Handwritten Machine Learning Notes.

Milind D Mali

- **Data Scientist.**

linkedin.com/in/milind-mali-585172236

Date: / /

Page

Machine learning

(main)

Supervised ML

- **Regression**
  - linear
  - Ridge & lasso
  - polynomial Reg.
  - Multi linear
  - Non-linear
  - OLS
  - Time series forecasting.
- **classification**
  - logistic
  - KNN
  - DT
  - Naive Byes
  - RF
  - XGB
  - GBM
  - SVM
  - ANN
  - catboost, adaboost

Unsupervised ML

- **clustering**
  - kmeans
  - DBScan
  - hierarchical
- **aprior**
- **Recommendation**
  - Content
  - Memory
    - User based
    - item based
- **Dimentionality Reduction**
  - PCA
  - tSNE
  - ICA
  - SVD
  - Auto Encoding.

Supervised ML →

We will start with Regression → linear

linear Regression

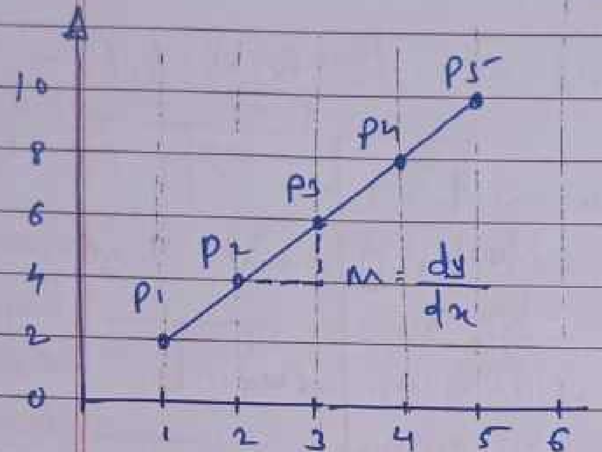
- Simple linear Regression (SLR)
- Multiple linear Regression (MLR)
- polynomial Regression

• Simple linear Regression (SLR) :- when for Continuous output we have only one feature then it is called SLR

for Ex.

CGPA	package
6.0	2 LPA
7.2	3 LPA
8.9	6 LPA.

Basics of linier Regression.



Equation of straight line

$$y = mx + c$$

find value of m and c

$P_2\ (2, 4) \longrightarrow x_1, y_1$

$P_3\ (3, 6) \longrightarrow x_2, y_2$

$$m = \text{slope} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{6-4}{3-2}$$

$$m = \cancel{4}/_1 = 4 = 2$$

slope means with every change in value of x how much differs in y

Intercept = c

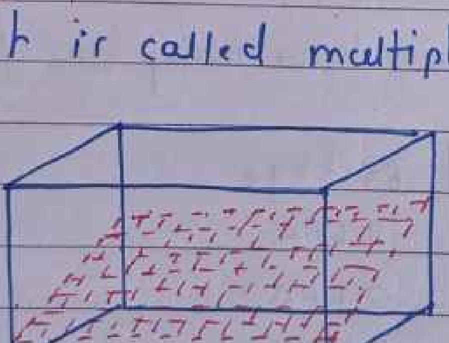
$P_4\ (x_3\, y_3) \longrightarrow (4, 8)$

$$y = mx + c$$
$$8 = 2(4) + c$$
$$8 - 8 = c = 0$$

Inference : The above line eq^n is function that relate x and y.
for given value of x we can find corresponding value of y.

what if we have two or more than two indepedent variable ?

$\longrightarrow$ then it is called multiple linear regrasion



(3D)

simple linear regression : $y = mx + c$

multiple linear regression

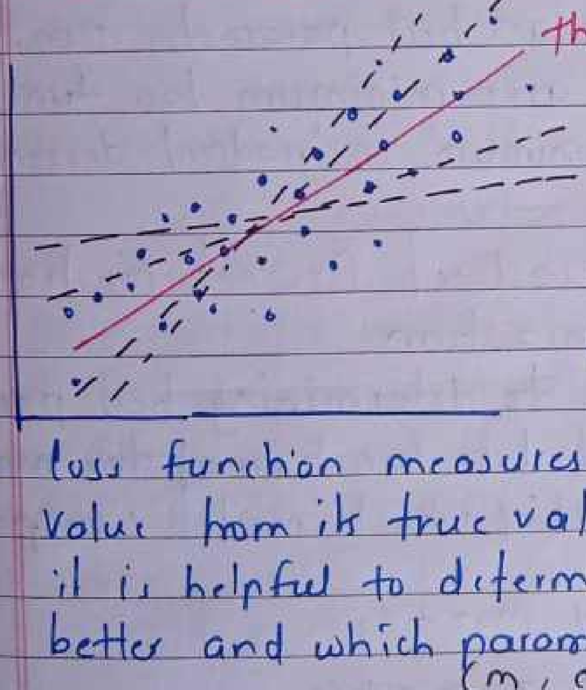
$$y = a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n + b.$$

Advantages: 1) simple to implement
2) perform well on the data with linear relationship.

Disadvantages: 1) Not suitable for data having non-linear relationship.
2) underfitting issue
3) sensitive to outlier

Mathematical Understanding :-

this is called best fit line

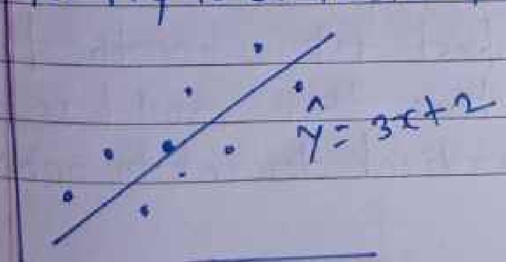


so here purpose is to find best fitted line (model) for linear regression.
{ this where loss function comes in picture }

- loss function measures how far an estimated value from its true value.
- it is helpful to determine which model performs better and which parameters (m, c) are better.

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

let try to understand with the ex. $m = 3$, $c = 2$



x	y	$\hat{y}$
2	10	8
3	14	11
4	18	14
5	22	17
6	26	20.

Now find its loss function.

$$loss = \frac{[(10-8)^2+(14-11)^2+(18-14)^2+(22-17)^2+(26-20)^2]}{5}$$

$$= \frac{4+9+16+25+36}{5} = \frac{90}{5} = 18$$

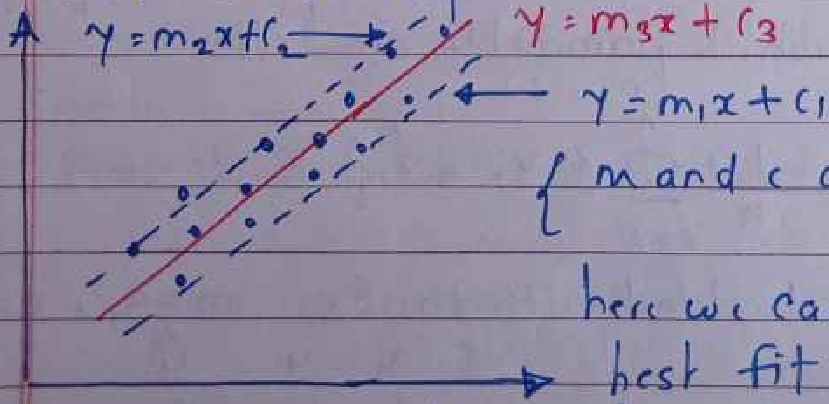
{reason to take square: so that positive and negative value may not cancel each other}

low loss value $\longrightarrow$ High Accuracy
high loss value $\longrightarrow$ low Accuracy

we can improve the model by some optimization technique called as "gradient descent" where repeate the process iteratively untill we get best parameter (m, c) for which model will give minimum loss function. Called as "global minimum" in "gradient descent"

How we can use gradient Descent for linear regression for optimization.

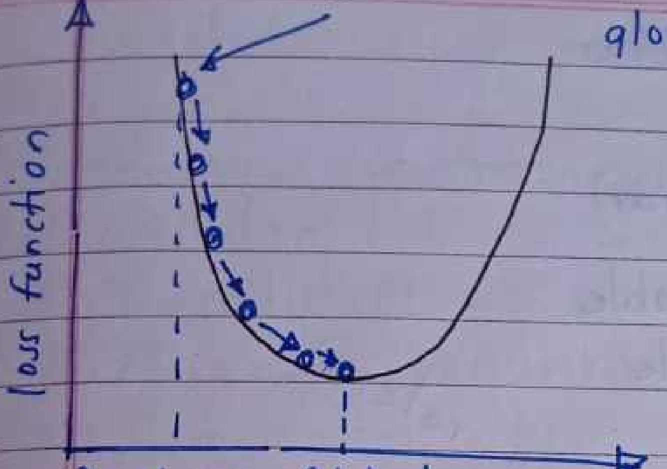
→ optimization refers to determining best parameter for model such that loss function of the model decrease as result of which model can predict more accurately.



{m and c are the parameters}

here we can find model 3 is best fit since the loss function is least and thus this model is optimum this where we used gradient descent to optimize model.

{this will step down untill reach down to global minimum by iterative process}



Random initial Value $(m_1 c_1)$

Global Minimum $(m_3 c_3)$

$$m = m_1 - L D_m$$
$$c = c_1 - L D_c$$

updated. initial

In machine learning

$$w = w - L\,dw$$
$$b = b - L\,db$$

m - slope
c - Intercept

w = weight
b = bias.

L - learning rate : it is magnitude of change that you want in parameter during iteration.

Dm : partial derevative of loss function w.r.t m
Dc : partial derevative of loss function w.r.t c.

$$D_m = \frac{\partial(\text{loss function})}{\partial m} = \frac{\partial}{\partial m}\left(\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right)$$

$$\boxed{= \frac{-2}{n}\sum_{i=0}^{n} x_i(y_i - \hat{y}_i)}$$

Ily $$D_c = \frac{\partial(\text{loss function})}{\partial c} = \frac{\partial}{\partial c}\left(\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right)$$

$$\boxed{= \frac{-2}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)}$$

Page

In terms weigh and bias

$$y = wx + b$$
(weight) (bias)

x - independent Variable
y - dependent Variable
w - weight
b - bias

updated, initial, learning rate(α)

$w = w - \alpha dw$ ← change in Loss function w.r.t w
$b = b - \alpha db$ ← change in loss function w.r.t. b.

Learning rate:- it is tuning parameter in an optimization algorithm that determine step size at each iteration while moving toward minimum of loss function.

$$dw = \frac{-2}{n}\sum_{i=0}^{n} x_i(y_i - \hat{y}_i)$$

$$db = \frac{-2}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i).$$

for Multiple linear Regression: for one target we have two or more than two independent Variables.

prediction Equation for MLR

$$\hat{y} = a_1x_1 + a_2x_2 + \cdots a_nx_n + b.$$

and,

Regression Equation

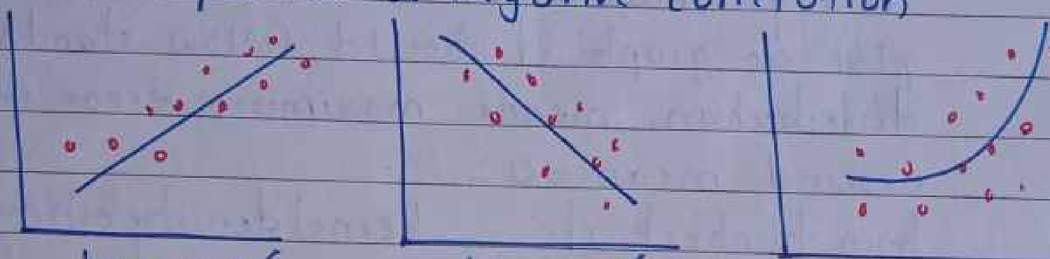
$$\underbrace{y}_{\text{Actual Value}} = \underbrace{a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n + b}_{\text{predicted value}} + \underbrace{\varepsilon}_{\text{Error}}$$

What are assumptions of linear Regression?

there are five main assumption in linear regression.

1. linear Relation between input and output.
2. No multicolinearity.
3. Normality of Residual.
4. Homoscedasticity.
5. No autocorrelation of error

1. **Assumption 1**:- there should be linear relation between individual feature and target (output) it could be positive or negative correlation



applicable to multiple linear problem as well. Not ok.
how to check this : scatter plot : (feature 1 vs target) (feature 2 vs target).

2. **Assumption 2**:- Multicollinearity.: it mean there all the feature should be independent or should not have any correlation among themselves.

why, what the problem?

In multi linear Regression model for 3D we draw a hyperplane.

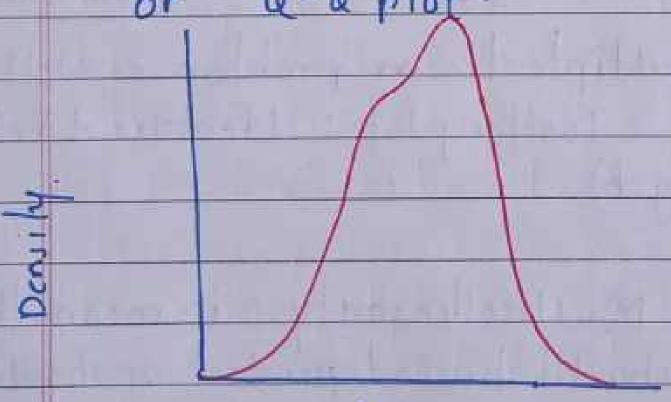
$$y = a_1x_1 + a_2x_2 + a_3x_3 + b.$$

where a_1 represent what will be the change in y with respect to x_1 assuming x_2 and x_3 constant. but if it violate this assumption then model will not perform good. (Ex of two physics scientist) how to check multicollinearity :-

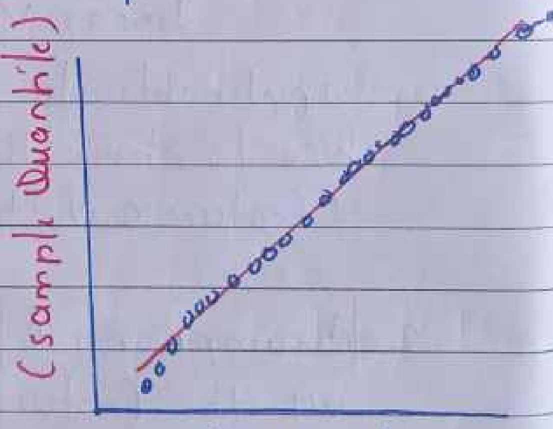
1) VIF (Variance Inflation Factor) :-
if it is around 1 then feature don't have the issue and if it is 5 or more than that then that particular feature have multicollinearity issue and need to remove it

2). another method is to find out correlation between all the features (Heatmap)

3. $\boxed{\text{Assumption 3.}}$ - Normality of Residual.
it says that when error (actual - predicted) plot or graph it should follow standard normal distribution. means maximum error should around mean = 0.
how to check it : kernel density estimation (kde) or Q-Q plot.



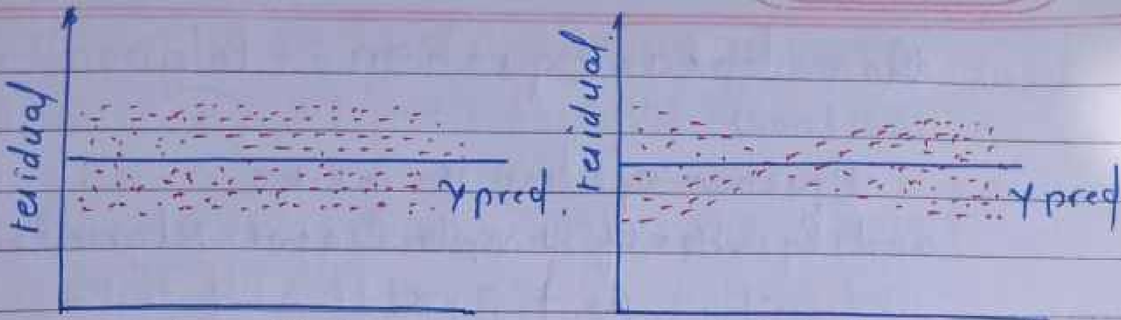
kde



Q-Q plot
(Therotical Quantile)

4. $\boxed{\text{Assumption 4}}$: $\underbrace{\text{Homo}}_{\text{same}}\underbrace{\text{scedasticity}}_{\text{scattered / spread}}$.

: spread of residual should be equally or uniformly scattered. if it is not it called as heteroscedasticity which is not desirable.

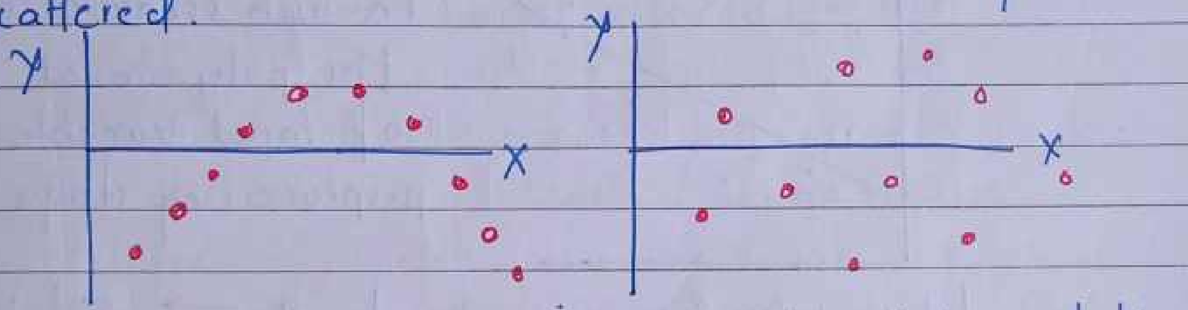


(desirable) (Not desirable)

how to check:- scatter plot (ypred, residual).

Assumption 5: No autocorrelation of Error.

if we plot the error there should not be any specific pattern instead it should randomly scattered.



positive Autocorrelation (Not desirable) Negative Autocorrelation (desirable).

how to check: pll. plot (residual)

Summary chart

	Assumption	Severity	prediction	Inference
1)	linear Relation	High	✓	✓
2)	Multicollinearity.	Medium	x	✓
3)	Normality.	low	x	✓
4)	Homoscedasticity	High	✓	✓
5)	Autocorrelation of Error.	–	–	–

Date: / /
Page

Non-linear Equation - Polynomial Regression

we know

Equation of line $y = mx + c$

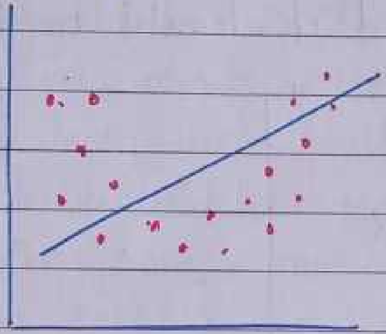
and Equation of simple Linear Regression

$$y = \beta_0 + \beta_1 x$$

and for multiple linear Equation

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_n x_n$$

this is applicable only when data is linear but what if data is not linear ?



In such scenario we extract the polynomial feature of input variable in preprocessing stage

let say for ex $x \mid y \rightarrow 5 \mid 10$

- For x we want to make polynomial of degree 2 then we will convert $x \rightarrow x^0, x^1, x^2, y$ so it will be 1, 5, 25
- this way we create a new data for training the extra polynomial feature try to extract their non-linear relationship.
- its formula become for simple polynomial regression.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

For degree 3

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3.$$

★ Now how we will know the perfect value for the degree _ since this is hyperparameter.

if we keep it low then may be it cause underfitting means it may be not able to learn the all attributes. and if we select very high then there is chance of overfitting or overlearned that's why our job is to find out optimum value

- In case if we have two features x_1, x_2, Y then for degree 2 our simple polynomial Equation would become

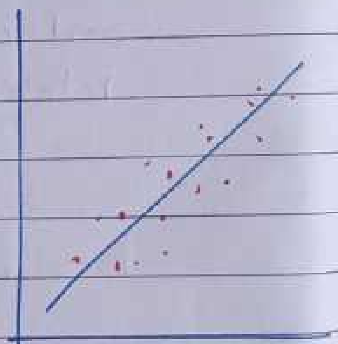
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2 + \beta_4 x_2^4$$

Interview Que: why polynomial Eqn essentially called as Linear Regression

Ans: when we talk about linear Regression we talk about relation between y and coefficients of features and degree of coefficient is still one and thus relation between y and coefficient is still linear

Ordinary least square :- (OLS Algorithm)

- it is method for estimating the parameters of linear regression model.
- it aim to find the values of the linear regression model's parameters (i.e coefficient) that minimize the sum of squared residual.
- the residuals are differences between observed values of dependent variable and predicted values of dependent variable given w.r.t independent variable
- OLS Algorithm assumes that the errors are normally distributed with zero mean and constant variance and that there is no multicollinearity (high correlation) among the independent variables.
- other method like generalized least square or weighted least square, should be used in case when these assumption are not meet.



let understand with problem

x	1	2	3	4	5	6	7
y	1.5	3.8	6.7	9.0	11.2	13.6	16

we will calculate the equation for the best fit line where all the point will be as close as possible by least square method.

x	y	xy	x^2
1	1.5	1.5	1
2	3.8	7.6	4
3	6.7	20.1	9
4	9.0	36	16
5	11.2	56	25
6	13.6	81.6	36
7	16	112	49
Σ 28	61.8	314.8	140

$\Sigma x = 28$ $\Sigma y = 61.8$

$\Sigma xy = 314.8$ $\Sigma x^2 = 140$

$n = 7$

(number of data points)

$$\boxed{y = mx + b}$$

$$m = \frac{n\Sigma xy - \Sigma x \Sigma y}{n\Sigma x^2 - (\Sigma x)^2} = \frac{7(314.8)-(28)(61.8)}{7(140)-(28)^2}$$

$$m = \frac{473.2}{196} = 2.4142857.$$

$$b = \frac{\Sigma y - m\Sigma x}{n} = \frac{61.8 - 2.4142857(28)}{7}$$

$$b = -0.828571.$$

to get linear equation we should plug value in

$$y = mx + b.$$

$\cancel{y = 2.41x}$ $\quad y = 2.41x - 0.83.$

let test it. for 2 $\qquad \hat{y} \qquad$ y_{act}

	$\hat{y}$	y_{act}
$y = 2.41(2) - 0.83 =$	3.99	3.8
$y = 2.41(5) - 0.83 =$	11.22	11.2
$y = 2.41(7) - 0.83 =$	16.09	16.

Syntax: statsmodel.api.OLS (y, x).

y: depedent variable $\qquad$ x - indepedent Variable.

- Import statsmodel.api as sm
  import pandas as pd.

- #reading data from csv
  df = pd.read_csv('train.csv')

- # defining the variables
  x = df['x'].tolist()
  y = df['Y'].tolist()

```
# adding the constant term x
x = sm.add_constant(x)

# performing regression and fitting model.
result = sm.OLS(y, x).fit()

# print summary table
print(result.summary())
```

What is Ridge Regression (L2 Regularization).

Ridge regression is model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization when issue of multicollinearity occurs, least square are unbiased and variance are large, this result in predicted values being far away from actual value.

Regularization:- it is technique used to calibrate Machine learning model to minimize adjusted loss function and avoid overfitting and underfitting.

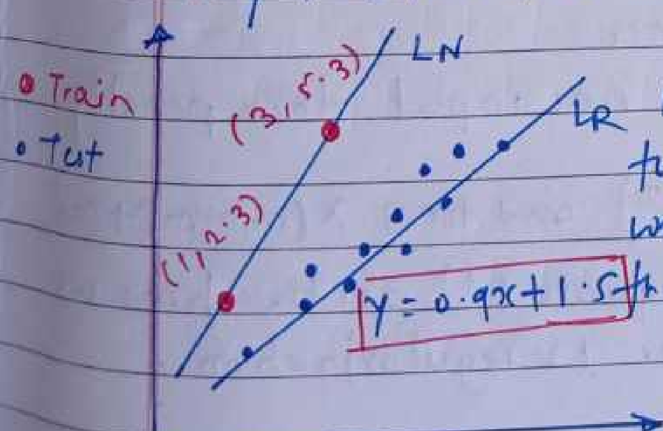
there are three types of regularization techniques:
1) Ridge Regression (L2 regularization)
2) Lasso Regression (L1 regularization)
3) Elastic Net (Combo of Ridge and Lasso)

· Ridge Regression:-

Overfitting:- means your train acc is too high but test accuracy is very low.

$y = mx + b$

m or slope which is coefficient of x is defining change in y, so to reduce overfitting means to reduce slope.



if we train the model only on two datapoint then best fit line would be the line which pass through both point

$y = 1.5x + 0.8$

Now we have to convey our model to choose LR and Not LN

Date: / /
Page

$$L = \underbrace{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}_{} + \underbrace{\lambda (m^2)}_{} \quad \text{—— (1)}$$

before we were only reducing this term which is nothing but Error / Residual

Now we add penalty which will add help for the first term
λ = hyperparameter
m = slope.

Now we will calculate loss for both line for $\lambda = 1$ with eqn (1)

Since line is passing through both point perfectly that's why 1st term will be zero

$L = 0 + 1(1.5)^2$

$\boxed{= 2.25}$

$(2.3 - 0.9 - 1.5)^2 +$
$(5.3 - 2.7 - 1.5)^2 + (0.9)^2$

$= (0.1)^2 + (1.1)^2 + (0.9)^2$

$= \boxed{2.03}$

here we getting significant reduction in loss for the new line
As our model can see this change it will select 2nd model although it will give bad accuracy on training since Variance is significantly reduced although bias increased.

why we called L^2 since
- if we have more than one input then penalty would be
$\lambda(m_1^2 + m_2^2)$ and for 3 $\lambda(m_1^2 + m_2^2 + m_3^2)$

Since we are doing square (^{2}) all the time we called it L2 Norm or L2 regularization.

Lasso Regression (L1 Regularization)

- this is also help to reduce overfitting.
- In Ridge regression we have seen.

$$L = \underbrace{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}_{MSE} + \underbrace{\lambda \|w\|^2}_{\substack{\text{penalty term} \\ \underbrace{(w_1^2 + w_2^2 + \cdots w_n^2)}_{\text{coefficient in MLR}}}}$$

Lasso is just another variation of Ridge

$$L = \underbrace{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}_{} + \lambda \|w\|$$

OR

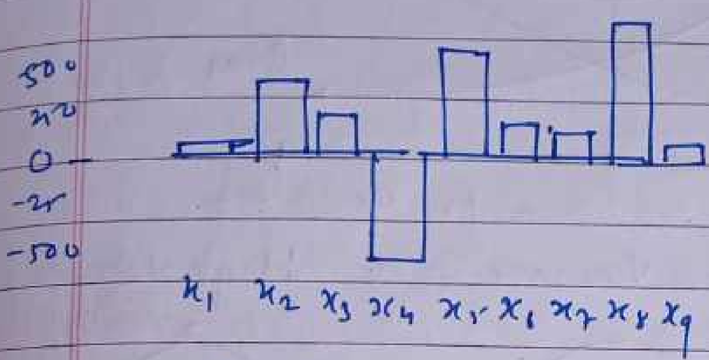
$$= MSE + \lambda [|w_1| + |w_2| + |w_3| + \cdots + |w_n|]$$

- In ridge regression for any value of λ there were always some value for coefficient of input feature but in lasso if you continuously increase value of λ at certain point you will zero value for some of coefficient which are not important so here we unknowing doing feature selection and it is advantage of Lasso.
- so when you are working on high dimensional data and some feature are not imp we should prefer Lasso over Ridge.

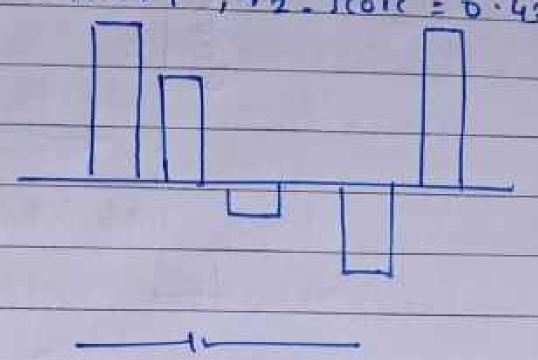
there are some keypoints need to discuss about Lasso.

1) How coefficient are affected. ?

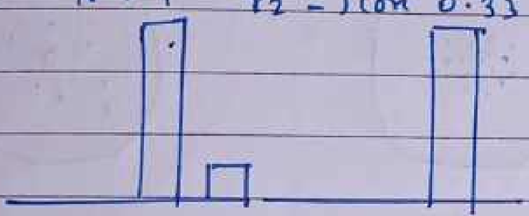
$\lambda = 0$, r2_score = 0.44



$\lambda = 0.1$, r2_score = 0.43



$\lambda = 1$ r2 - score 0.33



$\lambda = 10$ r2-score -0.01

{ All coefficient will be zero }

2. Higher coefficient are affected more

- generally as you increased λ coefficient value will be decrease gradually toward zero
- usually higher coefficients are affected first/rapidly penalize
- for it we should be cautious for selecting optimum value to do proper feature selection

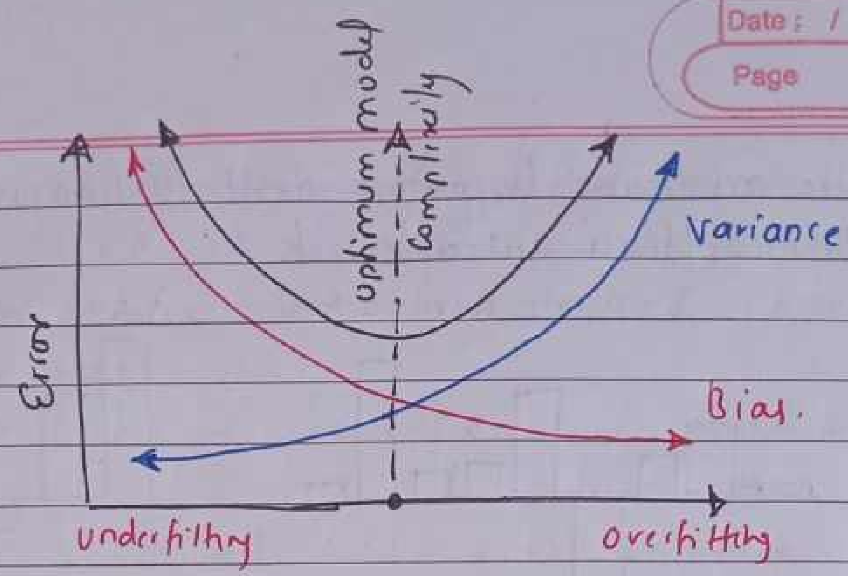
3. Impact on Bias and Variance

- As we know if λ increases, overfitting decreases (↓) which lead to increase in bias (↑)

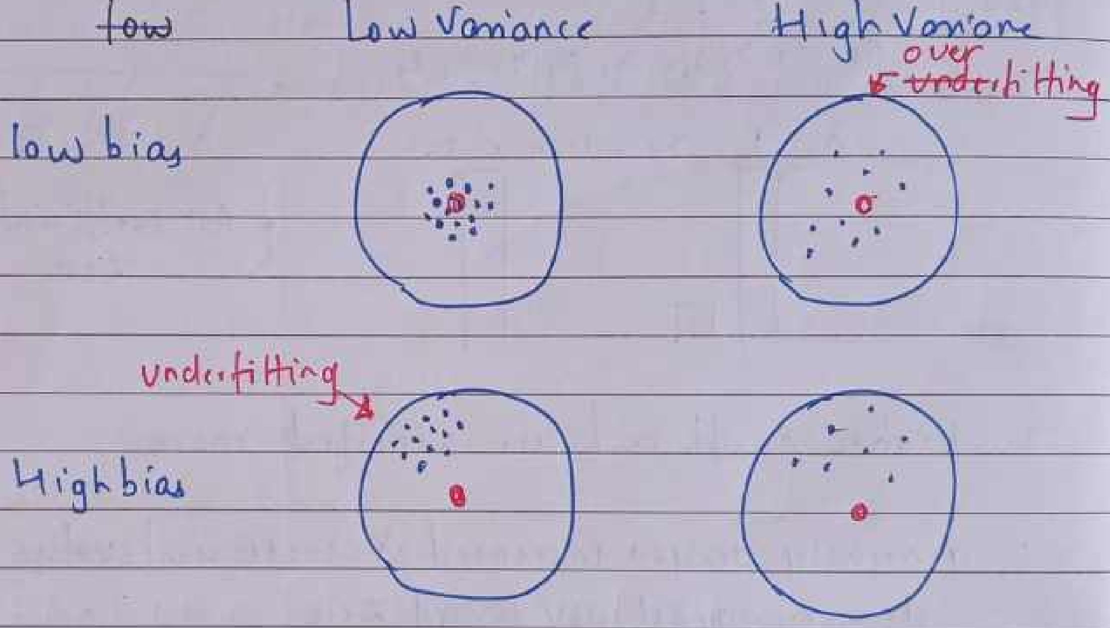
High Bias ↓ Underfitting

High Variance ↓ overfitting

fig a.



fig(B



Bias-Variance Tradeoff.

- it is important to understand prediction errors (bias and Variance) when it comes to accuracy in any ML Algorithm
- There is tradeoff between model's ability to minimize bias and variance which is reffered to as best solution for selecting a value of regularization constant
- proper understanding of these error would help to avoid the overfitting and underfitting of a dataset while training algorithm.

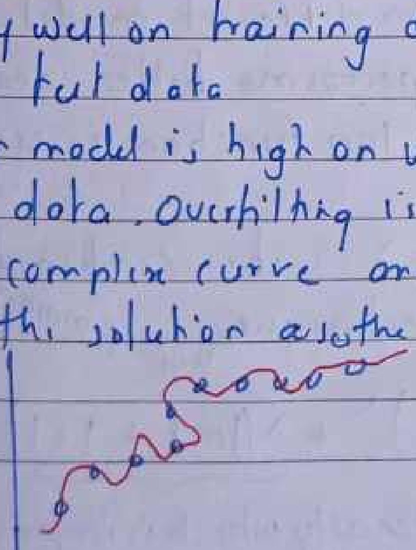
Date: / /
Page

Bias: Bias is known as difference between prediction values by ML model and correct values. Being high in biasing will give large error in training as well as testing. & thats why it is always recommended that algorithm should always be low biased to avoid underfitting.

- by high bias data is predicted is in straight line format thus not fitting accurately in the data in the dataset such fitting called as underfitting.
- this happen the hypothesis is too simple or linear in nature

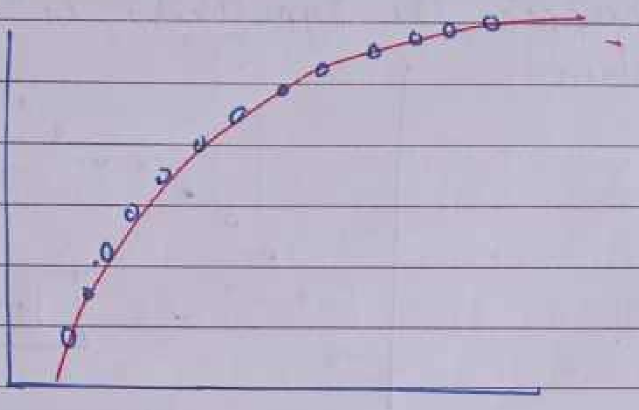
Variance: The variability of model prediction for given data point which tells us spread of our data is called variance of model.

- the model with high variance has a very complex fit to training data and thus not able to fit on the test data or data hasn't seen as result such model works very well on training data but has high error rates on test data
- when model is high on variance it is said to as overfitting of data. Overfitting is fitting the training set accurately via complex curve and high order hypothesis but is not the solution as the error with unseen data is high



high variance data look like follows

- Bias-Variance tradeoff :- if the algorithm is too simple (hypothesis with linear eqn) then it may be on high bias and low variance and thus it is error prone.
- if error fit too complex (hypothesis with high degree equation) then it may be on high variance and low bias, in the latter condition the new entries will not perform well. there is something between both of these condition known as tradeoff or bias-variance tradeoff



4) Effect of Regularization on loss function

loss function :-

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

- it measures how far an estimated value from its true value
- if we are training on different models LR, DT, RF to know which model performs better and which parameters are better loss function is useful.

Elastic Net :- $(L_1 + L_2)$: it is combination of both regularization technique.

$$L_1\,Reg = \frac{\sum(y_i - \hat{y}_i)^2}{n} + \underbrace{\overbrace{\lambda}^{\text{hyperparameter}}(|m| + |c|)}_{\text{penalty which is imposed on parameter}}$$

L_2 reg $= \dfrac{\Sigma(y_i - \hat{y}_i)^2}{n} + \underbrace{\lambda(m^2 + c^2)}_{\text{penalty}}$

Elastic ($L_1 + L_2$) :-

$$\frac{\Sigma(y_i - \hat{y}_i)^2}{n} + \lambda[(|m| + |c|) + (m^2 + c^2)]$$

adding combination percentage of Lasso & Ridge

$$\frac{\Sigma(y_i - \hat{y}_i)^2}{n} + \lambda[\underbrace{c(|m| + |c|)}_{\text{lasso}} + \underbrace{(1-c)(m^2 + c^2)}_{\text{Ridge}}]$$

c is between 0 to 1

$c = 1$ Lasso

$C = 0$ RIDGE

$c = 0.05$ = 50% Ridge & 50% Lasso.

Summary :- Ridge is majority is going to focus on regularization but Lasso is going to focus on feature selection as well

- if my job is only feature selection i will go for Lasso and if i want regularization then i will go for Ridge and if i want both the i would go for Elastic Net.

Date: / /
Page.

Time series forecasting.

Time series :- it is data which is index by time.
- the most important part of time series is sequence.

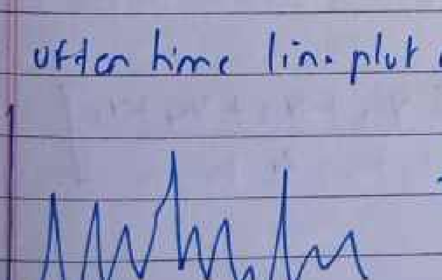
what we do with time series :- we capture the past data & based on that we predict for future this entire procedure is called forecasting

A). plotting time series
B). component of time series
C). forecasting in time series
  - (1) Data based prediction
  - (2) model based prediction

A) line plot :-

- line plot gives holistic view or overall view.
- line plot gives having understanding about long term i.e for long term what happened to my data.

- often time line plot are difficult to understand.



to overcome this we have technique called smoothing.

Date: / /
Page

how to do smoothing :-

moving average :- Ex i have rainfall data

Months	M_1	M_2	M_3	. . . .	M_{120}
data	y_1	y_2	y_3	. . . .	y_{120}

where M_1 - January
M_2 - february
M_3 - March.

for window size 3

$$y_2 = \frac{y_1 + y_2 + y_3}{3} \qquad y_3 = \frac{y_2 + y_3 + y_4}{3}$$ and so on —

- but it would not be same for first and last data point since they will not have one datapoint so their value will remain as it is.
- for window size 5 first two and last two will remain same.
- for even number of window size we wouldn't get balanced number

$y_1 \quad y_2 \quad y_3 \quad y_4 \quad y_5 \quad y_6 \quad y_7 \quad y_8$

$$y_3 = \frac{y_2 + y_3 + y_4 + y_5}{4}$$ {giving high priority to future data}

$$y_3 = \frac{y_1 + y_2 + y_3 + y_4}{4}$$ {giving high priority to past data}

Solution is :-

$$= \frac{1}{2}\left[\frac{y_1 + y_2 + y_3 + y_4}{4}\right] + \frac{1}{2}\left[\frac{y_2 + y_3 + y_4 + y_5}{4}\right]$$

• How to figure out best window size?

1. Way of domain knowledge.

Eg. sales of AC is based on season like in summer it sold out rapidly whereas in winter is not seem that like expected.

so here we can assume or try window size 3/4

2. When we don't have domain knowledge another way is by calculating size for all and check where it becomes smooth!

Component of time series:

it generally has three components.

1) Trend — holistic view - Moving average.

2) Seasonality - periodic change

3) Resid — Noise or Error - lesser the better.

Decomposition of T.S [Trend, Seasonality, Noise]

• Trend:- find optimum window size

• To find seasonality & Noise substract trend from original data.

let say we have data 2001 to 2010

Month	M_1	M_2	M_3	M_4	. . . .	M_{120}
data	Y_1	Y_2	Y_3	Y_4	. . . .	Y_{120}

Trend : $T_1 = M_1$, $T_2 = \frac{M_1 + M_2 + M_3}{3}$, $T_3 = \frac{M_2 + M_3 + M_4}{3}$

Seasonality : $S_1 = Jan = \frac{T_1 + T_{13} + T_{25} + \cdots T_{109}}{10}$

$S_2 = Feb = \frac{T_2 + T_{14} + T_{26} + \cdots T_{110}}{10}$

Noise : $N_1 = Y_1 - T_1 - S_1$, $N_2 = Y_2 - T_2 - S_2$, ... $N_{120} = Y_{120} - T_{120} - S_{120}$.

forecasting in time series.

Data based forecasting :-

① Simple Exponential :- it is applicable to those time series where there is no trend no seasonality.

② Double Exponential :- it is only performed good for those time series where it only have trend but not seasonability.

③ Holt winter's model - (Triple Exponential) it is performed well where there is both trend as well as seasonality (when you don't know whether time series have trend or seasonality then holt winter is best)

→ Simple Exponential :-

Day	D_1	D_2	D_3	D_{199}	D_{200}	D_{201}
temp	t_1	t_2	t_3	t_{199} ←	t_{200}	
					Today →	(predict)

what will be the easiest way to predict for tommorow's temp when we have data upto today?

① taking Avg :- $\frac{t_1 + t_2 + t_3 + \cdots t_{200}}{200} = x.$

but this not good method since temp varies a lot throught year summer to winter

(ii) another way is whatever is today's temp it will same for tomorrow.

$t_{day\,200} = 25.7°$

$t_{day\,201} = 25.7°.$

Simple Exponential:- instead of taking all 200 data into consideration. we will take recent data for prediction.

- Simple Exponential try to calculate the component in local Average.

Day	D_1	D_2	D_3	D_4	. . . . .	D_{199}	D_{200}
temp	t_1	t_2	t_3	t_4		t_{199}	t_{200}
	L_1	L_2	L_3	L_4		L_{199}	L_{200}

- local Average at Day 4 i.e L_4 will be combination of two component yesterday's Local Average and today temp

$$L_3 \swarrow L_4 \searrow t_4 \quad = L_4 = \frac{1}{2} L_3 + \frac{1}{2} t_4$$

Note $L_1 \rightarrow t_1$ since there is no previous record.
Simple Exponential is applicable second point onwards.

Mathematical Understanding :-

$$L_{200} = \frac{1}{2} L_{199} + \frac{1}{2} t_{200} \quad \text{—— (1)}$$

$$L_{199} = \frac{1}{2} L_{198} + \frac{1}{2} t_{199} \quad \text{—— (2)}$$

$$L_{200} = \frac{1}{2}\left[\frac{1}{2} L_{198} + \frac{1}{2} t_{199}\right] + \frac{1}{2} t_{200} \ldots \text{2 in (1)}$$

$$= \frac{1}{4} L_{198} + \frac{1}{4} t_{199} + \frac{1}{2} t_{200} \quad \text{—— (3)}$$

$$L_{198} = \frac{1}{2} L_{197} + \frac{1}{2} t_{198} \quad \text{—— (4)}$$

put (4) in (3)

$$\frac{1}{4}\left[\frac{1}{2} L_{197} + \frac{1}{2} t_{198}\right] + \frac{1}{4} t_{199} + \frac{1}{2} t_{200}$$

$$\frac{1}{8} L_{197} + \frac{1}{8} t_{198} + \frac{1}{4} t_{199} + \frac{1}{2} t_{200}$$

Date: / /
Page

if i rearrange this, we will get

$$L_{200} = \frac{1}{2} t_{200} + \frac{1}{4} t_{199} + \frac{1}{8} t_{198} + \frac{1}{16} t_{197} \ldots$$

$$\left(\tfrac{1}{2}\right)^1 \quad \left(\tfrac{1}{2}\right)^2 \quad \left(\tfrac{1}{2}\right)^3 \quad \left(\tfrac{1}{2}\right)^4$$

· from current datapoint weightage of coefficient reduce exponentially. whereas in simple average every coefficient were getting equal average.

let understand with short Example.

Day	1	2	3	4	5	6
Score	1	2	1	2	1	3

$$L_1 \rightarrow L_2 \rightarrow L_3 \rightarrow L_4 \rightarrow L_5 \rightarrow L_6$$

$\boxed{L_1 = 1}$, $L_2 = \frac{L_1 + S_2}{2} = \frac{1+2}{2} = \boxed{1.5}$

$L_3 = \frac{L_2 + S_3}{2} = \frac{1.5+1}{2} = \frac{2.5}{2} = \boxed{1.25}$ $\quad L_4 = \frac{L_3 + S_4}{2} = \frac{1.25+2}{2} = \frac{3.25}{2} = \boxed{1.625}$

$L_5 = 1.3125$, $L_6 = 2.1562$

All future prediction going to be last available local Average.

Double Exponential :- it contain trend but no seasonality.

· we can't always rely on local Average because for ex. if chocklate price increase by Rs 1 each day

Day	1	2	3	4	5	6
price	1	2	3	4	5	?

L_6

L_6 = around 4.2 which is not true.

Hence we need consider trend as well., that's why we need both trend as well as local Average

for local Avg

$$L_{200}$$

L_{199} {yesterday local Avg} Y_{200} {today's data}

∴ but local avg is also expected to change by some amount.

there is little modification in local Average

$$L_{200} = \frac{1}{2} L_{199} + \frac{1}{2} Y_{200}$$

$$L_{200} = \frac{1}{2}(L_{199} + T_{199}) + \frac{1}{2} Y_{200}$$

trend is added.

$$\underset{\text{(today's trend)}}{T_{200}} = \frac{1}{2} T_{199} + \frac{1}{2} \underbrace{(L_{200} - L_{199})}_{\text{change in local Average from yesterday to today}}$$

Raw data	Y_1	Y_2	Y_3	Y_4	. . . . .	Y_{200}
Local Avg	L_1	L_2	L_3	L_4	. . . . .	L_{200}
Trend	T_1	T_2	T_3	T_4	. . . . .	T_{200}

so from raw data we will calculate Local Avg & trend.

$L_1 = 0$
$T_1 = 0$

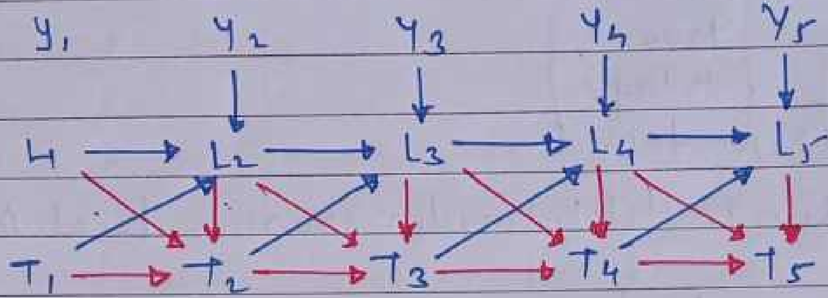
$$L_2 = \frac{1}{2}(L_1 + T_1) + \frac{1}{2}(Y_2)$$

$$T_2 = \frac{1}{2}(T_1) + \frac{1}{2}(L_2 - L_1).$$

Let us understand with Example, we will solve problem, both Simple Exponential & Double Exponential.

Day	1	2	3	4	5	$\frac{1+2}{2} = 1.5$
price (y)	1	2	3	4	5	$\frac{1.5+3}{2} = \frac{4.5}{2} = 2.2$
L(S)	1	1.5	2.25	3.125	4.0625	
L(D)	1	1.5	2.375	3.4687	4.6484	

Now we will calculate with Double Exponential.



$$L_2 = \tfrac{1}{2}(L_1 + T_1) + \tfrac{1}{2}(y_2)$$

$$T_2 = \tfrac{1}{2}(T_1) + \tfrac{1}{2}(L_2 - L_1)$$

$$L_2 = \tfrac{1}{2}(1+0) + \tfrac{1}{2}(2) = \tfrac{1}{2} + 1 = 1.5$$

$$T_2 = \tfrac{1}{2}(0) + \tfrac{1}{2}(1.5 - 1) = 0 + \frac{0.5}{2} = 0.25$$

$$L_3 = \frac{1}{2}(L_2 + T_2) + \frac{1}{2}(y_3)$$

$$= \frac{1}{2}(1.5 + 0.25) + \frac{1}{2}(3) = \tfrac{1}{2}(1.75) + 1.5$$

$$= 0.875 + 1.5$$

$$= 2.375$$

$$T_3 = \tfrac{1}{2} T_2 + \frac{1}{2}(L_3 - L_2)$$

$$= \tfrac{1}{2}(0.25) + \tfrac{1}{2}(2.375 - 1.5)$$

$$= 0.125 + 0.4375$$

$$= 0.5625$$

$L_4 = \frac{1}{2}(L_3 + T_3) + \frac{1}{2}(Y_4)$

$= \frac{1}{2}(2.375 + 0.5625) + \frac{1}{2}(4)$

$= \frac{2.9375}{2} + 2$

$= 1.4687 + 2 = 3.4687$

$T_4 = \frac{1}{2}(T_3) + \frac{1}{2}(L_4 - L_3)$

$= \frac{1}{2}(0.5625) + \frac{1}{2}(3.4687 - 2.375)$

$= 0.28125 + 0.5(1.0937)$

$= 0.28125 + 0.54685$

$= 0.8281$

$L_5 = \frac{1}{2}(L_4 + T_4) + \frac{1}{2}(Y_5)$

$= \frac{1}{2}(3.4687 + 0.8281) + \frac{1}{2}(5)$

$= 2.1484 + 2.5$

$= 4.6484$

$T_5 = \frac{1}{2}T_4 + \frac{1}{2}(L_5 - L_4)$

$= \frac{1}{2}(0.8281) + \frac{1}{2}(4.6484 - 3.4687)$

$= 0.41405 + 0.5(1.1797)$

$= 0.41405 + 0.58985$

$= 1.0039$

$$\left.\begin{array}{ccccc} T_1 & T_2 & T_3 & T_4 & T_5 \\ 0 & 0.25 & 0.56 & 0.82 & 1.0039 \end{array}\right\} \text{Trend is increased.}$$

	Actual	Predicted
Simple Exponential	4.0625	5
Double Exponential	4.6484	5

Date: / /
Page

Holt's winter model :- which contain both trend & seasonality.

- Here along with local Average we will find out trend as well as seasonlity.

time series data

	Y_1	Y_2	Y_3		Y_{200}
A	L_1	L_2	L_3		L_{200}
trend	T_1	T_2	T_3		T_{200}
seasonality	S_1	S_2	S_3		S_{200}

for 200 Months $\longrightarrow$ 16 yrs - 7 month
for seasonlity - periodic time of 12 month

let first convert it into double exponential problem. by subtracting seasonality from time series

$Y_1 - S_1,\ Y_2 - S_2,\ Y_3 - S_3 \quad - - - \quad Y_{200} - S_{200}$

$L_{200} = \frac{1}{2}(L_{199} + T_{199}) + \frac{1}{2}(Y_{200} - S_{200})$ —— (1)

{ Note Since $Y_{200} = Y_{200} - S_{200}$ }

$T_{200} = \frac{1}{2}(T_{199}) + \frac{1}{2}(L_{200} - L_{199})$ —— (2)

$S_{200} = \frac{1}{2}(S_{200-12}) + \frac{1}{2}(S_{200})$

From (1) we can get to know to get L_{200} we substract S_{200} from Y_{200} then now S_{200} we can do $Y_{200} - L_{200}$.

$S_{200} = \frac{1}{2}(S_{200-12}) + \frac{1}{2}(Y_{200} - L_{200})$ —— (3)

Date: / /
Page

if you are carefully observe then you find there is loop between ③ equations.

for L_{200} we need S_{200} for S_{200} we need L_{200}

Since seasonality is periodic

$S_{200} = S_{188}$

in eq^n ① $S_{200} = S_{188}$

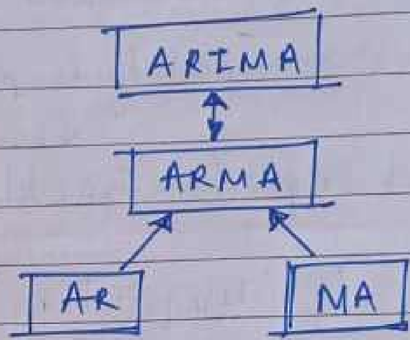
$$\boxed{\begin{aligned} L_{200} &= \tfrac{1}{2}(499 + T_{199}) + \tfrac{1}{2}(Y_{200} - S_{188}) \\ T_{200} &= \tfrac{1}{2}(T_{199}) + \tfrac{1}{2}(L_{200} - L_{199}) \\ S_{200} &= \tfrac{1}{2}(S_{200-12}) + \tfrac{1}{2}(Y_{200} - L_{200}) \end{aligned}}$$

Summary table.

Model	when we can apply	what should be future prediction	hyperparameter
Simple Exponential	No trend No seasonality.	last possible local Avg = all future prediction	1
Double Exponential	only trend No seasonality	last possible local Avg & changed trend	2
Tripple Exponential	both trend and seasonality.	local Average + trend + seasonality.	3.

Model Based forecasting :-

Basically there are three models.
1) AR
2) MA
3) ARMA
4) ARIMA



Here data must be stationary. it means distribution of data should be same across the time

(stationary) ✓ (Not stationary).

if we have stationary data then go for model based otherwise databased since there is no any condition

AR (Auto Regressive) : AR is completely depend upon past data and nothing else for future prediction

MA: Moving Average! future data is only depend only external outsiders factors.

ARMA:- future data only going to deal with both past data data and external factors

AR model :-

In regression $y = \alpha x + \beta + \varepsilon$

future y $y_t = \alpha y_{t-1} + \beta + \varepsilon$ Error

past y Beta

Date : / /
Page

Here Future data is only depend on past data
if it would depend on past two data

$$y_t = \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \beta + \varepsilon \quad AR(2)$$

$$y_t = \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \alpha_3 y_{t-3} + \beta + \varepsilon \quad \ldots A(3)$$

How to select Best possible model. AR1, AR2, AR3
~~AR4~~ based on test data accuracy whichever will perform good will finalize that.

MA : Moving Average Here my future data is only depend upon external factors.
Ex rate of some production which depends upon seasons. (grain or fruits) due to natural calamities.

data	y_1	y_2	y_3	...	y_t
External factors	ε_1	ε_2	ε_3	...	ε_t

$$y_t = \alpha \varepsilon_{t-1} + \beta + \varepsilon_t \quad \ldots \quad MA(1)$$

this year data

last year data

constant

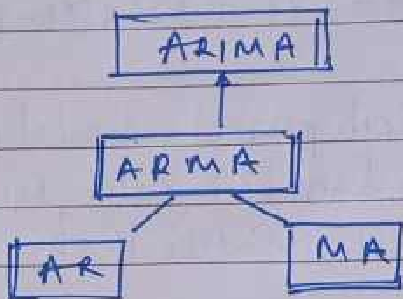
$$y_t = \alpha_1 \varepsilon_{t-1} + \alpha_2 \varepsilon_{t-2} + \beta + \varepsilon_t \quad \ldots \quad MA(2)$$

Date: / /
Page

Model based forecasting.

1. AR
2. MA
3. ARMA (AR + MA)
4. ARIMA (which is come by ARMA itself)

→



AR → future prediction are completely depend on past data, it is just copy of regression model.

$$\underbrace{y}_{\text{target variable}} = \underbrace{\alpha x + \beta}_{\text{prediction}} + \underbrace{\varepsilon}_{\text{error}}$$

$$y_t = \underbrace{\alpha y_{t-1}}_{\text{past data}} + \beta + \varepsilon \quad \text{———} \quad AR(1)$$

it it is depend on one past data

$$\underbrace{y_t}_{\text{tomorrow}} = \alpha_1 \underbrace{y_{t-1}}_{\text{today}} + \alpha_2 \underbrace{y_{t-2}}_{\text{yesterday}} + \alpha_3 \underbrace{y_{t-3}}_{\text{day before yesterday}} + \beta + \varepsilon$$

it is similar to SLR

SLR :	$y = \alpha x + \beta + \varepsilon$	$AR(1) = \alpha y_{t-1} + \beta + \varepsilon_t$
2 features	$y = \alpha_1 x_1 + \alpha_2 x_2 + \beta + \varepsilon$	$AR(2) = \alpha y_{t-1} + \alpha y_{t-2} + \beta + \varepsilon_t$

- one of the necessary condition for apply these data is data should be stationary i.e it means distribution of data should be same across all the data point.
- since among all four model ARMA, AR and MA are very critical for that we must sure that data should be stationary.

ARIMA
↑

ARMA
AR MA

since they are critical must sure for stationary data.

$\boxed{MA \longrightarrow \text{Moving Average}}$:- here future prediction are completely depend upon external factors.

production of n'u →	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
Error due to External → factor	ε_1	ε_2	ε_3	ε_4	ε_5	ε_6	ε_7

- when we say our data is only depending on last time point for example for 8th year

$$Y_8 = \alpha \varepsilon_7 + \beta + \varepsilon_8$$

so general formula is

$$Y_t = \alpha \varepsilon_{t-1} + \beta + \varepsilon_t \text{ ——— Moving Avg (MA)}$$

depend on External Error factors.

$$Y_t = \alpha y_{t-1} + \beta + \varepsilon_t \text{ ——— Auto Regressive (AR)}$$

depend on past data.

for MA(2) and MA(3) equation would be.

$MA(2)$: $y_t = \alpha_1 \varepsilon_{t-1} + \alpha_2 \varepsilon_{t-2} + \beta + \varepsilon$

$MA(3)$: $y_t = \alpha_1 \varepsilon_{t-1} + \alpha_2 \varepsilon_{t-2} + \alpha_2 \varepsilon_{t-3} + \beta + \varepsilon$.

- it is exactly similar to AR model only difference is instead of previous data here external factors are intrested.

$\boxed{\text{ARMA}}$: it is combination of AR and MA model.

$$\boxed{\begin{array}{c} ARMA(1,1) \\ AR(1) \quad + \quad MA(1) \end{array}}$$

$ARMA(p, q) = AR(p) + MA(q)$

$y_t = \alpha_1 y_{t-1} + \underline{\beta_1 \varepsilon_{t-1}} + \beta_0 + \varepsilon_t$.

if $\beta_1 \varepsilon_{t-1} = 0$ then it will be $AR(1)$

$y_t = \underline{\alpha_1 t_1 - 1} + \beta_1 \varepsilon_{t-1} + \beta_0 + \varepsilon_t$.

if $\alpha_1 t_1 - 1 = 0$ then it will be $MA(1)$

- thats why you dont need to put MA and AR model seperately.
- if you think your model mostly inclined to MA then we can put AR component zero if model inclined to AR then we can put MA component zero.

Date: / /
Page

- thing we need to take of for these three models
i) stationary :: if it is stationary we gonna directly used ARMA insted of using AR and MA model seperately.
ii) ARMA model has two hyperparameter (p, q)
p corresponds to AR and q corresponds to MA

how to choose best value of p and q.
by creating multiple model in range of (0 to 5) for each p and q

$p : \{0, 1, 2, 3, 4, 5\} = 6 \quad 6 \times 6 = 36$

$q : \{0, 1, 2, 3, 4, 5\} = 6$

since 0,0 is not possible pair we have 36 - 1 = 35

$\boxed{\text{ARIMA}}$: ARIMA doesn't care about data is stationary or Not.
this not a new model but extension of ARMA
ARIMA - Auto Regressive integrated moving average.

- if transformed data from original data follows ARMA model then original data will must follow ARIMA.

$$\underbrace{\begin{matrix} Y_1 & Y_2 & Y_3 & Y_4 & \cdots & Y_n \\ \downarrow & \downarrow & \text{transfored data} & & \downarrow & \downarrow \\ Z_1 & Z_2 & Z_3 & Z_4 & & Z_n \end{matrix}}$$

if this $\{Z_1, Z_2 \cdots Z_n\}$ follows ARMA then $(Y_1, Y_2 \cdots Y_n)$ will must follow ARIMA.

for ex.

Bank balance of Every month.

$y_1 \quad y_2 \quad y_3 \quad y_4 \quad \ldots \quad y_t$ ← (today)

{ here money is added every month not due to new added but due to interest }

what will interest added each month

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \quad \downarrow$

$Z_1 \quad Z_2 \quad Z_3 \quad Z_4 \quad \quad Z_t$

where $Z_1 = y_2 - y_1$
$Z_2 = y_3 - y_2$
$Z_3 = y_4 - y_3$

} this is newly constructed data if this follow ARMA the parent data will follow ARIMA

ARMA has two component (p, q) whereas
ARIMA has three component (p, q, d).

d – degree of substraction

d - for substraction of consecutive numbers.

d = 1	d = 2	d = 3
$Z_1 = y_2 - y_1$	$Z_1 = y_3 - y_1$	$Z_1 = y_4 - y_1$
$Z_2 = y_3 - y_2$	$Z_2 = y_4 - y_2$	$Z_2 = y_5 - y_2$
$\vdots$	$\vdots$	$\vdots$
Z_{100} $y_{101} - y_{100}$	$Z_{100} = y_{102} - y_{100}$	$Z_{100} = y_{103} - y_{100}$

what is lifecycle of ARIMA

Raw data → Substraction degree (Transformotion) → Apply ARMA on New Model.

Summary table.

Model	No of parameter	Conditions	depends on
AR	1	stationary	past
MA	1	stationary	External factor
ARMA	2	stationary	past + External factor
ARIMA	3	No need - since I'm transforming the data.	

forecasting

- **Data Based** — there is no previous condition
  - Simple Exponential — No trend No seasonality.
  - Double Exponential — trend ✓ but No seasonality.
  - Holt winters — both trend & seasonality.
  - { whenever we don't know which to used we can used directly Holt winter }
- **Model Based**
  - AR, MA, ARMA — { stationary data }
  - ARIMA
  - { when we are not sure directly go for ARIMA }

Evoluchion of time series Model.
there are two matrix we gonna used.
(1) Mean Square Error (2) MAPE score.

Mean square error.
let say i have so many future data.

Actual	Prediction
y_t	$\hat{y}_t$
y_{t+1}	$\hat{y}_{t+1}$
y_{t+2}	$\hat{y}_{t+2}$

$$MSE = \frac{(y_t - \hat{y}_t)^2 + (y_{t+1} - \hat{y}_{t+1})^2 + (y_{t+2} - \hat{y}_{t+2})^2}{n}$$

(2) MAPE Score. (Mean Absolute Percentage Error)
let compare 2 scenarios.

	S1	S2
Actual	3	1000
predicted	2	999.

which is more accurate ?
{ if you try mse for both it would be same }
· it means sometime we are not interested in amount of error but percentage of error

· formula for MAPE scores is

$$\frac{|y_A - y_P|}{y_A}$$ — this is value of percentage

final can = $\frac{|3-2|}{3} = 1/3 = 33.3\%$.

Date : / /
Page

Second case : $\left| \frac{1000 - 999}{100} \right| = 0.001$

0.01%

Since we are not only concerned about Absolute error always but absolute percentage error also.

Exercies :- Calculate the MSE and MAPE score

Actual	Predicted
5	10
20	15
30	25

$$MSE = \frac{(5-10)^2 + (20-15)^2 + (30-25)^2}{3}$$

$$= \frac{(-5)^2 + (5)^2 + (5)^2}{3} = 25$$

MAPE score :

$$\left| \frac{y_A - y_P}{y_A} \right| = \left| \frac{5-10}{5} \right| = -5/5 = -1 = 100\%$$

$$= \left| \frac{20-15}{20} \right| = 25\%$$

$$= \left| \frac{30-25}{25} \right| = 16.6\%$$

$$\text{Mean abs error} = \frac{100+25+16.6}{3} = \underline{\underline{47.2}}$$

Date: / /

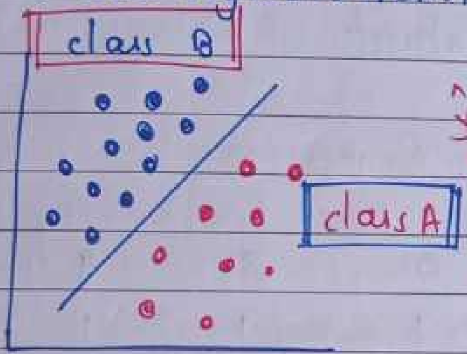
Page

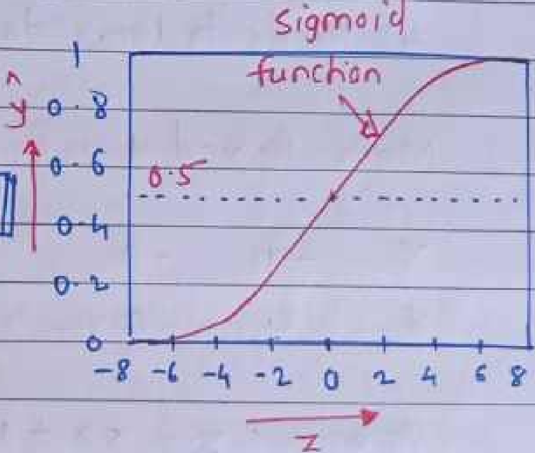
Classification Algorithm

1) Logistic Regression 2) SVM
3) KNN
4) Decision Tree 5) Random forest

1) logistic Regression:-

1. it is supervised learning model.
2. it is classification model and best for binary classification.
3. it uses sigmoid function





$$\text{Sigmoid function} = \hat{y} = \frac{1}{1+e^{-z}}$$

where $z = wx + b$.
$(mx + c)$ - eqⁿ of line

slope m $\longrightarrow$ weight w
intercept c $\longrightarrow$ bias b

- $\hat{y}$ = probability that $(y=1)$

$\hat{y} = P(y=1|x)$ ... {probability of y being 1 for given value of x}

x = input features
w = weights (it will be in the format)
{number of weight equal to number of feature in the dataset}
b = bias
$\hat{y} = \sigma(z)$.

Date: / /
Page

Advantages: 1) Easy to implement
2) perform well on data with linear relationship
3). less prone to overfitting for low dimensional dataset.

Disadvantages:- 1) High dimensional dataset causes overfitting.
2) difficult to capture complex relationship in dataset
3) sensitive to outlier
4) Needs to large dataset.

Math Behind logistic Regression.

X	-9	-8	0	8	9
y	0	0	1	1	1

Assume · $z = 5x + 10$ $\hat{y} = \dfrac{1}{1+e^{-z}}$

$x = -9$	$x = -8$	$x = 0$	$x = 8$	$x = 9$
$z = 5(-9)+10$ $= -35$	$z = 5(-8)+10$ $= -30$	$z = 5(0)+10$ $= 10$	$z = 5(8)+10$ $= 50$	$z = 5(9)+10$ $= 55$
$\hat{y} = \dfrac{1}{1+e^{35}}$	$\hat{y} = \dfrac{1}{1+e^{30}}$	$\hat{y} = \dfrac{1}{1+e^{10}}$	$\hat{y} = \dfrac{1}{1+e^{50}}$	$\hat{y} = \dfrac{1}{1+e^{55}}$
$\hat{y} = 0$	$\hat{y} = 0$	$\hat{y} = 1$	$\hat{y} = 1$	$\hat{y} = 1.$

Date: / /
Page

Inference :. if z value is large positive number.

$$\hat{y} = \frac{1}{1+0} \approx \hat{y} = 1.$$

if z is large negative number.

$$= \frac{1}{1 + (\text{large positive number})}$$

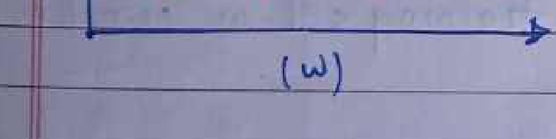
$$\hat{y} = 0.$$

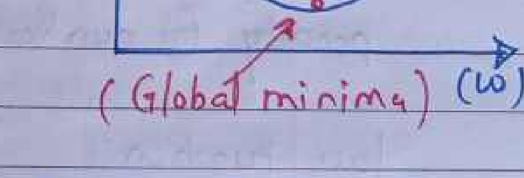
loss function & cost function for Logistic Regression.

- loss function measures how far an estimated value is from true value.

loss function for linear regression $= \frac{1}{n} \sum_{i=1}^{n} (y_i - y_{pred})^2$

if we use this function we will get many local minima instead of getting global minima.





- Binary cross entropy loss function (or) log loss.

$$L(y, \hat{y}) = -(y \log \hat{y} + (1-y) \log(1-\hat{y}))$$

Here,

$y \longrightarrow 0$ or 1

Illy $\hat{y} \longrightarrow 0$ to 1 (probability could be continuous) when $y=1$

$$L(1,\hat{y}) = -(1 \log \hat{y} + (1-1)\log(1-\hat{y}))$$
$$= -\log \hat{y}$$

- since we always won't smaller loss function value hence $\hat{y}$ should be very large (from 0 to 1) if it is the $-\log \hat{y}$ will be very large negative number or very small number.

when $y=0$

$$L(1,\hat{y}) = -(0 \log \hat{y} + (1-0)\log(1-\hat{y}))$$
$$= -\log(1-\hat{y}).$$

- since we want smaller loss function value, hence $\hat{y}$ should be very small the automatically $(1-\hat{y})$ will be very large thus $-\log(1-\hat{y})$ will be large negative number or very small number.

- Cost function is nothing but mean average of loss function
- loss function (L) mainly applies for single training set as compared to cost function (J) which deals with a penalty for number of training set or complete batch.

loss function:

$$L(y,\hat{y}) = -(y\log\hat{y} + (1-y)\log(1-\hat{y}))$$ — for single

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\left(L(y^{(i)},\hat{y}^{(i)})\right) =$$

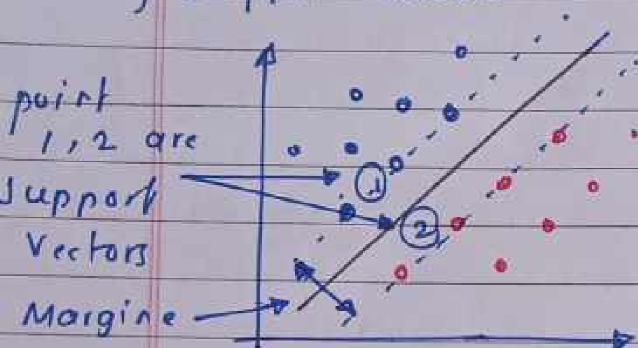
$$-\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})\right)$$

{ 'm' denotes number of data points in the training set }

Support Vector Machine (SVM)

Basic about SVM.

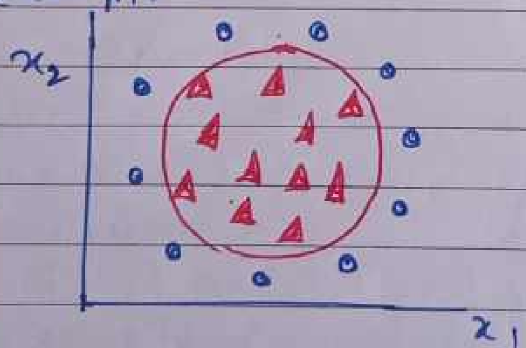
1) it is supervised ML model.
2) it can be used for both classification as well as regression but it is predominantly used for binary classification.
3) Hyperplane.
4) Support vectors



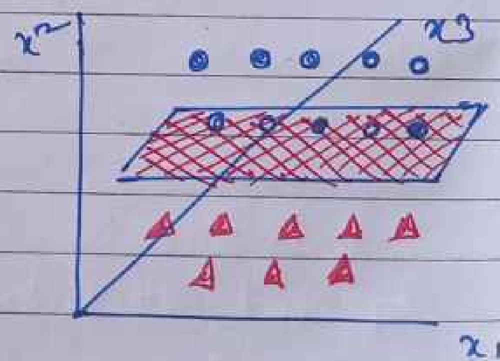
position of hyperplane depends upon support vectors

- for 2D data it is easy to draw hyperplane but where data point can't seperated by line need to convert into 3D where we can seperate the datapoint by hyperplane

Example :-



{Here Not easy to seperate by line}



2D → 3D
and seperate by hyperplane

Hyperplan :- Hyperplane is line (in 2D) or plane that seperate the data point into two classes

Support vectors :- these are the datapoints which are nearest to hyperplane if these datapoints changes position of hyperplane changes.

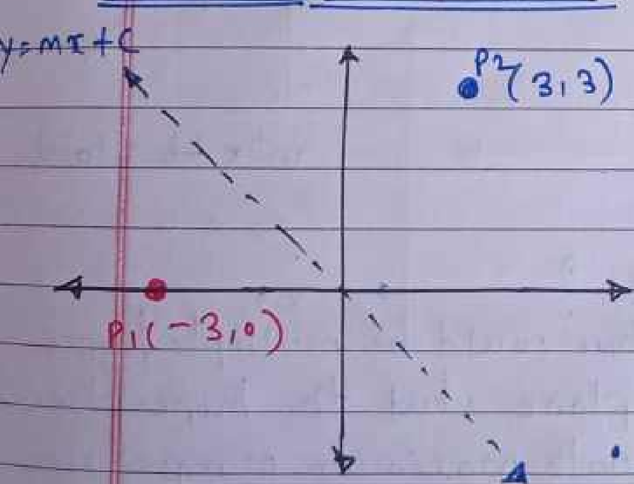
Advantages of SVM.

1) works fine with smaller dataset
2) works fine or efficiently where there is clear margine of seperation
3) works well with high dimensional data

Disadvantages

1) Not suitable for large dataset as training time would take very large.
2) Not suitable for noiser (outlier) dataset with overlapping classes.

Math Behind SVM.



let slope and intercept of hyperplane is,

$m = -1$

$c = 0$ {since passing through origine}

- let parameters of hyperplane save in W which is nothing but weight

$w \rightarrow (m, c) = (-1, 0)$

- let multiply x or P_1 by transpose of w

$$w^T x = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \begin{bmatrix} -3 & 0 \end{bmatrix} = 3 \quad \text{(positive)}$$

[Note: why transpose? → for matrix multiplication no of column of 1st Matrix must be equal ~~to~~ no. of rows of second matrix]

- positive value indicates all the points of hyperplane will be positive class.

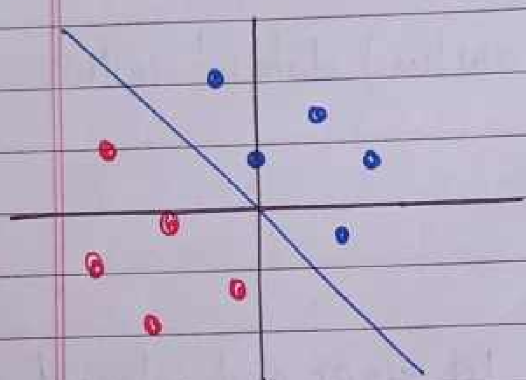
for $P_2(3, 3)$

$$w^T x = \begin{bmatrix} -1 \\ 0 \end{bmatrix} [3, 3]$$

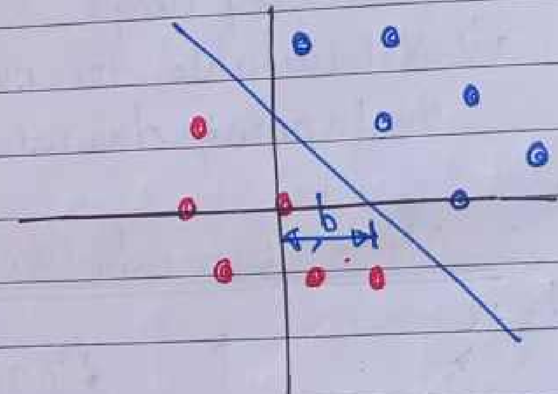
$$= -3 \quad \text{(Negative)}$$

Here for all the points which lie on the right side of hyperplane will be belong to negative class.

But Not all the time hyperplane will pass through Origine.

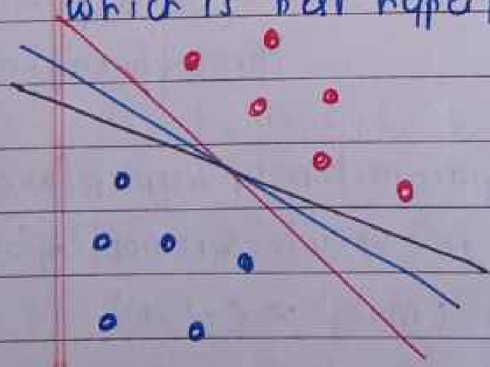


$w^T x = \text{label}$



$w^T x + b = \text{label}.$

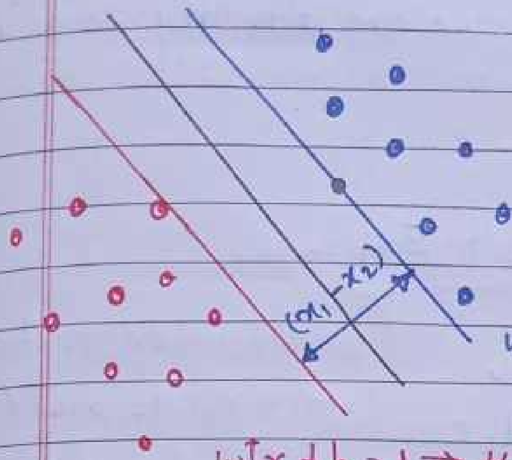
which is best hyperplane ?



there could be multiple hyper-plane, but the hyperplane with maximum margine size will be the best hyperplane.

→ optimization for maximum margine

$w^T x + b = \text{label}$



Equation of point or blue support vector & its output value any negative value.

$w^T x + b = -1$

$w^T x + b = 1 \Rightarrow$ this is equation of point or red support vector and its output value could be any positive value

to get margine let substract one from another.

$$
\begin{aligned}
& w^T x_1 + b = 1 \\
(-)\ & w^T x_2 + b = -1 \\
\hline
& w^T (x_1 - x_2) = 2
\end{aligned}
$$

$$w^T (x_1 - x_2) = 2$$

divide both side by $\|w\|$

$$\frac{\cancel{w^T} (x_1 - x_2)}{\|\cancel{w}\|} = \frac{2}{\|w\|}$$

$$(x_1 - x_2) = \frac{2}{\|w\|}$$ ← this is nothing but magnitude of vector.

and

$$y_i = \begin{cases} -1 & w^T x_1 + b \le -1 \\ 1 & w^T x_1 + b \ge 1 \end{cases} \quad \text{(label)}$$

So $\max\left(\frac{2}{\|w\|}\right)$ such that.

$$y_i = \begin{cases} -1 & w^T x_1 + b \le -1 \\ 1 & w^T x_1 + b \ge 1 \end{cases}$$

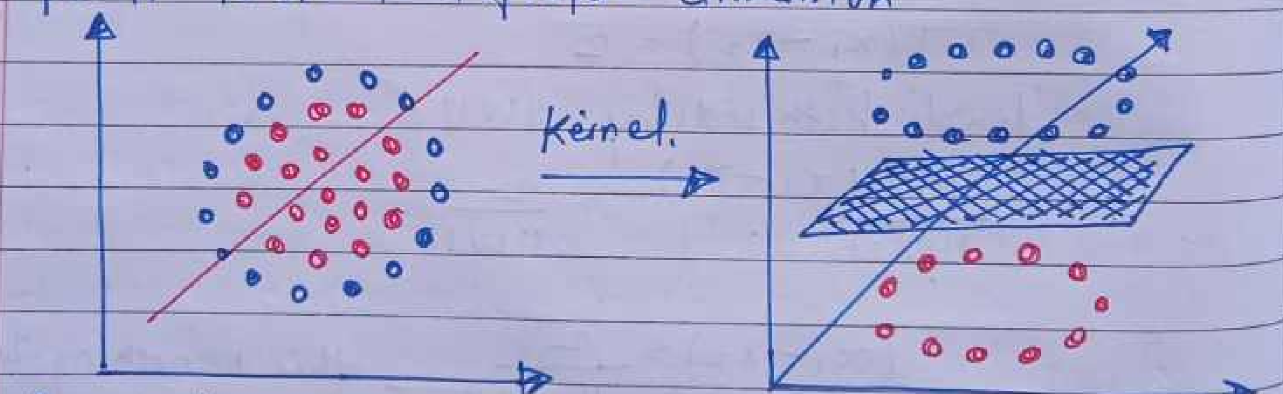
instead of using $\max \left(\frac{2}{\|w\|}\right)$ we can also try Min which make better sense

$$\min\left(\frac{\|w\|}{2}\right) + \underbrace{c * \sum \varepsilon_i}$$

(we all model to train with some error to avoid overfitting (i.e it will be good and train and will bad for test data)

c : Number of error
ε_i : Error magnitude

Kernels' in SVM : Generally function of the kernel is to transform the training set of data so that non-linear decision surface can be transformed to a linear equation in higher number of dimension space it return the inner product between two points in standard feature dimension



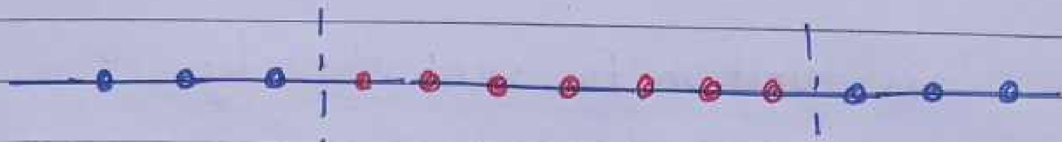
Types of svm kernel.

1) Linear
2) polynomial
3) Radial Basis function. (rbf)
4) sigmoid.

Suppose feature (x)

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
x^2	36	25	16	9	4	1	0	1	4	9	16	25	36

if you try to plot x on thin 1D line.

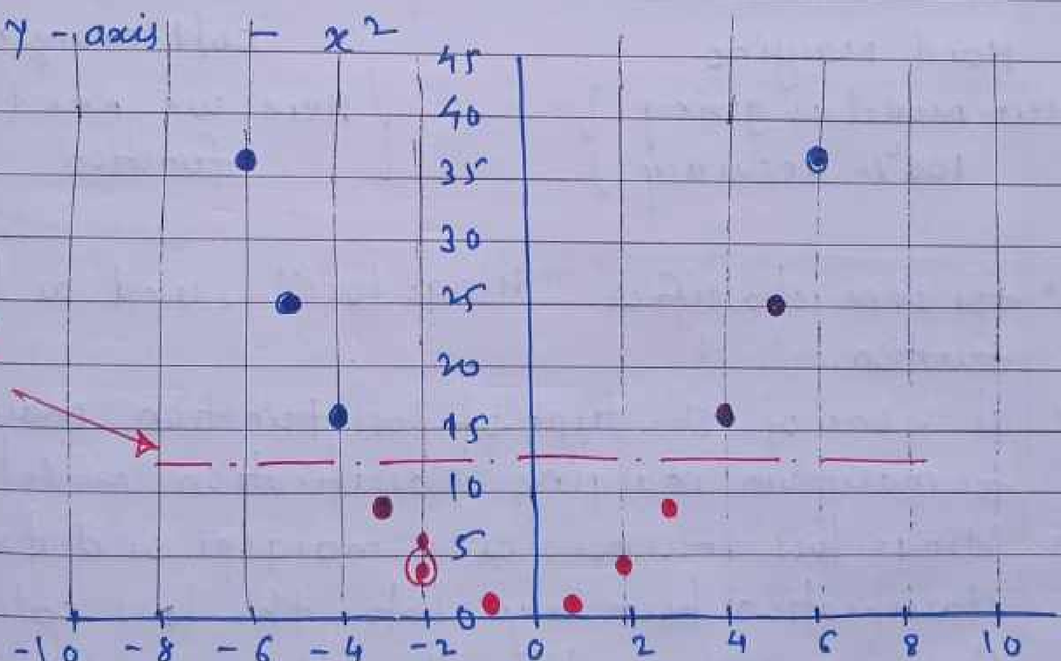


- we can see None of the line could seperate the two class perfectly.
- that's why we add another feature which is function of x i.e x^2

x-axis - x

y-axis - x^2

Now this is sepuable data.



1) Linear kernel :- $K(x_1, x_2) = x_1^T, x_2$

{best suitable for having too many features}

2) polynomial kernel.

$$K(x_1, x_2) = (x_1^T x_2 + r)^d.$$

degree

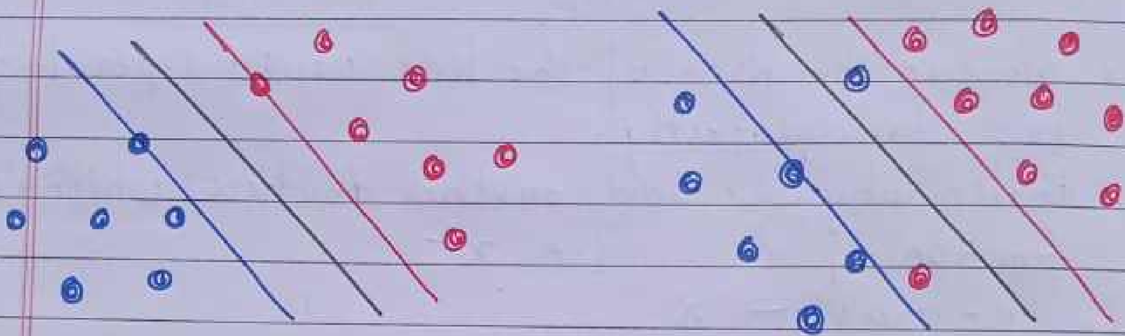
3.) radial basis function. (rbf kernel)

$$k(x_1, x_2) = \exp(-\gamma \cdot \|x_1 - x_2\|^2)$$

4. Sigmoid function

$$k(x_1, x_2) = \tanh(\gamma \cdot x_1^T x_2 + r)$$

Loss function for SVM classifier.



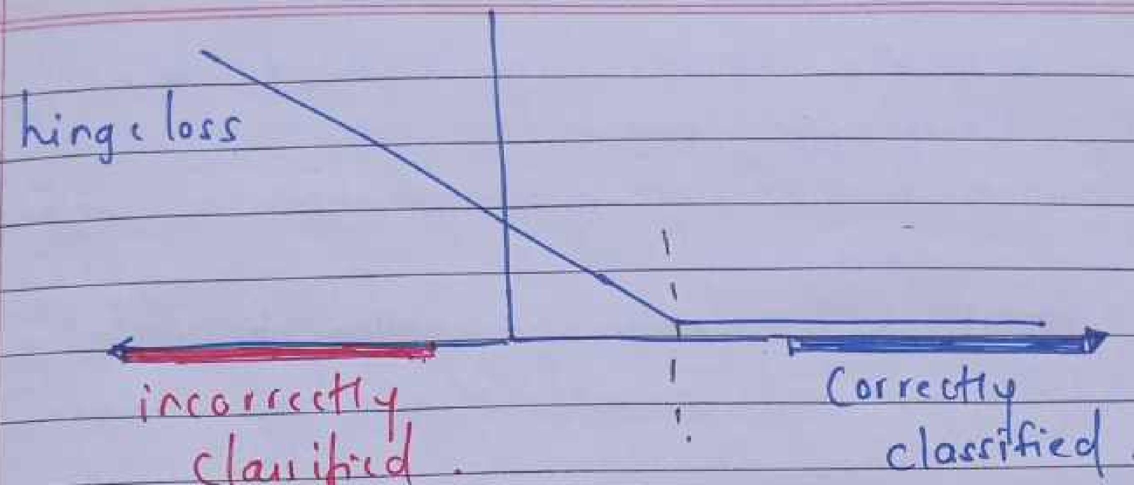
Hard Morgine
{Here model is giving 100% accuracy}

soft margine
{Here we need loss function}

- for SVM classifier "Hinge loss" is used as loss function.
- it is one of the type of loss function mainly used for maximum margine classification model.
- Hinge loss incorporates a margine or distance from classification boundary into the loss calculation Even if new observation classified correctly they can incure penalty if the margine from decision Boundary is not large enough.

$$L = \max(0, 1 - y_i(w^T x + b))$$

0 - for correct classification
1 - for wrong classification.



let test for wrong misclassification.

$y_i = 1$, $\hat{y}_1 = -1$ $\qquad$ $y_i = -1$ $\quad$ $\hat{y} = 1$

$L = (1 - 1(-1)$ $\qquad$ $L = (1 - (-1)(1)$

$= 1 + 1$ $\qquad$ $= 1 + 1$

$= 2$ $\qquad$ $= 2$

{ both are high loss value }

Now test for correct classification.

$y_i = 1$ $\quad$ $\hat{y}_1 = 1$ $\qquad$ $y_1 = -1$ $\quad$ $\hat{y}_1 = -1$

$(0 - (1)(1)$ $\qquad$ $(0 - (-1)(-1)$

$0 - 1$ $\qquad$ $0 - 1$

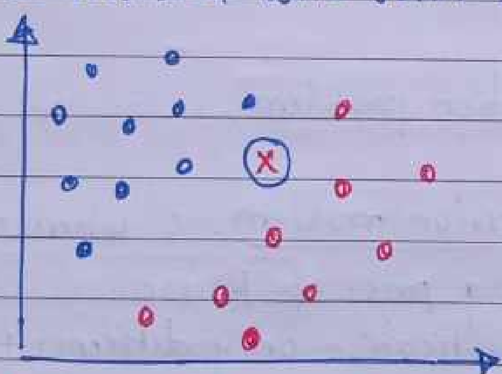
-1 $\qquad$ -1

{ both are low loss value }

KNN - K-Nearest Neighbor.

- The abbrevation KNN stands for "K-Nearest Neighbor" it is supervised machine learning algorithm this algorithm can be used to solve both classification as well as regression problem.
- the number of nearest neighbor to a new unknown Variable that has to be predicted or classified denoted by the symbol 'K'
- whenever new data will come if new data is close to 1 then prediction will be class 1 otherwise 0 for binary classification same principle for multiclassification problem.
- In general K is odd number

let understand with some example



Now we have to find new added ⓧ

Working of KNN Algorithm.

step 1: loading training as well as test data.

step 2: Next we choose value of K (hyperparameter) the nearest datapoints. k can be any integer.

step 3: for each data point (new) test data do the following.

3.1) Calculate distance between test data and each row of training data with the help of distance calculating method. like Euclidian, Manhatten or Hamming distance. The most commonly method to calculate is Euclidian.

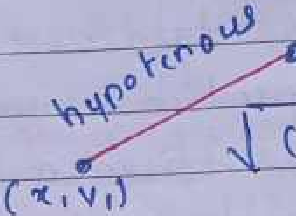
3.2) Now based on the distance value, sort them in ascending order.

3.3) Next it will choose the top k rows from sorted array.

3.4) Now it will assign a class to a test point based on most frequent class of these rows

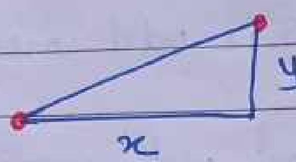
step 4 : End.

a) Eucledian distance :-



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

b) Manhattan distance



M. distance = $x + y$.

this is for classification problem.

Now let see for regression problem. : where output is continuous data. e.g price of house.

·). it is similar to classification only difference in last step where we were considering most frequent class here from k number of output (nearest 'k') we calculate their mean, that's it

Limitation : Not applicable to huge dataset since calculating distance would consume lot of time

- Sensitive to outliers
- Sensitive to missing values

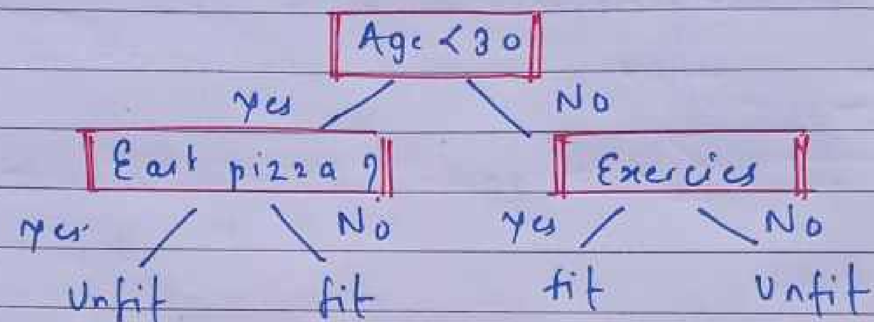
Desion Tree.

About decision Tree.

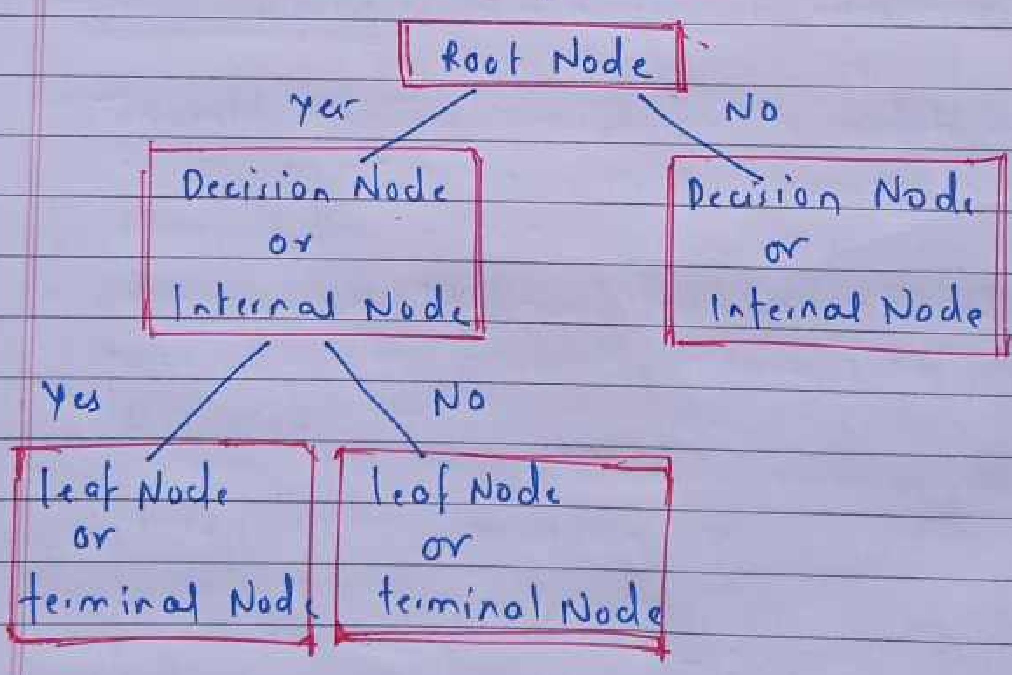
1) it is supervised ML Model
2) Used both classification & Regression.
3) Build Decision Nodes at each step.
4) Basis of Tree based model.

let understand with Example.

is person fit or Not ?



Structure & terminology of DT :-



Advantages :-

1). Can be used for both classification & Regression
2). Easy to interprete
3) No need for Normalization or scaling
4) Not sensetive to outliers.

Disadvantages:

1.) Overfitting issue
2) Small changes in the data alter the tree structure causing instability.
3) training time is relatively high.

Some concepts in DT :-

a) Entropy
b) Information Gain
c) Gini Impurity

Entropy : High	entropy : low
Information gain : low	Information gain : High
Gini Impurity : High	Gini Emparity : low

{ Entropy and gini impurity are inversly proportional to each other }

Entropy :- In ML Entropy is the quantative measures of randomness of information being processed.

Low entropy High Entropy.

- A high value of entropy means that randomness in the system is high and thus making accurate prediction is tough.
- A low value of Entropy means that randomness in the System is low and thus making accurate prediction is easier.

Entropy : $\sum_{i=1}^{c} -p_i \log_2 p_i$ c = number of classes
p_i = Probability of ith class

Date : / /
Page

- Information Gain :- once we find entropy to find which feature to be selected as root Node or internal Node we use information Gain.
- it is measure of how much information a feature provide about class low entropy leads to increased Information Gain and high entropy lead low information gain
- information gain computes the difference between entropy before split and average entropy after split of the dataset based on given value

$$\text{Info gain}(T, f) = \text{Entropy}(T) - \sum_{v \in F} \frac{|T_v|}{T} \cdot \text{Entropy}(T)$$

Target, feature

- Gini Impurity :- it is measure of impurity at Node
- The split made in decision tree is said to be pure if all the data point are accurately seperated into different class.
- it measure the likelihood that randomly selected data point would be incorrectly classified by specific Node.

formula : $1 - (P_Y^2 + P_N^2)$

P_Y = probability of class Y

P_N = probability of class N.

Decision Tree for regression.

Let understand with Example

Exp	Gap	Salary (K)
2	Yes	40
2.5	Yes	42
3	No	52
4	No	60
4.5	Yes	56
		$\bar{y} = 50$ ← Average.

Let take experience at root node
{Note - since exp is continuous data DT arrange it in ascending order}

- Now for comparison we will take two node example

≤ 2: 40 | 42, 52, 60, 56

≤ 2.5: 40, 42 | 52, 60, 56

- Now to decide which split is suitable we used one Concept called "variance reduction."

$$\text{Variance} = \frac{1}{n}\sum_{i=1}^{n}(y-\bar{y})^2 \leftarrow \text{(MSE formula)}$$

where $\bar{y}$ = Average output.

- Now we have to calculate variance at each node.
- 1st we will calculate variance of root.

$$\text{Variance(Root)} = \frac{1}{5}\left[(40-50)^2+(42-50)^2+(52-50)^2 + (60-50)^2+(56-50)^2\right]$$

$$= \frac{1}{5}\left[100+64+4+100+36\right]$$

$$\boxed{= 60.8}$$

Now we will colculate variance at each internal node or decision node

$$\text{Variance}(IN1) = \frac{1}{1}\left[(40-50)^2\right]$$

$$= 100$$

$$\text{Variance}(IN2) = \frac{1}{4}\left[(42-50)^2+(52-50)^2+(60-50)^2 + (50-56)^2\right]$$

$$= \frac{1}{4}\left[(-8)^2+(2)^2+(10)^2+(6)^2\right]$$

$$= \frac{1}{4}\left[64+4+100+36\right]$$

$$= 51$$

Variance reduction formula:-

$$= Var(Root) - \sum w_i \, Var(IN)$$

$$= 60.8 - \left[\frac{1}{5}(100) + \frac{4}{5}(51)\right]$$

$$= 60.8 - 20 - 40.8$$

$$= 0$$

Same we will colculate for second condition ≤ 2.5 whoever have large variance reduction we will finalize it for splitting.

$$var(IN1) = \frac{1}{2}\left[(40-50)^2+(42-50)^2\right]$$

$$= \frac{1}{2}\left[100+64\right]$$

$$= \frac{164}{2} = \; = 82$$

$$\text{Var IN2} = \frac{1}{2}\left[(52-50)^2 + (60-50)^2 + (56-50)^2\right]$$

$$= \frac{1}{3}\left[4 + 100 + 36\right]$$

$$= \frac{140}{3} = 46.66$$

Variance reduction for next split i.e ≤ 52.5

$$= \text{Var(Root)} - \sum w_i \text{Var (IN)}$$

$$= 60.8 - \left[\frac{2}{5}(82) + \frac{3}{5}(46.66)\right]$$

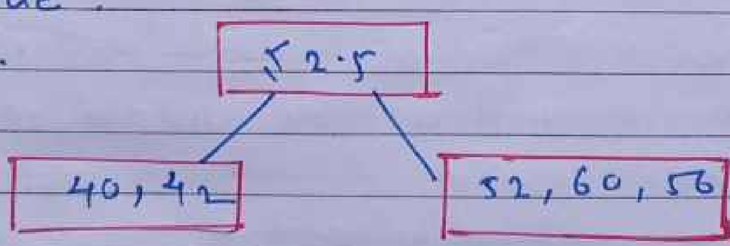
$$= 0.304$$

Var (split 2) > var (split 1) thats why we will select second split.

How to calculate o/p for the test data

- whichever leaf node your test data reached to take avg of all the numbers present in the same leaf Node.

for Ex.



if test data reached to 1st leaf then

$$\frac{40+42}{2} = 41$$

and it reached to second leaf then

$$\frac{52+60+56}{3} = 56$$

Date: / /

Page

Introduction to Ensemble Learning or Techniques.

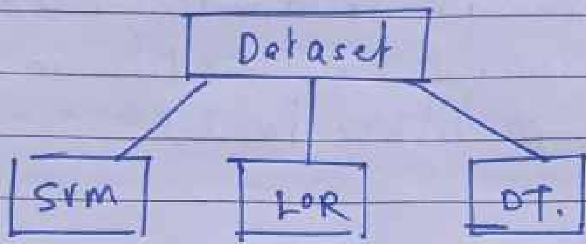
- Ensemble technique is a machine learning technique that combine several base models in order to produce one optimum predictive model.
- M1 M2 M3 M4 . . . . Mn
- Collectively these model called as Ensemble model. provided model should be different from each other.
- for making the models different we can adopt two way
  1. Algorithm choosed for the model should be different
     ex. LR, SVM, NB, KNN like
  2. Or if we select same algorithm for all the model then choose different data for all the model by some sampling method.

- In classification problem output is decided on majority count for the perticular class predicted by all the model.
- In regression mean or average of all the output predicted by model.

Types of Ensemble.

- Voting ensemble
- Bagging → Random forest
- Boosting
  - Adaboost
  - Gradient Boosting.
  - XgBoost
- stacking.

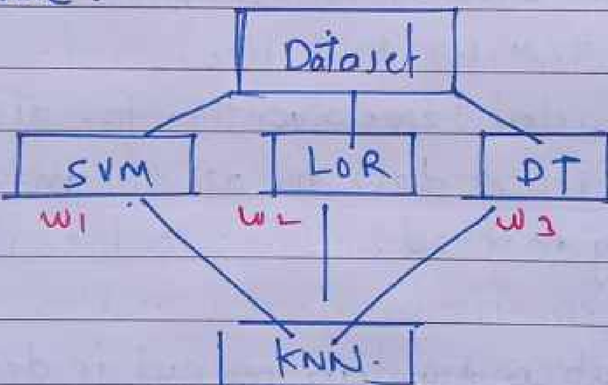
Voting Ensemble :- this depends on model should be different or particularly different algorithm.



For classification :- output is given by majority of count.

For regression :- output is calculated by mean of output given by all the algorithm.

stacking :- unlike voting here we give additional weightage to the models based on their performance.



- Here we add one more additional algorithm which is trained on the output of above models or ensemble model
- this newly added algorithm gives different weightage to each model based on their perforce based on the training data.
- and after this newly added model will predict the final result.

Bagging :- Bootstrap Aggregation

Here we use same algorithm for all the models just dataset is different. for each model.

$$\underbrace{\boxed{DT}\quad\boxed{DT}\quad\boxed{DT}\;\cdots\;\boxed{DT}}_{\text{Decision Tree}}$$

Random forest is special case of Bagging where base model is Decision tree. but we can other algorithm as well.

Boosting :- Boosting is an ensembling modeling technique that attempts to build a strong classifier from the number of weak classifier. it is done by building model by using weak model in series firstly a model is built from training data. Then second model is built which tries to correct the error present in the first model. This procedure is continued and the model are added until either the complete training data set predicted correctly or the maximum number of models are added.

training data - $\boxed{m_1} \rightarrow \boxed{m_2} \rightarrow \boxed{m_3} \cdots \boxed{m_n}$

Benefits :- 1) Improvement in performance.

2) it help to acchieve low bias and Low variance.

3) Robust to variance.

when to use :- always.

Disadvantages :- Here we have to train multiple models which increase computational complexity.

1) let understand bagging and Random forest deeply.

- A Bagging classifier is ensemble method that fits base classifier each on random subsets of the original dataset and then aggregate their individual prediction (either by voting or averaging) to form final prediction

- let understand bagging with Decision tree.
- Here we will create multiple Decision Tree, no. of decision tree is hyperparameter. let say '10'
- and suppose we 1000 rows dataset, Now how to distribute among the 10 models ?
- if we split it in 10 each model will get only 100 rows. since it is less data it will creat bad model.
- so each model should get enogh data and also different data.

there are two techniques for sampling :-

1) simple random sampling with replacement
2) simple random sampling without replacement.

Example 1 2 3 ← Container having 3 balls.

SRS w/o Replacement

1 2
1 3
2 3

{each row/column should not repeated twice}

SRS with replacement

1 1 1 2
2 2 1 3
3 3 2 3

{each row or column can repeate multiple time}.

so to select how many rows randomly for each model it depends on user generally people prefer 80% of the dataset

- so earlier due to splitting each model might get 100 only but by this technique each model will get 800 different dataset for training.

training data (1000 x 4 features)

DT1	DT2	DT3	DT10
(800x4)	(800x4)	(800x4)	(800x4)

{simple random sampling with replacement}.

- Now its time to aggregate result by combining method or voting.

Voting: If to predict from class 1 or class 0

DT 1 → 1
DT 2 → 0
DT 3 → 0
,
DT 10 → 1

out of 10
8 DT → 1
2 DT 0

{ so in this case due to majority input or test query belong to class 1 }

<u>hyperparameter</u> :- i) Number of DT's
ii) How much data you want to provide for each DT model.

Random forest :- if number of feature increases then complexity of creating decision tree also increases to avoid that this technique is used.

- instead of selecting random number of rows we select random number of columns.
- the each model will get lesser number of columns for this we will use simple random sampling w/o replacement.
- no of column is hyperparameter.
  generally for classification $= \sqrt{P}$
  regression problem $= P/2$
  where P is number of feature in original dataset
- so for the column we will do SRS w/o Replacement but for the rows we will do SRS with Replacement.

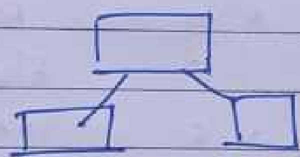
- why it is called random sampling ?

→ because everytime we select the features randomly.

- hyperparameters : 1) No. of trees
  2) No of features
  3) No of Datapoints.

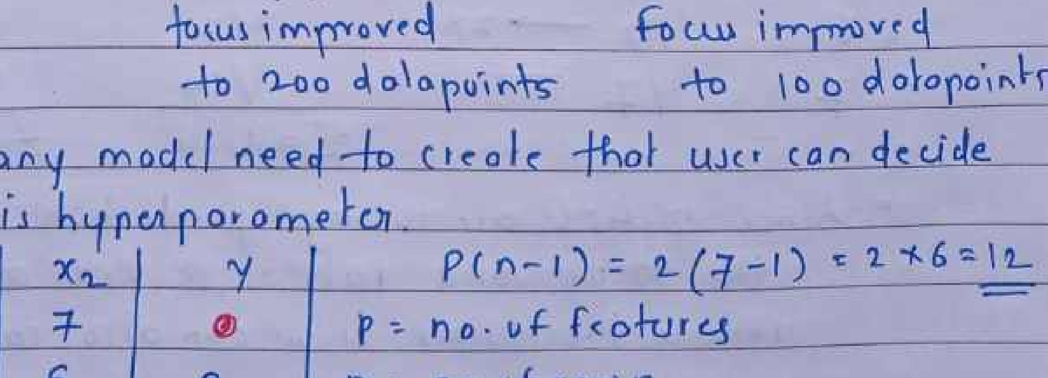
Boosting :- 1) Adaboost
2) Gradient boost
3) Xgboost.

1) Adaboost :- also called as Adaptive boosting is a technique in Machine learning used as an Ensemble method. The most common estimator used with Adaboost is decision trees with one level which means Decision trees with only 1 split. these tree are also called Decision stumps.



Boosting is nothing but wherever my model performing bad need improve focus there and wherever it is performing good need to focus less there.

workflow :→



- how many model need to create that user can decide i.e it is hyperparameter.

let understand with Example.

x_1	x_2	y
1	7	●
2	6	●
3	5	●
4	4	●
5	3	●
6	2	●
7	1	●

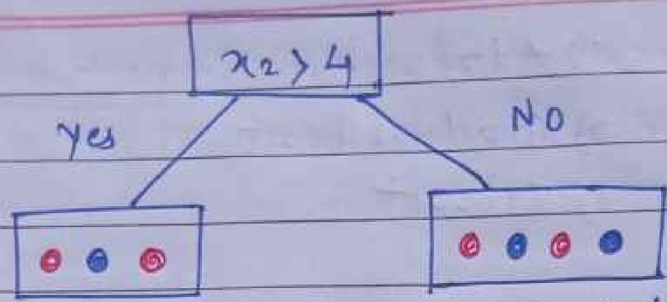
$P(n-1) = 2(7-1) = 2 \times 6 = \underline{12}$

P = no. of features

n = no. of rows

so with total 12 condition we can split DT for stump.

let say $x_2 > 4$.



predicted : Red   predicted : Blue.

	Actual class	Predicted.	
1	Red	Red	
• 2	blue	Red	→ wrong prediction
3	Red	Red	
• 4	Red	blue	→ wrong prediction
5	blue	blue	
• 6	Red	blue	→ wrong prediction
7	blue	blue	

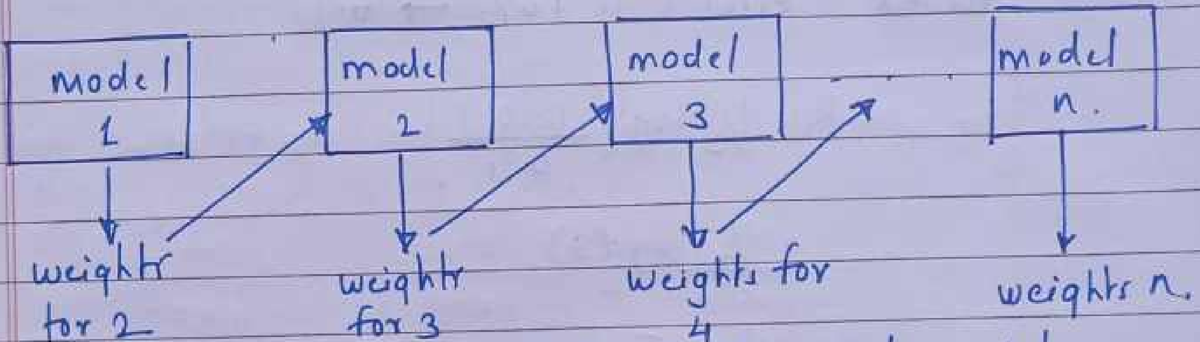
Now let understand my distribution of focus.

	earlier focus		New focus	updated focus
1	1/7	→ 0.5x	1/14	1/16
• 2	1/7	→ 2x	2/7	4/16
3	1/7	→ 0.5x	1/14	1/16
• 4	1/7	→ 2x	2/7	4/16
5	1/7	→ 0.5x	1/14	1/16
• 6	1/7	→ 2x	2/7	4/16
7	1/7	→ 0.5x	1/14	1/16
			$\Sigma = 1.14$	$\Sigma = 1$

- Since 2, 4, 6 are wrong prediction need improve the focus. and rest 1, 3, 5, 7 are correct will reduce the focus or we can also called it as weights.
- for ex. we will improve our focus by 2 for incorrect prediction and reduce by 2

- sum of updated focus should be not greter than 1 since new focus was 16/14 = 1.14 we try to Normalize it.
- to do so we divide New focus by 16/14 and we get new updated focus whose sum is equal to 1

- thus whichever point get higher focus / wt will get high priority in next model and those point with lesser focus/wt will get lesser priority.



- so we will create new dataset for next model based on weight received from last model.
- general inference is that since my first point - 1/16 it will repeate one time and my 2nd point - 4/16 it will repeat four times and same for 4, 6.

x_1	x_2	new wt	Bucket range
1	7	0.0625	0 - 0.0625
• 2	6	0.25	0.0625 - 0.3125 (big bucket)
3	5	0.0625	0.3125 - 0.375
• 4	4	0.25	0.375 - 0.625 (big bucket)
5	3	0.0625	0.625 - 0.6875
• 6	2	0.25	0.6875 - 0.9375 (big bucket)
7	1	0.0625	0.9375 - 1

2, 4, 6 comes in big range bucket ~~hence~~ there chance of getting focus is more while sampling with replacement.

Page

for this example we consider constant 2 for increasing or reducing the focus but how we decide this constant.

$$\text{constant} = e^{\lambda k}$$

where : $e = 2.718$

λ = learning rate (it could be from 0 to 1)

$$k = \frac{1}{2} \log\left(\frac{1 - \text{Error}}{\text{Error}}\right)$$

for Ex error = is 10% → 0.1

$$k = \frac{1}{2} \log\left(\frac{1 - 0.1}{0.1}\right)$$

$$= \frac{1}{2} \log(9)$$

$$k = \log\sqrt{9}$$

$$e^{\log\sqrt{9}} = \sqrt{9} = 3.$$

so instead of 2 for 10% Error we would multiply or divide by 3 to increase or decrease our focus.

Summary:-

step 1 :- Divide data into train and test

2 :- create weak learner (STUMP)

3 :- Calculate error, find multiplication factor, value of k.

4 :- update the focus and update the data, create new model with updated data.

5 :- repeat step 2 and 4 number of this repeatation equal to number of models.

6 : Combine models for voting.

Gradient Boosting :- let understand with example of regression problem which is easy to understand as GBM compare classification problem.

iq	cgpa	salary (LPA)
90	8	3
100	7	4
110	6	8
120	9	6
80	5	3

we will create gradient Boosting Model of 3 base model

$\boxed{M_1}$ (mean) $\boxed{M_2}$ (DT) $\boxed{M_3}$ (DT)

output of first model i.e M_1 will always be mean of output or target column of dataset

→ Here we have it is 4.8 → pred 1

iq	cgpa	salary	pred 1	psedo 1	pred 2	psedo 2	pred 3
90	8	3	4.8	-1.8	-1.8	-1.62	-1.62
100	7	4	4.8	-0.8	-0.8	-0.72	-0.72
110	6	8	4.8	3.2	3.2	2.88	2.88
120	9	6	4.8	1.2	1.2	1.08	1.08
80	5	3	4.8	-1.8	-1.8	-1.62	-1.62

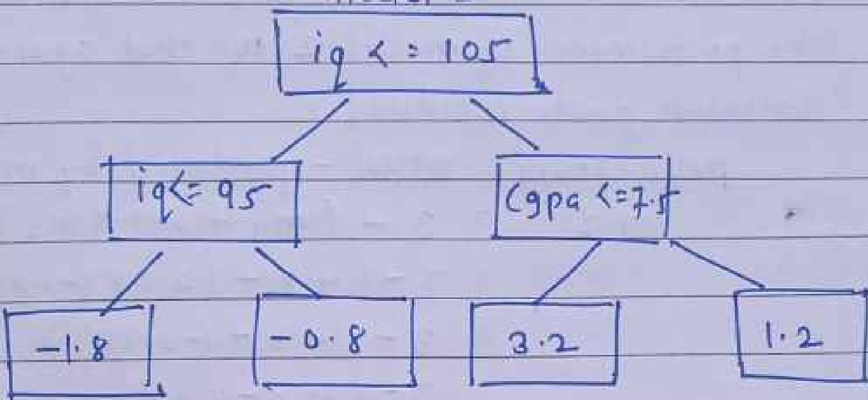
- In order to tell the error done by M_1 to M_2 we will calculate loss function of M_1

  actual - predicted = psedo - residual.

- Now we will train model 2 which is DT for which input will be same (iq, cgpa) but output will be psedo residual. and not salary.

Inference:- Here we are asking to model2 to tell us that how mistake or error is model 1 doings.

Decision Tree for model 2



Now we will do prediction of model 2 on my data i.e pred 2

Let see how would i predict if i would have GBM of only two based model.

prediction = $m_1 + m_2$

let calculate for student 1 (90, 8)

$= 4.8 + (-1.8)$

$= 4.8 - 1.8 = 3$

student 2 (100, 7)

$= 4.8 + (-0.8) = 4.8 - 0.8 = 4.$

student 3 = 8 | student 4 = 6 | student 5 = 3.

∴ which is all equal to actual target column, thus this problem of overfitting. like it will good for training set but will not the same for test data.

that's why in prediction we add learning rate.

prediction = $m_1 + lr \times m_2$

generally $lr = 0.1$

Now student 1 = $4.8 + (0.1)(-1.8) = 4.8 + (-0.18)$

$= 4.62$

with the help of learning rate we reduce the output and go close to actual output gradually. (stepwise)

Now we will calculate model 3 for which we should know the performance of model 2 for that again we will calculate psedo residual.

$$\text{psedo-resid} = \text{actual} - \text{predicted by } m_2$$

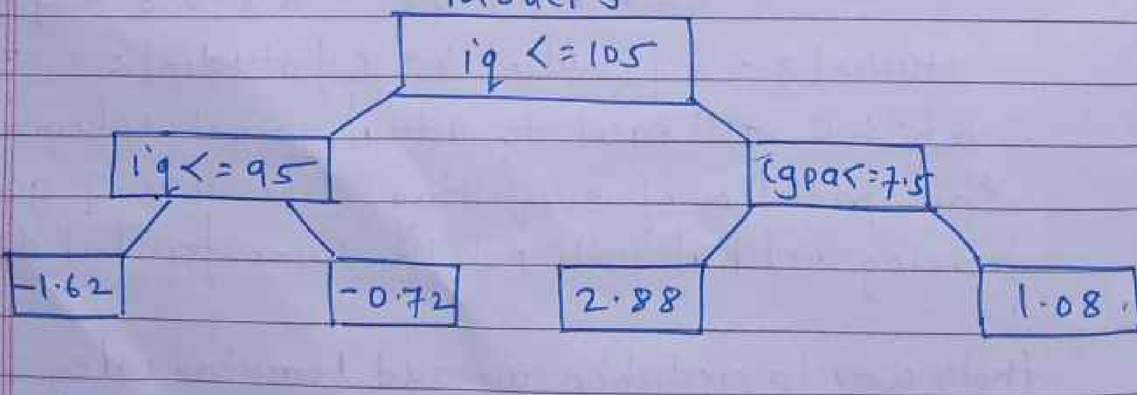
$$\text{st } 1 \quad = 3 - (m_1 + 0.1 \times m_2)$$
$$= 3 - (4.8 + (0.1 \times (-1.8)))$$
$$= 3 - (4.8 + (-0.18))$$
$$= 3 - (4.8 - 0.18)$$
$$= 3 - 4.62 = -1.62$$

same for studen 2/3/4/5 and it will be psedo-residual 2

so we can clearly observe after each shift of model our residual is shift towards zero

- Now we will create model 3 which is again DT. whose input col will be iq and cgpa but target will psedoresidual 2.
- model 3 combinely predict the mistakes of model 1 and model 2

Model 3



- Now well find prediction 3.

Now y-prediction by GBM will be

$$m_1 + 0.1 \times m_2 + 0.1 \times m_3$$

let calculate student x (60 , 4.9)

$$= 4.8 + \underline{0.1(-1.8)} + \underline{0.1(-1.62)}$$
$$= 4.8 - 0.18 - 0.162$$
$$= 4.458$$

earlier it would be 4.62 now it is 4.458

Difference between Adaboost Vs Gradient Boost

Adaboost	Gradient Boost
• Here we use dicision stump. (DT whose depth is 1) mean max leaf node could be 2	• In GBM max allowed leaf node could be from 8 to 32
• model are $(m_1, m_2 \ldots m_n)$ are prioritized based on their weights $w_1m_1 + w_2m_2 + w_3m_3$	Here we use learning rate concept which is same for all the models.

Gradient Boosting for classification problem:-

like popcorn	Age	favorite Color	Love Moview.
yes	12	Blue	yes
yes	87	gree	yes
No	44	blue	No
yes	19	Red	No
No	32	green	yes
No	14	blue	yes

- when we use gradient Boost for classification the initial prediction for every individual is the log(odds)

- log (odds) for target : $\log\left(\frac{\text{people love the movie}}{\text{people don't love the movie}}\right)$

$$= \log\left(\frac{4}{2}\right) = 0.7 \quad \text{(rounded value)}$$

∴ this is initial prediction.

- Just like with logistic Regression the easiest way to use the log (odd) for classification is to convert it to a probability. and we do it with logistic function.

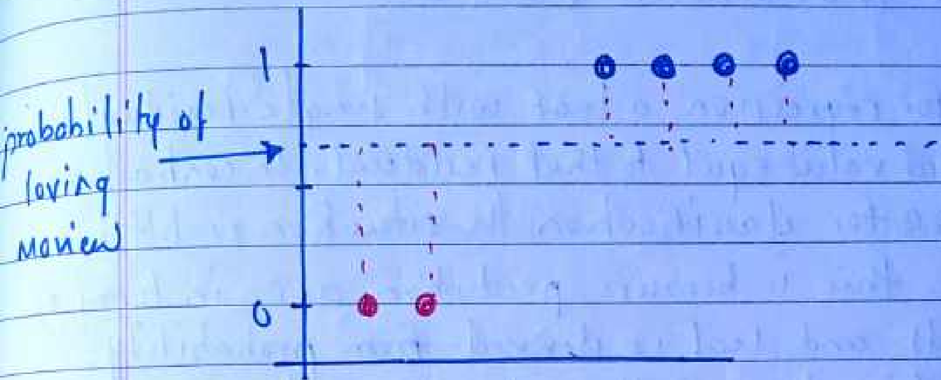
probability of loving Moview : $\frac{e^{\log(odd)}}{1+e^{\log(odds)}}$

$$= \frac{e^{\log(4/2)}}{1+e^{\log(4/2)}} = 0.7 \quad \text{(rounded value)}$$

- Now classifying everyone in training dataset is someone who love moview is pretty lame because two of people do not love movie.

we can measure the how bad initial prediction is by calculating psedo residuals the difference between observed and predicted values.

Residual = (observed − Predicted)



observed value could be 0 or 1 and predicted is 0.7 if we calculate residual for initial prediction.

likes popcorn	Age	faverait Color	Love Movie	Residual
yes	12	Blue	yes	0.3
yes	87	green	yes	0.3
NO	44	blue	NO	-0.7
yes	19	Red	NO	-0.7
No	32	green	yes	0.3
No	14	blue	yes	0.3

Now we will build new tree for next model using "likes popcorn and age" as input and Residual as output or target

Red

−0.7   Age > 37

−3.3

0.3, −0.7   0.3, 0.3, 0.3   1.4

−1

- Note:- Just like when we used Gradient boost for regression, we are limiting the number of leaves that we will allow in the tree, here we are allowing no. of leaves to 3.
- In practice people often set the maximum number of leaves to be between 8 and 32

- In GB for regression a leaf with single residual had output value equal to that residual. In contrast when we use GB for classification the situation is little complex. this is because prediction are in terms of log(odd) and leaf is derived from probability. So we can't just add them together to get new log(odd's) prediction without some sort of prediction

-o- for output of leaf 1st

$$\frac{\Sigma \text{ Residual}}{\Sigma [\text{Prev. Probability}_i \times (1 - \text{prev probability})}$$

$$\therefore \quad \frac{-0.7}{0.7 \times (1-0.7)} = -3.3$$

-o- for output of leaf 2nd

$$\frac{0.3 + (-0.7)}{(0.7 \times (1-0.7) + (0.7 \times (1-0.7)} = -1$$

-o- Output value for leaf 3.

$$\frac{0.3 + 0.3 + 0.3}{[(0.7 \times (1-0.7)) + (0.7 \times (1-0.7)) + (0.7 \times (1-0.7))]} = 1.4$$

Now we are ready to update our predictions by combining the initial leaf with new tree.

Date: / /
Page

prediction = initial leaf + learning rate x Current leaf

- let take learning rate is 0.8 which is very large but it for illustrative purpose however 0.1 is most common.

Now predict for each person, person 1

$$0.7 + (0.8 \times 1.4) = 1.8$$

$$\text{probability} = \frac{e^{1.8}}{1+e^{1.8}} = 0.9$$

Similar we do for all the person.

like popcorn	Age	faverail color	love Movie	Residual	predicted	Residual.
yes	12	Blue	yes	0.3	0.9	0.1
yes	87	green	yes	0.3	0.5	0.5
No	44	blue	No	-0.7	0.5	-0.5
yes	19	Red	No	-0.7	0.1	-0.1
No	32	green	yes	0.3	0.9	0.1
No	14	blue	yes	0.3	0.9	0.1

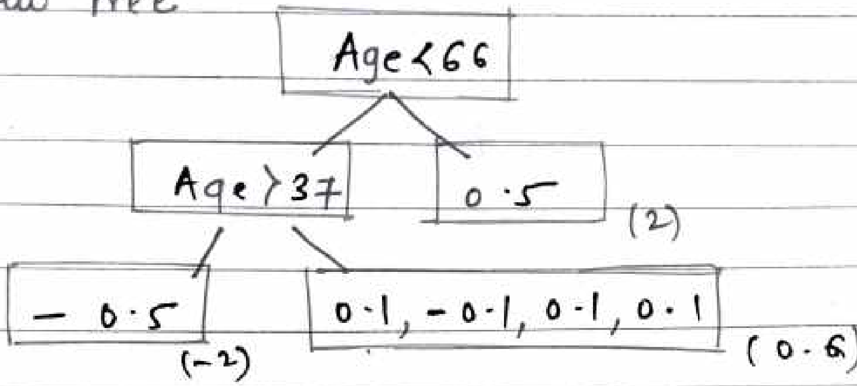
this new predicted probabilities are worst than before & that's why we build lot of tree and not just one

- And now just like before we calculate new residuals since Residual = observed - predicted.

for person 1st : $(1-0.9) = 0.1$
2nd : $(1-0.5) = 0.5$
⋮
6th : $(1-0.9) = 0.1$

put in above table

Now with the help of new residual we will build new tree



output of leaf 3rd (2nd person) : $\dfrac{\Sigma \text{ Residual}}{\Sigma[\text{prev prob} \times (1 - \text{prev probability})]}$

$= \dfrac{0.5}{0.5 \times (1 - 0.5)} = 2$

output of leaf 1st (3rd person) : $\dfrac{-0.5}{0.5 \times (1 - 0.5)} = -2$

output of leaf 2 (1, 4, 5, 6 person) : $\dfrac{0.1 + (-0.1) + (0.1) + (0.1)}{(0.9 \times (1-0.9)) + (0.9 \times (1-0.9)) + (0.9 \times (1-0.9)) + (0.9 \times (1-0.9))}$

$= 0.6$

Now we have calculated all of the output values for this tree we can combine it with everything else we've done so far

this process repeats until we have made the maximum number of trees specified or the residual get super small.

if we need to classify a new person a somone who loves or not love the movie

Date: / /
Page

log(odd) prediction. first leaf or model + 0.8 x 2nd model + 0.8 x 3rd model

$= 0.7 + (0.8 \times 1.4) + (0.8 \times 0.6) = \underline{\underline{2.3}}$

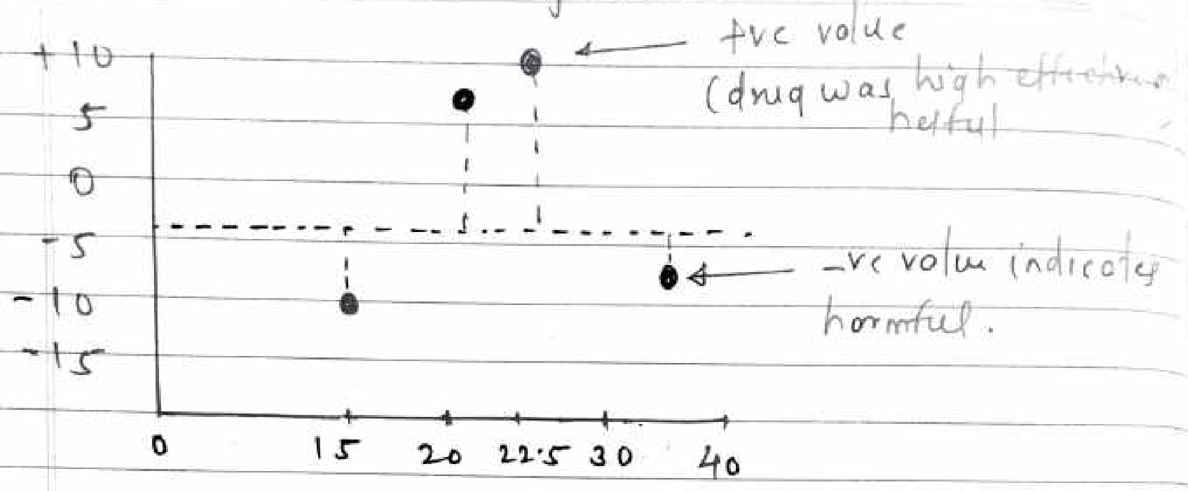
Now convert 2.3 into probobility

$$\text{probobility} = \frac{e^{2.3}}{1+e^{2.3}} = 0.9$$

and thus predicted probability that this individual love the movies is 0.9

- Since we have keep threshold 0.5 and $0.9 > 0.5$ we classify the new person to class 1 which means he will love the movies

XGBoost for Regression.



initial prediction is 0.5 wheather it is XGboost classification or regression.

Residual = observed - predicted

let build 1st DT with residual, the tree start with single leaf where all residual will stored.

$\boxed{-10.5, 6.5, 7.5, -7.5}$ 1st leaf

we will calculate similarity score for residual

$$\text{Similarity score} = \frac{[\text{Sum of Residue}]^2}{\text{Number of Residual} + \lambda}$$

where λ = regularization parameter.

Assume $\lambda = 0$

$$= \frac{[-10.5 + 6.5 + 7.5 + (-7.5)]^2}{4 + 0}$$

$$= \frac{(-4)^2}{4} = \frac{16}{4} = 4$$

Now question is whether or not we do better job of clustering similar residuals if we split them into two groups a] first for threshold we will consider two operation with lowest dosages and take their average for splitting

Dosage < 15

$\boxed{-10.5,\ 6.5,\ 7.5,\ -7.5}$ SS = 4

leaf ① $\boxed{-10.5}$ SS = 110.25

② leaf $\boxed{6.5,\ 7.5,\ -7.5}$ SS = 14.08

similarity score for 1st leaf : $\dfrac{(-10.5)^2}{1+\lambda} = 110.25$

Similarity score for 2nd leaf $= \dfrac{[6.5+\cancel{7.5}+(-\cancel{7.5})]}{3+\lambda}$

$= \dfrac{(6.5)^2}{3} = 14.08$

Note: when residual are very different they cancel out each other and similarity score is very less In contrast Residual are same or only one they don't cancel each other similarity score is high.

- Now we need to quantify how much better the leaves cluster similar Residuals than the root.

Gain = left leaf similarity + Righ leaf similarity - Root similarity.

Gain : $110.25 + 14.08 - 4 = \underline{\underline{120.33}}$.

b] Now we will shift dosage threshold to 22.5

Dosage < 22.5

$\boxed{-10.5,\ 6.5,\ 7.5,\ -7.5}$

$\boxed{-10.5,\ 6.5}$ SS = 8

$\boxed{7.5,\ -7.5}$ SS = 0

again we will calcute "Similarity score" for each leaf.

$$SS_{left} = \frac{[(-10.5) + 6.5]^2}{2+0} = 8$$

$$SS_{Righ} = \frac{[7.5 + (-7.5)]^2}{2+0} = 0.$$

Gain ~~Gain~~ = SS Left + SS Right − Root SS

= 8 + 0 − 4 = $\underline{\underline{4}}$

Gain (Dosage > 22.5) < Gain (Dosage > 15)

c] Now we will shift threshold value to 30.

Dosage < 30

$-10.5, 6.5, 7.5, -7.5$	
$SS = 4.08$: $-10.5, 6.5, 7.5$	-7.5: $SS = 56.25$

$$SS_{left} = \frac{[-10.5 + 6.5 + 7.5]^2}{3+0} = 4.08$$

$$SS_{righ} = \frac{[-7.5]^2}{1} = 56.25$$

Gain = SSleft + SSRight − SS Root

= 4.08 + 56.25 − 4.

= $\underline{\underline{56.33}}$.

Gain (Dosage < ~~30~~ 22.5) < Gain (Dosage < 30) < Gain (Dosage<1

that's why we will split the root node of DT with threshold of 15 since it is giving highest Gain.

Dosage < 15 ss = 4

-10.5 ss = 110.25

Dosage < 22.5

6.5, 7.5, -7.5 ss = 14.08

Since there is only one residual in left leaf we can't split it ahead. howeve we can split right leaf further.

Dosage < 22.5

6.5, 7.5, -7.5 ss 14.08

6.5

7.5, -7.5

ss left = 42.25
ss Righ = 0

$\}$ see the previous formula.

Gain = ssleft + ss Right - ss Root
= 42.25 + 0 - 14.08
= $\underline{\underline{28.17}}$

Now again will shift threshold value to 30

Dosage < 30

6.5, 7.5, -7.5 ss = 14.08

6.5, 7.5 ss = 98

-7.5 ss = 56.25

ss left : 98
ss Right : 56.25

Gain = ~~98~~ ssleft + ss Right - ss Root = $\underline{\underline{140.17}}$

Gain (Dosage < 22.5) < Gain (Dosage < 30)

∴ so this is perfect split.

to keep this example we are limiting it to two levels and not splitting further however in default it will allow up to 6 levels

- Now we will see how to prune this tree. In XG boost we prune based on its gain value.

- if we consider any random number like 130 this number called as γ
  here $\gamma = 130$.

- then we calculate the difference between γ and gain association with lowest branch of tree.
- if difference is negative we will remove the branch and if it is positive we will not remove the branch.

- so lowest branch gain 140.17
  $\gamma = 130$
  Gain $- \gamma$ = ~~140.13~~ $140.17 - 130 = \underline{\underline{10.17}}$
  +ve value we will not remove the branch.

- ~~if number would be 150 (i.e. $\gamma = 150$)~~ the difference of γ and gain is negative at root but because we are not removing lowest branch hence we will not remove this as well

- if γ would be 150 the we had to remove lowest branch as well as root and whole tree would gone and it would be extreme pruning.

all this we did for $\lambda = 0$.

Now we will repeat all this step for $\lambda = 1$.

λ is regularization parameter which mean it is intended to reduce the prediction sensitivity to individual observation.

$$\boxed{-10.5,\ 6.5,\ 7.5,\ -7.5} = 3.2$$

$$\boxed{-10.5}\ 55.12 \qquad \boxed{6.5,\ 7.5,\ -7.5}\ 10.56$$

again similarity score at root

$$SS_{Root} = \frac{[-10.5 + 6.5 + 7.5 + (-7.5)]^2}{4+1}$$

$$= 3.2$$

$$SS_{left} = \frac{(-10.5)^2}{1+1} = 55.12$$

$$SS_{Right} = \frac{[6.5 + 7.5 + (-7.5)]^2}{3+1} = 10.56$$

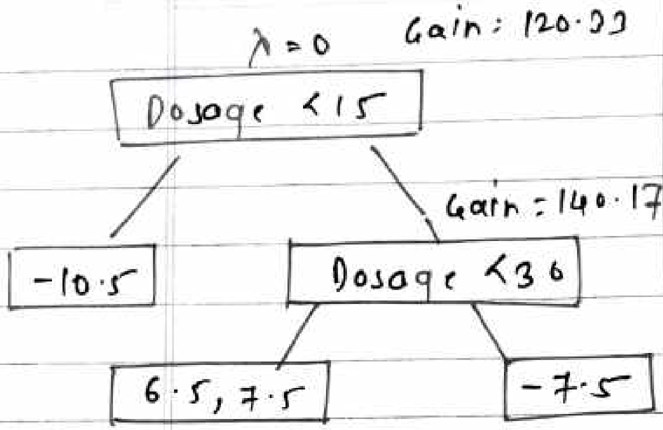
Similarity score while $\lambda = 1$ are very less than similarity score while $\lambda = 0$, the amount of decrease is inversly proportional to the number of residual in the node.

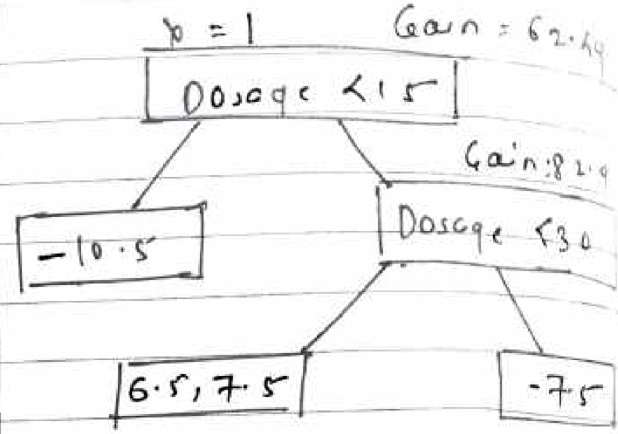
$$Gain = SS_{left} + SS_{Right} - SS_{Root}$$

$$= 55.12 + 10.56 - 3.2$$

$$= 62.48$$

and $\lambda = 0$ it was 120.33.



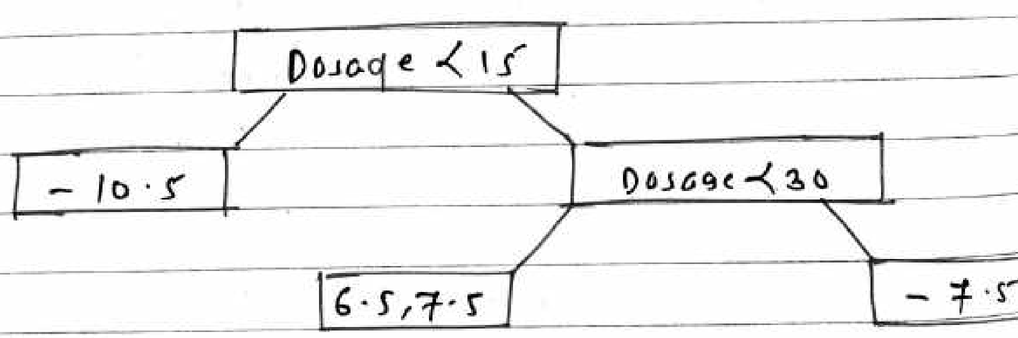


For pruining
For number : γ = 130
due to the difference we didn't prune

For pruining
since lowest branch gain is less than 130 we will remove it and eventully prune whole tree.

Note :- setting γ = 0 doesn't turn off pruining if difference become negative it anyway will prune

on other hand λ = 1 λ did what it was supposed to do, it prevent overfitting training data.



to calculate output :- $\dfrac{[\text{Sum of Residual}]}{\text{Number of Residual} + \lambda}$

o/p for 1st datapoint $= \dfrac{-10.5}{1+\lambda}$.

Date: / /
Page

if $\lambda = 0$ mean w/o regularization = -10.5
if $\lambda = 1$ with regularization factor = -5.25

if $\lambda > 0$ then it will reduce the amount that this individual observation adds to overall prediction.

thus λ (lambda) regularization parameter will reduce the prediction sensitivity to this individual observation

output of leaf ②: $\frac{6.5 + 7.5}{2 + 0} = 7$

when $\lambda = 0$ o/p of value is simply avg of residual of leaf.

output of leaf ③: -7.5

Since we have build new tree we can make new prediction
like gradient boosting we have to use learning rate here also, default value is 0.3. rest is same

$\therefore$ initial leaf + learning rate $\times$ DT.

prediction for Dosage 10

$$0.5 + 0.3(-10.5) = -2.65$$

Now we can see new residual is smaller than before this means we are taking small step in right direction

Dosage = 20 = 2.6
Dosage = 22.5 = -1.75 } smaller than before.

Now we will build another tree based on new residual and make new prediction that can give us even smaller residual and then again another tree based on newest residual and we keep building trees untill residual are super small or we reach maximum number.

Summary:-

1) we calculate similarity score and gain to determine how to split the data

2) and we prune the tree by calculating the difference between gain value and user defined tree complexity parameter γ

3) $\text{Gain} - \gamma \begin{cases} \text{if positive then do not prune} \\ \text{if negative then prune} \end{cases}$

4) then we calculate output for remaining ~~values~~ leaves

5) λ (lambda): Regularization parameter and when $\lambda > 0$ it result in more pruning by shrinking the similarity score and smaller output value for the leaves

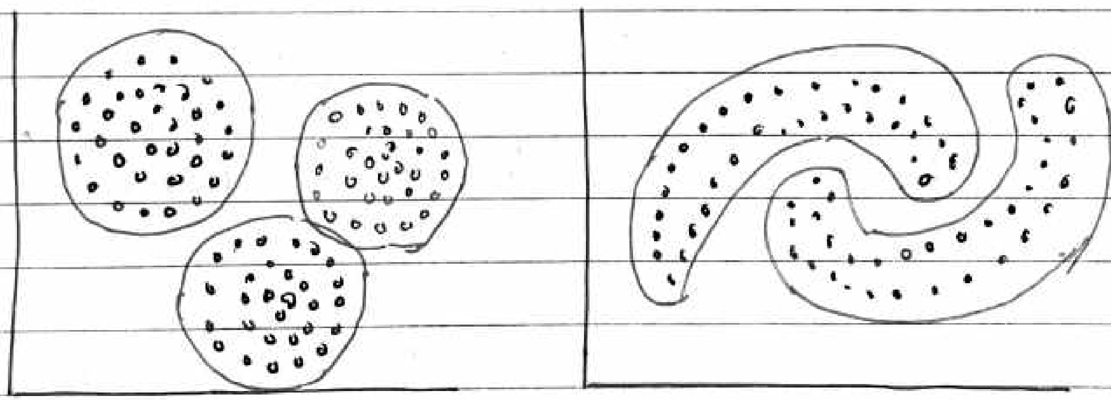
$$\text{Similarity Score} = \frac{[\text{Sum of residual}]^2}{\text{No of residual} + \lambda}$$

$$\text{output value} = \frac{\text{Sum of Residual}}{\text{Number of residual}}$$

Unsupervised Machine learning.

Clustering: it is basically type of unsupervised learning method. An unsupervised learning method is a method in which we draw reference from datasets consisting of input data without labelled responces. Generally it is used as a process to find meaningfull structure, explanotory underlying processes, generative features and grouping inherent in a set of example.

Clustering:- is task of dividing a population or datapoint into number of groups such that datapoint in some group are more similar to other datapoints in the same group and dissimilar to data points in other groups. it is basically collection of object on the basis of similarity and dissimilarity between them



k-means clustering algorithm mainly perform two task.

1) Determine the best value for k centre points or centroid by iterative process.
2) Assign each data point to its closest k centre. Those datapoints which are near to particular k-centre, creates a cluster.

* How does the k-means Algorithm work?

step1: Select the number of k to decide the number of clusters

step2: select random k point or centroids

step3: Assign each datapoint to their closest centroid, which will form the predefined clusters.

step4: calculate variance and place a new centroid of each cluster.

step5: Repeat the third step which mean reassign each datapoint to the new closest centroid of each cluster.

step6: if any reassignment occurs then go to step4 else go to finish.

step 7: model is reddy.

* **Drawback:** Before starting the process you must mention the number of cluster.

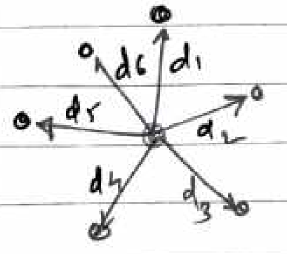
* How to choose best value of K "Number of cluster"

Elbow Method: This method use the concept of WCSS value. WCSS stands for within cluster sum of square, which defines the total variation within cluster. The formula to calculate the value of WCSS is given below. (for 3 cluster)

$$WCSS = \sum_{P_i \in cluster 1} distance(P_i C_1)^2 + \sum_{P_i \in cluster 2} distance(P_i C_2)^2 + \sum_{P_i \in cluster 3} distance(P_i C_3)^2.$$

where

$\sum Pi/cluster\,1$ distance $(Pi\,C_1)^2$: it is sum of square of distances between each datapoint and its centroid within custer 1 and same for other two terms



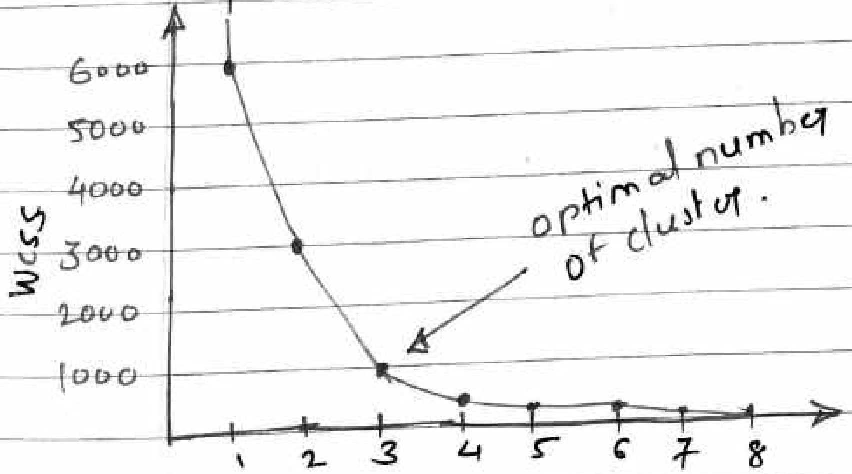
$= d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$

∴ to measure the distance we can use any method Eucledian ~~method~~ distance or manhattan distance.

cluster 1

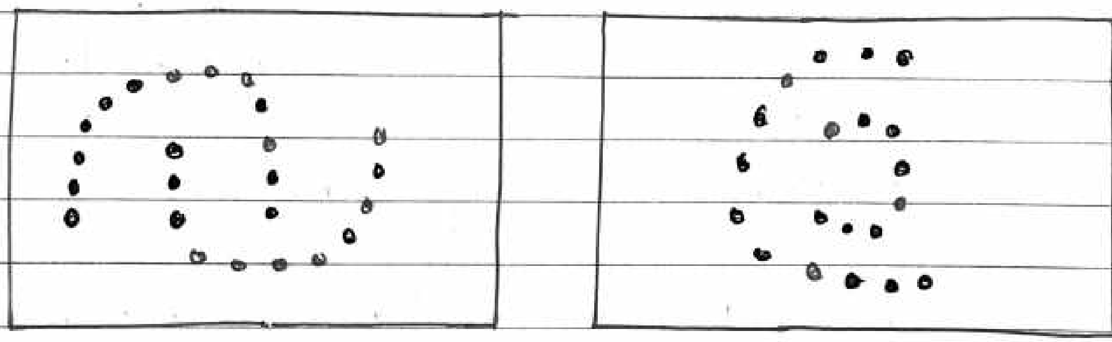
To find optimal value of clusters the elbow method follow below steps.

1) Executes K-means clustering on a given dataset for different k values (range from 1-10)
2) for each value of k calculate wcss value
3) plot curve between calculated wcss and Number of cluster
4) sharp point of bend or a point of ~~black~~ plot like an arm or elbow then that point considered as best value of K



DBSCAN it stands for Density Based spatial clustering of application with Noise.

- whenever we want to understand shape and structure there DB-SCAN is used.
- k-means and hierarchical clustering both fail in creating clusters of arbitrary shapes they are not able to form clusters based on varying densities. That why we need DBSCAN clustering.

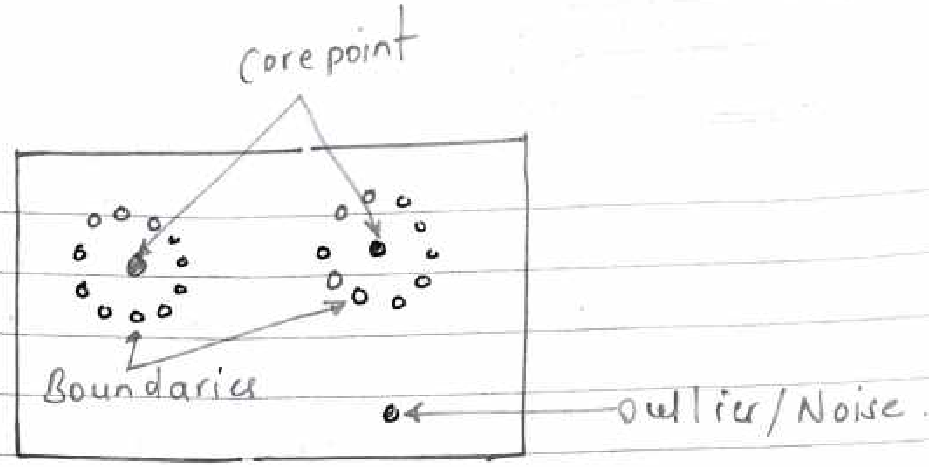


- The main idea behind DBSCAN is that point belong to a cluster if it is close to many points from that cluster.

There are two key parameter of DBSCAN.

1) eps (ε): also called as epsilon the distance that specifies the neighborhood. Two points are considered to be neighbors if the distance between them are less than or equal to eps.
2) Min pts: Minimum number of datapoints to define a cluster.

Based on the two parameter points are classified as Core point, border or boundary point and Noise or outlier.



Core point :- A point is core point if there are atleast min pts number of points (including the point itself) in its surrounding area with radius epsilon (ε)

Boundary point :- A point is boundary point if it is reachable from core point and there are less min pts number of ~~pts~~ points within its surrounding area. or in simpler words the points which covered the core point.

Noise/outlier :- A point is an outlier if it is not core point and not reachable from any core points.

Pros and cons of DBSCAN.

Pros :- 1) Does not require to specify number of cluster beforehand.

2) perform well with arbitrary shapes cluster.

3) it is robust to outlier and able to detect outlier.

Cons :- In some cases determining appropriate distance of neighborhood (eps) is not easy and it requires domain knowledge.

cluster validation technique :-

- elbow method can only used for kmeans but silhouette score can be used everywhere

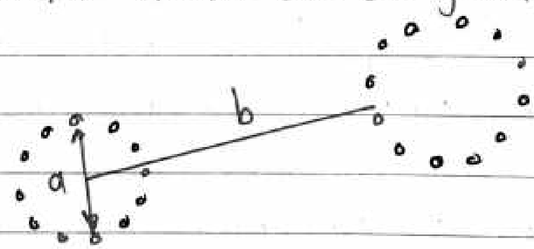
silhouette coefficient :- silhouette coefficient or score is a metric used for calculate the goodness of clustering technique

- for best cluster distance between the cluster must be greater than distance within the cluster.
- it's value ranges from -1 to 1

1 : Means cluster are well apart from each other and clearly distinguished.

0 : cluster are indifferent or we can say distance between cluster is not significant.

-1 : Means cluster are assigned in wrong way.



$$\text{silhouette score} = \frac{(b-a)}{\max(a, b)}$$

where,

a = avg intra-cluster distance i.e the average distance between each point within a cluster

b = avg inter cluster distance i.e avg distance between all the clusters.

hierorchical clustering :- Hierorchical clustering is another unsupervised machine learning algorithm which is used to group the unlabelled dataset into cluster and also known as hierarchical cluster analysis or HCA

- HCA is in form of tree and this tree shaped structure called as dendogram.
- result of kmeans and HCA may look similar but they both work differently. As here no requirment of predetermined the number of clusters as we did in k-means.
- HCA technique has two approches.

1. Agglomerative :- it is bottom up approch in which algorithm starts with taking all the points as single cluster and merging them untill one cluster is left
2. Divisive :- it is reverse of agglomerative algorithm as it is topdown approch.

How HCA works :-

step 1 :- Create each datapoint as single cluster for n number of datapoint n number of clusters.

step 2 : Take two ~~datapoints~~ closest datapoint or cluster and merge them to form one cluster So there will be n-1 cluster.

step 3 : Again take two closest cluster and merge them together to form one cluster now there will be n-2 cluster.

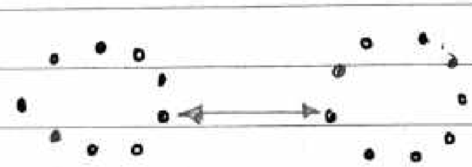
step 4 : Repeate step 3 untill one cluster left

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

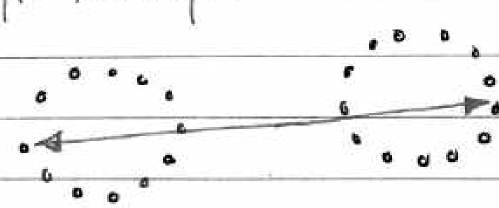
Measure of distance between two clusters :-

since we have seen closest distance between two cluster is very crucial for HCA there are various ways to calculate the distance between two clusters and these ways decide the rule for clustering these measures are called as linkage methods some of the popular linkage method are given below.

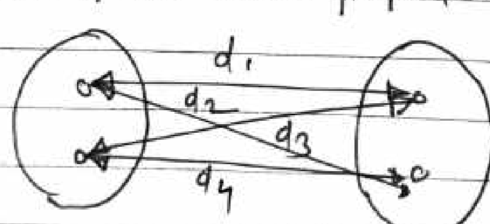
1. Single linkage :- it is shortest distance between the closest point



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

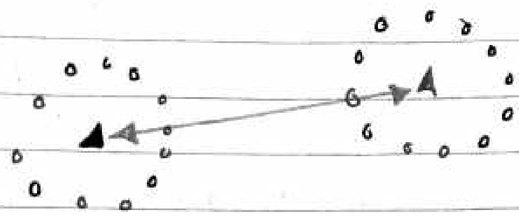


3) Average linkage :- it is linkage method in which distance between each pair of datapoint is added up and the divided by total number of datapoint to calculate avg distance between two clusters, it is also one of the most popular linkage method.



$$= \frac{d_1 + d_2 + d_3 + d_4}{4}.$$

4) Centroid linkage :- it is linkage method in which the distance between the centroid of the cluster is calculated.



Appriori - Association rule Mining.

Apriori Algorithm uses frequent itemset to generate association rule and it is designed to work on the databases that contain transaction. with the help of these association rule it determine how strongly or weakly two or more object are connected it is iterative process for finding the frequent itemset from large dataset.

Steps of Apriori :

step1 : Determine the support of itemset in the transactional database and select the minimum support and confidence.

step2 : Take all the support in the transaction with higher support value than the minimum or selected support value.

step3 : find all the rules of this subset that have higher confidence value than the threshold or minimum confidence value.

let understand with Example

1	Milk	Egg	bread	Butter
2	Milk	Butter	egg	ketchup
3	Bread	Butter	ketchup	
4	Milk	bread	butter	
5	Bread	butter	Cookies	
6	Milk	bread	butter	Cookies
7	Milk	Cookies		
8	milk	bread	butter	
9	bread	butter	Egg	Cookies
10	milk	butter	bread	
11	milk	bread	butter	
12	milk	bread.	Cookies	ketchup

- Here objective is to use this transaction data is to find affinities between products that is which product sell together often
- The support level be at 33% and confidence at 50%

$$x \rightarrow y \quad \text{support} = \frac{\text{frequenc}(x,y)}{N}$$

$$\text{confidence} = \frac{\text{frequency}(x,y)}{\text{freq}(x)}$$

step 1

1 item set	freq.
Milk	9
bread	10
butter	10
Egg	3
ketchup	3
Cookies	5

→

frequent 1 item set	frequeny
Milk	9
Bread	10
Butter	10
Cookies	5

{this satisfies min support which is 33%.}

$$\frac{x \text{ item bought}}{\text{total no. of transaction}} \quad = \text{for milk} = \frac{9}{12}$$

$$\boxed{= 0.75}$$

step 2 : From this one item set Now we will calculate frequency for two item set.

2 item set	Frequency
milk, bread	7
milk, butter	7
milk, cookies	3
bread, butter	9
bread, cookies	4
butter, cookies	3

Frequent 2 itemset	Frequeny
milk, bread	7
milk, butter	7
bread, butter	9
bread, cookies	4

based on min support threshold

step3 Now from this two item set we will calculate frequency of three itemset.

uniqu item are: Milk, Bread, butter, Cookies

3-item set ~~frequency~~	frequency	
milk, bread, butter	6	✓
milk, bread, cookies	1	
bread, butter, cookies	3	
milk, Butter, cookies	2	

based on support thrushold.

milk, bread, butter — is only valid with support of 0.33. or 33%.

Now ik time to identify association rules.

- subset creation

frequent 3-item set = i => Milk, bread, butter.

Non empty subset are:

(Milk), (bread), (butter), (milk, bread), (milk, butter), (bread, butter).

How to form association rules.

for every non-empty subset s of i the association rule is

$s \rightarrow (i-s)$

if support (i)/support (s) >= min_confidence.

for ex. milk → (bread, butter)

$$\frac{\text{support (milk, bread, butter)}}{\text{support (milk)}} \geq \text{min. confidence.}$$

		Support	Confidence	Status
1)	milk → (bread, butter)	50%	66.67%	valid.
2)	bread → (milk, butter)	50%	60%	valid
3)	butter → (milk, bread)	50%	60%	valid.
4)	(milk, bread) → butter	50%	85.7%	valid
5)	(milk, butter) → bread	50%	85.7%	valid
6)	(bread, butter) → milk	50%	66.67%	valid

$$\text{lift} = \frac{\text{Support}}{\text{Support}(A) \times \text{support}(B)}$$

this says how likely item B is purchased when item A is purchased while controlling for how popular item B is.

$$\text{Support} = \frac{\text{freq}(A, B)}{N}$$

.. { how popular an item is as measured by proportion of transaction in which itemset appears }

$$\text{Confidence} = \frac{\text{freq}(A, B)}{\text{freq}(A)}$$

.. { this says how likely item B is purchased when item A is purchased }

Recommendation System.

A recommendation system is subclass of information filtering system that seeks to predict the rating or the preferences a user might give to an item. In simple words it is an algorithm that suggest relevant item to users Eg. Netflix, Amazon.

There are three main type of recommendation system

1) Content Based recommendation system.
2) Popularity based recommendation system
3) Collaborative recommendation system.
   A) User-based collaborative filtering.
   B) item-based collaborative filtering.

1) Content Based recommendation system:-

it is type of recommendation system that suggest item to users based on characteristics or features of item themselves it analyze the content or attribute of item such as text, genre, actors or other metadata to identify similarities or pattern and uses this similarity to make recommendation.

~~The~~ it assumes that users will be interested in items that are similar to those they have liked or engage in the past.

2) Popularity Based recommendation System:-

it is simple type of recommendation system that suggest item to uses based on their popularity or overall popularity among all the users. it relies on the assumption that that popular item are more likely to be preferred by users and therefore recommends item that already

or have high number of ratings or engagment metrics.
- it typically rank the items based on their popularity metrics such as total number of views, rating, likes or sales and ~~ranks top~~ recommends top rank items to users, without considering user preference and behaviour.

Advantage :- 1) simplicity and ease of implimentation
2) Not needed complex Algorithm or user specific data

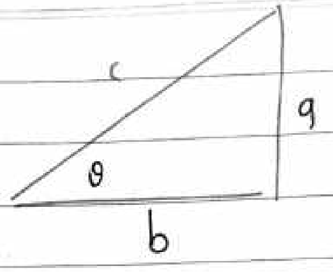
Disadvantage :- it does not take into account individual user preference interest or specific item characteristics.

<u>Collaborative Recommendation System</u> :- it is type of recommendation system that suggest item to users based on some pattern or user interaction or behaviours. it uses past behaviour of users such as rating, reviews or purchase history to identify similarities or pattern among users and item and make recommendation based on these similarities.

1) <u>User based colaborative filtering</u> :- In this approch similarities among the users are used to make recommendation Users who have similar behaviour or preference in the past are considered to have similar taste and item liked or rated highly by users with similar behaviours are recommend to target user.

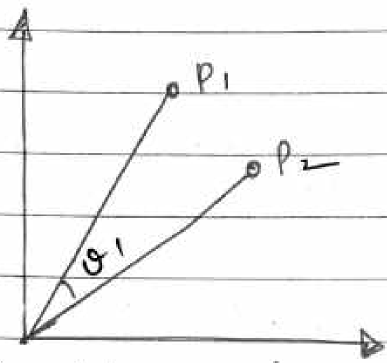
2) <u>item based collaborative filtering</u> :- In this approch similarities among the items are used to make recommendation. item that often liked or rated highly by same user in the past are considered to be similar and item similar to those liked or rated highly by the target user are recommended.

1) cosine similarity :

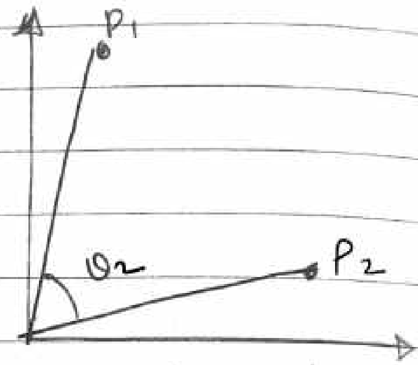


$\cos\theta = \frac{b}{c}$

as θ increase cosinθ decreases



since θ_1 is less cosine value will be high means similarity will be high



since θ_2 is high Cosine value will be less and thus similari
will be low

cosine distance = 1 - cosine similarity

Principal component Analysis

PCA is unsupervised learning Algorithm that is used for dimentionality reduction in machine learning. it is statistical process that converts the observations of correlated features into set of linearly uncorrelated features with the help of orthogonal transformation. these new transformed feature are called principle components. it is one of the popular tool that is used for EDA and predictive modelling. it is technique to draw strong patterns from the given dataset by reducing the variance.

- PCA generally try to find lower dimensional surface to project high dimensional data

- PCA works by considering the variance of each attribute because high attribute show good split between the classes and hence it reduces dimentionality some real world application of PCA are image processing, Movie recommendation system, optimizing the power allowcation in various communication channels. it is feature extraction technique so it contain important variable and drop least imp variable.

- The PCA algorithm is based on some methemotical concept such as
1) Variance and covariance
2) Eigen values and eigen vectors

- Some common term used in PCA algorithm.
1) Dimentionality :- it is number of feature or variables present in the given dataset, more easily it is number

of columns present in the dataset

Correlation :- it signifies that how strongly two variables are related to each other. such as if one changes, the other variable also gets changed. The correlation value ranges from -1 to +1. Here -1 occurs if variables are inversly proportional to each other and +1 indicates that variables are directly proportional to each other.

Orthogonal :- it defines that variables are not correlated to each other and hence correlation between the pair of variable zero.

Eigenvectors :- if there is square matrix M, and a non zero vector v is given. then v will be eigenvector if Av is the scaler multiple of v

Covariance Matrix :- A matrix containing the covariance between the pair of variables is called the covariance of Matrix.

Principal Components in PCA :- As described above the transformed new features or the output of PCA are the principal components The number of these PCs are either equal to or less than the original features present in the dataset. Some properties of these principal components are given below.

- The component component must be linear Combination of the original features.

- These components are orthogonal i.e the correlation between a pair of variable is zero.

- The importance of each component decrease when going to 1 to n it means the 1PC has most importance and n PC will have the least importance

Steps for PCA Algorithm :-

1. Getting the dataset :- firstly we need to take the input dataset and divide it into two subparts X and Y where X is training set and Y is the validation set.

2) Representing data into structure :- Now we will represent our dataset into a structure such as we will represent the two dimensional matrix of independent variable X. Here each row corresponds to the data items and the column corresponds to the feature. The number of columns is the dimension of the dataset.

3) standardizing the data :- in this step we will standardize our dataset such as particular column, the feature with high variance are more important compared to the features with lower variance.
if the importance of the feature is ~~depe~~ independent of the variance of the feature, then we will divide each data item in a column with the std deviation of the column. Here we will name the matrix as Z.

4) Calculating covariance of Z :- To calculate the covariance of Z we will take the matrix Z, and will transpose it. After transpose, we will multiply it by Z. The output matrix will be the covariance matrix of Z

5) Calculating the eigen values and eigen vectors. Now we need to calculate eigen value and eigen vectors for the resultant covariance matrix Z. Eigen vectors or the covariance matrix are the direction of the axes with high information And the coefficient of these eigen vectors are defined as eigen values.

6) Sorting the eigen vectors :- In this step we will take all the eigen values and will send them descending order, which means from largest to smallest And simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues The resultant matrix will be named as P.

7) Calculating the new features or principal components:- Here we will calculate the new features to do this we will multiply the P matrix to the Z in the resultant matrix Z, each observation is the linear combination of original features. each column of the the Z matrix is independent of each other.

8) Remove less or unimportant features from the new dataset: The new feature set has occured, so we will decide here what to keep and what to remove. It means we will keep only relevant or important features in the new dataset and unimportant features will be removed out.

Date: / /
Page

Application of PCA.

1) PCA is mainly used as the dimensionality reduction technique in various AI application such as. Computer vision, image compression etc.

2) it can also used for finding hidden patterns if data has high dimensions. some field where PCA is used are finance, data mining, psychology. etc.