Date: / /
Page

# Machine learning

(main)

## Supervised ML

| Regression | classification |
|---|---|
| • linear | logistic |
| • Ridge & lasso | KNN |
| • polynomial Reg. | DT |
| • Multilinear | Naive Byes |
| • Non-linear | RF |
| • OLS | XGB |
| • Timeseries forecasting | GBM |
| | SVM |
| | ANN |
| | catboost, adaboost |

## Unsupervised ML

| clustering | apriori | Recommendation | |
|---|---|---|---|
| • kmeans | | Content | Memory |
| • DBscan | | | |
| • hierarchical | | | 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 |
|---|---|---|

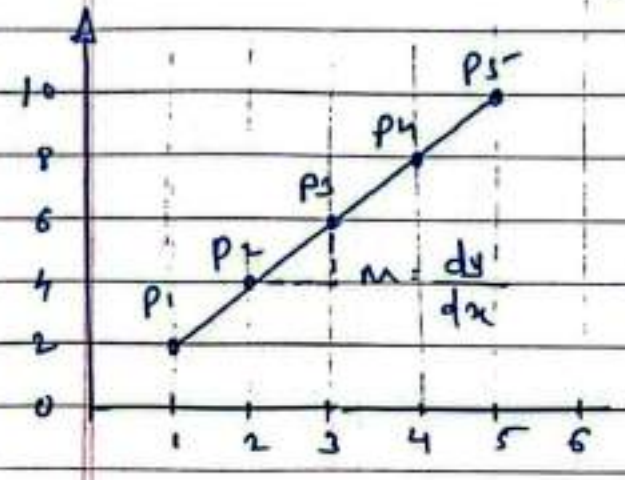
- Simple linear Regression (SLR) :- when for continuous output we have only one feature then it is called SLR

for Ex.

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

Page

# Basics of linier Regression.




Equation of straigh line

$y = mx + c$

find value of m and c

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

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

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

$m = \frac{2}{1} = 2$

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

Intercept = c

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

$y = mx + c$

$8 = 2(4) + c$

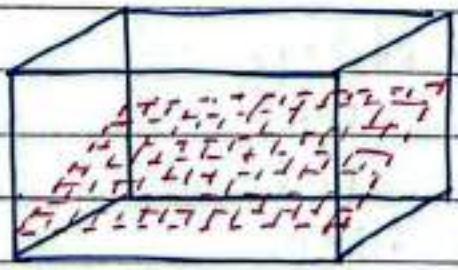
$8 - 8 = c = 0$

Inference : The above line eqn 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 independent variable ?

→ then it is called multiple linear regression



(3D)

simple linear regression : $y = mx + c$

multiple linear regression

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

Advantages : 1) simple to implement

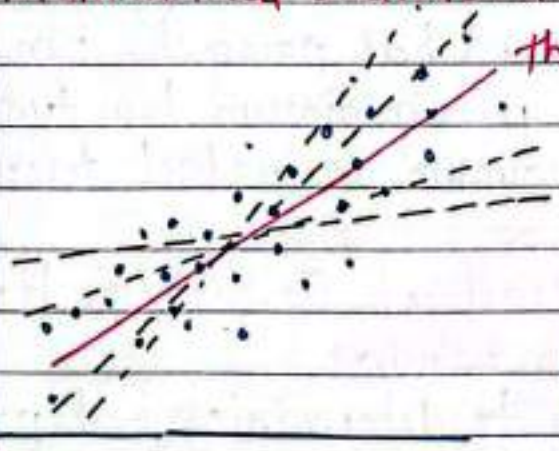
2) perform well on the data with linear relationship.

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

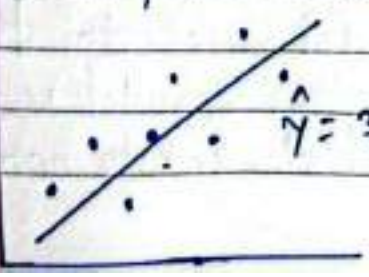
so here purpose is to find best fitted line (model) for linear regression.

{ 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 parameter are better.

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

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.

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

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

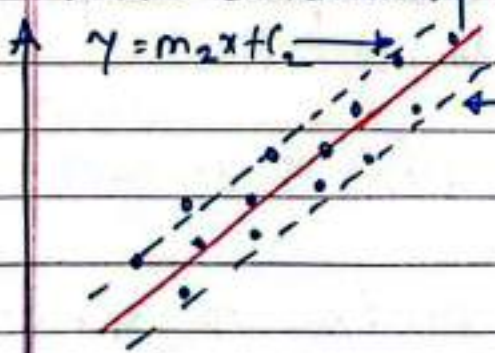
{ reason to take square :. so that positive and negative value may not cancel each other }

low loss value ⟶ High Accuracy
high loss value ⟶ low Accuracy

We can improve the model by some optimization technique called as "gradient descent" where repeat the process iteratively until 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.

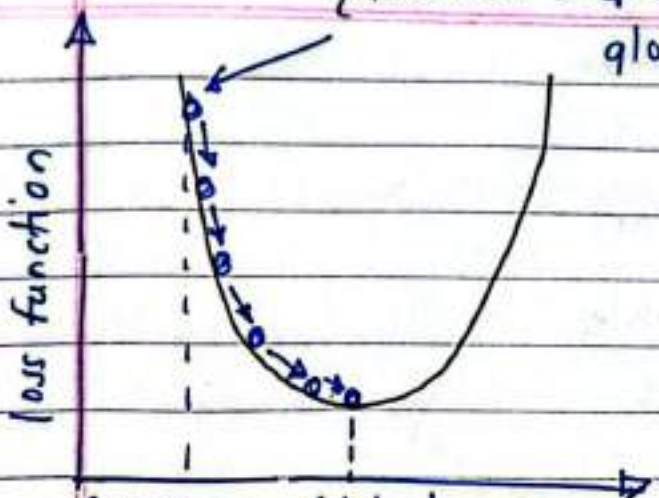
⟶ optimization refers to determining best parameter for model such that loss function of the model decreases as result of which model can predict more accurately.




{ 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

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




updated. → $m = m_1 - L\,Dm$, $c = c_1 - L\,Dc$ ← initial

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

In machine learning

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

w = weight
b = bias.

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

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

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

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

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

Interms weigh and bias

$$y = wx + b$$

(weight) (biar)

x - independent Variable

y - dependent Variable

w - weight

b - bias

updated, initial, learning rate($\alpha$), change in Loss function w.r.t w

$$w = w - \alpha dw$$

change in lors function w.r.t. b.

$$b = b - \alpha db$$

Learning rate:- it is tuning porometer in an optimize on olgorithm that determine step size at each iteratio while moving toword minimum of lors function.

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

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

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

prediction Equation for MLR

$$\hat{y} = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + b.$$

and,

Regression Equation

$$y = \underbrace{a_1 x_1 + a_2 x_2 + a_3 x_3 + \cdots + a_n x_n + b}_{\text{predicted value}} + \varepsilon$$

$y$: Actual value.

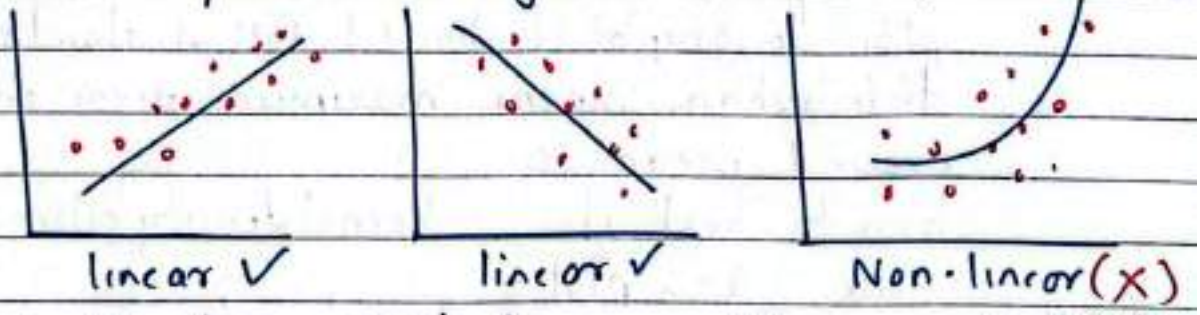
$\varepsilon$: Error

# 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 pasitive 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 $a_i$ 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)

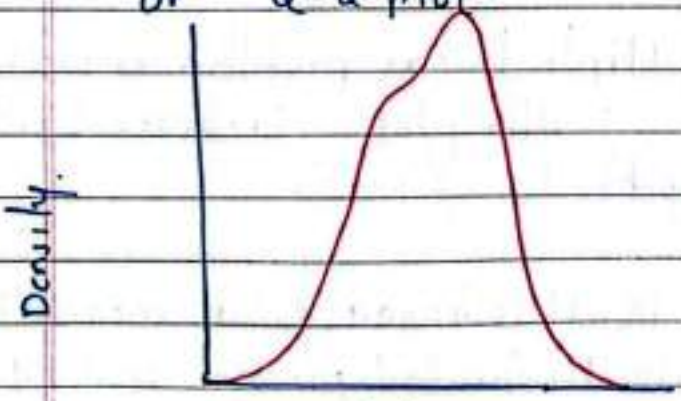
Date: / /
Page

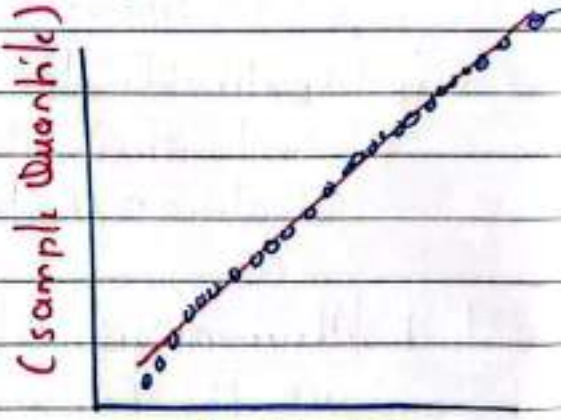
how to check multicollinearity :-

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 perticular feature have multicollinearity issue and need to remove it

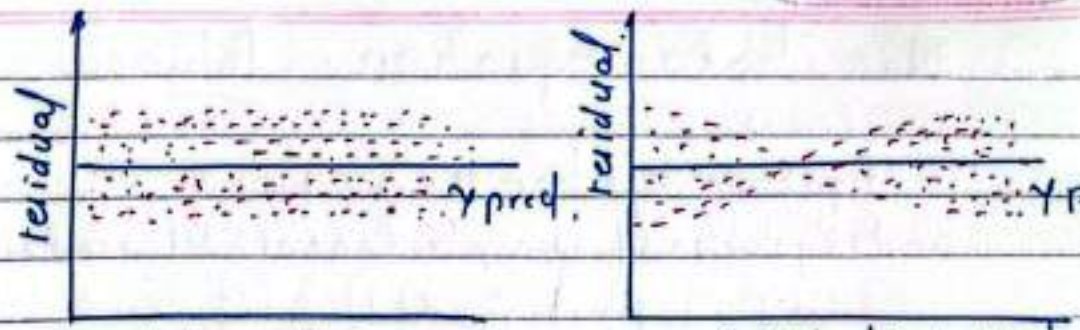
2) another method is to find out correlation between all the features (Heatmap)

3 **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




4. **Assumption 4**: Homoscedasticity.
same scattered / spread.
∴ spread of residual should be equally or uniformly scattered. if it is not it called as heteroscedasticity which is not desirable.

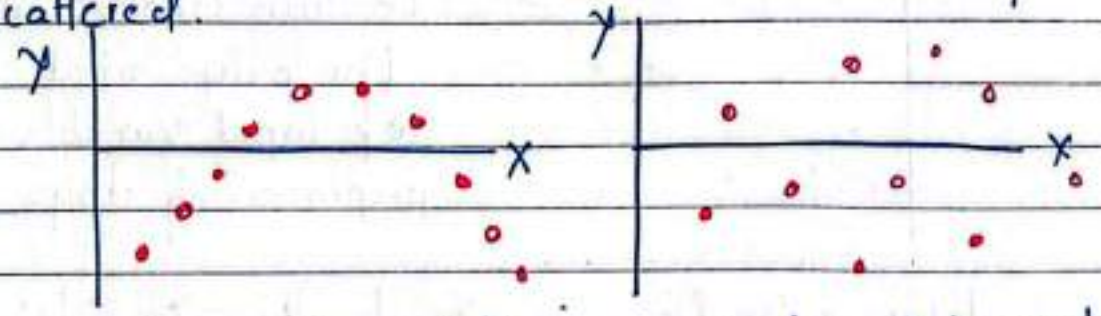



(desirable) (Not desirable)

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 | - | - | - |

Date: / /
Page

# Non-linear Equation - Polynomial Regression

we know

Equation of line $y = mx + c$

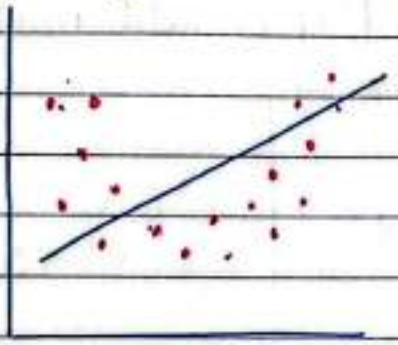
and Equation of simple Linear Regression

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

and for multiple linear Equation.

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

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 | y → 5 | 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 chances 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 models 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 Algorithom 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 | $\Sigma x = 28$ $\Sigma y = 61.8$ |
| 2 | 3.8 | 7.6 | 4 | $\Sigma xy = 314.8$ $\Sigma x^2 = 140$ |
| 3 | 6.7 | 20.1 | 9 | |
| 4 | 9.0 | 36 | 16 | $n = 7$ |
| 5 | 11.2 | 56 | 25 | (number of data points) |
| 6 | 13.6 | 81.6 | 36 | |
| 7 | 16 | 112 | 49. | $y = mx + b$ |
| $\Sigma$ 28 | 61.8 | 314.8 | 140. | |

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

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

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

$$b = -0.8285714.$$

to get linear equation we should plug value. in

$$y = mx + b.$$

~~$y = 2.41x$~~ $y = 2.41x - 0.83$.

| let fit. for x | $\hat{y}$ | $y_{act}$ |
|---|---|---|
| $y = 2.41(2) - 0.83 =$ | 3.99 | 3.8 |
| $y = 2.41(5) - 0.83 =$ | 11.22 | 11.2 |
| $y = 2.41(7) - 0.83 =$ | 16.04 | 16. |

Syntax: statsmodel.api.OLS(y, x).
y: depended variable x - independent 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()

Date: / /
Page

```
# 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 squares are unbiased and variance are large this result in predicted values being far away from actual values.

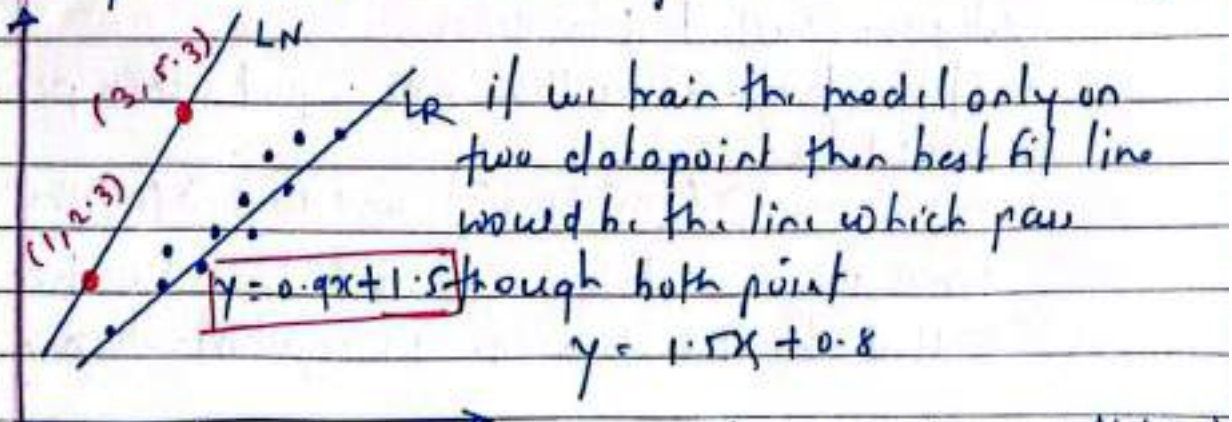
Regularization:- it is technique used to calibrate machine learning model to minimize adjusted loss function and avoid overfitting and underfitting.

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 define 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

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

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

$\lambda$ = 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.

Date : / /
Page

# Lasso Regression (L1 Regularization)

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

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

coefficient in MLR

Lasso is just another variation of Ridge

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

OR

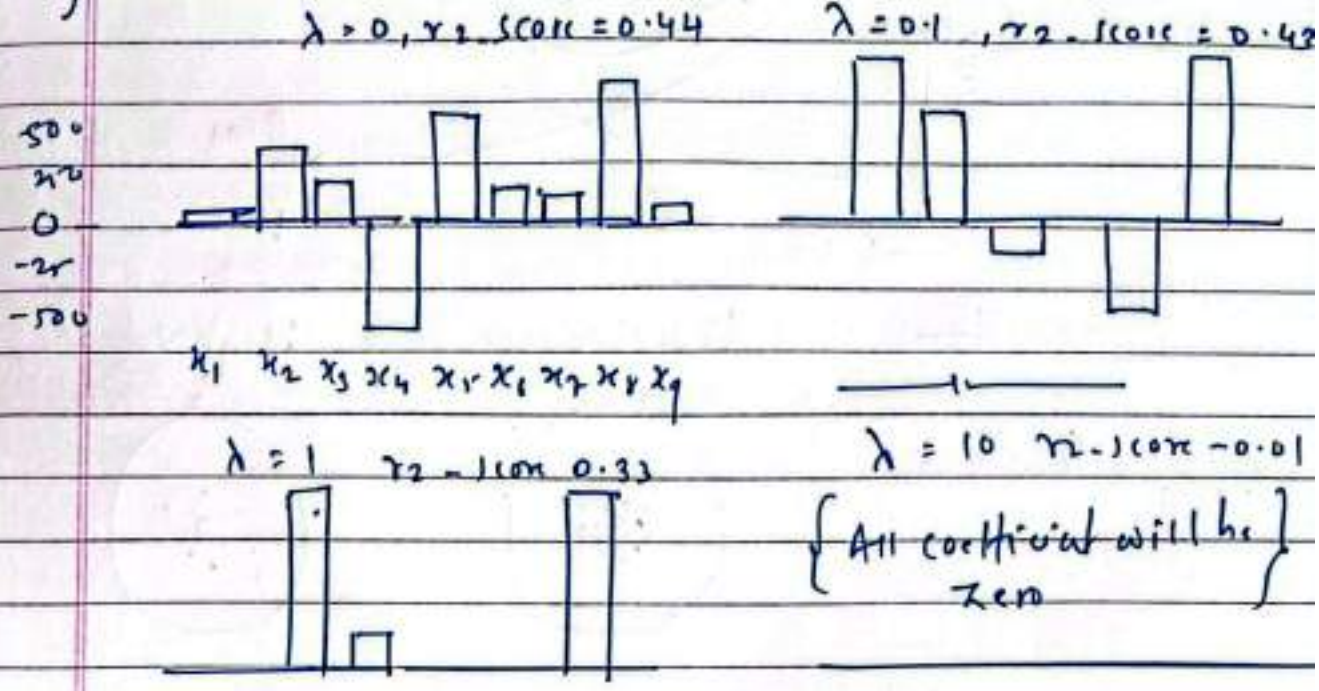
$$= MSE + \lambda [|w_1| + |w_2| + |w_3| + \cdots + |w_n|]$$

- In ridge regression for any value of $\lambda$ there were always some value for coefficient of input feature but in lasso if you continously increase value of $\lambda$ 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.

Date: / /
Page

there are some keypoints need to discuss about Lasso.

1) How coefficient are affected. ?




2. Higher coefficient are affected more

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

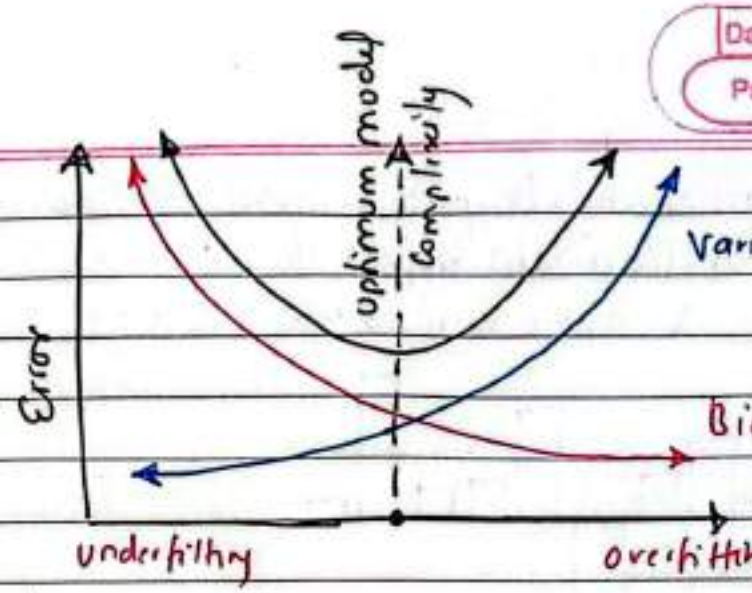
3. Impact on Bias and Variance

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

| High Bias | High Variance |
|---|---|
| ↓ | ↓ |
| Underfitting | overfitting |

Date: / /
Page

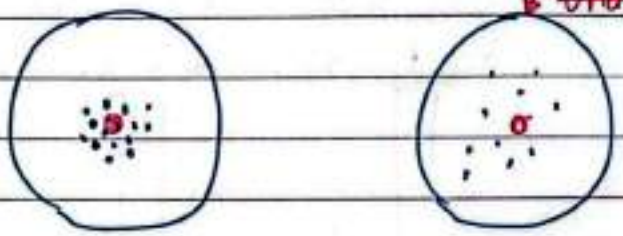
fig a.




low
Low Variance
High Variance
over underfitting

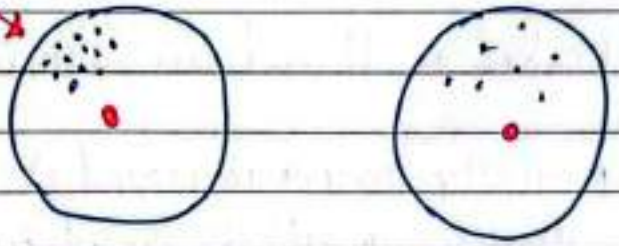
fig(b

low bias



underfitting

High bias



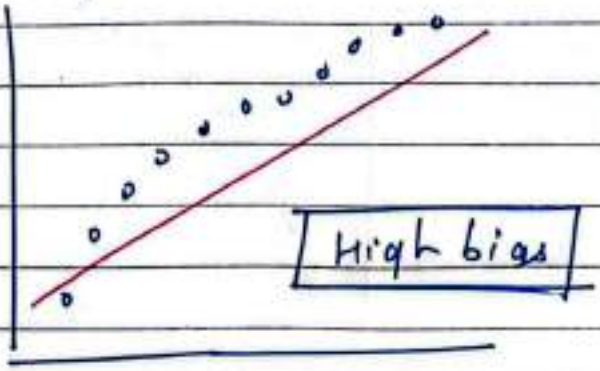
## Bias-Variance Tradeoff.

- 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 referred 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

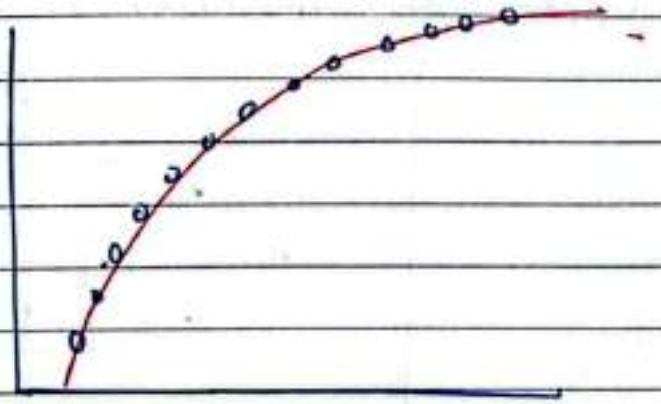



**Variance** . The variability of model prediction for given data point which tells us spread of our data is called variance of model.

- 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 rate 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 eq^n) 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 perform better and which parameters are better loss function is useful.

Elastic Net :- (L1+L2) : it is combination of both regularization technique

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

hyperparameter

L2 reg = $\frac{\sum (y_i - \hat{y}_i)^2}{n} + \underbrace{\lambda(m^2 + c^2)}_{\text{penalty}}$

Elastic (L1+L2) :-

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

adding combination percentage of Lasso & Ridge

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

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 then i would go for Elastic Net.

Date: / /
Page.

# Time series forecasting.

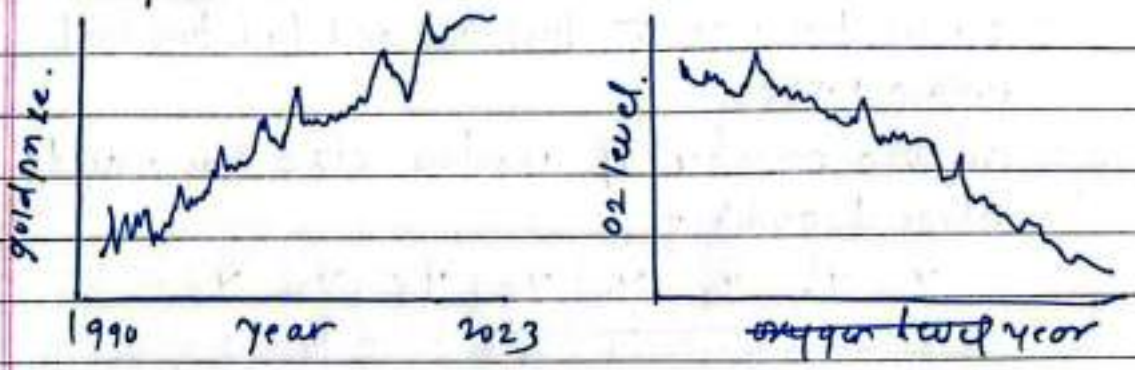
Time series :- it is data which is index by time.

- 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
  - C1) Data based prediction
  - C2) 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.

how to do smoothing :-

moving average :- Eg 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

$y_2 = \frac{y_1 + y_2 + y_3}{3}$ $y_3 = \frac{y_2 + y_3 + y_4}{3}$ 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

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

$y_3 = \frac{y_2 + y_3 + y_4 + y_5}{4}$ {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 but 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 whereas it becomes smooth.

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

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

② 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.

| Days | $D_1$ | $D_2$ | $D_3$ | $D_4$ | ... | $D_{199}$ | $D_{200}$ |
|---|---|---|---|---|---|---|---|
| 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

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

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

**Mathematical Understanding :-**

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

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

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

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

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

put (4) in (3)

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

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

if i rearrange this, we will get

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

$(\frac{1}{2})^1 \quad (\frac{1}{2})^2 \quad (\frac{1}{2})^3 \quad (\frac{1}{2})^4$

- 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

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

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

$L_3 = \frac{L_2 + S_3}{2} = \frac{1.5+1}{2} = \frac{2.5}{2} = 1.25$

$L_4 = \frac{L_3 + S_4}{2} = \frac{1.25+2}{2} = \frac{3.25}{2} = 1.625$

$L_5 = 1.3125$, $L_6 = 2.1562$

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 chocolate price increase by Rs 1 each day

Day: 1 2 3 4 5 6

price: 1 2 3 4 5 9

$L_6$

$L_6$ = around 4.2 which is not true.

Hence we need consider a 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.

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

(today's trend)

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

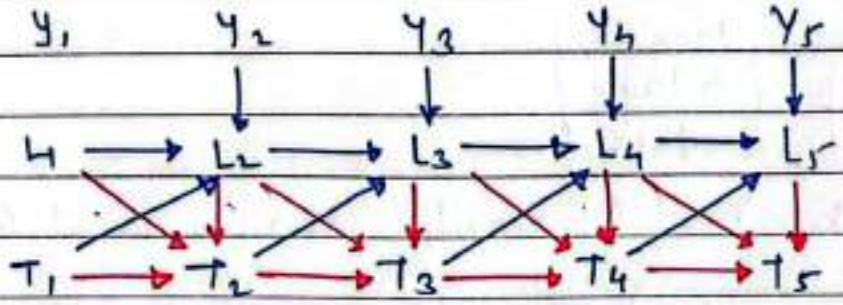
Date: / /
Page

| Day | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| price(y) | 1 | 2 | 3 | 4 | 5 |
| L(S) | 1 | 1.5 | 2.25 | 3.125 | 4.0625 |
| L(D) | 1 | 1.5 | 2.375 | 3.4687 | 4.6484 |

$\frac{1+2}{2} = 1.5$

$\frac{1.5+3}{2} = \frac{4.5}{2} = 2$

Now we will calculate with Double Exponential.




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

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

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

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

$L_3 = \frac{1}{2}(L_2 + T_2) + \frac{1}{2}(Y_3)$

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

$= 0.875 + 1.5$

$= 2.375$

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

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

$= 0.125 + 0.4375$

$= 0.5625$

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

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

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

$= 1.4687 + 2 = 3.4687$

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

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

$= 0.28125 + 0.5\,(1.0937)$

$= 0.28125 + 0.54685$

$= 0.8281$

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

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

$= 2.1484 + 2.5$

$= 4.6484$

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

$= \frac{1}{2}(0.8281) + \frac{1}{2}(4.6484 - 2.4687)$

$= 0.41405 + 0.5\,(1.1797)$

$= 0.41405 + 0.58985$

$= 1.0039$

| $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | } Trend is increased. |
|---|---|---|---|---|---|
| 0 | 0.25 | 0.56 | 0.82 | 1.0039 | |

| | Actual | Predicted. |
|---|---|---|
| Simple Exponential | 4.063 | 5 |
| Double Exponential | 4.6484 | 5 |

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

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

time series data.

| | | | | |
|---|---|---|---|---|
| | $Y_1$ | $Y_2$ | $Y_3$ | $Y_{200}$ |
| A | $L_1$ | $L_2$ | $L_3$ | $L_{200}$ |
| rend | $T_1$ | $T_2$ | $T_3$ | $T_{200}$ |
| sonality | $S_1$ | $S_2$ | $S_3$ | $S_{200}$ |

for 200 Months ⟶ 16 yr - 7 month
for seasonality — periodic time of 12 months

Let first convert it into double Expontial problem.
by subitracting seasonality from time series

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

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

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

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

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

↑

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 $Y_{200} - L_{200}$.

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

Date : 11
Page

if you are carefully observe then you find there is loop between ② 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^ ① $S_{200} = S_{188}$

$$L_{200} = \frac{1}{2}(L_{199} + T_{199}) + \frac{1}{2}(Y_{200} - S_{188})$$

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

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

Summary table.

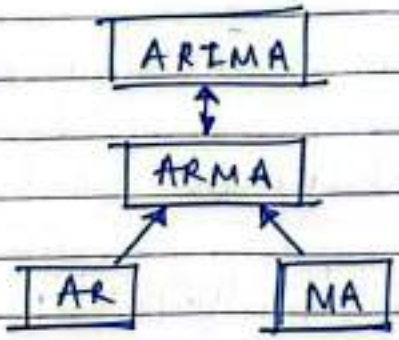
| Model | when we can apply | what should be future prediction | hyperparameter |
|---|---|---|---|
| Simple Exponential | No trend No seasonality. | last possible local Avg = all future prediction | 1 |
| Double Exponential | only trend No seasonality | last possible local Avg & changed trend | 2 |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

Date: / /
Page

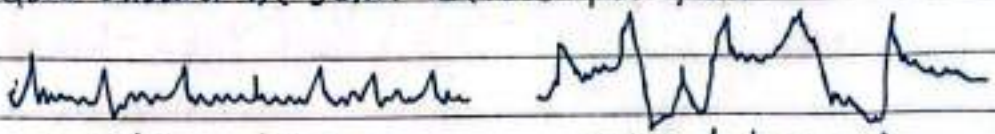
## Model Based forecasting :-

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 data based 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, data

Date: / /
Page

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

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

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

How to select Best possible model. AR1, AR2, AR3 ~~AR+~~ ~~obi~~ 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 fruit) due to natural calamities.

| data | $y_1$ | $y_2$ | $y_3$ | ... | $y_t$ |
|---|---|---|---|---|---|
| External factors | $\varepsilon_1$ | $\varepsilon_2$ | $\varepsilon_3$ | ... | $\varepsilon_t$ |

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

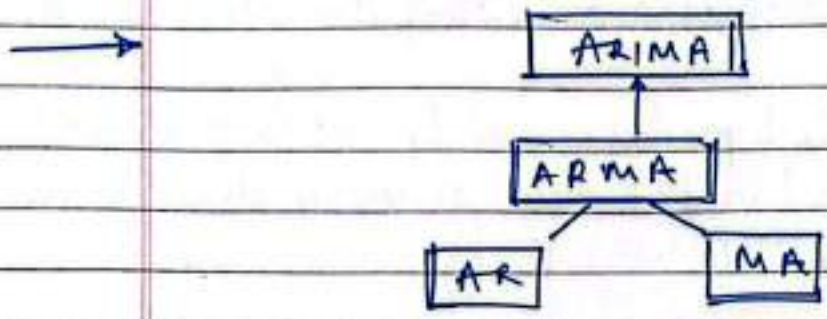
this year data / last year data | constant

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

Date: / /
Page

Model based forecasting.

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




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

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

target variable ↗ prediction ↖ Error.

$y_t = \alpha y_{t-1} + \beta + \varepsilon$ ——— AR(1)

past data — it it is depend on one past data

$y_t = \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \alpha_3 y_{t-3} + \beta + \varepsilon$

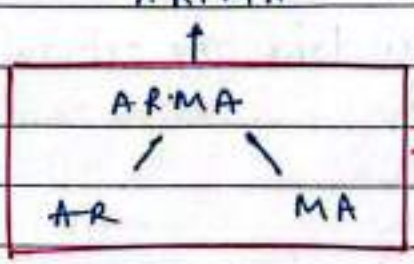
tomorrow ↑ today ↑ yesterday ↑ day before yesterday

it is similar to SLR

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

Date: / /
Page

- 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.




← Since they are critical must sure for stationary data.

MA → Moving Average :- here future prediction are completely depend upon external factor.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| production of milk → | $Y_1$ | $Y_2$ | $Y_3$ | $Y_4$ | $Y_5$ | $Y_6$ | $Y_7$ |
| Error due to external factor → | $\varepsilon_1$ | $\varepsilon_2$ | $\varepsilon_3$ | $\varepsilon_4$ | $\varepsilon_5$ | $\varepsilon_6$ | $\varepsilon_7$ |

- 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

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

depend on External Error factor.

$$Y_t = \alpha(y_{t-1}) + \beta + \varepsilon_t$$ ——— AR to 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 introduced.

**ARMA**: it is combination of AR and MA model.

| ARMA(1,1) |
|---|
| AR(1) + MA(1) |

ARMA(P, q) = AR(P) + MA(q)

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

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

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

if $\alpha_1 t_{-1} = 0$ then it will be MA(1)

- thats why you don't 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.

- thing we need to take of for these three models
i) stationary :: if it is stationary we gonna directly used ARMA instead 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  6 x 6 = 36
q : {0, 1, 2, 3, 4, 5} = 6

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

**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.

$Y_1$ $Y_2$ $Y_3$ $Y_4$ . . . . $Y_n$.
↓ ↓ transformed data ↓ ↓
$Z_1$ $Z_2$ $Z_3$ $Z_4$ $Z_n$

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

Date: / /
Page

for ex.

Bank balance of Every month

$y_1$ $y_2$ $y_3$ $y_4$ . . . . $y_t$ ← (today)

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

what will interest added each month

↓ ↓ ↓ ↓ ↓

$z_1$ $z_2$ $z_3$ $z_4$ $z_t$

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 then perfect data will follow AREMA

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 |
|---|---|---|
| $z_1 = y_2 - y_1$ | $z_1 = y_3 - y_1$ | $z_1 = y_4 - y_1$ |
| $z_2 = y_3 - y_2$ | $z_2 = y_4 - y_2$ | $z_2 = y_5 - y_2$ |
| : : : | : : : | : : : |
| $z_{100}$ $y_{101}$ $y_{100}$ | $z_{100} = y_{102} - y_{100}$ | $z_{100} = y_{103} - y_{100}$ |

what is lifecycle of ARIMA

Transformation

Raw data → Substraction degree → Apply ARMA on New Model.

Page

## Summary table.

| Model | No of parameter | Condition | 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, ARIMA

{ stationary data }

{ when we are not sure directly go for ARMA }

# Evaluation 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

first can : $\frac{|3-2|}{3} = \frac{1}{3} = 33.3\%$.

Date : / /
Page

Second case : $\left|\frac{1000-999}{100}\right| = 0.00|$

$0.01\%$

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

Exercise :- 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 :

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

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

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

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

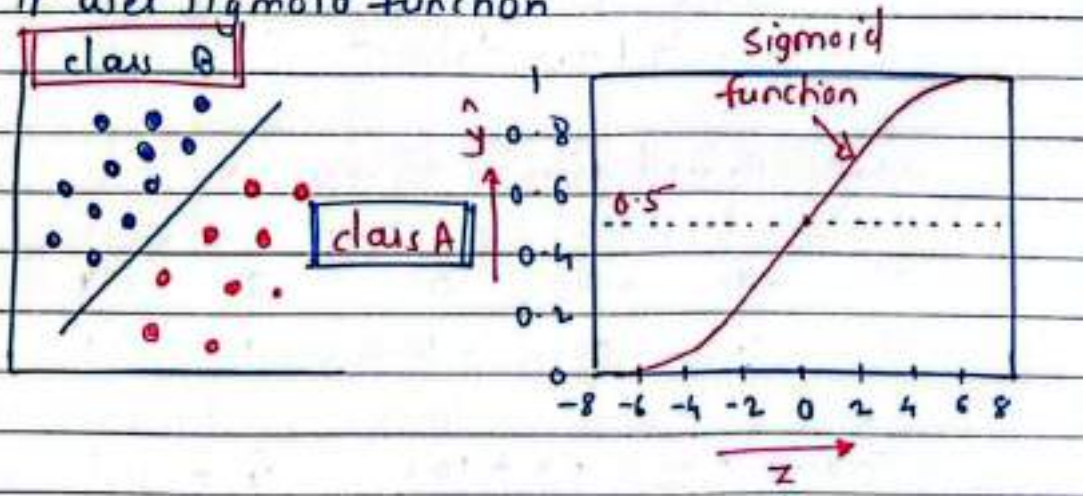
Date: / /
Page

# 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




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

where $z = wx + b$.
$(mx + c)$ - eq^n of line

slope m → weight w
intercept c → 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 this format)
{number of weight equal to number of feature in the dataset}
b = bias
$\hat{y} = \sigma(z)$

Date: / /
Page

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

**Disadvantage :-** 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 $z = 5x + 10$ $\quad \hat{y} = \frac{1}{1+e^{-z}}$

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

Date: / /
Page

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

**Disadvantage :-** 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 $z = 5x + 10$ $\quad \hat{y} = \frac{1}{1+e^{-z}}$

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

Date : / /
Page

Inference :- if z value is large positive number.

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

if z is large negative number.

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

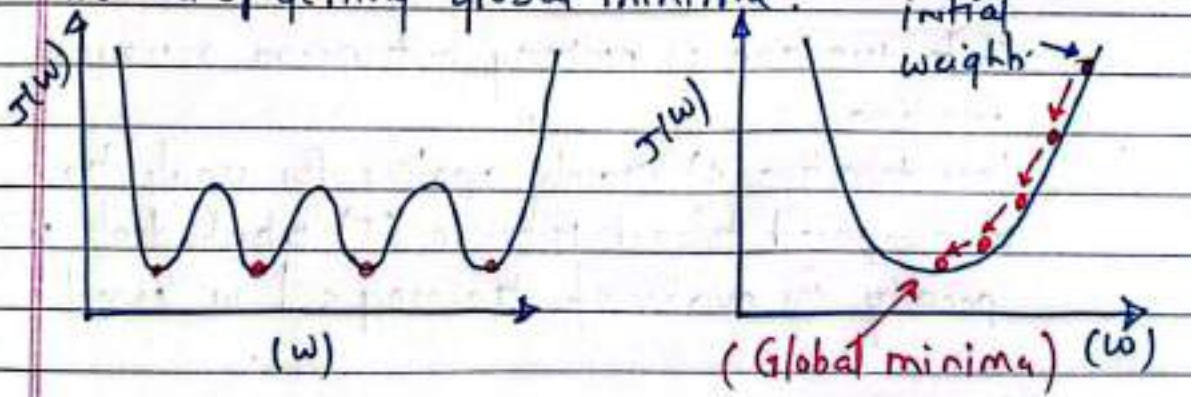
$$\hat{y} = 0.$$

loss function & cost function for Logistic Regression.

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

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

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 $1$

& $\hat{y} \longrightarrow 0$ 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:-

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

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

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

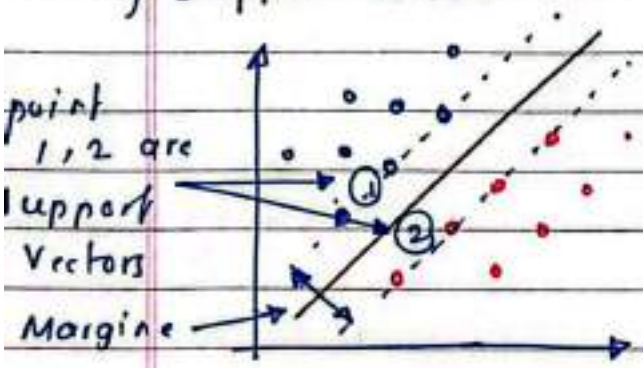
{'m' denotes number of data-points in the training set}.

Date: / /
Page

# Support Vector Machine (SVM)

Basic about SVM.

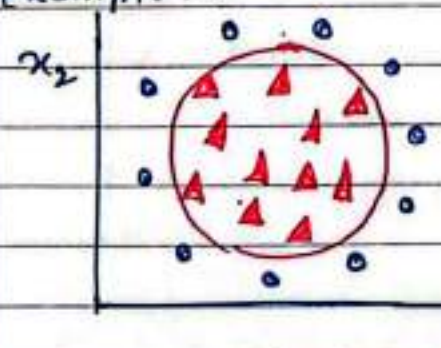
1) it is supervised ML model.
2) it can be used for both classification as well as regression but it is predominantly used for binary classification.
3) Hyperplane.
4) Support vectors



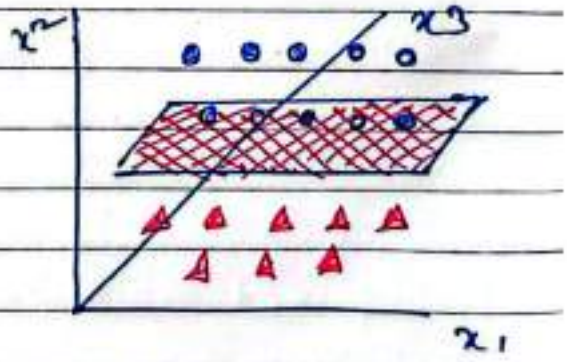

position of hyperplane depends upon support vectors.

- 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

Date : / /
Page

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

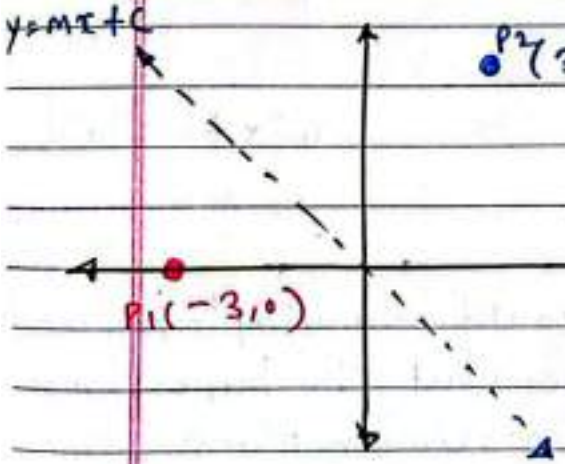
**Advantages of SVM.**

1) works fine with smaller dataset
2) works fine or efficiently where there is clear margin 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 (owlier) 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 ~~row~~ no. of rows of Second Matrix ]

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

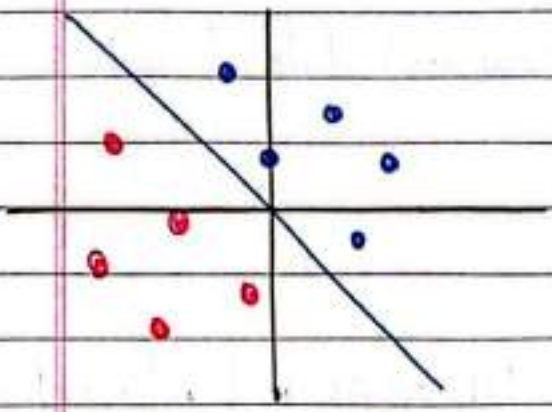
for P2(3,3)

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

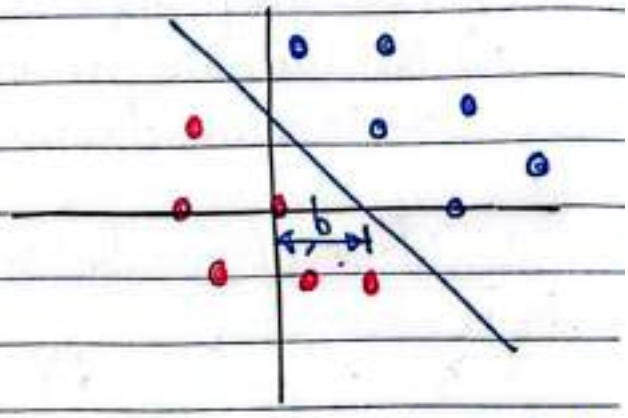
$$= -3 \text{ (Negative)}$$

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 Origin.

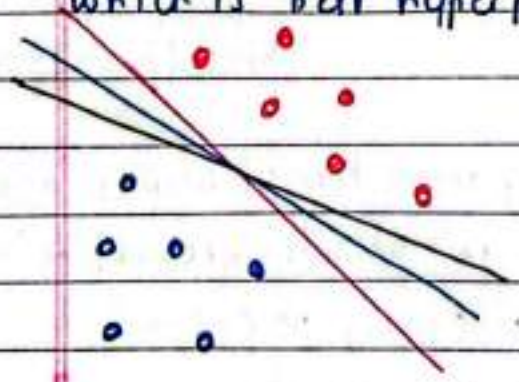


$w^T x = \text{label}$




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

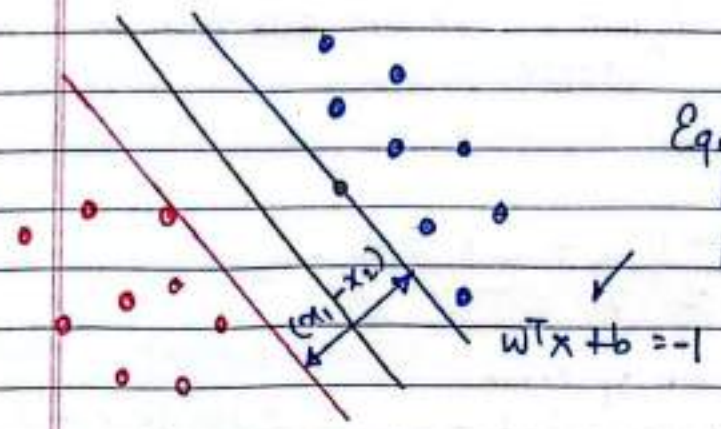
which is best hyperplane ?



there could be multiple hyperplane, but the hyperplane with maximum margin 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.

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

to get margin let substract one from another.

$$w^T x_1 + b = 1$$
$$(-)\ w^T x_2 + b = -1$$
$$w^T (x_1 - x_2) = 2$$

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

divide both side by $\|w\|$

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

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

and

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

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

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

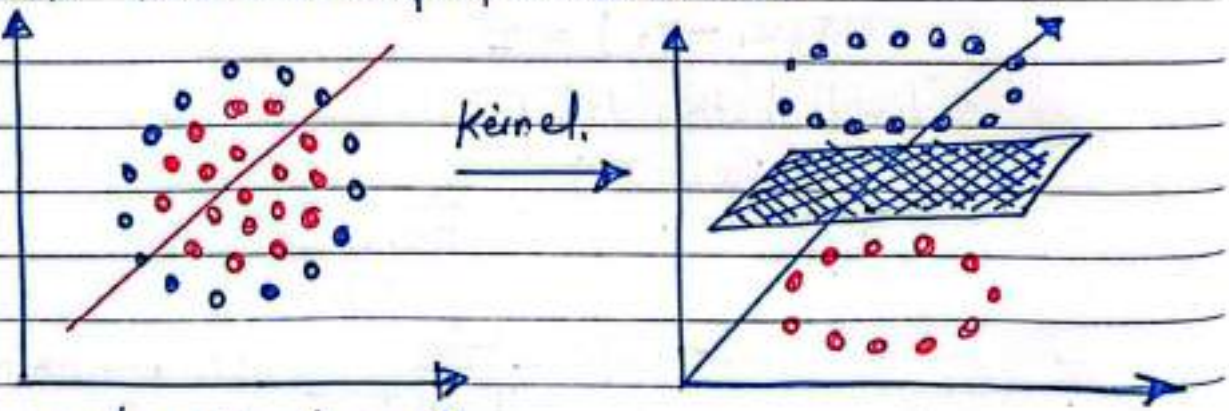
instead of using $\max_{\lambda}\left(\frac{2}{\|w\|}\right)$ we can also try Min which make better sense

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

(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
$\xi_i$ : 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



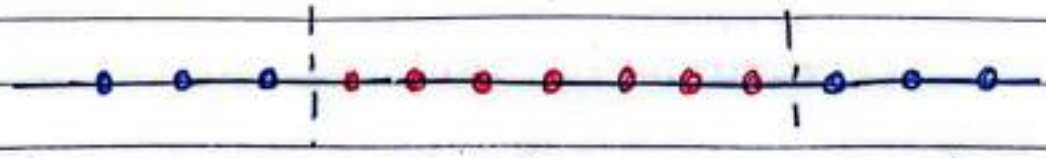

Type of SVM kernel.

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 |

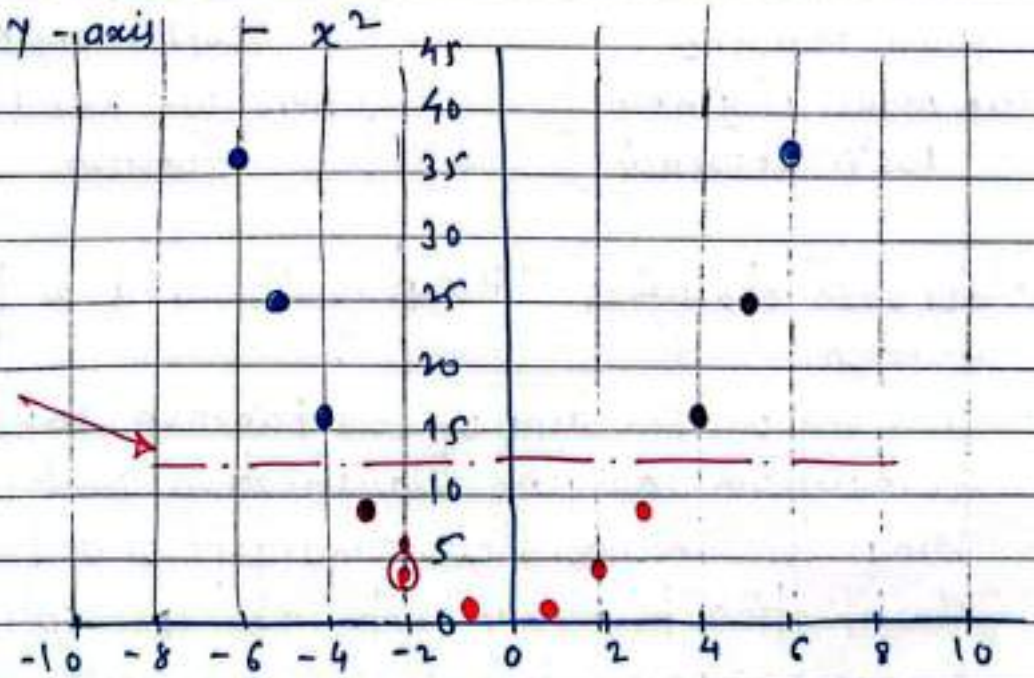
if you try to plot x on this 1D line.



- we can see None of the line could seperate the two class pufectly.
- 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 separable data.

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

{best suitable for having too many features}

2) polynomial kernel.

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

d : ...

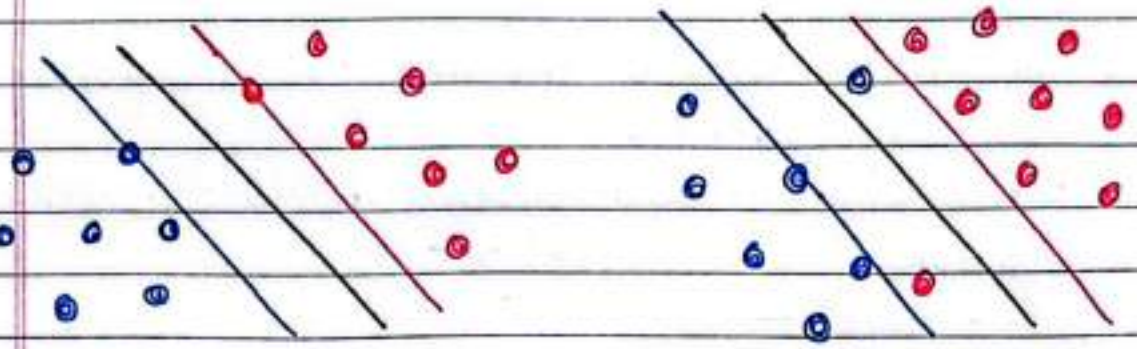
3.) radial basis function. (rbf kernel)

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

4. Sigmoid function

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

## Loss function for SVM classifier.



Hard Margine
{Here model is giving 100% accuracy}

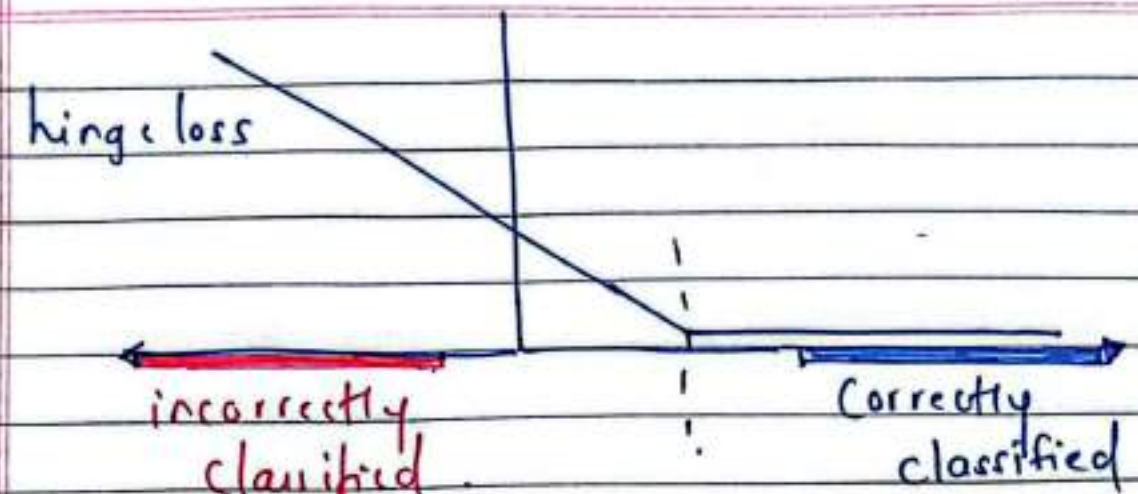
Soft margine
{Here we need loss function}

- 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 tal for misclassification.

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

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

$= 1 + 1$ $\quad$ $= 1 + 1$

$= 2$ $\quad$ $= 2$

{ both are high loss value }

Now tal for correct classification.

$y_i = 1$ $\quad$ $\hat{y}_i = 1$ $\quad$ $y_i = -1$ $\quad$ $\hat{y}_i = -1$

$(0 - (1)(1)$ $\quad$ $(0 - (-1)(-1)$

$0 - 1$ $\quad$ $0 - 1$

$-1$ $\quad$ $-1$

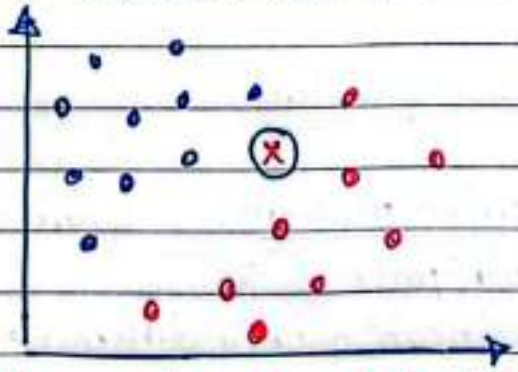
{ both are low loss value }

Date :
Page

# 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 other wise 0 for binary classification some principle for multiclassification problem.
- In general K is odd number

let understand with some example



Now we have to find new added (x)

Working of KNN Algorithm.

step 1: Lodding training as wellas 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.

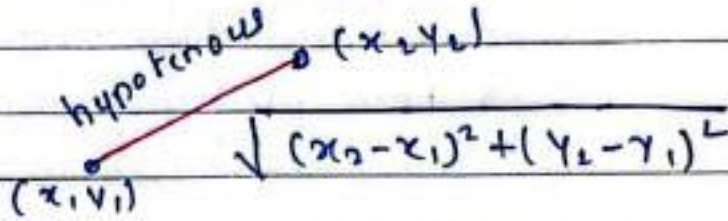
3.2) Now based on its distance value, sort them in ascending order.

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) Euclidian distance :-




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

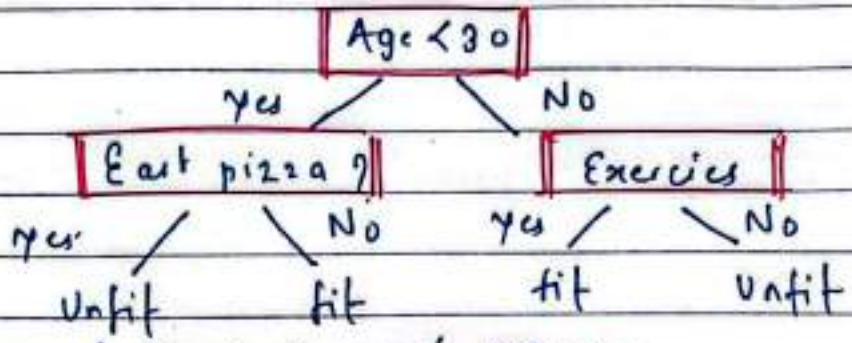
# Desion Tree.

Ahow decision Tree.

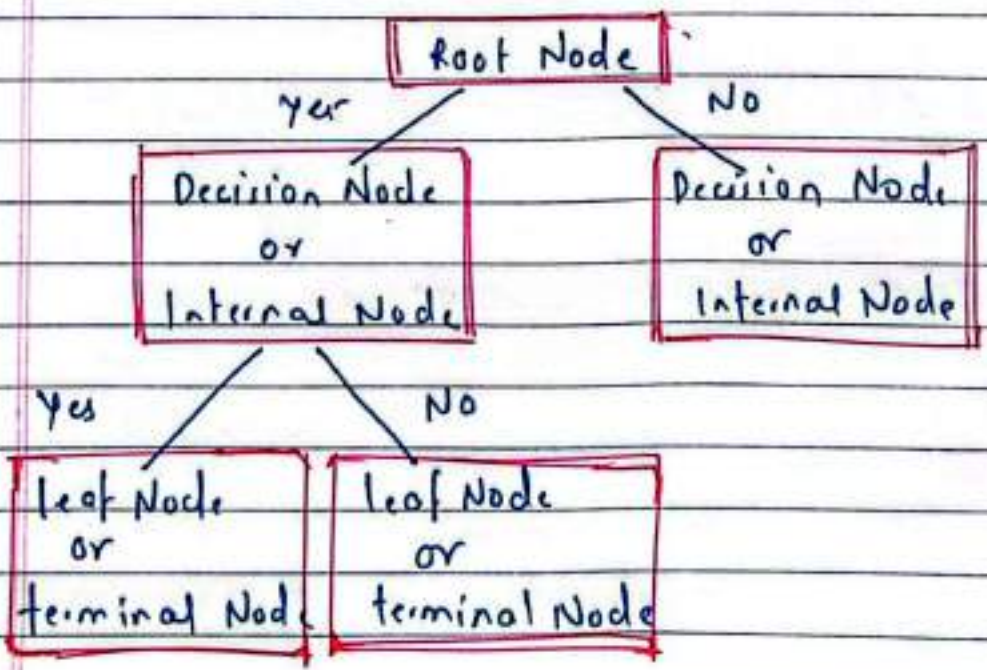
1) it is supervised ML Model
2) Used both classification & Regression.
3) Build Decision Nodes at each step.
4) Basis of Tree based model.

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.

Disdvantages:

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 :-

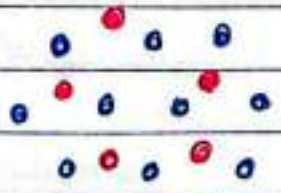
a) Entropy
b) Information Gain
c) Gini Impurity

| | |
|---|---|
| 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 mean that randomness in the system is low and thus making accurate prediction is easier.

Entropy: $\sum_{i}^{c} -p_i \log_2 p_i$ $c$ = 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 leeds 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

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

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.

Date: / /
Page

# Decision Tree for regression.

Let understand with Example

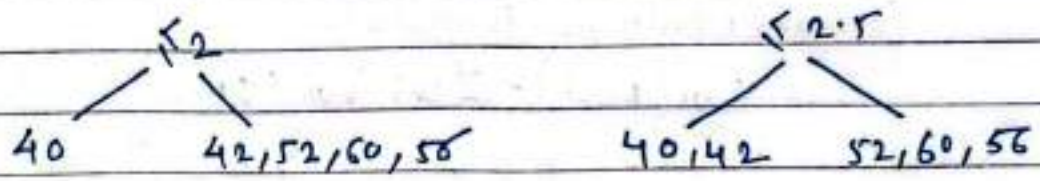
| Exp | Gap | Salary (K) |
|---|---|---|
| 2 | Yes | 40 |
| 2.5 | Yes | 42 |
| 3 | No | 52 |
| 4 | No | 60 |
| 4.5 | Yes | 56 |

$\bar{y} = 50$ ← Average.

Let take experience at root node

{Note:- Since exp is continuous data or arrange it in ascending order

- Now for comparison we will take two node example




- 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 at root.

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

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

$$= 60.8$$

Now we will calculate variance of each internal node or decision node

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

$$= 100$$

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

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

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

$$= 51$$

Variance reduction formula:-

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

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

$$= 60.8 - 20 - 40.8$$

$$= 0$$

Same we will calculate for second condition $\leq 2.5$ whoever have large variance reduction we will finalize it for splitting.

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

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

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

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

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

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

Variance reduction for next split i.e ≤ 2.5

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

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

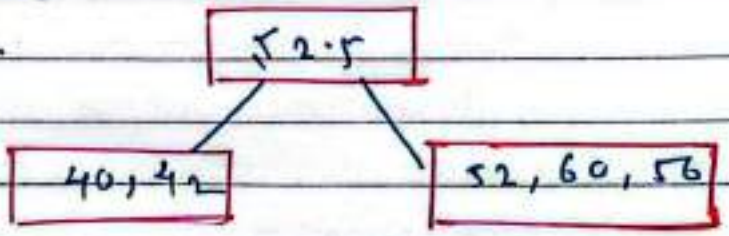
$$= 0.304$$

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

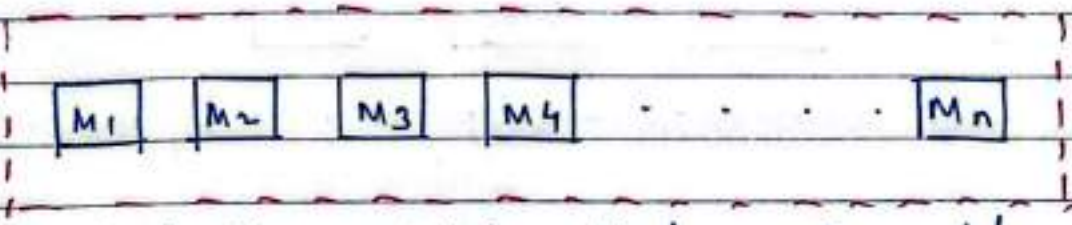
$$\frac{40+42}{2} = 41$$

and it reached to second leaf then

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

# 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.
-




- 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.

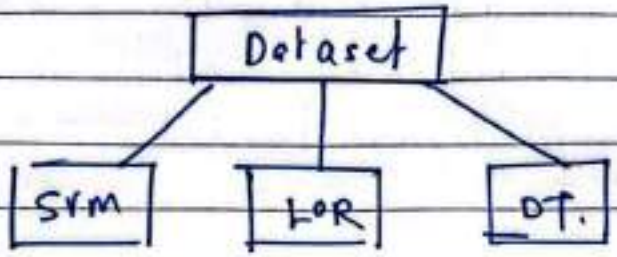
- In classification problem output is decided on majority count for the perticular class predicted by all the model.
- In regression mean or average of all the output predicted by model.

## Types of Ensemble.

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

Page

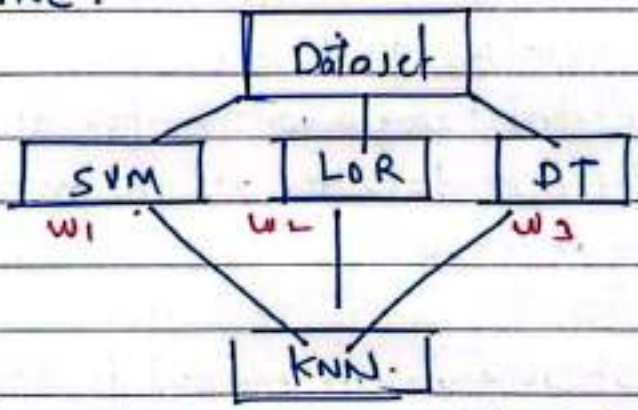
**Voting Ensemble :-** this depends on model should be different or particularly different algorithm.




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.



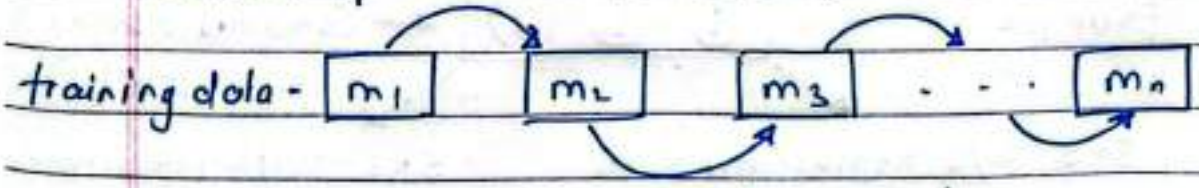

- Here we add one more additional algorithm which is trained on the output of above models or ensemble model
- this newly added algorithm gives different weightage to each model based on their perforce based on the training data.
- and after this newly added model will predict the final result.

**Bagging** :- Bootstrap Aggregation

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


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.




Benefits :- 1) Improvement in performance.
2) it help to acchieve low bias and low variance.
3) Robust to variance.

when to use :- always.

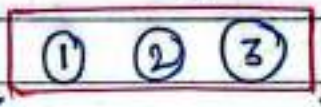
Disadvantages :- Here we have to train multiple models which increase computational complexity.

1) let understand bagging and Randomforest 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 

← Container having 3 balls.

SRS W/o Replacement

1 2
1 3
2 3

{each row/column should not repeated twice}

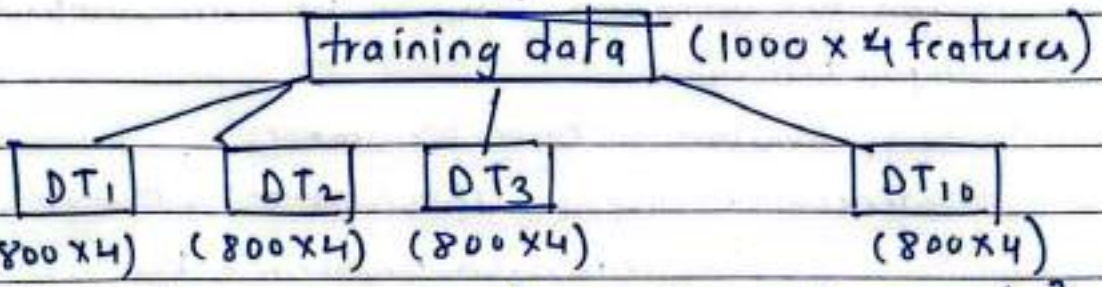
SRS with replacement

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.




{ simple random sampling with replacement }.

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

Voting: it to predict from class 1 or class 0

DT1 → 1
DT2 → 0
DT3 → 0
.
.
DT10 → 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 }

hyperparameter :- 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 complexicity 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.

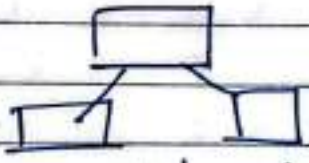
- why it is called random sampling ?

→ because everytime we select the features randomly.

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

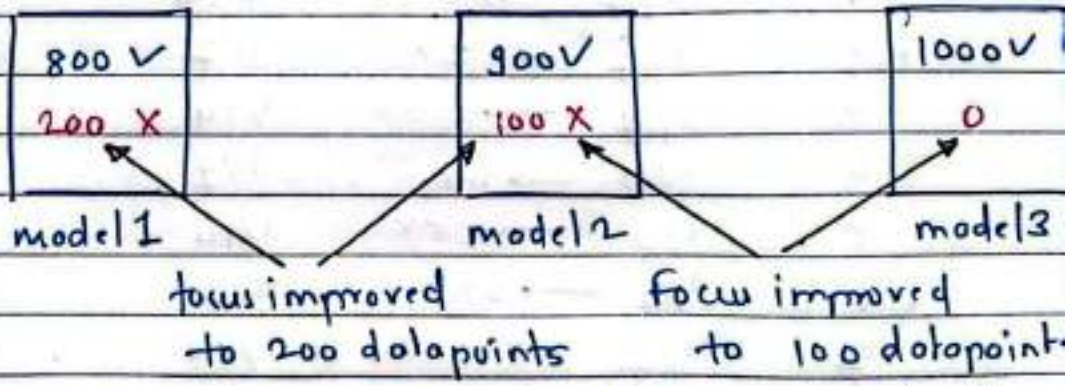
Boosting :- 1) Adaboost
2) Gradient boost
3) Xgboost.

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 | 🔵 |

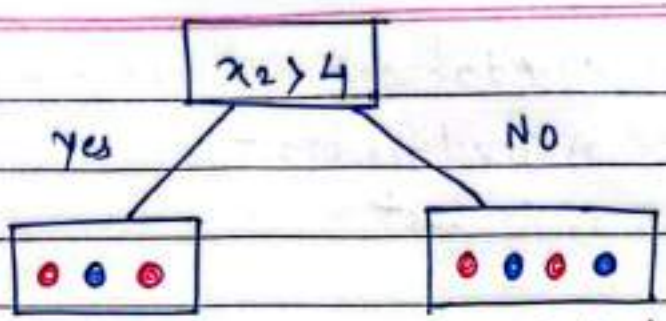
$P(n-1) = 2(7-1) = 2 \times 6 = 12$

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 | |

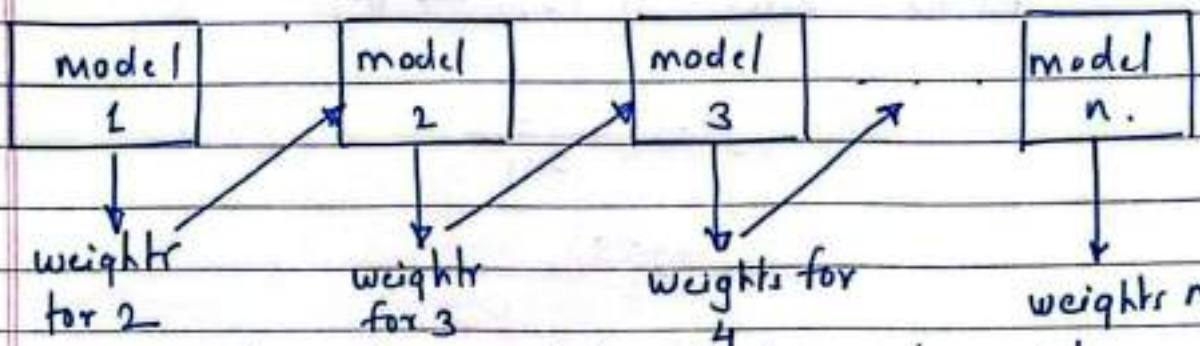
Now let understand my distribution of focus.

| | 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 there 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 come in big range bucket hence their chance of getting focus is more while sampling with replacement.

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

constant : $e^{\lambda k}$

where : $e = 2.718$

$\lambda$ = learning rate (it could be from 0 to 1)

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

for Ex error : is 10% $\rightarrow$ 0.1

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

$= 1/2 \log(9)$

$k = \log\sqrt{9}$

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

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.

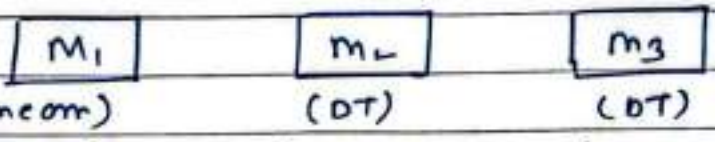
5 :- repeat step 2 and 4 number of this repetation equal to number of models.

6 : Combine models for voting.

**Gradient Boosting :-** let understand with example of regression problem which is easy to understand & 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




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 | psedo1 | pred 2 | psedo2 | 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 |

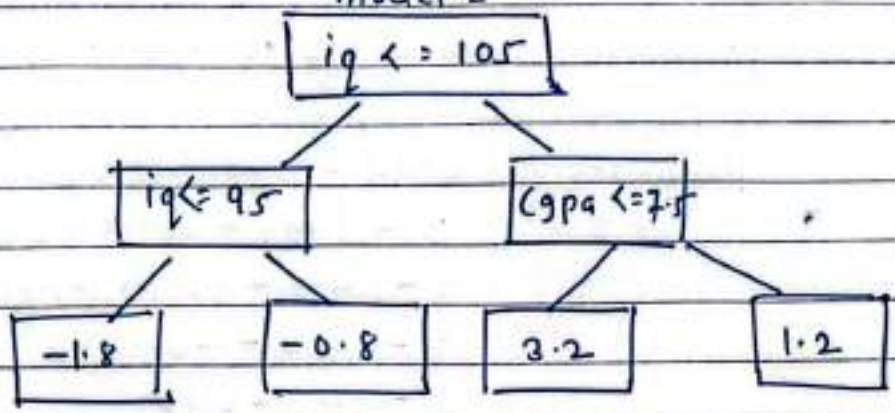
- In order to fall the error done by $M_1$ to $M_2$ we will calculate loss function of $M_1$

  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 doing.

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 : $4.8 + (0.1)(-1.8) = 4.8 + (-0.18)$

$= 4.62$

Date: / /
Page

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 pseudo residual.

pseudo-resid : actual - predicted by m2

st 1 : $3 - (m_1 + 0.1 \times m_2)$

$= 3 - (4.8 + (0.1 \times (-1.8)))$

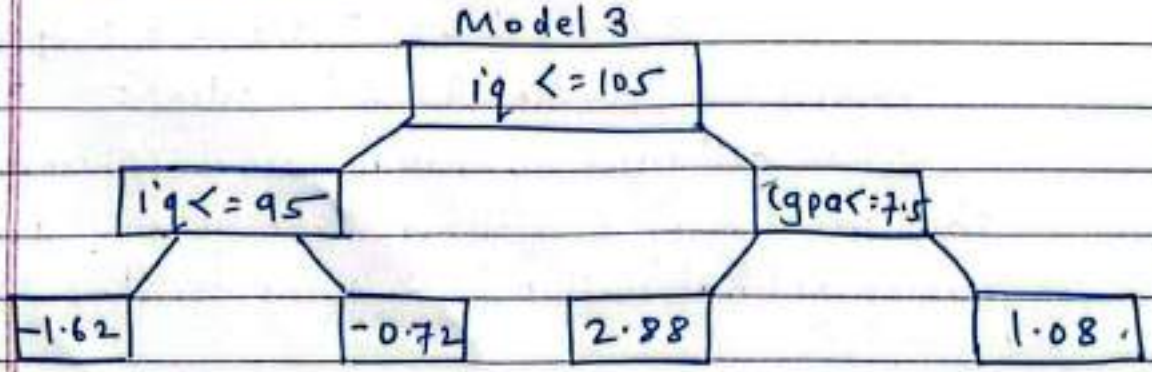
$= 3 - (4.8 + (-0.18))$

$= 3 - (4.8 - 0.18)$

$= 3 - 4.62 = -1.62$

same for ~~stu~~ student 2/3/4/5 and it will be pseudo 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 pseudo residual 2.
- model 3 combinely predict the mistakes of model 1 and model 2




- Now will 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)

$= 4.8 + 0.1(-1.8) + 0.1(-1.62)$

$= 4.8 - 0.18 - 0.162$

$= 4.458$

earlier it would be 4.62 now it is 4.458

## Difference between Adaboost Vs Gradient Boost

| Adaboost | Gradient Boost |
|---|---|
| • Here we use division 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 ot $(m_1, m_2 \cdots m_N)$ are prioritized based on their weights $w_1 m_1 + w_2 m_2 + w_3 m_3$ | Here we use learning rate concept which is same for all the models. |

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)

$= 4.8 + 0.1(-1.8) + 0.1(-1.62)$

$= 4.8 - 0.18 - 0.162$

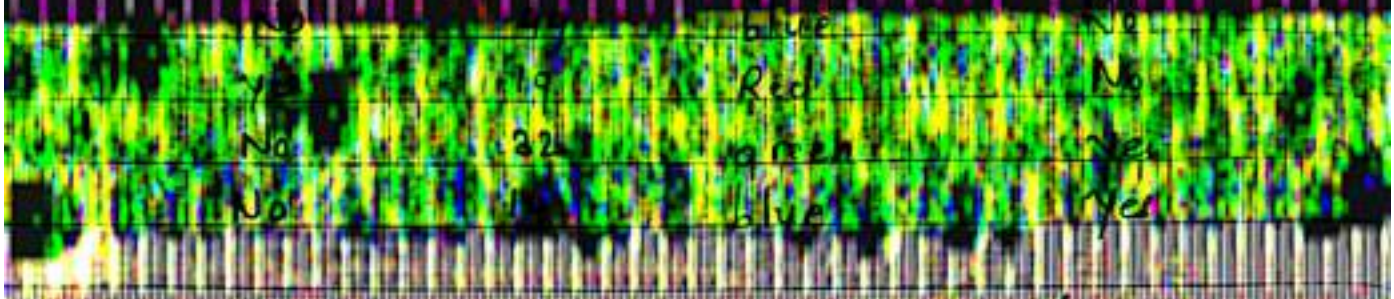
$= 4.458$

earlier it would be 4.62 now it is 4.458

## Difference between Adaboost Vs Gradient Boost

| Adaboost | Gradient Boost |
|---|---|
| • Here we use division 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 ot $(m_1, m_2 \cdots m_N)$ are prioritized based on their weights $w_1 m_1 + w_2 m_2 + w_3 m_3$ | Here we use learning rate concept which is same for all the models. |

# Gradient Boosting for classification problems



- when we use gradient Boost for classification the initial prediction for every individual is the log(odds)
- log(odds) for target : $\log\left(\frac{\text{people love the movies}}{\text{people don't love the movies}}\right)$

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

∴ this is initial prediction

- Just like with logistic Regression the correct 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 Movies $= \frac{e^{\log(odd)}}{1 + e^{\log(odds)}}$

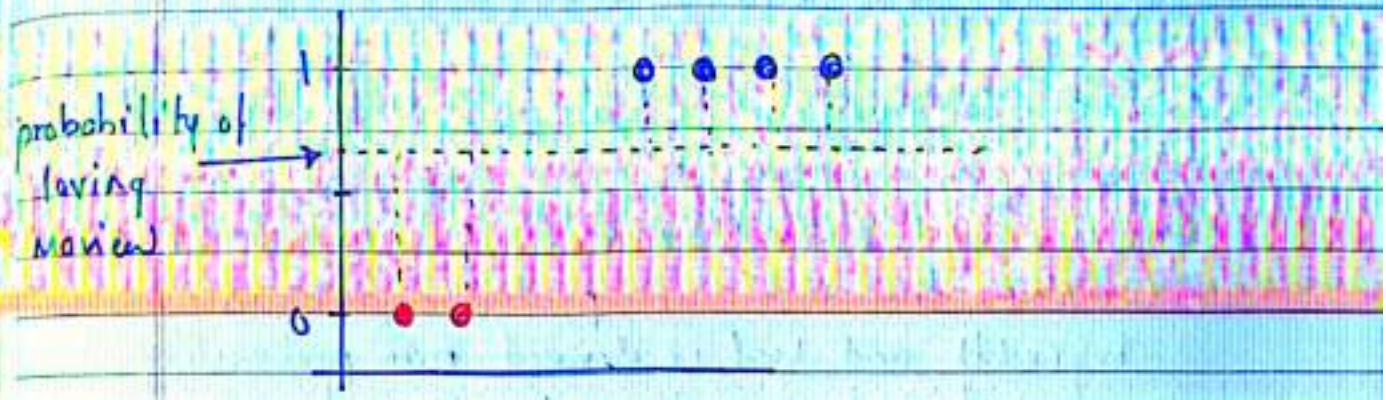
$= \frac{e^{\log(4/2)}}{1 + e^{\log(4/2)}} = 0.7$ (rounded value)

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

Date: / /
Page

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

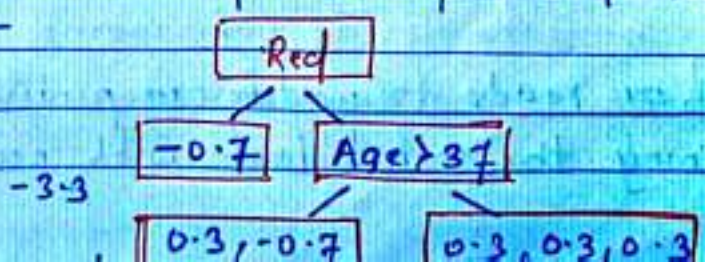
Residual = (observed − Predicted)




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

| likes popcorn | Age | favorait Color | love Movies | 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 "like popcorn and age" as input and Residual as output or target



- 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 predictions are in terms of log(odd) and leaf is derived from probability. so we can't just add them together to get new log(odds) prediction without some sort of prediction

-:- for output of leaf 1st

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

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

-:- for output of leaf 2nd

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

-:- 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.

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 | favorai 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 |

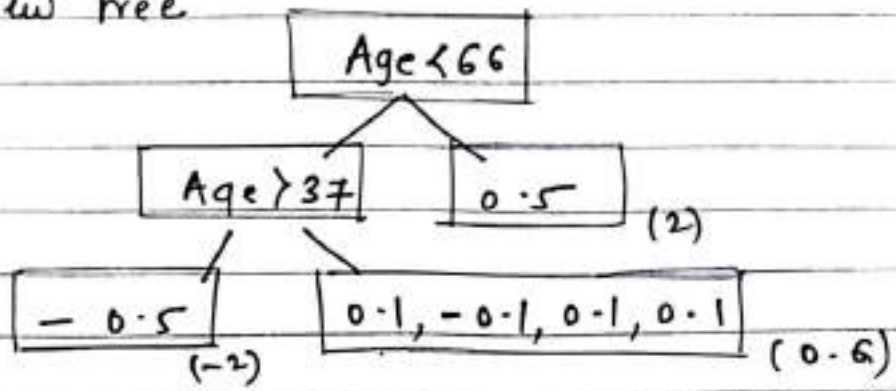
this new predicted probabilities are wonk than before & thats why we build lot of tree and not just one

· 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) : $\frac{\Sigma \text{Residual}}{\Sigma[\text{prev prob} \times (1 - \text{prev probability})]}$

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

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

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

$= 0.6$

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

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 someone who loves or not love the movie

Date: / /
Page

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

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

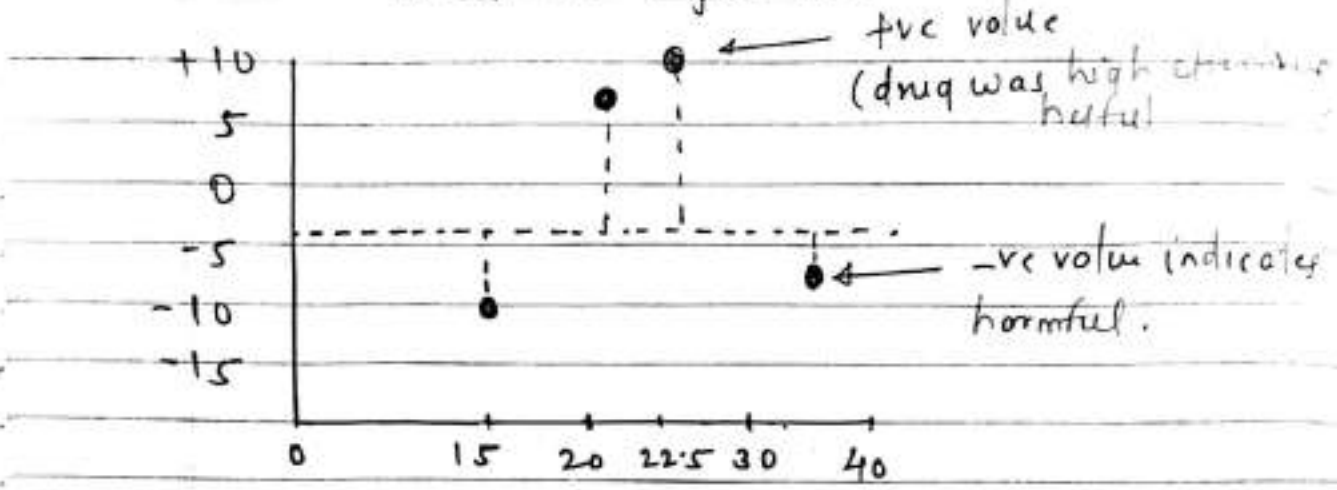
Now convert 2.3 into probability

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

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 whether it is XGboost classification or regression.

Residual = observed - predicted

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

[-10.5, 6.5, 7.5, -7.5] 1st leaf

we will calculate similarity score for residual

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

where $\lambda$ = 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

| −10.5, 6.5, 7.5, −7.5 | SS = 4 |
|---|---|

leaf ①

| −10.5 | SS = 110.25 |
|---|---|

② leaf

| 6.5, 7.5, −7.5 | SS = 14.08 |
|---|---|

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

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

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

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 + Right leaf similarity − Root similarity.

Gain : 110.25 + 14.08 − 4 = 120.33.

b] Now we will shift dosage threshold to 22.5

Dosage < 22.5

| −10.5, 6.5, 7.5, −7.5 |
|---|

| −10.5, 6.5 | SS = 8 |
|---|---|

| 7.5, −7.5 | SS = 0 |
|---|---|

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

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

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

Gain ~~Gain~~ = SS Left + SS Right - Root ss

= 8 + 0 - 4 = 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

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

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

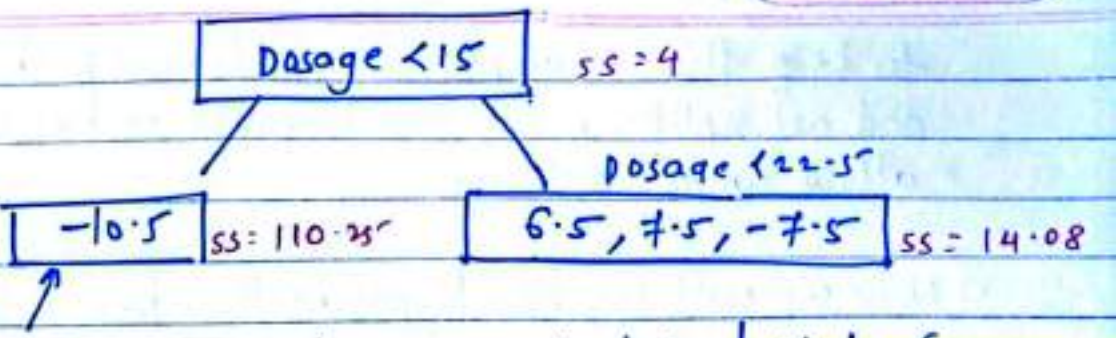
Gain = SS left + SS Right - SS Root

= 4.08 + 56.25 - 4.

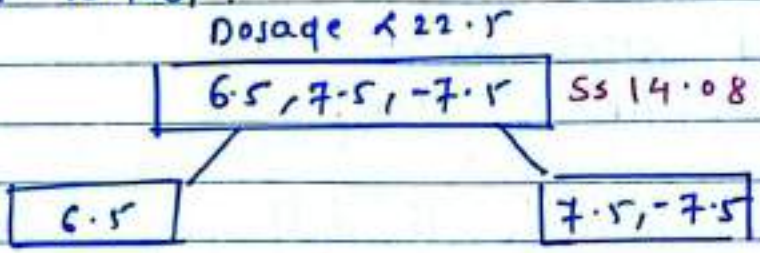
= 56.33.

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

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



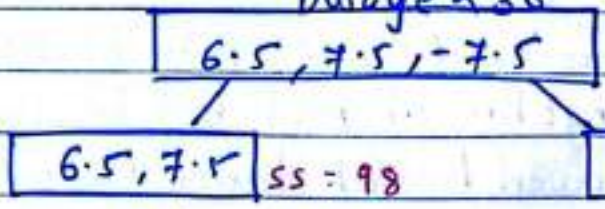

Since there is only one residual in left leaf we can't split it ahead. however we can split right leaf further.




ss left = 42.25
ss Right = 0
} See the previous formula.

Gain = ssleft + ss Right - ss Root
= 42.25 + 0 - 14.08
= 28.17

Now again will shift threshold value to 30




ss left = 98
ss Right = 56.25

Gain = 98 ssleft + ss Right - ss Root = 140.17

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

∴ so this is perfect split.

to keep this example we are limiting it to two leaves 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 $\gamma$
  here $\gamma = 130$.

- then we calculate the difference between $\gamma$ 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 - 3~~ 140.17 - 130 = 10.17
  +ve value we will not remove the branch.

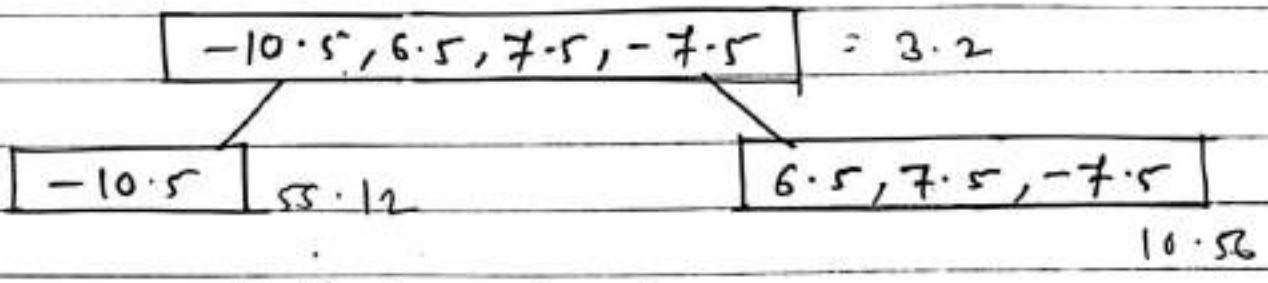
- ~~if number would be 150 (i.e $\gamma = 150$)~~ the difference of $\gamma$ and gain is negative at root but because we are not removing lowest branch hence we will not remove this as well

- if $\gamma$ would be 150 the we had to remove lowest branch as well as root and whole tree would go and it would be extreme pruning.

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

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

$\lambda$ is regularization parameter which mean it is intended to reduce the prediction sensitivity to individual observation.




again similarity score at root

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

$$= 3.2$$

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

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

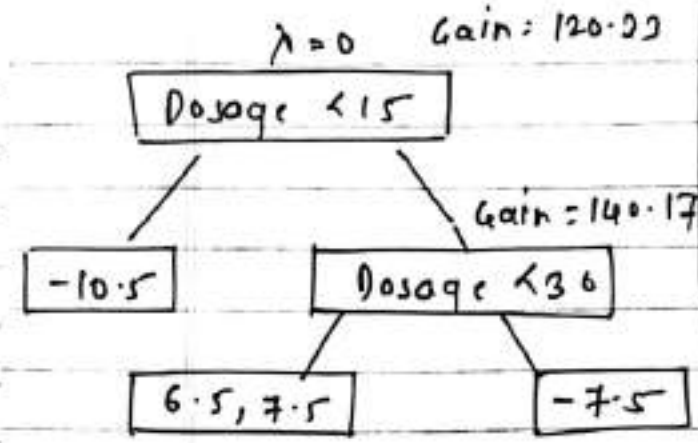
Similarity score while $\lambda = 1$ are very less than similarity score while $\lambda = 0$, the amount of decrease is inversely proportional to the number of residual in the node.

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

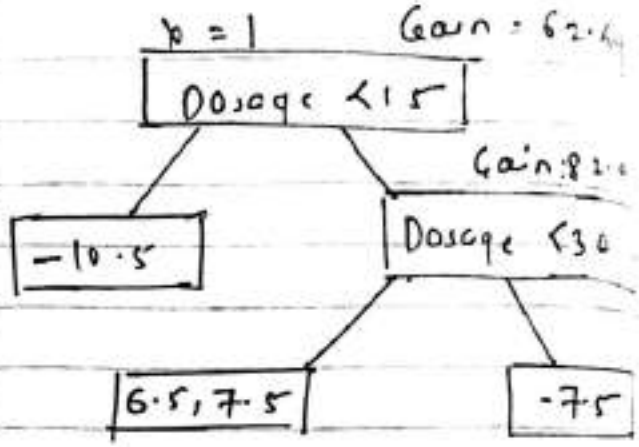
$= 55.12 + 10.56 - 3.2$

$= 62.48$

and $\lambda = 0$ it was 120.33.



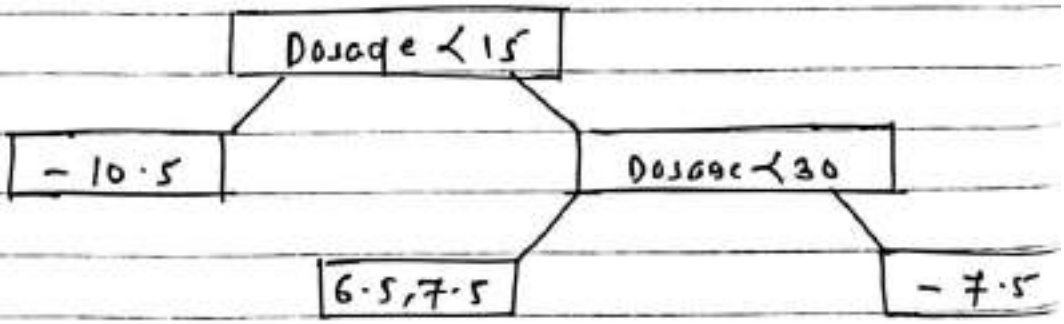

for pruning
For number $\gamma = 130$
due to the difference we didn't prune




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

Note:- setting $\gamma = 0$ doesn't turn off pruning if difference become negative it anyway will prune

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




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

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

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 $\lambda$ (lambda) regularization parameter will reduce the prediction sensitivity to this individual observation

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

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

output of leaf ⑧ : -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

∴ initial leaf + learning rate × 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.