\ldots(31)$$

Putting the value of e^{-H1} in equation (30)

$$= \frac{1-\mathrm{H1_{final}}}{\mathrm{H1_{final}}} \times (\mathrm{H1_{final}})^2$$

$$= \mathrm{H1_{final}} \times (1-\mathrm{H1_{final}})$$

$$= 0.593269992 \times (1 - 0.593269992)$$

$$\mathbf{\frac{\partial H1_{final}}{\partial H1} = 0.2413007085923199}$$

We calculate the partial derivative of the total net input to H1 with respect to w1 the same as we did for the output neuron:

$$\mathrm{H1} = \mathrm{H1_{final}} \times \mathrm{w5} + \mathrm{H2_{final}} \times \mathrm{w6} + \mathrm{b2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (32)$$

$$\frac{\partial \mathrm{y1}}{\partial \mathrm{w1}} = \frac{\partial(\mathrm{x1} \times \mathrm{w1} + \mathrm{x2} \times \mathrm{w3} + \mathrm{b1} \times 1)}{\partial \mathrm{w1}}$$

$$= \mathrm{x1}$$

$$\mathbf{\frac{\partial H1}{\partial w1} = 0.05} \ldots\ldots\ldots\mathbf{(33)}$$

So, we put the values of $\frac{\partial \mathrm{E_{total}}}{\partial \mathrm{H1_{final}}}$, $\frac{\partial \mathrm{H1_{final}}}{\partial \mathrm{H1}}$, and $\frac{\partial \mathrm{H1}}{\partial \mathrm{w1}}$ in equation (13) to find the final result.

$$\frac{\partial \mathrm{E_{total}}}{\partial \mathrm{w1}} = \frac{\partial \mathrm{E_{total}}}{\partial \mathrm{H1_{final}}} \times \frac{\partial \mathrm{H1_{final}}}{\partial \mathrm{H1}} \times \frac{\partial \mathrm{H1}}{\partial \mathrm{w1}}$$

$$= 0.0364908241736793 \times 0.2413007085923199 \times 0.05$$

$$\mathbf{Error_{w1} = \frac{\partial E_{total}}{\partial w1} = 0.000438568} \ldots\ldots\ldots.\mathbf{(34)}$$

Now, we will calculate the updated weight $\mathrm{w1_{new}}$ with the help of the following formula

$$w1_{new} = w1 - \eta \times \frac{\partial E_{total}}{\partial w1} \text{ Here } \eta = \text{learning rate} = 0.5$$

$$= 0.15 - 0.5 \times 0.000438568$$

$$\mathbf{w1_{new} = 0.149780716} \ldots\ldots\ldots(35)$$

In the same way, we calculate $w2_{new}$,$w3_{new}$, and w4 and this will give us the following values

$w1_{new}$=0.149780716
$w2_{new}$=0.19956143
$w3_{new}$=0.24975114
$w4_{new}$=0.29950229

We have updated all the weights. We found the error 0.298371109 on the network when we fed forward the 0.05 and 0.1 inputs. In the first round of Backpropagation, the total error is down to 0.291027924. After repeating this process 10,000, the total error is down to 0.0000351085. At this point, the outputs neurons generate 0.159121960 and 0.984065734 i.e., nearby our target value when we feed forward the 0.05 and 0.1.

2.5.1 Difference Between a Shallow Net & Deep Learning Net:

Sl.No	Shallow Net's	Deep Learning Net's
1	One Hidden layer(or very less no. ofHidden Layers)	Deep Net's has many layers of Hiddenlayers with more no. of neurons in each layers
2	Takes input only as VECTORS	DL can have raw data like image, textas inputs
3	Shallow net's needs more parametersto have better fit	DL can fit functions better with lessparameters than a shallow network
4	Shallow networks with one Hidden layer (same no of neurons as DL) cannot place complex functions overthe input space	DL can compactly express highly complex functions over input space
5	The number of units in a shallow network grows exponentially withtask complexity.	DL don't need to increase it size(neurons) for complex problems

6	Shallow network is more difficult to train with our current algorithms (e.g.it has issues of local minima etc)	Training in DL is easy and no issue oflocal minima in DL

4.6 The Vanishing Gradient Problem

The Problem, Its Causes, Its Significance, and Its Solutions

The problem:

As more layers using certain activation functions are added to neural networks, the gradients of the loss function approaches zero, making the network hard to train.

Why:

Certain activation functions, like the sigmoid function, squishes a large input space into a small input space between 0 and 1. Therefore, a large change in the input of the sigmoid function will cause a small change in the output. Hence, the derivative becomes small.

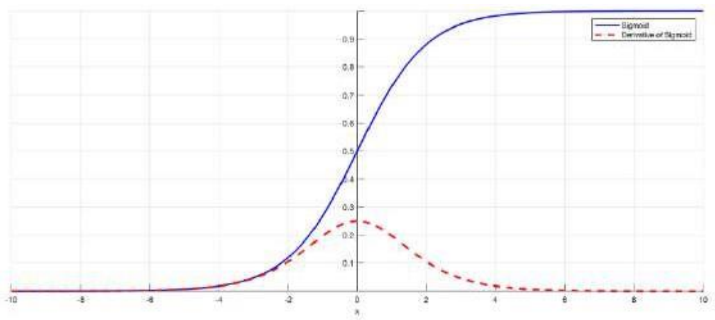


Image 1: The sigmoid function and its derivative

As an example, Image 1 is the sigmoid function and its derivative. Note how when the inputs of the sigmoid function becomes larger or smaller (when |x| becomes bigger), the derivative becomes close to zero.

Why it's significant:

For shallow network with only a few layers that use these activations, this isn't a big problem. However, when more layers are used, it can cause the gradient to be too small for training to work effectively.

Gradients of neural networks are found using backpropagation. Simply put, backpropagation finds the derivatives of the network by moving layer by layer from the final layer to the initial one. By the chain rule, the derivatives of each layer are multiplied down the network (from the final layer to the initial) to compute the derivatives of the initial layers.

However, when *n* hidden layers use an activation like the sigmoid function, *n* small derivatives are multiplied together. Thus, the gradient decreases exponentially as we propagate down to the initial layers.

A small gradient means that the weights and biases of the initial layers will not be updated effectively with each training session. Since these initial layers are often crucial to recognizing the core elements of the input data, it can lead to overall inaccuracy of the whole network.

Solutions:

The simplest solution is to use other activation functions, such as ReLU, which doesn't cause a small derivative.

Residual networks are another solution, as they provide residual connections straight to earlier layers. As seen in Image 2, the residual connection directly adds the value at the beginning of the block, **x**, to the end of the block (F(x)+x). This residual connection doesn't go through activation functions that "squashes" the derivatives, resulting in a higher overall derivative of the block.

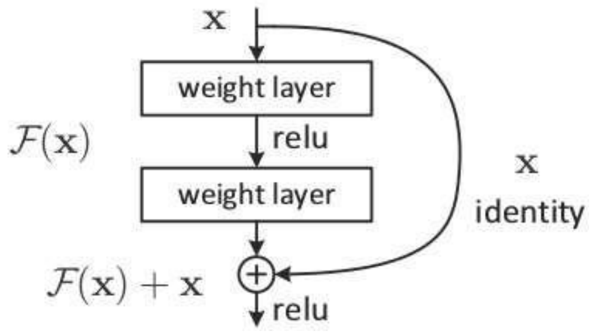

x
weight layer
relu
$\mathcal{F}(\mathbf{x})$
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Image 2: A residual block

Finally, batch normalization layers can also resolve the issue. As stated before, the problem arises when a large input space is mapped to a small one, causing the derivatives to disappear. In Image 1, this is most clearly seen at when |x| is big. Batch normalization reduces this problem by simply normalizing the input so |x| doesn't reach the outer edges of the sigmoid function. As seen in Image 3, it normalizes the input so that most of it falls in the green region, where the derivative isn't too small.

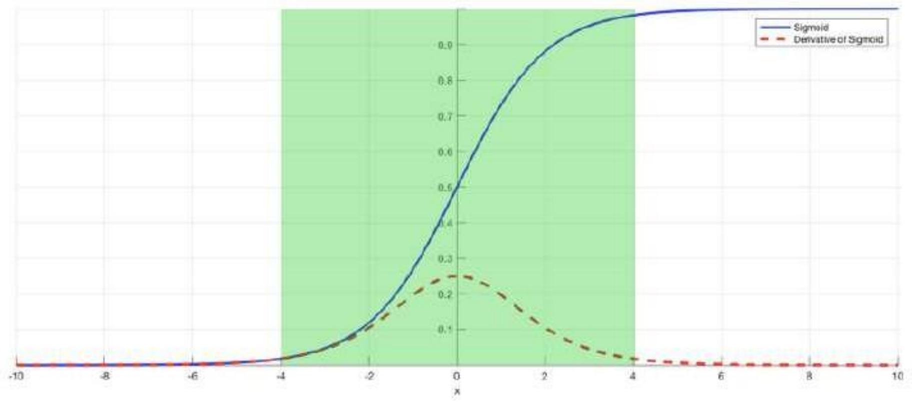


Image 3: Sigmoid function with restricted inputs

4.7 Hyperparameters in Machine Learning

Hyperparameters in Machine learning are those parameters that are explicitly defined by the user to control the learning process. These hyperparameters are used to improve the learning of the model, and their values are set before starting the learning process of the model.

- Here the prefix "hyper" suggests that the parameters are top-level parameters that are used in controlling the learning process.
- The value of the Hyperparameter is selected and set by the machine learning engineer before the learning algorithm begins training the model.
- **Hence, these are external to the model, and their values cannot be changed during the training process**.

Some examples of Hyperparameters in Machine Learning

- The k in kNN or K-Nearest Neighbour algorithm
- Learning rate for training a neural network
- Train-test split ratio
- Batch Size
- Number of Epochs
- Branches in Decision Tree
- Number of clusters in Clustering Algorithm

Model Parameters:

Model parameters are configuration variables that are internal to the model, and a model learns them on its own. For example**, W Weights or Coefficients of independent variables in the Linear regression model.** or **Weights or Coefficients of independent variables in SVM, weight, and biases of a neural network, cluster centroid in clustering.** Some key points for model parameters are as follows:

- They are used by the model for making predictions.
- They are learned by the model from the data itself
- These are usually not set manually.
- These are the part of the model and key to a machine learning Algorithm.

Model Hyperparameters:

Hyperparameters are those parameters that are explicitly defined by the user to control the learning process. Some key points for model parameters are as follows:

- These are usually defined manually by the machine learning engineer.
- One cannot know the exact best value for hyperparameters for the given problem. The best value can be determined either by the rule of thumb or by trial and error.
- Some examples of Hyperparameters are **the learning rate for training a neural network, K in the KNN algorithm,**

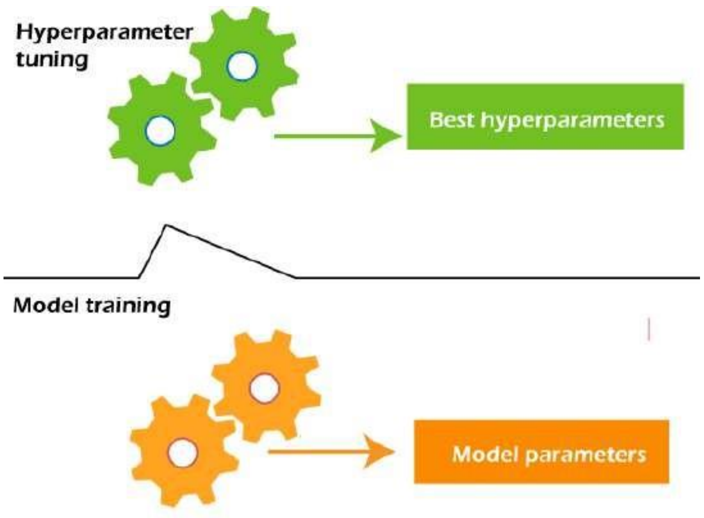
Categories of Hyperparameters

Broadly hyperparameters can be divided into two categories, which are given below:

1. **Hyperparameter for Optimization**
2. **Hyperparameter for Specific Models**

Hyperparameter for Optimization

The process of selecting the best hyperparameters to use is known as hyperparameter tuning, and the tuning process is also known as hyperparameter optimization. Optimization parameters are used for optimizing the model.



Some of the popular optimization parameters are given below:

- **Learning Rate:** The learning rate is the hyperparameter in optimization algorithms that controls how much the model needs to change in response to the estimated error for each time when the model's weights are updated. It is one of the crucial parameters while building a neural network, and also it determines the frequency of cross-checking with model parameters. Selecting the optimized learning rate is a challenging task because if the learning rate is very less, then it may slow down the training process. On the other hand, if the learning rate is too large, then it may not optimize the model properly.

- **Batch Size:** To enhance the speed of the learning process, the training set is divided into different subsets, which are known as a batch. **Number of Epochs:** An epoch can be defined as the complete cycle for training the machine learning model. Epoch represents an iterative learning process. The number of epochs varies from model to model, and various models are created with more than one epoch. To determine the right number of epochs, a validation error is taken into account. The number of epochs is increased until there is a reduction in a validation error. If there is no improvement in reduction error for the consecutive epochs, then it indicates to stop increasing the number of epochs.

Hyperparameter for Specific Models

Hyperparameters that are involved in the structure of the model are known as hyperparameters for specific models. These are given below:

- **A number of Hidden Units:** Hidden units are part of neural networks, which refer to the components comprising the layers of processors between input and output units in a neural network.

It is important to specify the number of hidden units hyperparameter for the neural network. It should be between the size of the input layer and the size of the output layer. More specifically, the number of hidden units should be 2/3 of the size of the input layer, plus the size of the output layer.

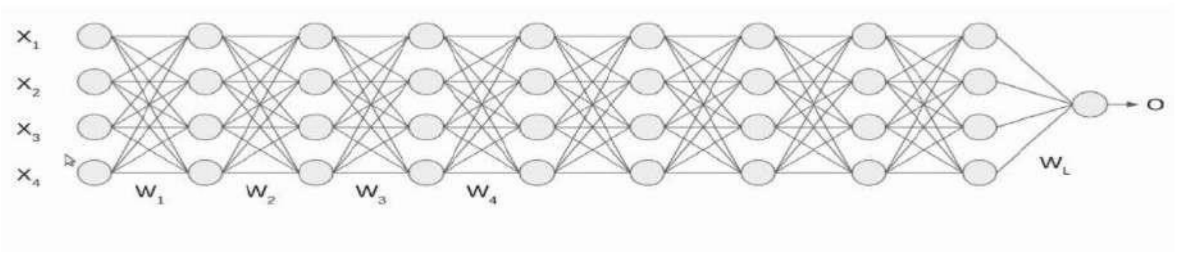
For complex functions, it is necessary to specify the number of hidden units, but it should not overfit the model.

- **Number of Layers:** A neural network is made up of vertically arranged components, which are called layers. There are mainly **input layers, hidden layers, and output layers**. A 3-layered neural network gives a better performance than a 2-layered network. For a Convolutional Neural network, a greater number of layers make a better model.

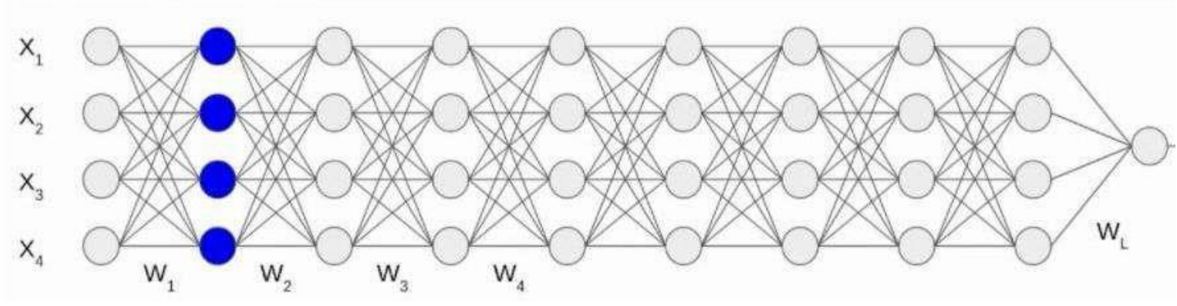
4.8 Batch Normalization:

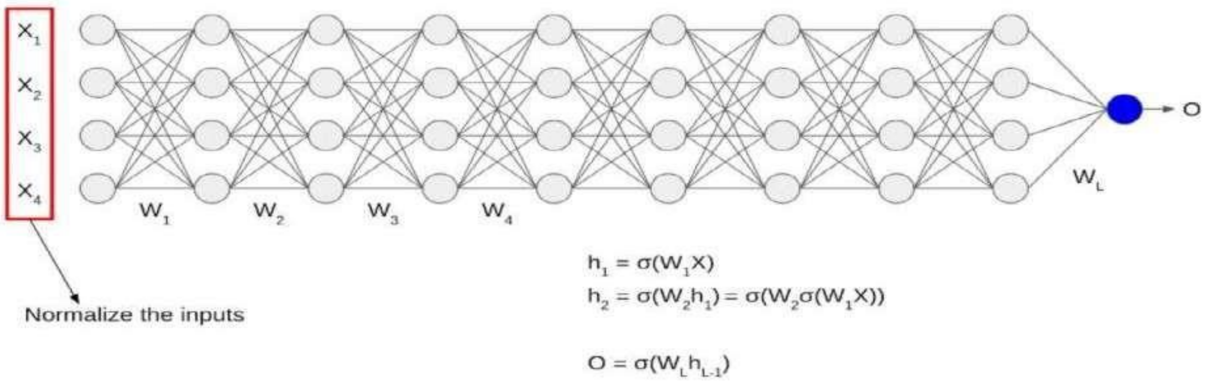
- ❖ It is a method of adaptive reparameterization, motivated by the difficulty of training very deep models.In Deep networks, the weights are updated for each layer.
- ❖ So the output will no longer be on the same scale as the input (even though input is normalized).
- ❖ Normalization - is a data pre-processing tool used to bring the numerical data toa common scale without distorting its shape.
- ❖ when we input the data to a machine or deep learning algorithm we tend to change the values to a balanced scale because, we ensure thatour model can generalize appropriately.(Normalization is used to bring the input into a balanced scale/ Range).

Let's understand this through an example, we have a deep neural network as shown in the following image.



Initially, our inputs X1, X2, X3, X4 are in normalized form as they are coming from the pre-processing stage. the input passes through the first layer, it transforms, as a sigmoid function applied over the dot product of
X and the weight matrix W.





$$h_1 = \sigma(W_1 X)$$
$$h_2 = \sigma(W_2 h_1) = \sigma(W_2 \sigma(W_1 X))$$
$$O = \sigma(W_L h_{L-1})$$

Image Source: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-batch-normalization/

- Even though the input X was normalized but the output is no longer on the same scale.
- The data passes through multiple layers of network with multiple times(sigmoidal) activation functions are applied, which leads to an internal co-variate shift in the data.
- This motivates us to move towards Batch Normalization
- Normalization is the process of altering the input data to have mean as zero and standard deviationvalue as one.

Procedure to do Batch Normalization:

(1) Consider the batch input from layer h, for this layer we need to calculate the mean of this hiddenactivation.After calculating the mean the next step is to calculate the standard deviation of the hidden activations.

(2) Now we normalize the hidden activations using these Mean & Standard Deviation values. To dothis, we subtract the mean from each input and divide the whole value with the sum of standard deviation and the smoothing term (ε).

(3) As the final stage, the re-scaling and offsetting of the input is performed. Here two components of the BN algorithm is used, γ(gamma) and β (beta). These parameters are used for re-scaling (γ) andshifting(β) the vector contains values from the previous operations.
These two parameters are learnable parameters, Hence during the training of neural network,the optimal values of γ and β are obtained and used. Hence we get the accurate normalization of eachbatch.

4.9 Regularization

Definition: - "any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error."

- In the context of deep learning, most regularization strategies are based onregularizing estimators.
- Regularization of an estimator works by trading increased bias for reducedvariance.<u>An effective regularizer is one that makes a profitable trade, reducing variancesignificantly while not overly increasing the bias</u>.
- Many regularization approaches are based on limiting the capacity of models, such as neural networks, linear regression, or logistic regression, by adding a parameter norm penalty Ω(θ) to the objective function J. We denote the regularized objective function by $\tilde{J}$

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta)$$

where α ∈ [0, ∞) is a hyperparameter that weights the relative contribution of the normpenalty term, Ω, relative to the standard objective function J. Setting α to 0 results in no regularization. Larger values of α correspond to more regularization.

The parameter norm penalty Ω that penalizes only the weights of the affine transformation at each layer and leaves the biases unregularized.

L2 Regularization

One of the simplest and most common kind of parameter norm penalty is L2 parameter & it’s also called commonly as weight decay. This regularization strategy drives the weights closerto the origin by adding a regularization term

$\Omega(\theta) = \frac{1}{2} || w || \frac{1}{2}$.

L2regularization is also known as ridge regression or Tikhonov regularization. To simplify, weassume no bias parameter, so θ is just w. Such a model has the following total objective function.

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \frac{\alpha}{2} \boldsymbol{w}^\top \boldsymbol{w} + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

with the corresponding parameter gradient

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

To take a single gradient step to update the weights, we perform this update

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \epsilon \left(\alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) \right).$$

Written another way, the update is

$$\boldsymbol{w} \leftarrow (1 - \epsilon\alpha) \boldsymbol{w} - \epsilon \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

We can see that the addition of the weight decay term has modified the learning rule to multiplicatively shrink the weight vector by a constant factor on each step, just before performing the usual gradient update. This describes what happens in a single step.The approximation ^J is Given by

$$\hat{J}(\boldsymbol{\theta}) = J(\boldsymbol{w}^*) + \frac{1}{2} (\boldsymbol{w} - \boldsymbol{w}^*)^\top \boldsymbol{H} (\boldsymbol{w} - \boldsymbol{w}^*),$$

[illegible]

Where H is the Hessian matrix of J with respect to w evaluated at w∗.

The minimum of ˆJ occurs where its gradient ∇wˆJ(w) = H(w − w∗) is equal to '0'To

study the eff ect of weight decay,

$$\alpha\tilde{w} + H(\tilde{w} - w^*) = 0$$
$$(H + \alpha I)\tilde{w} = Hw^*$$
$$\tilde{w} = (H + \alpha I)^{-1} Hw^*$$

As α approaches 0, the regularized solution ˜w approaches w*. But what happens as α grows? Because H is real and symmetric, we can decompose it into a diagonal matrix Λ and an orthonormal basis of eigenvectors, Q, such that H = $Q\Lambda Q^T$. Applying Decomposition to theabove equation, We Obtain

$$\begin{aligned}\tilde{w} &= (Q\Lambda Q^\top + \alpha I)^{-1} Q\Lambda Q^\top w^* \\ &= \left[Q(\Lambda + \alpha I)Q^\top\right]^{-1} Q\Lambda Q^\top w^* \\ &= Q(\Lambda + \alpha I)^{-1}\Lambda Q^\top w^*.\end{aligned}$$

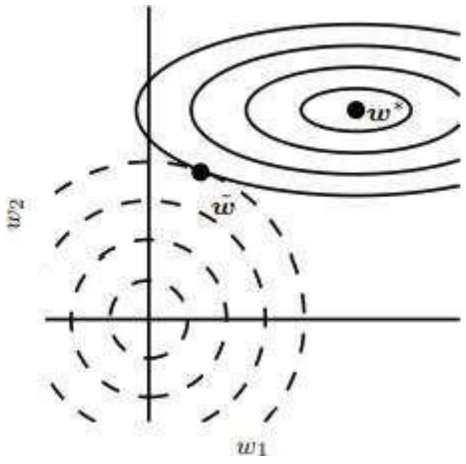


Figure 2: Weight updation effect

The solid ellipses represent contours of equal value of the unregularized objective. The dotted circles represent contours of equal value of the L 2 regularizer. At the point w˜, these competing objectives reach an equilibrium. In the first dimension, the eigenvalue of the Hessian of J is small. The objective function does not increase much when moving horizontally away from w∗ . Because the objective function does not express a strong preference along this direction, the regularizer has astrong effect on this axis. The regularizer pulls w1 close to zero. In the second dimension, the objective function is very sensitive to movements away from w∗ . The corresponding eigenvalue is large, indicating high curvature. As a result, weight decay affects the position of w2 relatively little.

L1 Regularization

While L2 weight decay is the most common form of weight decay, there are other ways to penalize the size of the model parameters. Another option is to use L1 regularization.

- L1 regularization on the model parameter w is defined as the sum of absolute values of theindividual parameters.

$$\Omega(\boldsymbol{\theta}) = ||\boldsymbol{w}||_1 = \sum_i |w_i|,$$

L1 weight decay controls the strength of the regularization by scaling the penalty Ω using a positive hyperparameter α. Thus, the regularized objective function J˜(w; X, y) is given by

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha||\boldsymbol{w}||_1 + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

with the corresponding gradient as

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \text{sign}(\boldsymbol{w}) + \nabla_{\boldsymbol{w}} J(\boldsymbol{X}, \boldsymbol{y}; \boldsymbol{w}), \quad \longrightarrow \quad \text{Eq-1}$$

By inspecting equation 1, we can see immediately that the effect of L 1 regularization is quite different from that of L 2 regularization. Specifically, we can see that the regularization contribution to the gradient no longer scales linearly with each wi ; instead it is a constant factorwith a sign equal to sign(wi).

Quadratic approximation of the L 1 regularized objective function decomposes into a sum over the parameters

$$\hat{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{w}^*; \boldsymbol{X}, \boldsymbol{y}) + \sum_i \left[\frac{1}{2} H_{i,i}(\boldsymbol{w}_i - \boldsymbol{w}_i^*)^2 + \alpha|w_i|\right].$$

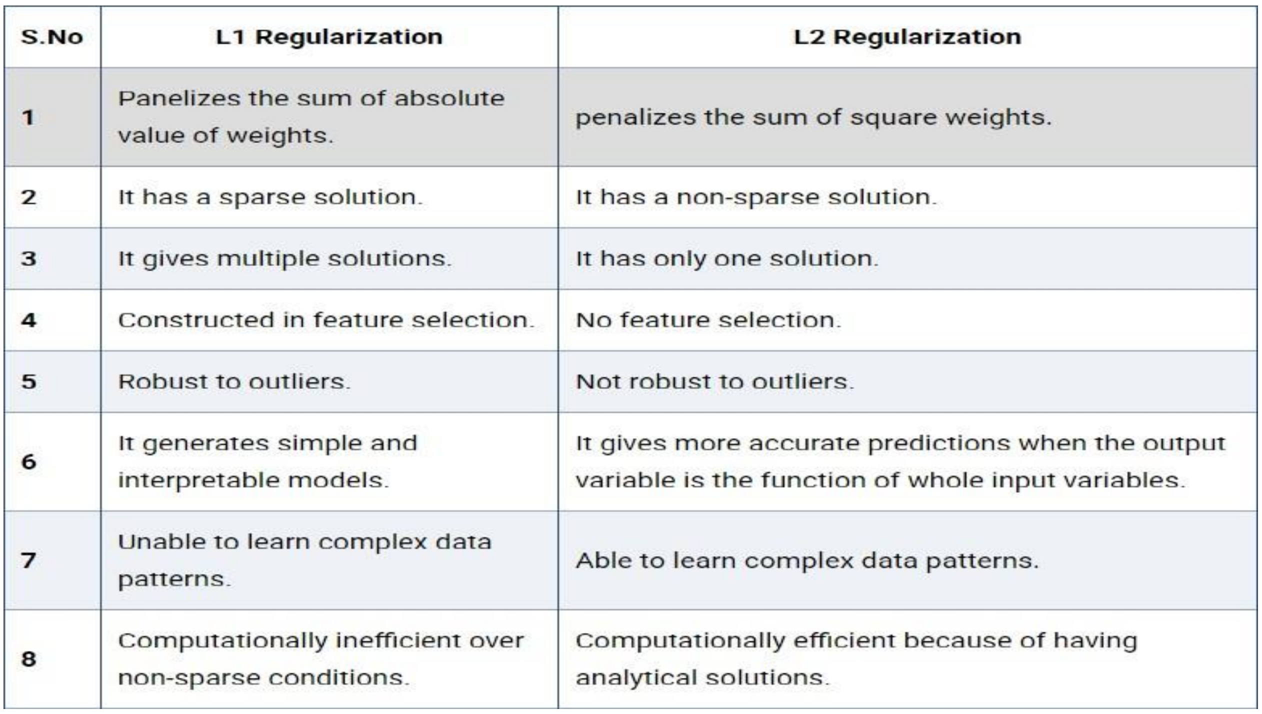
The problem of minimizing this approximate cost function has an analytical solution with the following form:

$$w_i = \text{sign}(w_i^*) \max\left\{|w_i^*| - \frac{\alpha}{H_{i,i}}, 0\right\}.$$

Consider the situation where w ∗ i > 0 for all i. There are two possible outcomes:

1. The case where $w_i^* \leq \frac{\alpha}{H_{i,i}}$. Here the optimal value of w_i under the regularized objective is simply $w_i = 0$. This occurs because the contribution of $J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})$ to the regularized objective $\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})$ is overwhelmed—in direction i—by the L^1 regularization, which pushes the value of w_i to zero.
2. The case where $w_i^* > \frac{\alpha}{H_{i,i}}$. In this case, the regularization does not move the optimal value of w_i to zero but instead just shifts it in that direction by a distance equal to $\frac{\alpha}{H_{i,i}}$.

Difference between L1 & L2 Parameter Regularization



S.No	L1 Regularization	L2 Regularization
1	Panelizes the sum of absolute value of weights.	penalizes the sum of square weights.
2	It has a sparse solution.	It has a non-sparse solution.
3	It gives multiple solutions.	It has only one solution.
4	Constructed in feature selection.	No feature selection.
5	Robust to outliers.	Not robust to outliers.
6	It generates simple and interpretable models.	It gives more accurate predictions when the output variable is the function of whole input variables.
7	Unable to learn complex data patterns.	Able to learn complex data patterns.
8	Computationally inefficient over non-sparse conditions.	Computationally efficient because of having analytical solutions.

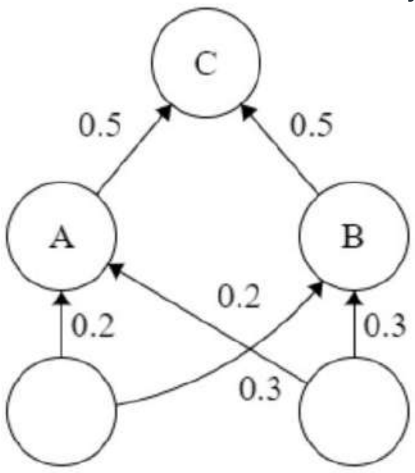
Difference between Normalization and Standardization

Normalization	**Standardization**
This technique uses minimum and max values for scaling of model.	This technique uses mean and standard deviation for scaling of model.
It is helpful when features are of different scales.	It is helpful when the mean of a variable is set to 0 and the standard deviation is set to 1.
Scales values ranges between [0, 1] or [-1, 1].	Scale values are not restricted to a specific range.
It got affected by outliers.	It is comparatively less affected by outliers.
Scikit-Learn provides a transformer called MinMaxScaler for Normalization.	Scikit-Learn provides a transformer called StandardScaler for Normalization.
It is also called Scaling normalization.	It is known as Z-score normalization.
It is useful when feature distribution is unknown.	It is useful when feature distribution is normal.

4.10 Dropout in Neural Networks

A Neural Network (NN) is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Since such a network is created artificially in machines, we refer to that as Artificial Neural Networks (ANN).

Problem: When a fully-connected layer has a large number of neurons, co-adaptation is more likely to happen. Co-adaptation refers to when multiple neurons in a layer extract the same, or very similar, hidden features from the input data. This can happen when the connection weights for two different neurons are nearly identical.

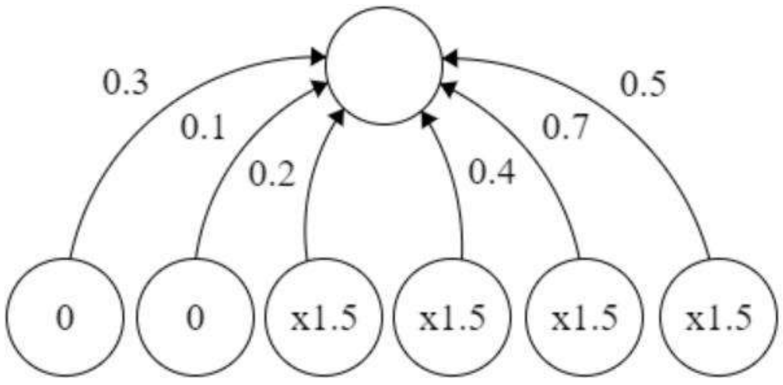


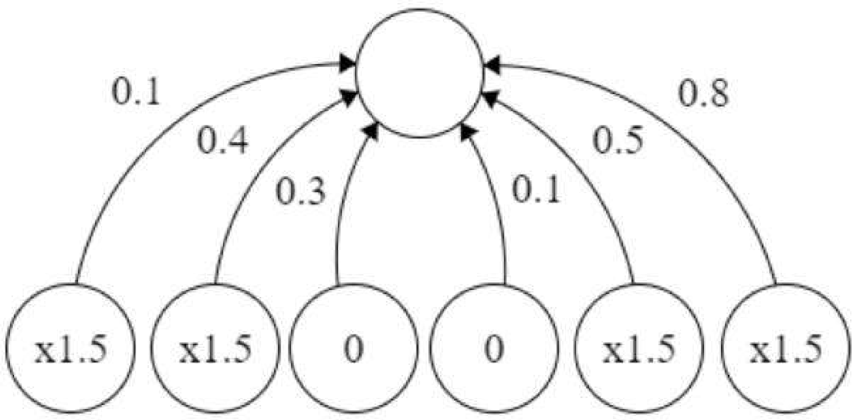
This poses two different problems to our model:

- Wastage of machine's resources when computing the same output.
- If many neurons are extracting the same features, it adds more significance to those features for our model. This leads to overfitting if the duplicate extracted features are specific to only the training set.

Solution to the problem: As the title suggests, we use dropout while training the NN to minimize co-adaptation. In dropout, we randomly shut down some fraction of a layer's neurons at each training step by zeroing out the neuron values. The fraction of neurons to be zeroed out is known as the dropout rate, . The remaining neurons have their

values multiplied by so that the overall sum of the neuron values remains the same.





The two images represent dropout applied to a layer of 6 units, shown at multiple training steps. The dropout rate is 1/3, and the remaining 4 neurons at each training step have their value scaled by x1.5. Thereby, we are choosing a random sample of neurons rather than training the whole network at once. This ensures that the co- adaptation is solved and they learn the hidden features better.

Why dropout works?

- By using dropout, in every iteration, you will work on a smaller neural network than the previous one and therefore, it approaches regularization.
- Dropout helps in shrinking the squared norm of the weights and this tends to a reduction in overfitting.

5.1 Machine Learning Life Cycle

- The Machine Leaning (ML) model management and the delivery of highly performing model is as important as the initial build of the model by choosing right dataset. The concepts around model retraining, model versioning, model deployment and model monitoring are the basis for machine learning operations that helps the data science teams deliver highly performing models.
- The use of machine leaning has increased substantially in enterprise data analytics scenarios to extract valuable insights from the business data. Hence, it is very important to have an ecosystem to build, test, deploy and maintain the enterprise grade machine learning models in production environments.
- The ML model development involves data acquisition from multiple trusted sources, data processing to make suitable for building the model, choose algorithm to build the model, build model, compute performance metrics and choose best performing model.
- The model maintenance plays critical role once the model is deployed into production. The maintenance of machine learning model includes keeping the model up to date and relevant in tune with the source data changes as there is a risk of model becoming outdated in course of time.
- Machine learning model lifecycle refers to the process that covers right from source data identification to model development, model deployment and model maintenance. At high level, the entire activities fall under two broad categories, such as ML model development and ML model operations.
- The machine learning lifecycle process is shows in Fig. 5.1.1 and it includes the following phases :
  1. Business goal identification
  2. ML problem framing
  3. Data processing (Data collection, data preprocessing, feature engineering)
  4. Model development (Training, tuning, evaluation)
  5. Model deployment (Inference, prediction)
  6. Model monitoring.
- **Business goal :** An organization considering ML should have a clear idea of the problem and the business value to be gained by solving that problem. We must be able to measure business value against specific business objectives and success criteria.
- **ML problem framing :** In this phase, the business problem is framed as a machine learning problem : What is observed and what should be predicted (known as a

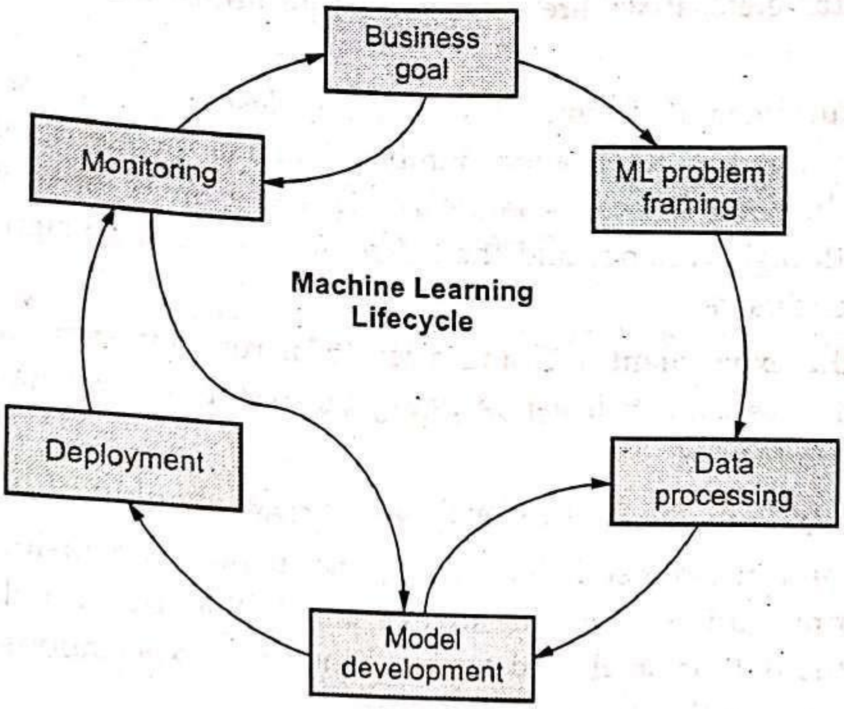


Fig. 5.1.1 Machine learning lifecycle process

label or target variable). Determining what to predict and how performance and error metrics must be optimized is a key step in this phase.

- **Data processing :** Training an accurate ML model requires data processing to convert data into a usable format. Data processing steps include collecting data, preparing data and feature engineering that is the process of creating, transforming, extracting, and selecting variables from data.
- **Model development :** Model development consists of model building, training, tuning and evaluation. Model building includes creating a pipeline that automates the build, train and release to staging and production environments.
- **Deployment :** After a model is trained, tuned, evaluated and validated, we can deploy the model into production. we can then make predictions and inferences against the model.
- **Monitoring :** Model monitoring system ensures your model is maintaining a desired level of performance through early detection and mitigation.

5.2 Guidelines for Machine Learning Experiments

- **Aim of the study :** What are the objectives (e.g. assessing the expected error of an algorithm, comparing two learning algorithm on a particular problem, etc.).
- **Selection of the response variable :** what should we use as the quality measure (e.g. error, precision and recall, complexity, etc.)
- **Choice of factors and levels :** What are the factors for the defined aim of the study (factors are hyperparameters when the algorithm is fix and want to find

best hyperparameters, if we are comparing algorithms, the learning algorithm is a factor).

- **Choice of experimental design :** Use factorial design unless we are sure that the factors do not interact. Replication number depends on the dataset size; it can be kept small when the dataset is large. Avoid using small datasets which leads to responses with high variance and the differences will not be significant and results will not be conclusive.
- **Performing the experiment :** Doing a few trial runs for some random settings to check that all is as expected, before doing the factorial experiment. All the results should be reproducible.
- **Statistical analysis of the data :** Conclusion we get should not be due to chance.
- Conclusions and recommendations : One frequently conclusion is the need for further experimentation. There is always a risk that our conclusions be wrong, especially if the data is small and noisy. When our expectations are not met, it is most helpful to investigate why they are not.

5.2.1 Dataset Preparation

- Machine learning is about learning some properties of a data set and applying them to new data. This is why a common practice in machine learning to evaluate an algorithm is to split the data at hand in two sets, one that we call a training set on which we learn data properties and one that we call a testing set, on which we test these properties.
- In training data, data are assign the labels. In test data, data labels are unknown but not given. The training data consist of a set of training examples.
- The real aim of supervised learning is to do well on test data that is not known during learning. Choosing the values for the parameters that minimize the loss function on the training data is not necessarily the best policy.
- The training error is the mean error over the training sample. The test error is the expected prediction error over an independent test sample.
- Problem is that training error is not a good estimator for test error. Training error can be reduced by making the hypothesis more sensitive to training data, but this may lead to over fitting and poor generalization.
- Training set : A set of examples used for learning, where the target value is known.
- Test set : It is used only to assess the performances of a classifier. It is never used during the training process so that the error on the test set provides an unbiased estimate of the generalization error.
- Training data is the knowledge about the data source which we use to construct the classifier.

- In a dataset, a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. Data points in the training set are excluded from the test (validation) set. Usually, a dataset is divided into a training set, a validation set (some people use 'test set' instead) in each iteration or divided into a training set, a validation set and a test set in each iteration.
- In machine learning, we basically try to create a model to predict the test data. So, we use the training data to fit the model and testing data to test it. The models generated are to predict the results unknown which is named as the test set.

5.3 Cross Validation (CV) and Resampling

- Validation techniques in machine learning are used to get the error rate of the ML model, which can be considered as close to the true error rate of the population. If the data volume is large enough to be representative of the population, you may not need the validation techniques.
- In machine learning, model validation is referred to as the process where a trained model is evaluated with a testing data set. The testing data set is a separate portion of the same data set from which the training set is derived. The main purpose of using the testing data set is to test the generalization ability of a trained model.
- Cross-validation is a technique for evaluating ML models by training several ML models on subsets of the available input data and evaluating them on the complementary subset of the data. Use cross-validation to detect overfitting, ie, failing to generalize a pattern.
- In general, ML involves deriving models from data, with the aim of achieving some kind of desired behavior, e.g., prediction or classification.
- But this generic task is broken down into a number of special cases. When training is done, the data that was removed can be used to test the performance of the learned model on ``new" data. This is the basic idea for a whole class of model evaluation methods called **cross validation.**
- Types of cross validation methods are holdout, K - fold and leave-one-out.
- The holdout method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The function approximate fits a function using the training set only.
- K - fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other k-1 subsets are put together to form a training set. Then the average error across all k trials is computed.

- Leave-one-out cross validation is K - fold cross validation taken to its logical extreme, with K equal to N, the number of data points in the set. That means that N separate times, the function approximate is trained on all the data except for one point and a prediction is made for that point.

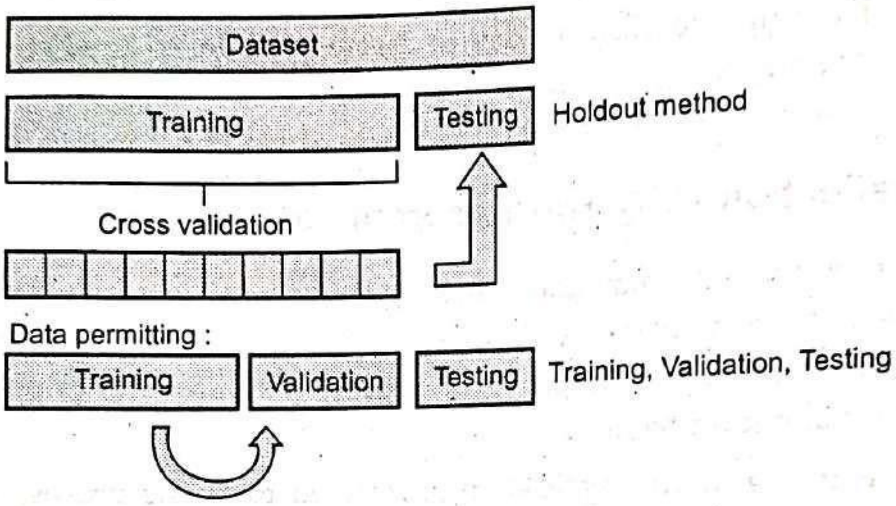


Fig. 5.3.1

5.3.1 K - Fold Cross Validation

- K - fold CV is where a given data set is split into a K number of sections/folds where each fold is used as testing set at some point.
- Lets take the scenario of 5-Fold cross validation (K = 5). Here, the data set is split into 5 folds.
- In the first interaction, the first fold is used to test the model and the rest are used to train the model. In the second iteration, 2nd fold is used as the testing set while the rest serve as the training set. This process is repeated until each fold of the 5 folds has been used as the testing set.
- K - fold cross validation is performed as per the following steps :

1. Partition the original training data set into k equal subsets. Each subset is called a fold. Let the folds be named as $f_1, f_2, \ldots f_k$.
2. For i = 1 to i = k

- Keep the fold f_i as validation set and keep all the remaining k – 1 folds in the cross validation training set.

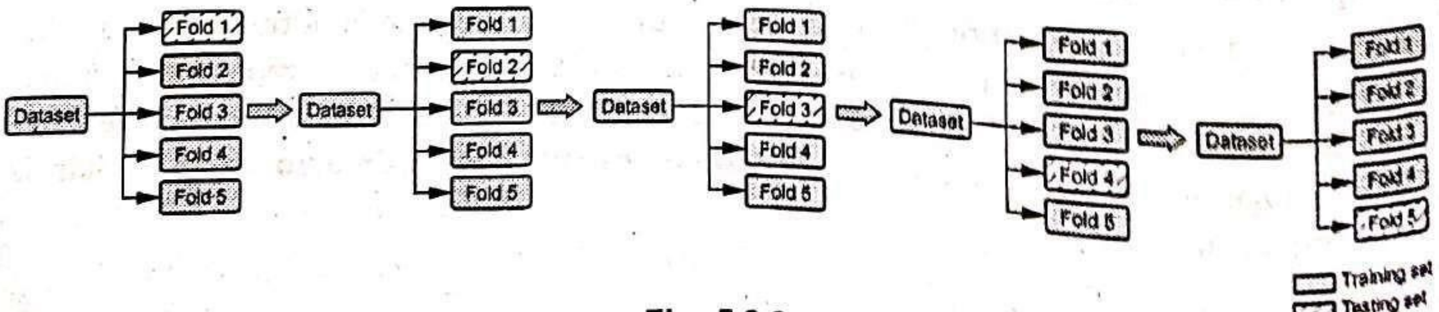


Fig. 5.3.2

3. Estimate the accuracy of your machine learning model by averaging the accuracies derived in all the k cases of cross validation.

- In the k - fold cross validation method, all the entries in the original training data set are used for both training as well as validation. Also, each entry is used for validation just once.
- The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test set exactly once and gets to be in a training set $k-1$ times. The variance of the resulting estimating is reduced as k is increased.
- The disadvantage of this method is that the training algorithm has to be rerun scratch k times, which means it takes k times as much computation to make an evaluation. A variant of this method is to randomly divide the data into a test and training set k different times.
- The advantage of doing this is that you can independently choose how large each test set is and how many trials you average over.

5.3.2 Bootstrapping

- Bootstrapping is a method of sample reuse that is much more general than cross-validation. The idea is to use the observed sample to estimate the population distribution. Then samples can be drawn from the estimated population and the sampling distribution of any type of estimator can itself be estimated.
- The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method. For example, it can provide an estimate of the standard error of a coefficient, or a confidence interval for that coefficient.
- Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of X and Y respectively, where X and Y are random quantities. We will invest a fraction α of our money in X and will invest the remaining $1-\alpha$ in Y.
- We wish to choose α to minimize the total risk, or variance, of our investment. In other words, we want to minimize Var $(\alpha X + (1-\alpha)Y)$.
- One can show that the value that minimizes the risk is given by,

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}},$$

where $\sigma_X^2 = \text{Var}(X)$, $\sigma_Y^2 = \text{Var}(Y)$ and $\sigma_{XY} = \text{Cov}(X,Y)$

- But the values of σ_X^2, σ_Y^2 and σ_{XY} are unknown.

- We can compute estimates for these quantities, σ_X^2, σ_Y^2 and $\hat{\sigma}_{XY}$, using a data set that contains measurements for X and Y.
- We can then estimate the value of α that minimizes the variance of our investment using,

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

- To estimate the standard deviation of $\hat{\alpha}$, we repeated the process of simulating 100 paired observations of X and Y and estimating α 1,000 times.
- We thereby obtained 1,000 estimates for α, which we can call $\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{1000}$.
- For these simulations the parameters were set to $\sigma_X^2 = 1$, $\sigma_Y^2 = 1.25$ and $\sigma_{XY} = 0.5$ and so we know that the true value of α is 0.6.
- The mean over all 1,000 estimates for α is,

$$\hat{\alpha} = \frac{1}{1000}\sum_{r=1}^{1000}\hat{\alpha}_r = 0.5996,$$

very close to $\alpha = 0.6$ and the standard deviation of the estimates is,

$$\sqrt{\frac{1}{1000-1}\sum_{r=1}^{1000}(\hat{\alpha}_r - \hat{\alpha})^2} = 0.083$$

- This gives us a very good idea of the accuracy of $\hat{\alpha}$: $SE(\hat{\alpha}) \approx 0.083$.
- So roughly speaking, for a random sample from the population, we would expect $\hat{\alpha}$ to differ from α by approximately 0.08, on average.
- There are three forms of bootstrapping which differ primarily in how the population is estimated.
  1. Nonparametric (Resampling)
  2. Semiparametric (Adding noise)
  3. Parametric. (Simulation)

1. **Nonparametric bootstrap :** In the nonparametric bootstrap a sample of the same size as the data is taken from the data with replacement. If we measure 10 samples, we create a new sample of size 10 by replicating some of the samples that we have already seen and omitting others.

2. **Semiparametric bootstrap :** The resampling bootstrap can only reproduce the items that were in the original sample. The semiparametric bootstrap assumes that the population includes other items that are similar to the observed sample by sampling from a smoothed version of the sample histogram. It turns out that this can be done very simply by first taking a sample with replacement from the observed sample and then adding noise.

3. **Parametric bootstrap :** Parametric bootstrapping assumes that the data comes from a known distribution with unknown parameters. We estimate the parameters from the data that you have and then you use the estimated distributions to simulate the samples.

5.4 Measuring Classifier Performance

- A binary classification rule is a method that assigns a class to an object, on the basis of its description.
- The performance of a binary classifier can be assessed by tabulating its predictions on a test set with known labels in contingency table or confusion matrix, with actual classes in rows and predicted classes in columns.
- Measures of performance need to satisfy several criteria :
  1. They must coherently capture the aspect of performance of interest;
  2. They must be intutive enough to become widely used, so that the same measures are consistently reported by researches, enabling community-wide conclusions to be drawn;
  3. They must be computationally tractable, to match the rapid growth in scale of modem data collection.
  4. They must be simple to report as a single number for each method-dataset combination.
- Performance metrics for binary classification are designed to captured tradeoffs between four fundamental population quantities : True positives, false positives, true negatives and false negatives.
- The evaluation measures in classification problems are defined from a matrix with the numbers of examples correctly and incorrectly classified for each class, named confusion matrix. The confusion matrix for a binary classification problem is shown below.

True class	Predicted class	
	Positive	Negative
Positive	True positive	False negative
Negative	False positive	True negative

- A confusion matrix contains about actual and predicted classifications done by a classification system. Performance of such systems is commonly using data in the matrix. Confusion matrix is also called a contingency table.
  1. **False positives :** Examples predicted as positive, which are from the negative class.

2. **False negatives :** Examples predicted as negative, whose true class is positive.
3. **True poisitives :** Examples correctly predicted as pertaining to the positive class.
4. **True negatives :** Examples correctly predicted as belongings to the negative class.

- The evaluation measure most used in practice is the accuracy rate. It evaluates the effectiveness of the classifier by its percentage of correct predictions.

$$\text{Accuracy rate} = \frac{|\text{True negatives}| + |\text{True positives}|}{|\text{False negatives}| + \text{False positives}| + |\text{True negatives}| + |\text{True positives}|}$$

- The complement of accuracy rate is the error rate, which evaluates a classifier by its percentage of incorrect predictions.

$$\text{Error rate} = \frac{|\text{False negatives}| + |\text{False positives}|}{|\text{False negatives}| + |\text{False positives}| + |\text{True negatives}| + |\text{True positives}|}$$

Error rate = 1 – (Accuracy rate)

- The recall and specificity measures evaluate the effectiveness of a classifier for each class in the binary problem. The recall is also known as sensitivity or true positive rate. Recall is the proportion of examples belonging to the positive class which were correctly predicted as positive.
- The specificity is a statistical measures of how well a binary classification test correctly identifies the negative cases.

$$\text{Recall (R)} = \frac{|\text{True positive}|}{||\text{True positive}| + |\text{False negative}|}$$

$$\text{Specificity} = \frac{|\text{True negative}|}{|\text{False positive}| + |\text{True positive}|}$$

- True positive Rate (TPR) is also called sensitivity, hit rate and recall.

$$\text{Sensitivity} = \frac{\text{Number of true positives}}{\text{Number of true positive} + \text{Number of false positive}}$$

- A statistical measure of how well a binary classification test correctly identifies a condition. Probability of correctly labeling members of the target class.
- No single measures tells the whole story. A classifier with 90 % accuracy can be useless if 90 percent of the population does not have cancer and the 10 % that do are misclassified by the classifier. Use of multiple measures recommended.

5.4.1 Accuracy and ROC Curves

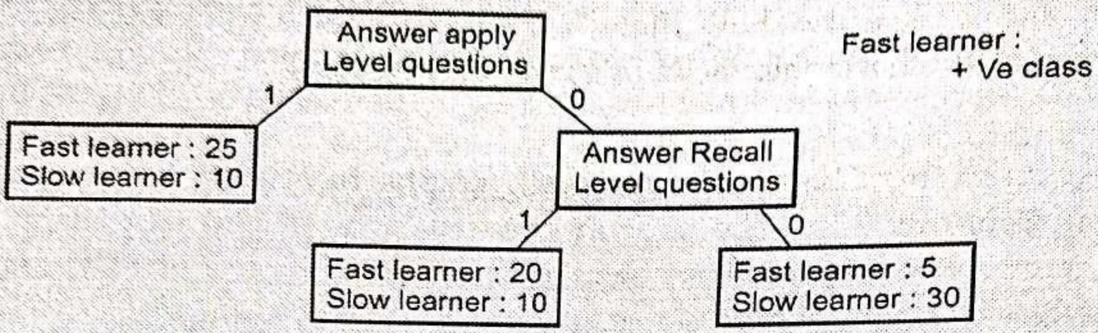
- Binary classification accuracy metrics quantify the two types of correct predictions and two types of errors. Typical metrics are accuracy (ACC), precision, recall, false positive rate, F1-measure. Each metric measures a different aspect of the predicative model.

- Accuracy (ACC) measures the fraction of correct predictions. Precision measures the fraction of actual positives among those examples that are predicted as positive. Recall measures how many actual positives were predicted as positive. F1-measure is the harmonic mean of precision and recall.

ROC Curve

- Receiver Operating Characteristics (ROC) graphs have long been used in signal detection theory to depict the tradeoff between hit rates and false alarm fates over noisy channel. Recent years have seen an increase in the use of ROC graphs in the machine learning community.
- An ROC plot plots true positive rate on the Y-axis false positive rate on the X-axis; a single contingency table corresponds to a single point in an ROC plot.
- The performance of a ranker can be assessed by drawing a piecewise linear curve in an ROC plot, known as an ROC curve. The curve starts in (0, 0), finishes in (1, 1) and is monitorically non-decreasing in both axes.
- A useful technique for organizing classifiers and visualizing their performance. Especially useful for domains with skewed class distribution and unequal classification error costs.
- It allows to create ROC curve and a complete sensitivity/specificity report. The ROC curve is a fundamental tool for diagnostic test evaluation.
- In a ROC curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100 Specificity) for different cut-off points of a parameter. Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. The area under the ROC curve is a measure of how well a parameter can distinguish between two diagnostic groups.
- Each point on an ROC curve connecting two segments corresponds to the true and false positive rates achieved on the same test set by the classifier obtained from the ranker by splitting the ranking between those two segments.
- An ROC curve is convex if the slopes are montonically non-increasing when moving along the curve from (0, 0) to (1, 1). A concavity in an ROC curve, i.e., two or more adjacent segments with increasing slopes, indicates a locally worse than random ranking. In this we would get better ranking performance by joining the segments involved in the concavity, thus creating a coarser classifier.

Example 5.4.1 *i) Find contingency table ii) Find recall iii) Precision iv) Negative recall v) False positive rate*



Answer apply Level questions
1
0
Fast learner : 25 Slow learner : 10
Answer Recall Level questions
1
0
Fast learner : 20 Slow learner : 10
Fast learner : 5 Slow learner : 30
Fast learner : + Ve class

Solution : Contingency table

	Predicted			**Total**	
Faster Learner	25	20	5	50	
Slow Learner	10	10	30	50	Actual
Total	35	30	35	100	

$$\text{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \text{ or } \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \text{ or } \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Calculate precision and recall

Precision = 25/35 = 0.714

Recall = 25/30 = 0.833

False positive rate = (False positive) / (false positive + true negative)

= 10/(10 + 30) = 0.25

Example 5.4.2 *Consider following confusion matrix and calculate following i) Sensitivity of classifier ii) Specificity of classifier.*

Confusion Matrix		**Predicted**		**Total**
Actual	+	8	10	18
	–	4	8	12
Total		12	18	30

Solution : Given data : TP = 8, FN = 10, FP = 4, TN = 8

- Sensitivity (SN) is calculated as the number of correct positive predictions divided by the total number of positives. It is also called recall (REC) or true positive rate (TPR)

$$\text{Sensitivity (SN)} = \frac{TP}{TP+FN} = \frac{8}{8+10} = 0.444$$

- Specificity (SP) is calculated as the number of correct negative predictions divided by the total number of negatives. It is also called true negative rate (TNR).

$$\text{Specificity (SP)} = \frac{TN}{TN+FP} = \frac{8}{8+4} = 0.666$$

Example 5.4.3 *Consider the following 3-class confusion matrix. Calculate precision and recall per class. Also calculate weighted average precision and recall for classifier.*

	Predicted		
Actual	15	2	3
	7	15	8
	2	3	45

Solution :

	Predicted			
Actual	15	2	3	20
	7	15	8	30
	2	3	45	50
	24	20	56	100

$$\text{Classifier Accuracy} = \frac{15+15+45}{100} = \frac{75}{100} = 0.75$$

Calculate per-class precision and recall :

First class $= \frac{15}{24} = 0.63$ and $\frac{15}{20} = 0.75$

Second class $= \frac{15}{20} = 0.75$ and $\frac{15}{30} = 0.50$

Third class $= \frac{45}{56} = 0.8$ and $\frac{45}{50} = 0.9$

Example 5.4.4 *Prove that : i) FPR = 1 – TPR ii) FNR = 1 – TPR*

Solution : i) FPR = 1–TPR

False Positive Rate (FPR) = 1 – True Negative Rate (TNR)

$$\text{FPR} = \text{FP/N} = \text{FP/ (FP + TN)}$$

$$\text{FPR} = 1 - \text{TNR}$$

ii) FNR = 1 – TPR

False Negative Rate = 1 – True Positive Rate

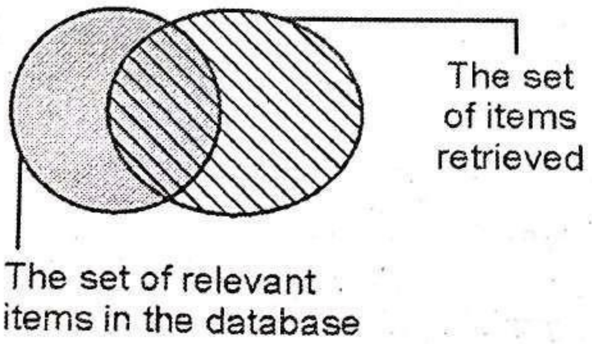
$$\text{FNR} = \text{FN (FN + TP)}$$

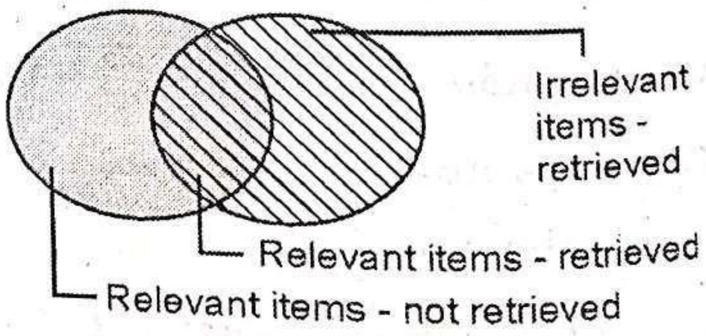
$$\text{FNR} = 1 - \text{TPR}$$

5.4.2 Precision and Recall

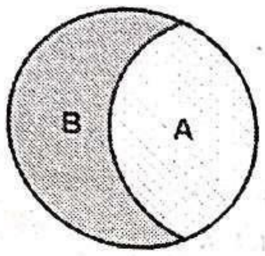
- **Relevance :** Relevance is a subjective notion. Different users may differ about the relevance or non-relevance of particular documents to given questions.
- In response to a query, an IR system searches its document collection and returns a ordered list of responses. It is called the retrieved set or ranked list. The system employs a search strategy or algorithm and measure the quality of a ranked list.
- A better search strategy yields a better ranked list and better ranked lists help the user fill their information need.
- Precision and recall are the basic measures used in evaluating search strategies. As shown in the first two figures, these measures assume :

1. There is a set of records in the database which is relevant to the search topic
2. Records are assumed to be either relevant or irrelevant.
3. The actual retrieval set may not perfectly match the set of relevant records.


The set of items retrieved
The set of relevant items in the database


Irrelevant items - retrieved
Relevant items - retrieved
Relevant items - not retrieved

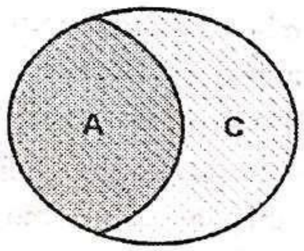
- **Recall** is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.



A = Number of relevant records retrieved.
B = Number of relevant records not retrieved.

$$\text{Recall} = \frac{A}{A+B} \times 100\ \%$$

Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.



A = Number of relevant records retrieved.
C = Number of irrelevant records retrieved.

$$\text{Precison} = \frac{A}{A+C} \times 100\ \%$$

- As recall increases, the precision decreases and recall decreases the precision increases.

Example 5.4.5 *Assume the following :*

A database contains 80 records on a particular topic

A search was conducted on that topic and 60 records were retrieved.

Of the 60 records retrieved, 45 were relevant.

Calculate the precision and recall scores for the search.

Solution : Using the designations above :

A = The number of relevant records retrieved,

B = The number of relevant records not retrieved, and

C = The number of irrelevant records retrieved.

In this example A = 45, B = 35 (80 – 45) and C = 15 (60 – 45).

$$\text{Recall} = \frac{45}{45+35} \times 100\ \%$$

$$\text{Recall} = \frac{45}{80} \times 100\ \%$$

$$\textbf{Recall} = \mathbf{56.25\ \%}$$

$$\text{Precision} = \frac{A}{A+C} \times 100\ \%$$

$$\text{Precision} = \frac{45}{45+15} \times 100\ \% = \frac{45}{60} \times 100$$

$$\textbf{Precision} = 75\ \%$$

Example 5.4.6 *20 found documents, 18 relevant, 3 relevant documents are not found, 27 irrelevant are as well not found. Calculate the precision and recall and fallout scores for the search.*

Solution : Precision : 18/20 = 90 %

Recall : 18/21 = 85.7 %

Fall-out : 2/29 = 6.9 %

- Recall is a non-decreasing function of the number of docs retrieved. In a good system, precision decreases as either the number of docs retrieved or recall increases. This is not a theorem, but a result with strong empirical confirmation.
- The set of ordered pairs makes up the precision-recall graph. Geometrically when the points have been joined up in some way they make up the precision-recall curve. The performance of each request is usually given by a precision-recall curve. To measure the overall performance of a system, the set of curves, one for each request, is combined in some way to produce an average curve.
- Assume that set R_q containing the relevant document for q has been defined. Without loss of generality, assume further that the set R_q is composed of the following documents :

$$R_q = \{d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123}\}$$

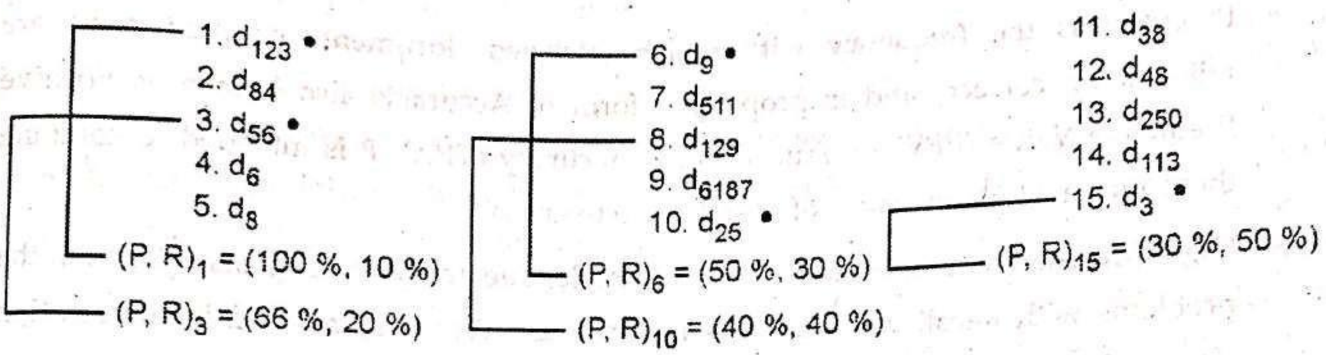
There are ten documents which are relevant to the query q.

- For the query q, a ranking of the documents in the answer set as follows.

Ranking for query q :

1. d_{123}	*	6. d_9	*	11. d_{38}	
2. d_{84}		7. d_{511}		12. d_{48}	
3. d_{56}	*	8. d_{129}		13. d_{250}	
4. d_6		9. d_{187}		14. d_{113}	
5. d_8		10. d_{25}	*	15. d_3	*

- The documents that are relevant to the query q are marked with star after the document number. Ten relevant documents, five included in Top 15 .



- Fig 5.4.1 shows the curve of precision versus recall. By taking various numbers of the top returned documents (levels of recall), the evaluator can produce a precision-recall curve.

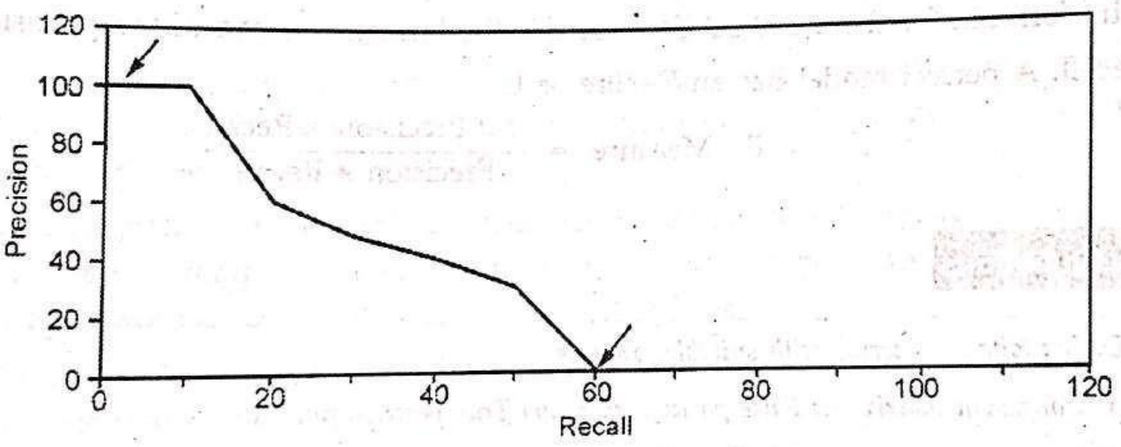


Fig. 5.4.1 Precision versus recall curve

- The precision versus recall curve is usually plotted based on 11 standard recall level: 0 %,10 %,....,100 %.
- In this example : The precisions for recall levels higher than 50 % drop to 0 because no relevant documents were retrieved. There was an interpolation for the recall level 0 %.
- Since the recall levels for each query might be distinct from the 11 standard recall levels.

5.4.3 F - Measure

- The F measure is a measure of a test's accuracy and is defined as the weighted harmonic mean of the precision and recall of the test. The F - measure or F - score is one of the most commonly used "single number" measures in Information Retrieval, Natural Language Processing and Machine Learning.
- F-measure comes from Information Retrieval (IR) where Recall is the frequency with which relevant documents are retrieved or 'recalled' by a system, but it is known elsewhere as Sensitivity or True Positive Rate (TPR).

- Precision is the frequency with which retrieved documents or predictions are relevant or 'correct', and is properly a form of Accuracy, also known as Positive Predictive Value (PPV) or True Positive Accuracy (TPA). F is intended to combine these into a single measure of search 'effectiveness'.
- High precision and low accuracy is possible. due to systematic bias. One of the problems with Recall, Precision, F - measure and Accuracy as used in Information Retrieval is that they are easily biased.
- The F-measure balances the precision and recall. The result is a value between 0.0 for the worst F-measure and 1.0 for a perfect F - measure.
- The formula for the standard F1 - score is the harmonic mean of the precision and recall. A perfect model has an F-score of 1.

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

Review Questions

1. *Define following terms with suitable example :*

   i) Confusion matrix ii) False positive rate iii) True positive rate.
2. *What is a contingency table/matrix ? What is the use of it ?*
3. *Explain true positive, true negative false positives, false negatives and class ratio.*
4. *What is a contingency table ? What does it represent ?*
5. *What is multiple linear regression ? How will it be different from simple linear regression ?*

5.5 Multiclass Classification

- Multiclass classification is a machine learning classification task that consists of more than two classes, or outputs. For example, using a model to identify animal types in images from an encyclopedia is a multiclass classification example because there are many different animal classifications that each image can be classified as. Multiclass classification also requires that a sample only have one class.
- Each training point belongs to one of N different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new point belongs.

- There are many scenarios in which there are multiple categories to which points belong, but a given point can belong to multiple categories. In its most basic form, this problem decomposes trivially into a set of unlinked binary problems, which can be solved naturally using techniques for binary classification.
- Common model for classification is the Support Vector Machine (SVM). An SVM works by projecting the data into a higher dimensional space and separating it into different classes by using a single (or set of) hyperplanes. A single SVM does binary classification and can differentiate between two classes. In order to differentiate between K classes, one can use (K – 1) SVMs. Each one would predict membership in one of the K classes.

5.5.1 Weighted Average

- **Mean Average Precision (MAP)** is also called average precision at seen relevant documents. It determine precision at each point when a new relevant document gets retrieved. Average of the precision value obtained for the top k documents, each time a relevant doc is retrieved.
- Avoids interpolation, use of fixed recall levels. MAP for query collection is arithmetic averaging. Average precision - recall curves are normally used to compare the performance of distinct IR algorithms.
- Use P = 0 for each relevant document that was not retrieved. Determine average for each query, then average over queries :

$$\text{MAP} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{Q_j}\sum_{i=1}^{Q_j}P(\text{doc}_i)$$

where Q_j = Number of relevant document for query j

N = Number of queries

$P(\text{doc}_i)$ = Precision at i^{th} relevant document

Precision - recall appropriateness :

- Precision and recall have been extensively used to evaluate the retrieval performance of IR algorithms. However, a more careful reflection reveals problems with these two measures :
- First, the proper estimation of maximum recall for a query requires detailed knowledge of all the documents in the collection.
- Second, in many situations the use of a single measure could be more appropriate.
- Third, recall and precision measure the effectiveness over a set of queries processed in batch mode.

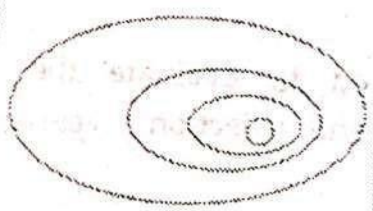
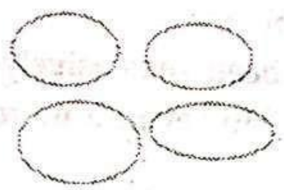
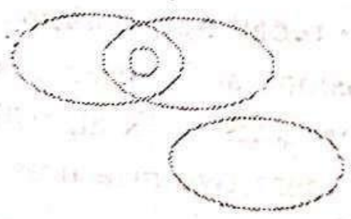
- Fourth, for systems which require a weak ordering though, recall and precision might be inadequate.

Single value summaries :

- Average precision - - recall curves constitute standard evaluation metrics for information retrieval systems. However, there are situations in which we would like to evaluate retrieval performance over individual queries. The reasons are two fold :
  1. First, averaging precision over many queries might disguise important anomalies in the retrieval algorithms under study.
  2. Second, we might be interested in investigating whether a algorithm outperforms the other for each query.
- In these situations, a single precision value can be used.

5.5.2 Multiclass Classification Techniques

- Each training point belongs to one of N different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new point belongs. The multi-class classification problem refers to assigning each of the observations into one of k classes.
- A common way to combine pair wise comparisons is by voting. It constructs a rule for discriminating between every pair of classes and then selecting the class with the most winning two-class decisions. Though the voting procedure requires just pair wise decisions, it only predicts a class label.
- Example of multi-label classification is as follows :

1. Is it eatable ? 2. Is it sweet ? 3. Is it a fruit ? 4. Is it a banana ?	1. Is it a banana ? 2. Is it an apple ? 3. Is it an orange ? 4. Is it a pineapple ?	1. Is it a banana ? 2. Is it yellow ? 3. Is it sweet ? 4. Is it round ?
		
Nested/Hierarchical	Exclusive/Multi-class	General/Structured

- Fig. 5.5.1 and 5.5.2 shows binary and multiclass classification.

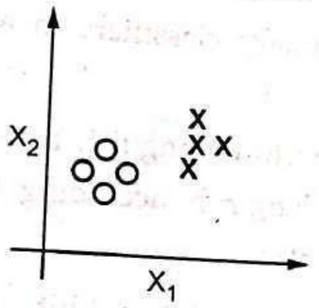


Fig. 5.5.1 Binary classification

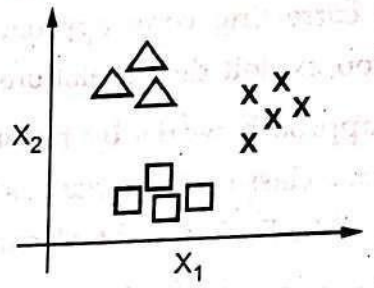


Fig. 5.5.2 Multiclass classification

- Multiclass classification through binary classification :

1. One Vs All (OVA) :

- For each class build a classifier for that class vs the rest. Build N different binary classifiers.
- For this approach, we require N = K binary classifiers, where the k^th^ classifier is trained with positive examples belonging to class k and negative examples belonging to the other K – 1 classes.
- When testing an unknown example, the classifier producing the maximum output is considered the winner, and this class label is assigned to that example.
- It is simple and provides performance that is comparable to other more complicated approaches when the binary classifier is tuned well.

2. All-Vs-All (AVA) :

- For each class build a classifier for those class vs the rest. Build N (N – 1) classifiers, one classifier to distinguish each pair of classes i and j .
- A binary classifier is built to discriminate between each pair of classes, while discarding the rest of the classes.
- When testing a new example, a voting is performed among the classifiers and the class with the maximum number of votes wins.

3. Calibration

- The decision function f of a classifier is said to be calibrated or well-calibrated if P (x is correctly classified | f(x) = s) ≈ s
- Informally f is a good estimate of the probability of classifying correctly a new datapoint x which would have output value x. Intuitively if the "raw" output of a classifier is g you can calibrate it by estimating the probability of x being well classified given that g(x)=y for all y values possible.

4. Error-Correcting Output-Coding (ECOC)

- Error correcting code approaches try to combine binary classifiers in a way that lets you exploit de-correlations and correct errors.
- This approach works by training N binary classifiers to distinguish between the K different classes. Each class is given a codeword of length N according to a binary matrix M . Each row of M corresponds to a certain class.
- The following table shows an example for K = 5 classes and N = 7 bit code words.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Class 1	0	0	0	0	0	0	0
Class 2	0	1	1	0	0	1	1
Class 3	0	1	1	1	1	0	0
Class 4	1	0	1	1	0	1	0
Class 5	1	1	0	1	0	0	1

- Each class is given a row of the matrix. Each column is used to train a distinct binary classifier. When testing an unseen example, the output codeword from the N classifiers is compared to the given K code words, and the one with the minimum hamming distance is considered the class label for that example.

Example 5.5.1 *Consider the following three-class confusion matrix.*

	Predicted		
	15	2	3
Actual	7	15	8
	2	3	45

Calculate precision and recall per class. Also calculate weighted average precision and recall for the classifier.

Solution :

	Predicted			
	15	2	3	20
Actual	7	15	8	30
	2	3	45	50
	24	20	56	100

Classifier accuracy $= \frac{15+15+45}{100} = \frac{75}{100} = 0.75$

Calculate per-class precision and recall :

First class $= \frac{15}{24} = 0.63$ and $\frac{15}{20} = 0.75$

Second class $= \frac{15}{20} = 0.75$ and $\frac{15}{30} = 0.50$

Third class $= \frac{45}{56} = 0.8$ and $\frac{45}{50} = 0.9$

Example 5.5.2 *Prove with an example : Accuracy = 1 – Error rate.*

Solution : Accuracy is the percent of correct classifications. Error rate is the percent of incorrect classifications. Classification accuracy is a misleading measure of performance when the data are not perfectly balanced. This is because a classifier may take advantage of an imbalanced dataset and trivially achieve a classification accuracy equal to the fraction of the majority class.

Review Questions

1. *Explain construction of multi-class classifier,*

   i) One Vs all approach *ii) One Vs one approach*

   iii) Error correcting output codes approach.

2. *Explain any two approaches to construct multiclass classifier.*

5.6 t - Test

- When a small sample (size < 30) is considered, the tests are inapplicable because the assumptions we made for large sample tests, do not hold good for small samples.
- In case of small samples it is not possible to assume,
  i) That the random sampling distribution of a statistics normal
  ii) The sample values are sufficiently close to population values to calculate the S.E. of estimate.
- Thus an entirely new approach is required to deal with problems of small samples. But one should note that the methods and theory of small samples are applicable to large samples but its converse is not true
- When sample sizes are small, as is often the case in practice, the Central Limit Theorem does not apply. One must then impose stricter assumptions on the population to give statistical validity to the test procedure. One common assumption is that the population from which the sample is taken has a normal probability distribution to begin with.
- Degree of freedom (df) : By degree of freedom we mean the number of classes to which the value can be assigned arbitrarily or at will without voicing the restrictions or limitations placed.
- For example, we are asked to choose any 4 numbers whose total is 50. Clearly we are at freedom to choose any 3 numbers say 10, 23, 7 but the fourth number, 10 is fixed since the total is 50 [50 – (10 + 23 + 7) = 10]. Thus we are given a restriction, hence the freedom of selection of number is 4 – 1 = 3.
- The degree of freedom (df) is denoted by ν(nu) or df and it is given by $\nu = n - k$, where n = number of classes and k = number of independent constrains.

5.6.1 t - Test for Single Mean

- When the sample values come from a normal distribution, the exact distribution of "t" was worked out by W. S. Gossett. He called it a **t - distribution.**
- Unfortunately, there is not one t - distribution. There are different t - distributions for each different value of n. If n = 7 there is a certain t - distribution but if n = 13 the t - distribution is a little different. We say that the variable t has a t - distribution with n–1 degrees of freedom.
- Suppose a simple random sample of size n is drawn from a population. If the population from which the sample is taken follows a normal distribution, the distribution of the random variable,

$$t = \frac{\overline{x}-\mu_0}{s/\sqrt{n}}$$

follows **Student's t - Distribution** with n – 1 degrees of freedom.

- The sample mean is $\overline{x}$ and the sample standard deviation is s.
- The **degrees of freedom** are the number of free choices left after a sample statistic such as is calculated. When you use a t - distribution to estimate a population mean, the degrees of freedom are equal to one less than the sample size.

  d.f. = n – 1

Assumptions :

1. Population is normal although this assumption can be relaxed if sample size is "large".
2. Random sample was drawn from the population of interest.

- Based on the comparison of calculated 't' value with the theoretical 't' value from the table, we conclude :

Shape of student's t - distribution

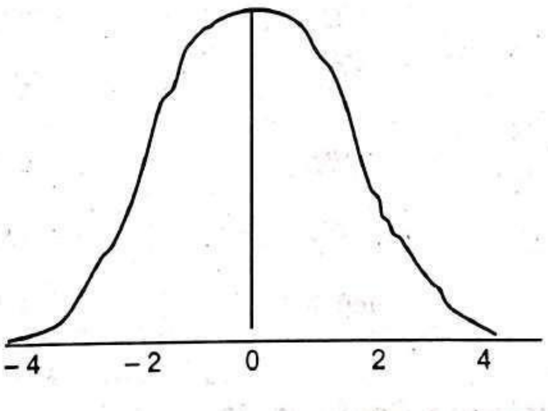


Fig. 5.6.1

5.6.2 Properties of Students t - Distribution

1. The t - distribution is different for different degrees of freedom.
2. The t - distribution is centered at 0 and symmetric about 0.
3. The total area under the curve is 1. The area to the left of 0 is 1/2 and the area to the right of 0 is 1/2.
4. As the magnitude of t increases the graph approaches but never equals 0.
5. The area in the tails of the t - distribution is larger than the area in the tails of the normal distribution.
6. The shape of the t-distribution is dependent on the sample size n.

6. As sample size n increases, the distribution becomes approximately normal.
7. The standard deviation is greater than 1.
8. The mean, median, and mode of the t-distribution are equal to zero.
9. The area in the tails of the t - distribution is a little greater than the area in the tails of the standard normal distribution, because we are using s as an estimate of σ, thereby introducing further variability.
10. As the sample size n increases the density of the curve of t get closer to the standard normal density curve. This result occurs because as the sample size n increases, the values of s get closer to σ, by the law of large numbers.

T - critical values

- Critical values for various degrees of freedom for the t - distribution are (compared to the normal)

n	Degrees of freedom	$t_{0.025}$
6	5	2.571
16	15	2.131
31	30	2.042
101	100	1.984
1001	1000	1.962
Normal	"Infinite"	1.960

5.6.3 t - Test for Correlation Coefficients

- The correlation coefficient, ρ (rho), is a popular statistic for describing the strength of the relationship between two variables.
- The correlation coefficient is the slope of the regression line between two variables when both variables have been standardized by subtracting their means and dividing by their standard deviations. The correlation ranges between plus and minus one.
- When ρ is used as a descriptive statistic, no special distributional assumptions need to be made about the variables (Y and X) from which it is calculated.

- When hypothesis tests are made, you assume that the observations are independent and that the variables are distributed according to the bivariate-normal density function.
- However, as with the t-test, tests based on the correlation coefficient are robust to moderate departures from this normality assumption.
- The population correlation ? is estimated by the sample correlation coefficient r. Note we use the symbol R on the screens and printouts to represent the population correlation.
- t - test for correlation coefficients formul

$$t = r\sqrt{\frac{n-2}{1-r^2}}$$

With degrees of freedom equal to n – 2.

- The steps to be followed for the t - test for correlation coefficient is listed below :

1. State the null hypothesis and alternative hypothesis.

   $H_0 = \rho = 0$

   $H_a = \rho \neq 0$

   Here ρ is the population correlation coefficient.
2. State the significance level.
3. Find the test statistic of correlation coefficient with the above-defined formula.
4. To make a decision, use the critical value approach or the p - value approach
5. Finally, state the conclusion.

- The above test is conducted with the supposition that the association is linear between the variables and originate from a normal distribution that is bivariate.
- The t-test is always used for population correlation coefficient of zero. So, in order to test the population correlation coefficient other than zero, z-test for correlation coefficient is used to test the significance of the correlation coefficient.

5.7 McNemar's Test

- The McNemar test is a non-parametric test for paired nominal data. It is used for finding a change in proportion for the paired data. It compare the performance of two classifiers on N items from a single test set.
- McNemar's test is used to compare the performance of two classifiers on the same test set. This test works if there are a large number of items on which A and B make different predictions.

- McNemar's test is applied to 2 × 2 contingency tables with matched pairs of subjects to determine whether the row and column marginal frequencies are equal.
- The three main assumptions for the test are :
  1. We must have one nominal variable with two categories and one independent variable with two connected groups.
  2. The two groups in the dependent variable must be mutually exclusive.
  3. Sample must be a random sample.

5.8 K - fold CV Paired t Test

- We use k - fold cross-validation to get K training/validation set pairs. To train the two classification algorithms on the training sets T_i; where i = 1; ...; K and test on the validation sets V_i.
- The error percentages of the classifiers on the validation sets are recorded as p_i^1 and p_i^2.
- If the two classification algorithms have the same error rate, then we expect them to have the same mean, or equivalently, that the difference of their means is 0.
- The difference in error rates on fold i is $p_i = p_i^1 - p_i^2$. This is a paired test; that is, for each i, both algorithms see the same training and validation sets.
- When this is done K times, we have a distribution of p_i containing K points. Given that p_i^1 and p_i^2 are both (approximately) normal, their difference p_i is also normal. The null hypothesis is that this distribution has 0 mean ($H_0 : \mu = 0$ vs. $H_1 : \mu \neq 0$)
- We define :

$$m = \frac{\sum_{t=1}^{N} p_i}{K}, \quad S^2 = \frac{\sum_{t=1}^{K} (p_i - m)^2}{K-1}$$

Under the null hypothesis that $\mu = 0$, we have a statistic that is t-distributed with K – 1 degrees of freedom :

$$\frac{\sqrt{k}(m-0)}{S} = \frac{\sqrt{k}(m)}{S} \sim t_{k-1}$$

- Thus the K - fold cv paired t - test rejects the hypothesis that two classification algorithms have the same error rate at significance level α if this value is outside the interval $(-t_{\alpha/2, K-1}, t_{\alpha/2, K-1})$.
- If we want to test whether the first algorithm has less error than the second, we need a one-sided hypothesis and use a one-tailed test :

$$H_0 : \mu \geq 0 \text{ vs. } H_1 : \mu < 0$$

- If the test rejects, out claim that the first one has significantly less error is supported.

- Advantage is that each test set is independent of others. But the training sets still overlap. This overlap may prevent the test from obtaining a good estimate of the amount of variation that would be observed if each training set were completely independent of previous training sets.
- The variance in the t statistic maybe sometimes underestimated, the means are occasionally poorly estimated and this may result in large t values.

5.9 Two Marks Questions with Answers

Q.1 Define Bootstrapping.

Ans. : Bootstrapping is a method of sample reuse that is much more general than cross-validation. The idea is to use the observed sample to estimate the population distribution. Then samples can be drawn from the estimated population and the sampling distribution of any type of estimator can itself be estimated.

Q.2 What is confusion matrix ?

Ans. : The evaluation measures in classification problems are defined from a matrix with the number of examples correctly and incorrectly classified for each class, named confusion matrix.

Q.3 What is cross-validation ?

Ans. : Cross-validation is a technique for validating the model efficiency by training it on the subset of input data and testing on previously unseen subset of the input data.

Q.4 Explain McNemar's test.

Ans. :

- The McNemar test is a non-parametric test for paired nominal data. It is used for finding a change in proportion for the paired data. It compare the performance of two classifiers on N items from a single test set.
- McNemar's test is used to compare the performance of two classifiers on the same test set. This test works if there are a large number of items on which A and B make different predictions.

Q.5 What is K - fold cross-validation ?

Ans. : K-fold cross-validation approach divides the input dataset into K groups of samples of equal sizes. These samples are called **folds**. For each learning set, the prediction function uses k – 1 folds and the rest of the folds are used for the test set.

Q.6 What is a T - test ?

Ans. : The t - test compares the means (averages) of two populations to determine how different they are from each other. The test generates a t-score and p-value, which quantify exactly how different each population is and the likelihood that this difference can be explained by chance or sampling error.

Q.7 List the applications of cross-validation.

Ans. :

- This technique can be used to compare the performance of different predictive modeling methods.
- It has great scope in the medical research field.
- It can also be used for the meta-analysis, as it is already being used by the data scientists in the field of medical statistics.

Q.8 Explain merits and demerits of t-test.

Ans. : Merits :

1. Easy to gather data.
2. Determine source data.
3. Essential for generalization.

Demerits :

1. It may contains small amount of noise.
2. If the data collected violates the assumption of the t - test, then the output is unreliable.
3. T-test cannot be used for multiple comparisons

❑❑❑