Dataset

Train

Test

1 2 3 4 5 6 7

k = 5

k Fold

1 2 3	4 5 6	4 5	7	1 2

odds: The odds that India wins the match against pakistan is 4:1 with

India → 4 matches
Pakistan → 1 match

Total number of matches = 5

$P(\text{India})(\text{wins}) = 4/5 \longrightarrow P(\text{india})(\text{lose}) = 1/5$

$P(\text{Pak})(\text{wins}) = 1/5$

$P(\text{India})(\text{win}) = 4/5$

$P(\text{India})(\text{Lose}) = 1/5$

$$\text{odds} = \left(\frac{P}{1-P}\right) \rightarrow \text{eq (1)}$$

$$\frac{P(\text{India})(\text{win})}{P(\text{India})(\text{Lose})} = \frac{(4/5)}{(1/5)} = (4/1)$$

odds → 4 : 1 = (4/1)

$$odds = \left(\frac{p}{1-p}\right) \rightarrow$$

$$(odds) = \left(\frac{p}{1-p}\right)$$

$$\log(odds) = \log\left(\frac{p}{1-p}\right) \rightarrow \text{eq (2)}$$

Log of odds = Logit function

(10) $\log(odds) = \log\left(\frac{p}{1-p}\right)$

$p \rightarrow$ Range $\rightarrow 0 \longleftrightarrow 1$

$$\log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p) \rightarrow$$

$\log(a/b)$
$\Downarrow$
$\log(a) - \log(b)$

$= \log(0) - \log(1-0) = \log(0) - \log(1)$

$= (-\infty) - 0 = -\infty$

when
$p = 0$
$\log(0)$
$= -\infty$
$\log(1)$
$= 0$

(11) $\log(\text{odds}) = \log(P/1-P)$

$P \rightarrow$ Range $\rightarrow 0 \rightarrow 1$

$\log(a/b) = \log(a) - \log(b)$

$\log(P/1-P) = \log(P) - \log(1-P)$

$= \log(1) - \log(1-1) = \log(1) - \log(0)$

$= 0 - (-\infty) = (+\infty)$

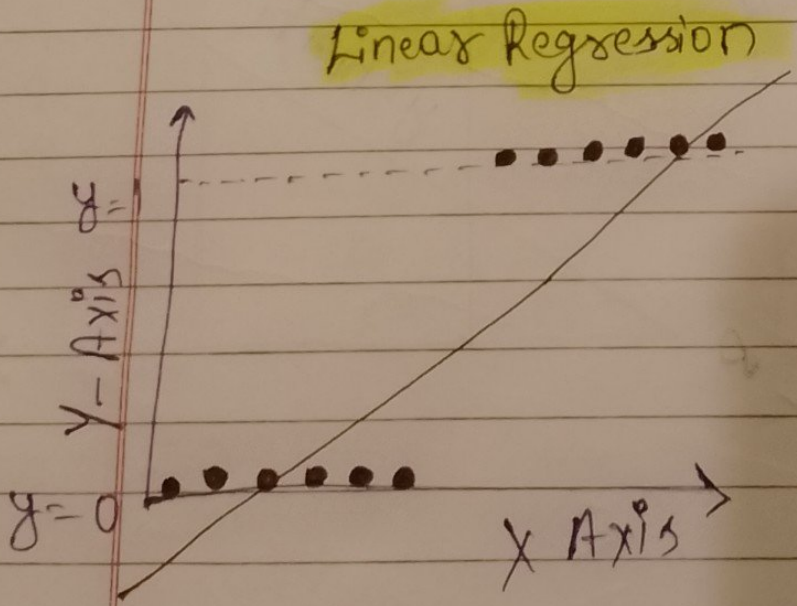
when $P=1$, $\log(0) = -\infty$, $\log(1) = 0$

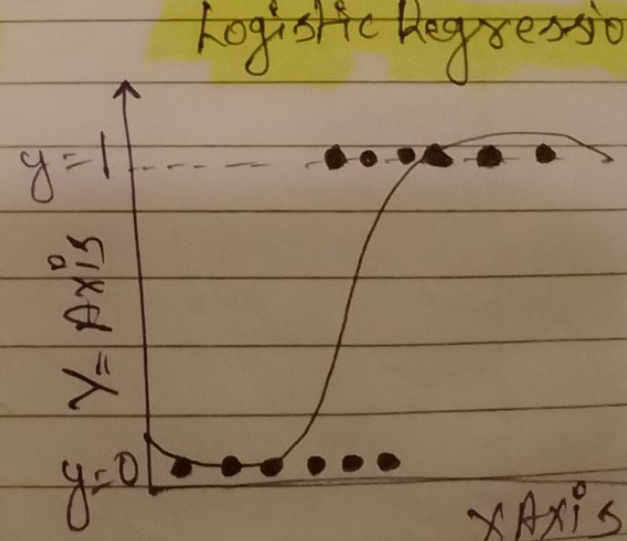
(12) $P=0 \longrightarrow \log(P/1-P) \rightarrow -\infty$

$P=1 \longrightarrow \log(P/1-P) \rightarrow +\infty$

~~$y = b_0 + b_1 x_1 \rightarrow$ Linear Regression~~

(13) Logistic Regression

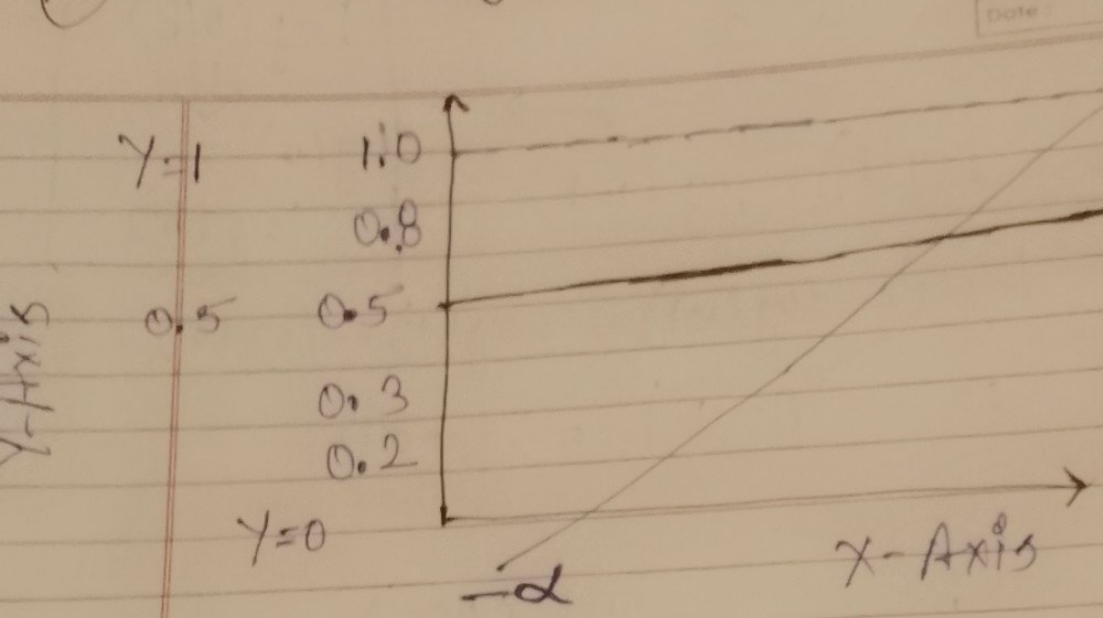




Logistic Model

$$P = \frac{1}{1+e^{-(b_0+b_1 x)}}$$

(14) Logistic Regression



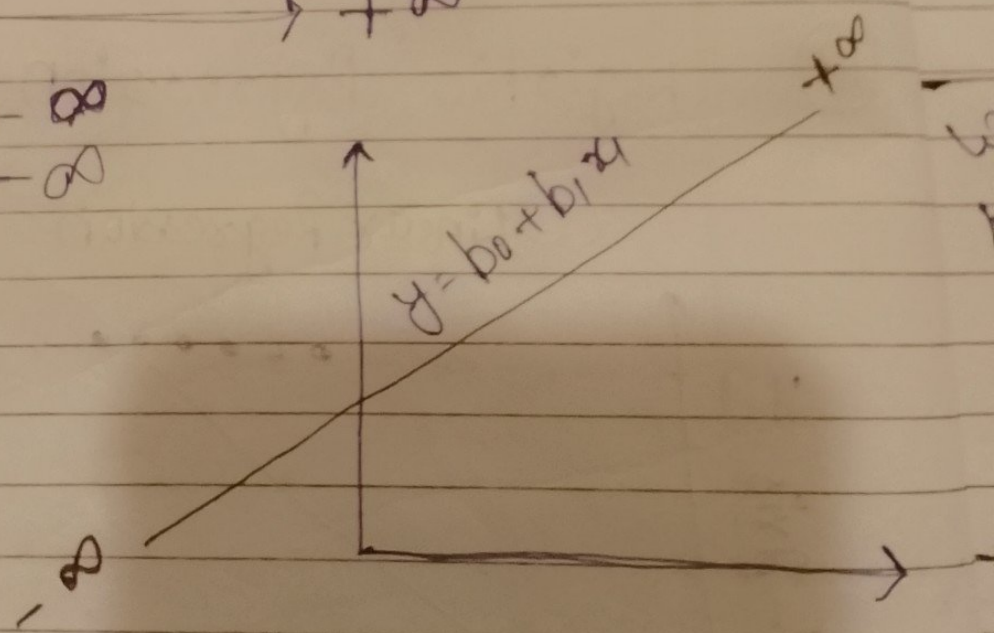
(15) $y = b_0 + b_1 x_1$

$x_1 \rightarrow -\infty$

$x_1 \rightarrow +\infty$

$y \rightarrow +\infty$

$-\infty$



(16)

Linear Reg → Magic box Transformation → classification

$y = b_0 + b_1 x_1$

Mathematical equation → $\left(\frac{1}{1+e^{-z}}\right)$ → going to capture Linear Regression

This is exponential
Value = $e = 2.713$.

(17)

$$\frac{1}{1+e^{-(b_0+b_1x_1)}}$$

$z = b_0 + b_1 x_1$

x_1 : $-\infty \rightarrow +\infty$

$y = b_0 + b_1 x_1$: $-\infty \rightarrow +\infty$

when $b_0 + b_1 x_1 = y \rightarrow (-\infty)$

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

$= \frac{1}{\text{Large number}}$

$= 0$

e^{∞} = large number

$(2.713)^{\infty}$ = large number

1 + large no. = large number

(18)

$$\frac{1}{1+e^{-(b_0+b_1x_1)}}$$

when $b_0+b_1x_1 = y$

$b_0+b_1x_1 \Rightarrow (+\infty)$

$$\frac{1}{1+e^{-(+\infty)}} = \frac{1}{1+e^{-\infty}} = \frac{1}{1+\frac{1}{e^{\infty}}} = \frac{1}{1+\frac{1}{\text{largeno}}}$$

$$\frac{1}{1+0} = 1$$

x_1 → root → ∞

$y = b_0 + b_1x_1$

→ $-\infty$

→ $+\infty$

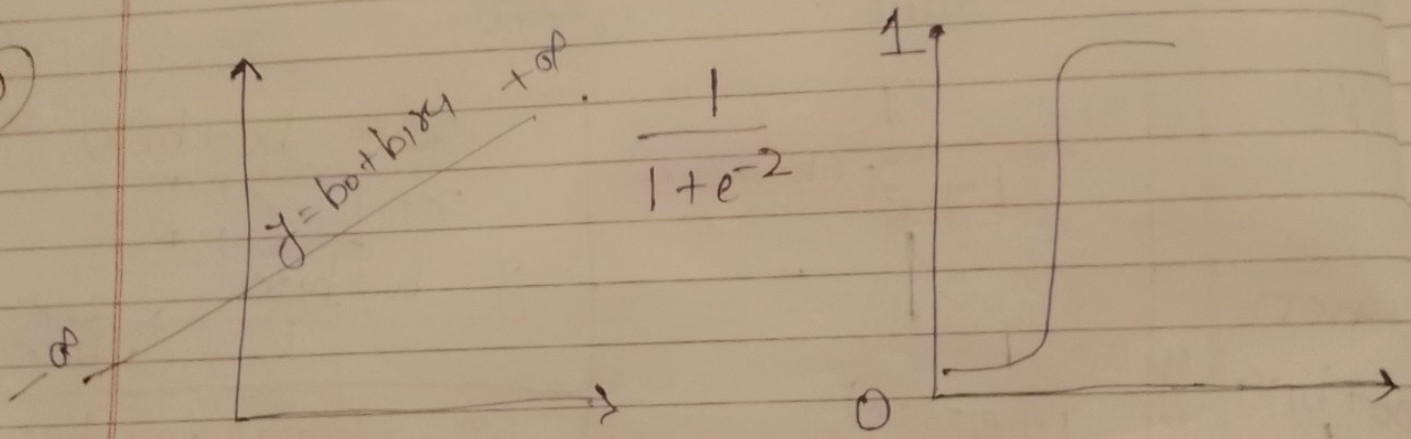
e^{∞} = large number

$(2.713)^{\infty}$ = large number

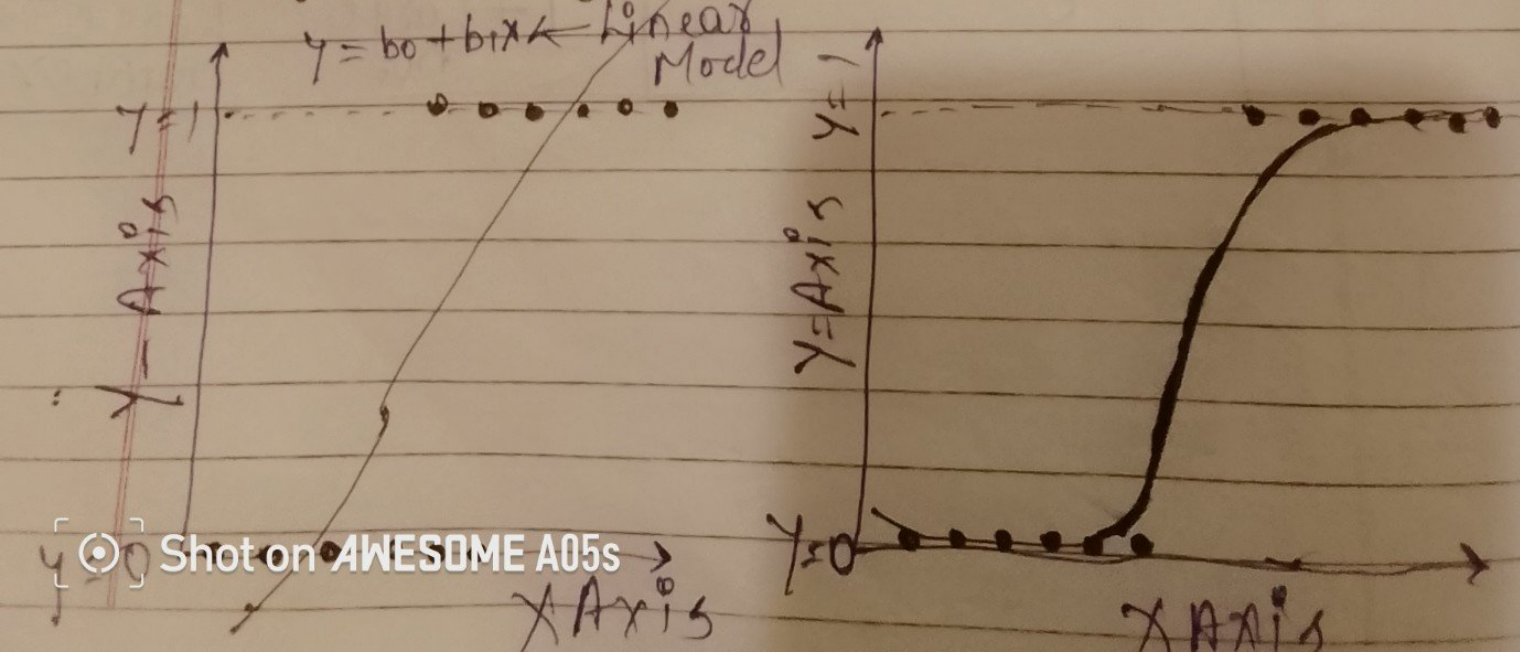
$1 + \text{large no.} = \text{large number}$

(21)

(19)



(20) Logistic Regression



Logistic Model

$$P = \frac{1}{1 + e^{-(b_0 + b_1 x)}}$$

(21) $P = 0 \longrightarrow \log(P/1-P) \longrightarrow -\infty \rightarrow b_0 + b_1 x_1 (x_1) \rightarrow -\infty$

$P = 1 \longrightarrow \log(P/1-P) \longrightarrow +\infty \hookrightarrow b_0 + b_1 x_1 (x_1) \rightarrow +\infty$

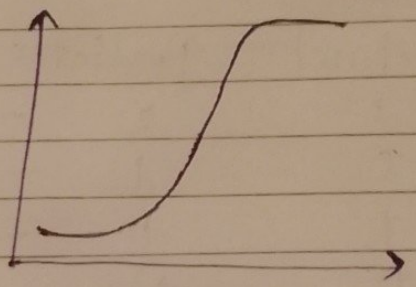
$$\log(P/1-P) = b_0 + b_1 x_1$$

→ Logit function

↓

Logistic Regression

(22) $\left(\frac{1}{1+e^{-z}}\right)$ → Sigmoid equation



→ Sigmoid curve

(23) $y = 1$ $y = b_0 + b_1 x$ ← Linear Model

Y Axis

$y = 0$ X Axis

$y = 1$

0.95

0.85

0.40

0.30

$y = 0$

Logis

$P = \frac{1}{1+e^{-(}}$

X Axi

(24) $P = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(b_0+b_1x_1)}}$

$x_1 = -\infty \Rightarrow P = 0$

$x_1 = 1000$

$x_1 = +\infty \Rightarrow P = 1$

$x_1 = 1$ lacs $\Rightarrow P = 0.85$

(25) Regression → Residual = A − P

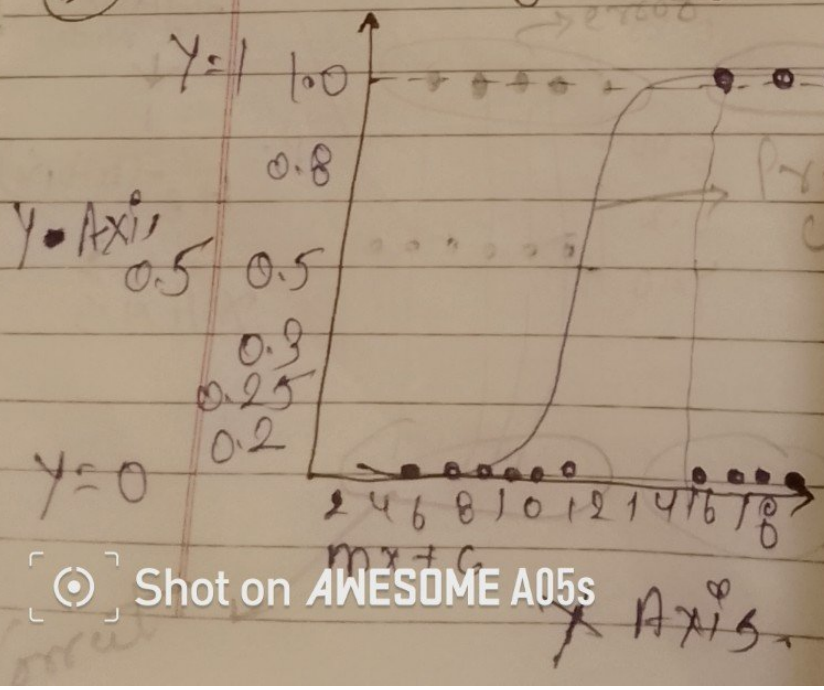
classification → Correctly classified
→ Incorrectly classified

0 → Non Diabetic
1 → Diabetic

Actual	Predict	
1	0	→ Mis Classified
0	1	→ Mis Classified
1	1	→ Correctly classified
0	0	→ Correctly cl

(26) Logistic Regression

Blue : class 0
Red : class 1



Logistic Model

$$P = \frac{1}{1+e^{-(b_0+b_1x)}}$$

(27)

(28)

Bernoulli class distribution

x	0	1
P(x)	1-P	P

$= P^y * (1-P)^{1-y}$

when y=0

$= P^0 * (1-P)^{1-0}$

$= 1 * (1-P)$

P = Probability of success

1-P = Probability of failure

y → class

y = 0 class

y = 1 class

(3) Bernoulli distribution

y	0	1
P(y)	1-P	P

$= P^y * (1-P)^{1-y}$

when y=1

$= P^1 * (1-P)^{1-1}$

$P^1 * (1-P)^0$

$= P \times 1 = P$

P = Probability of success

1-P = Prob. of failure

y → class

y = 0 class

y = 1 class

(3c) Bernoulli distribution

y	0	1
P(y)	1-P	P

$P^y * (1-P)^y$

when y=0

$= P^0 * (1-P)^{1-0}$

$P^0 * (1-P)^1$

$1 * (1-P) = 1-P$

P = Probability of success

1-P = Prob. of failu

y → class

y = 0 class

y = 1 clas

(36) when $y = 1$ $P^y \cdot (1-P)^{1-y}$ ⟷ P

y	0	1
P(y)	1-P	P

(37) $f(x) = (P^y)(1-P)^{1-y}$

Likelihood comes because Bernoulli theorem has two outcomes and in classification also we have two outcomes

$f(x) \rightarrow P^y \times (1-P)^{1-y}$

↳ Probability of success × Probability of failure

Likelihood is defined as the Probability of success × Probability of failure

$L = (P^y)(1-P)^{1-y}$

(38) $L = (P^y)(1-P)^{1-y}$

$\log(L) = \log[(P^y) \cdot (1-P)^{1-y}]$

$\log(L) = \log(P^y) + \log(1-P)^{1-y}$

$\log(L) = y\log(P) + (1-y) \cdot \log(1-P)$

⇓

Log Likelihood

$\log(a \cdot b)$ ⇓ $\log a + \log b$

$\log(a^b)$ ⇓ $b \log a$

(39)

(40)

when y model any P

Int Mo

(39)

Null Model	Fitted Model	Full Model	Saturated Model
When you build model without any Predictors	If you build Model with atleast one Predictor	If you build Model with all predictors	If you build model which Predictors everything right (Perfect Model)
Intercept Model			

(40) log(L) ⟶ Saturated Model

⇓

logL = 0

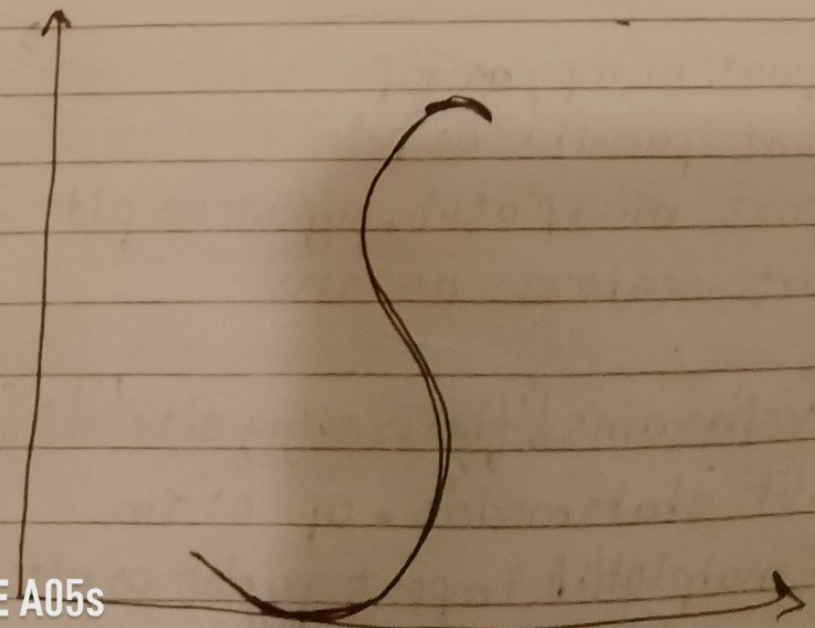
(-ve) ← saturated

null model (logL) - 300 ⟵ 0 logL

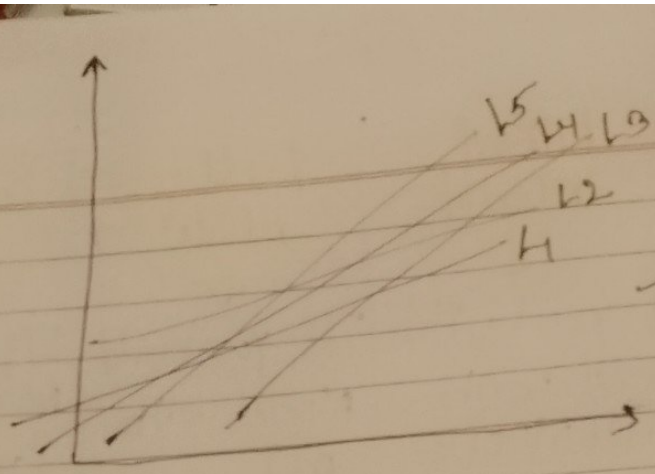
Good model should have log likelihood closest to 0

far from null model

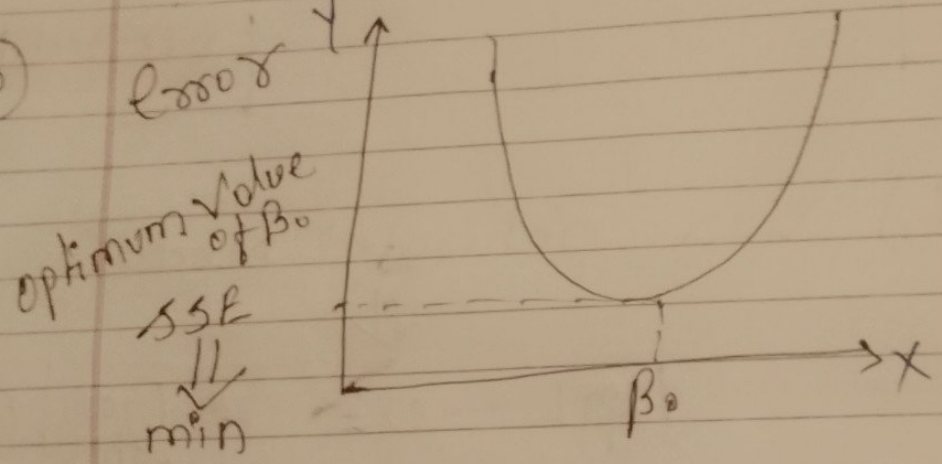
(41)



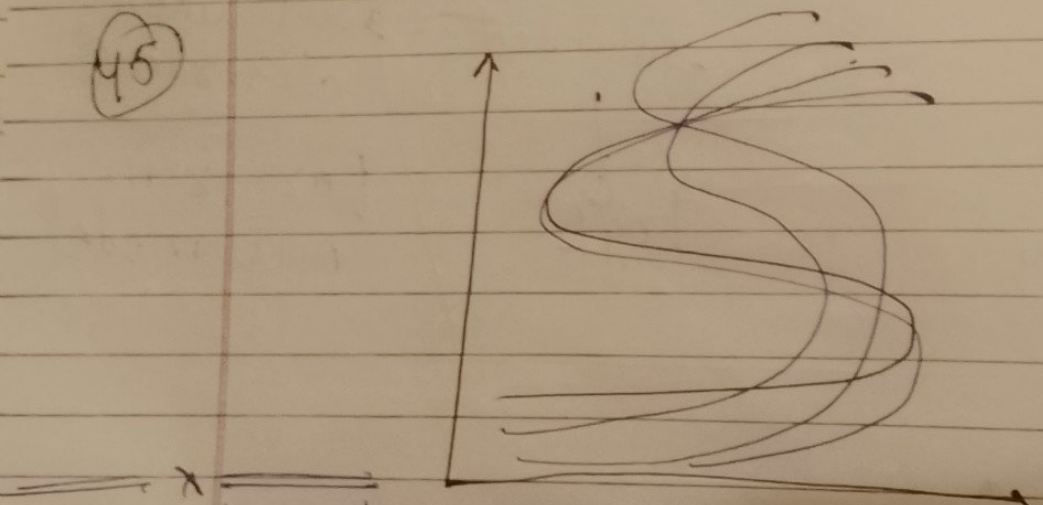
42



43



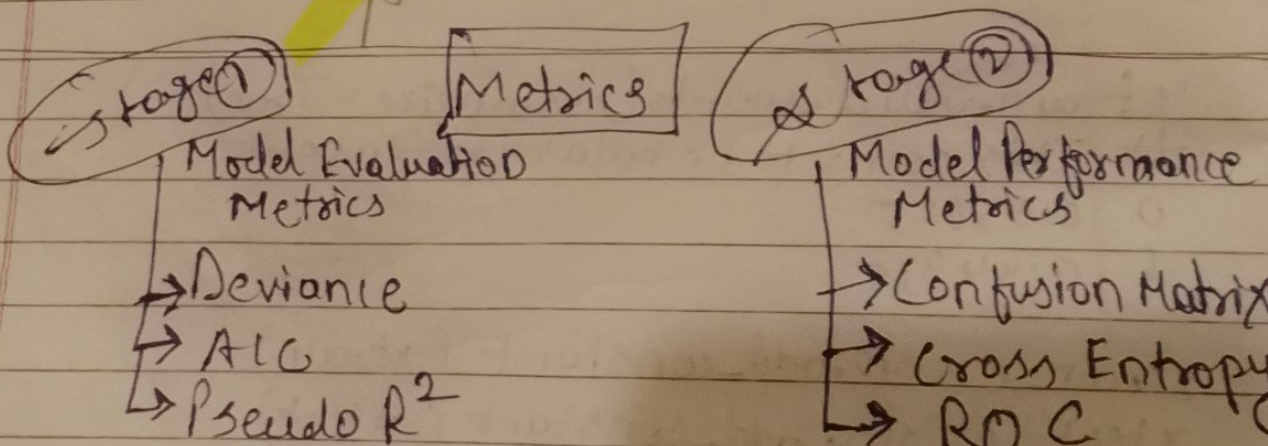
45



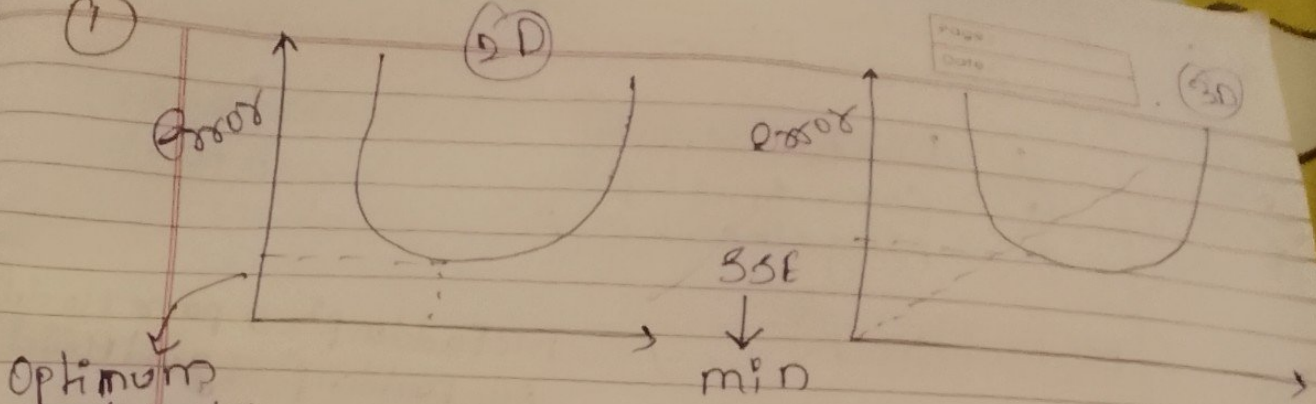

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

plt.rcParams['figure.figsize'] = [15,8]
import statsmodels.api as sm
from matplotlib import pyplot as plt
```

print(logreg.summ
optimization terminated
current function value: 0.
Iterations 8

Linear regression based on variation
Logistic regression based on probability

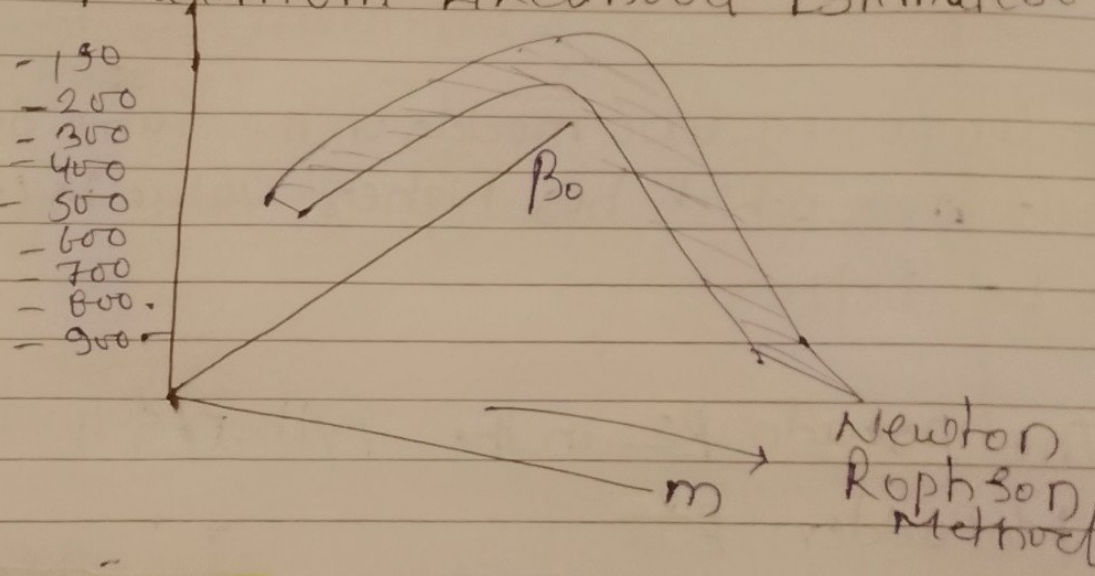


(1)



Optimum Value of B_0 when SSE = min

(2) Maximum Likelihood Estimates



(3) Metrics

Stage (1)

This tells you about how good the model was built

Model Evaluation Metrics

- Deviance ✓
- AIC ✓
- Pseudo R2 ✓

Stage (2)

How good the predict were don

Model Performance Metrics

- Confusion Matrix ✓
- Cross Entropy ✓
- RUC

④ Pseudo R^2

This called as R^2 because it is based on the Probabilities

The Pseudo R^2 are

① Mc Fadden R^2
② Cox-Snell R^2
③ Nagelkerke R^2

In linear regression R^2 based on the variations

range 0 to 1

⑤ Mc Fadden R^2

This defined as

$$R^2_{McFadden} = 1 - \frac{\ln \text{likelihood of full Model}}{\ln \text{likelihood of null Model}} \quad \frac{\text{Num}}{\text{Den}} \approx 0$$

Mc Fadden $R^2 = 0$ poor Model
Mc Fadden $R^2 = 1$ good Model

If comparing two models on the same data, the model which has higher value is considered to be better

The pseudo R2 in the python output is the Mcfadden R^2

This is done so that the values are bounded.

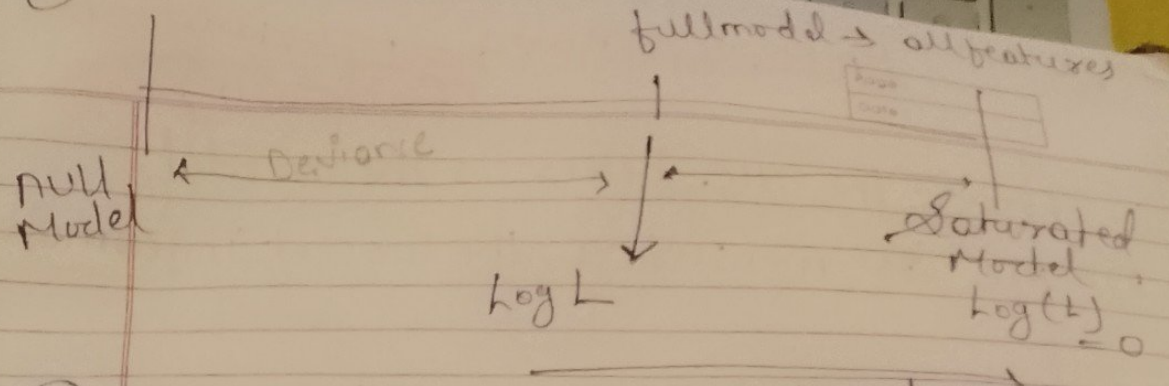
Pseudo $R^2 = 0$ (poor model)

Pseudo $R^2 = 1$ (overfit model)

Null Model

⑥ ⑦ ⑧

⑥



⑦ closer to zero

⑧ $D = -2\ln\left[\frac{\text{Likelihood of fitted Model}}{\text{Likelihood of Saturated model}}\right]$

$D = -2\log\left[\frac{\text{Likelihood of fitted model}}{\text{Likelihood of Saturated model}}\right]$

$\log(1) = 0$ ← $\frac{\text{Likelihood of fitted model}}{\text{Likelihood of saturated model}} = 1$

fitted model — Saturated Model

① Deviance is a measure

② It measures the difference between the fitted model and Saturated model.

③ Deviance should tends towards zero

⑨

① If Deviance is low it means that the fitted model is closer to the Saturated model

② If the deviance is high it means the fitted model is far off from the Saturated model

(10)

AIC (Akaike Information Criteria)

(1) Lower the AIC better is the model

(2) AIC is measuring the loss of information

(11) The Akaike Information Criteria (AIC) is a relative measure of model evaluation for a given dataset

It is given by

L : log likelihood
k : parameters to be estimated

The AIC gives a trade off between the model accuracy and model complexity
i.e it prevents us from overfitting

$$AIC = -2 \log L + 2k$$

k is the no. of Parameters.
logL → Log likelihood of the model

(12)

(13)

False
False
True
True

(12) AIC $= -2\log L + 2k$

lower the AIC better is the model

→ AIC is positive value

→ $AIC_{M_1} = 1000$, $AIC_{M_2} = 2000$ — M_1 is better than M_2

→ AIC is Relative Measure

(13) Errors (Reg) A − P

(Class) → Misclassified

	Actual	Predicted	
False Positive (FP)	0	1	Misclassified
False Negative (FN)	1	0	
True Negative (TN)	0	0	Correctly classified
True Positive (TP)	1	1	

(14) Errors (Reg) A − P

(Class) Misclassified

	Actual	Model Predicted
FP	0	1
FN	1	0
TN	0	0
TP	1	1

0 → Negative

1 → Positive

Mismatch → false

Correctly → True

Precision − out of TP + FP, how many are actually are true

Recall − , how many are actually

(24)

Predict

Actual	0(N)	1(P)
0	TN	FP
1	FN	TP

Recall (Positive) = $\frac{TP}{TP+FN}$

Recall (Negative) = $\frac{TN}{TN+FP}$

(25)

Predictive.

Actual	0(N)	1(P)
0	TN	FP
1	FN	TP

Precision (positive) = $\frac{TP}{TP+FP} = \frac{1}{1+1} = \frac{1}{2} = 0.5$

Precision (Negative) = $\frac{TN}{TN+FN}$

Precision = Precision is the ratio of correctly predicted positive class to the total positive. It is defined as (TP/TP+FP)

Recall = is also a ~~Recall~~ Sensitivity and True Positive rate. It is the ratio of correctly predicted **positive observations to all the observations (TP/TP+FN)

Note ⇒ Precision and Recall are inversely associated with each other. It means that if the precision increases the recall will go down and vice-versa.

A high recall indicates that it has a very low False negative where as high precision indicates very low to false positive.

(26)

$$F1\ score_{(Pos)} = 2 \times \frac{Recall_{(Pos)} * Precision_{(Pos)}}{Recall_{(Pos)} + Precision_{(Pos)}}$$

$$F2\ score_{(Neg)} = 2 \times \frac{Recall_{(Neg)} * Precision_{(Neg)}}{Recall_{(Neg)} + Precision_{(Neg)}}$$

F1 score is harmonic mean of Precision and Recall.

(27)

Actual \ Prediction	0(N)	1(P)
0	TN	FP
1	FN	TP

$$Precision_{(Pos)} = \frac{TP}{TP+FP} \qquad Precision_{(Neg)} = \frac{TN}{TN+FN}$$

$$F1score(P) = 2 \times \frac{R \times P}{R+P} \qquad F1score(N) = 2 \times \frac{R \times P}{R+P}$$

$$Recall_{(Pos)} = \frac{TP}{TP+FN} \qquad Recall_{(Neg)} = \frac{TN}{TN+FP}$$

(28)

COEFFICIENTS OF LOGISTIC REGRESSION

Interpretation

$$\log(odds) = b_0 + b_1 x_1$$

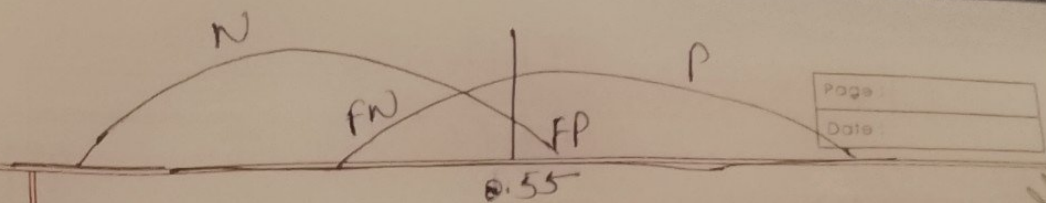
$$y = b_0 + b_1 x_1$$

1 unit of change in x, brings b_1 units of change in y

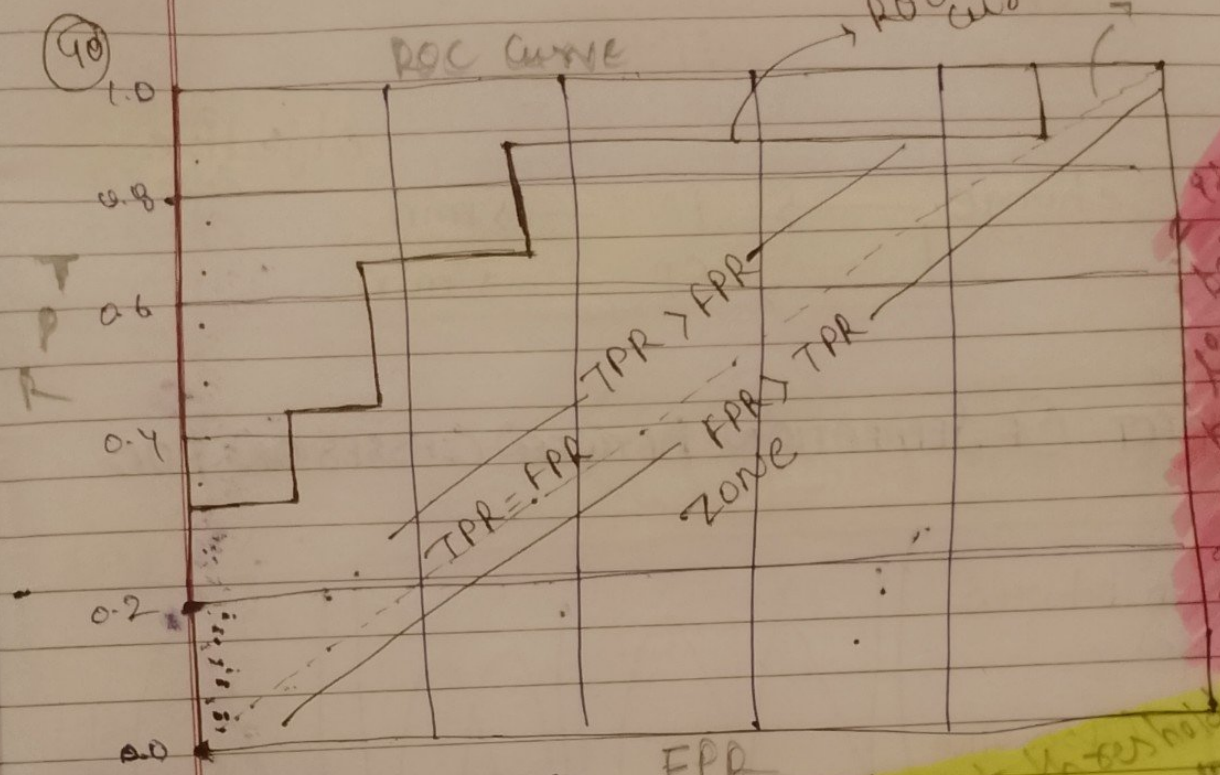
1 unit of change in → Brings b_1 units of change in log(odds)

(29) first

(30)



(90)

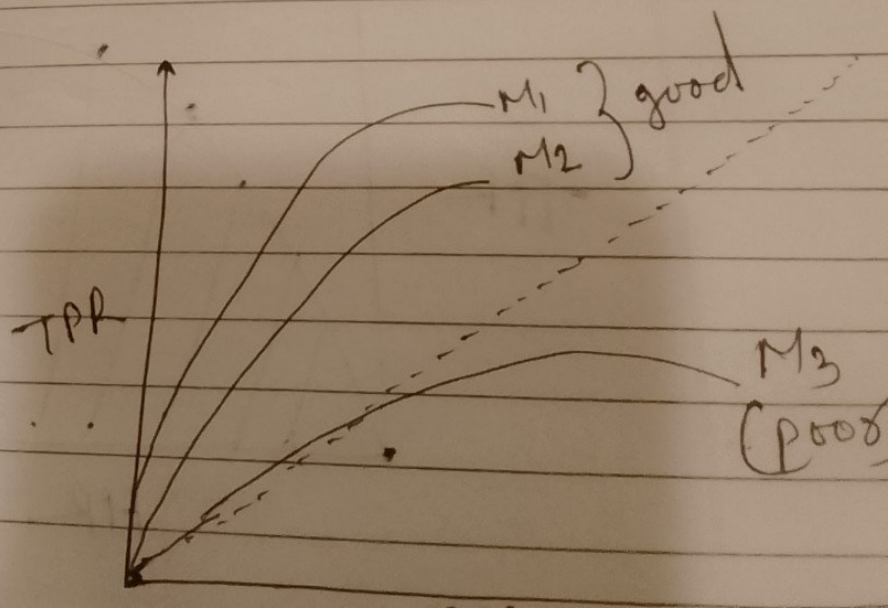


If ROC curve is below random line then model is bad and above random line then model is good

Q. Do you plot Y-threshold on ROC curve ans is NO we plot ROC curve to TPR & FPR

y TPR	x FPR	Threshold
0.33	0	0.78
0.33	0.09	0.73
0.44	0.091	0.62
0.44	0.18	0.61
0.66	0.18	0.55
0.67	0.36	0.46
0.88	0.36	0.35
0.89	0.91	0.159
1	0.91	0.157

(91)



42

$$TPR = \frac{TP}{TP+FN}$$

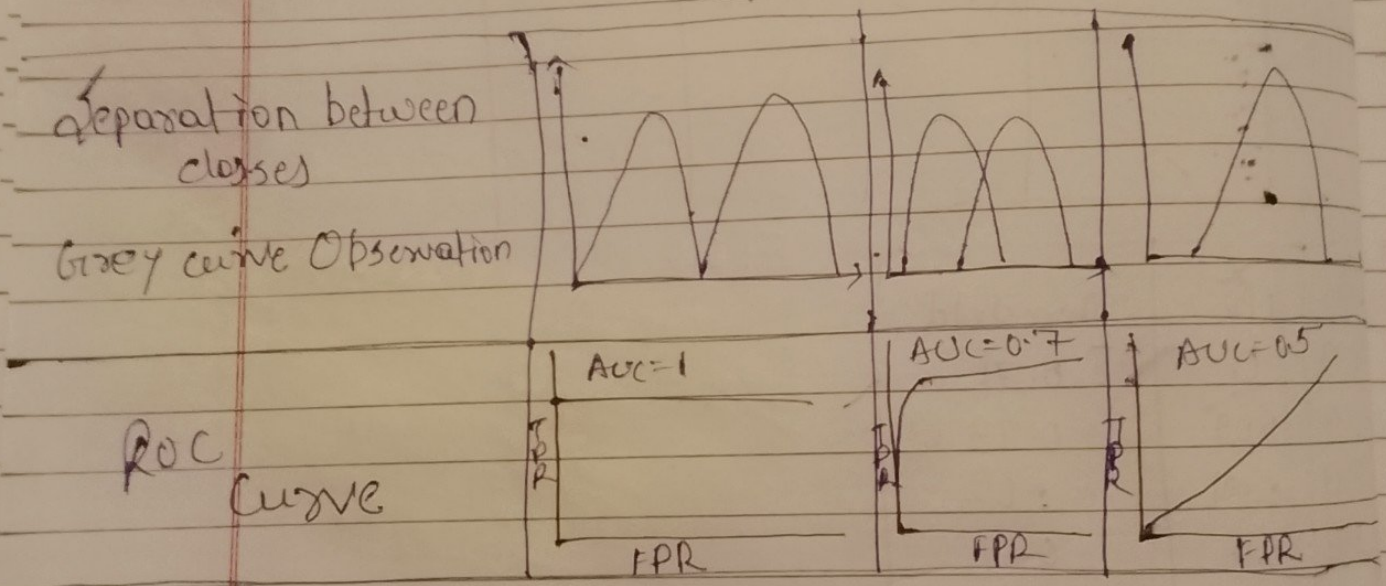
$$FPR = \frac{FP}{FP+TN}$$

objective

change → FN → min

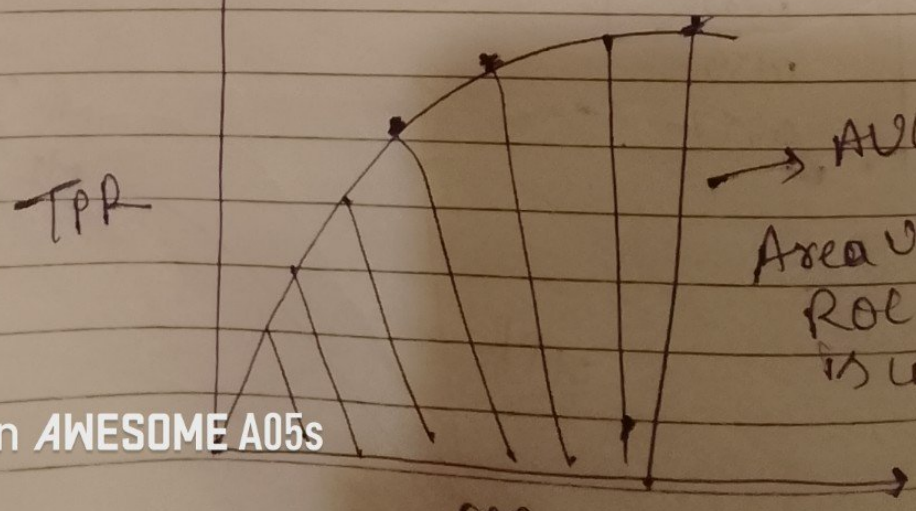
→ FP → min

43 EFFECT OF SEPARATION BETWEEN CLASSES ON ROC



44

AUC = [Area Under Curve]



→ AUC

Area under the ROC curve is called AUC

45

Kappa
<0
0-0.2
0.2-0.4
0.4-0.6
0.6-0.8
0.8-

46

R1
R2
R3
R4
R5
⋮
R100

47

45)

kappa	Interpretation
<0	No Consistency
0-0.2	Slight-Consistency
0.2-0.4	Fair Consistency
0.4-0.6	Moderate Consistency
0.6-0.8	Substantial Consistency
0.8-1	Almost perfect Consistency

Reliability → Consistency of performance which is measured as when — kappa — score

≤ 0.4] poor consistency

> 0.4] Good Consistency

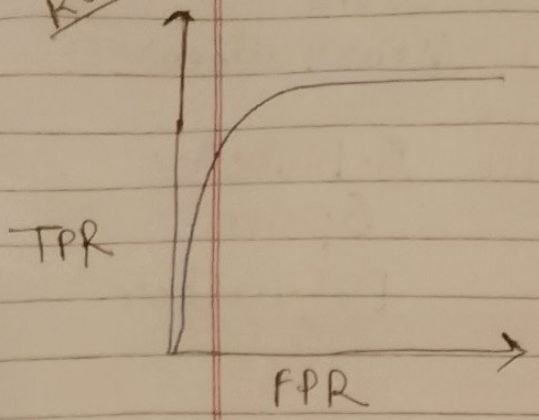
46)

	C_1	C_2	C_3	C_4	C_5	y	P
R_1						0	0
R_2						1	0
R_3						1	1
R_4						0	1
R_5						0	0
⋮						0	1
R_{100}						1	0

47)

(1) TPR FPR Threshold

ROC



FPR → min.
TPR → max.

TPR	FPR	Threshold

Good model

(2) Youden's Index

Diff is high can I get a good model.

FPR	TPR	Threshold	Difference
0	0.89	0.78	0.89
0.01	0.89	0.73	0.88
0.02	0.91	0.62	0.89
0.02	0.95	0.61	0.93
0.03	0.95	0.55	0.92
0.04	0.96	0.46	0.92
0.06	0.97	0.35	0.91
0.09	0.97	0.159	0.88
0.09	1	0.157	0.91

1. The objective is FPR should be as Min. as possible And TPR should be as max as possible
2. Hence the difference between TPR-FPR will be max
3. Take that Threshold comes pending to the max. diff and develop the model

(3) Imbalanced Dataset

(4) 2 classes

Negative (0)

(i) No covid
(ii) No heart disease
(iii) No Diabetics
HAM
Non Defaulter

Positive (1)

(i) Covid
(ii) Heart disease
(iii) Diabetic
Spam
Defaulter

(5)

Balanced Dataset → Target Variable (y)
100

y

non-diabetic: 60, 70, 80, 85

Diabetic: 40, 30, 20, 15

fairly balanced

(6) Balanced Dataset → Target Variable (y)
100

y

non-diabetic: 90, 91, 92, ..., 99

diabetic: 10, 9, 8, ..., 1

90 : 10 is considered as imbalanced dataset

Imbalanced dataset.

This is not a Rule but a guideline.

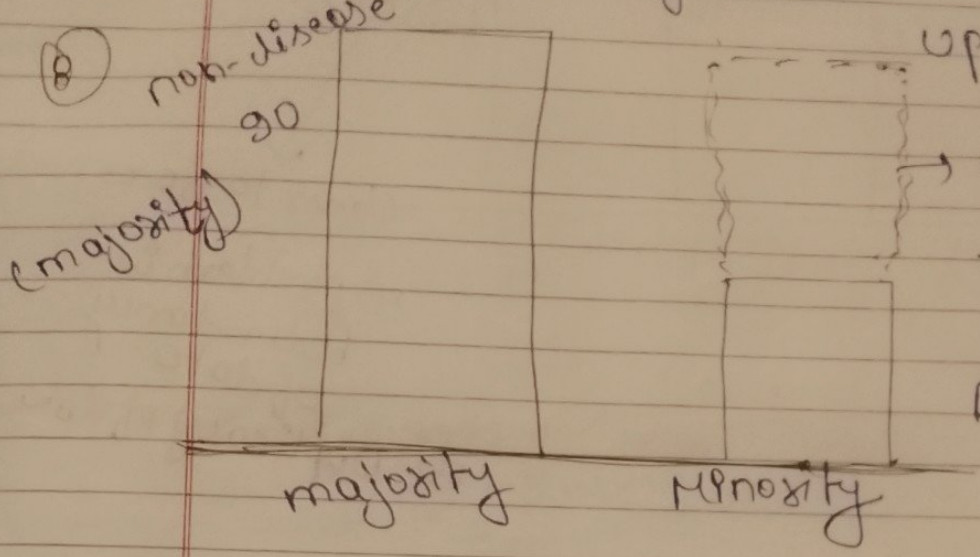
HAM

(1)

(7) Handling imbalanced Data

(a) Up sample minority class
(b) Down sample majority class
(c) change the performance metric
(d) Try Synthetic sampling approach
(e) Use different algorithm

(8)



Up sample minority class
→ closer to 90 or 90
⇒ problem is redundancy of data
Duplicacy of data!

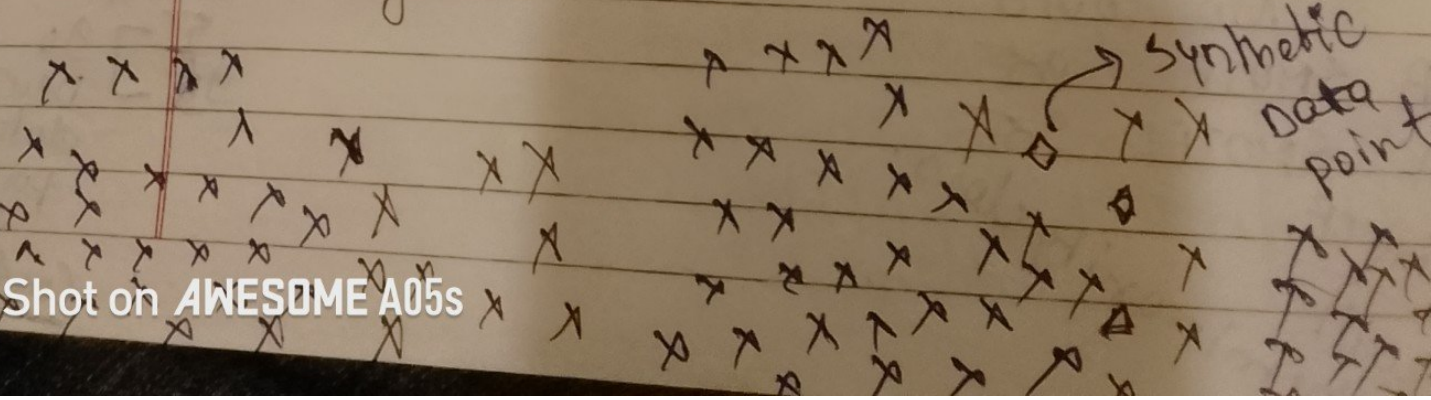
(9) SMOTE Imbalance Dataset

Synthetic minority oversampling Technique

(1) Synthesizing data points for the minority class. Synthesizing means generating new data points based on existing data points.

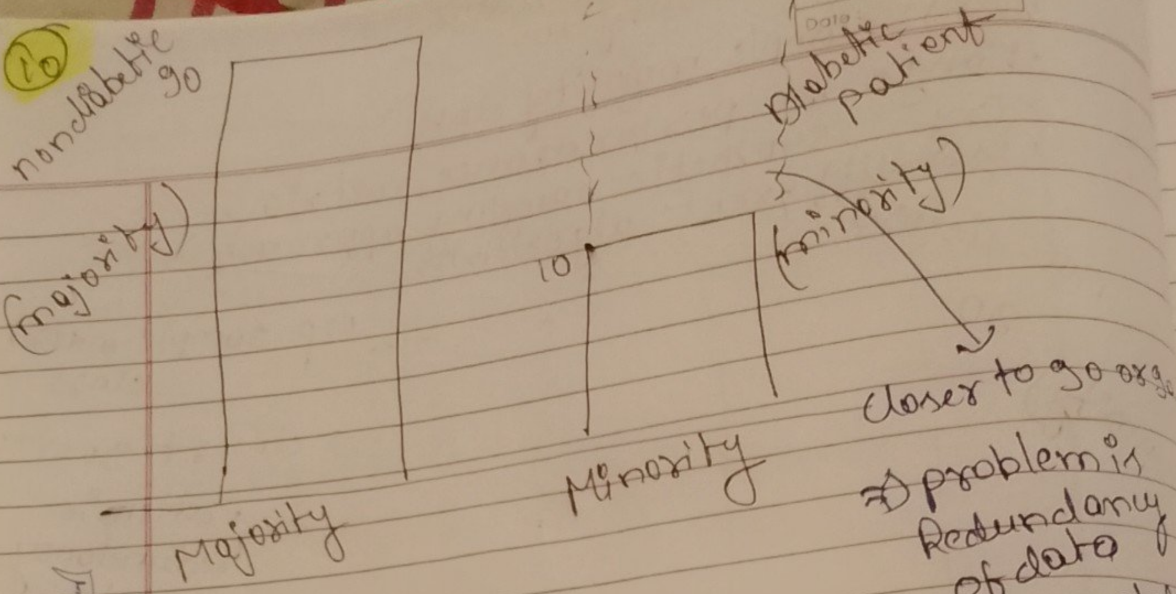
(2) Function randomly picks up a data point from minority class and applies kNN technique.

(3) New datapoint which is called Synthetic data point is added between the point picked up and its neighbors



UP Sampling minority data

(10)



closer to 90 or 9...

⇒ problem is Redundancy of data

⇒ Duplicacy of data

(11)

1. So Using SMOTE you will see the better performance of the model
2. SMOTE is balancing the number of observation in negative and positive class
3. SMOTE is applied only on Training data
4. So when SMOTE is applied on Training data: Model starts learning the pattern well and hence gives better Result.

(12)

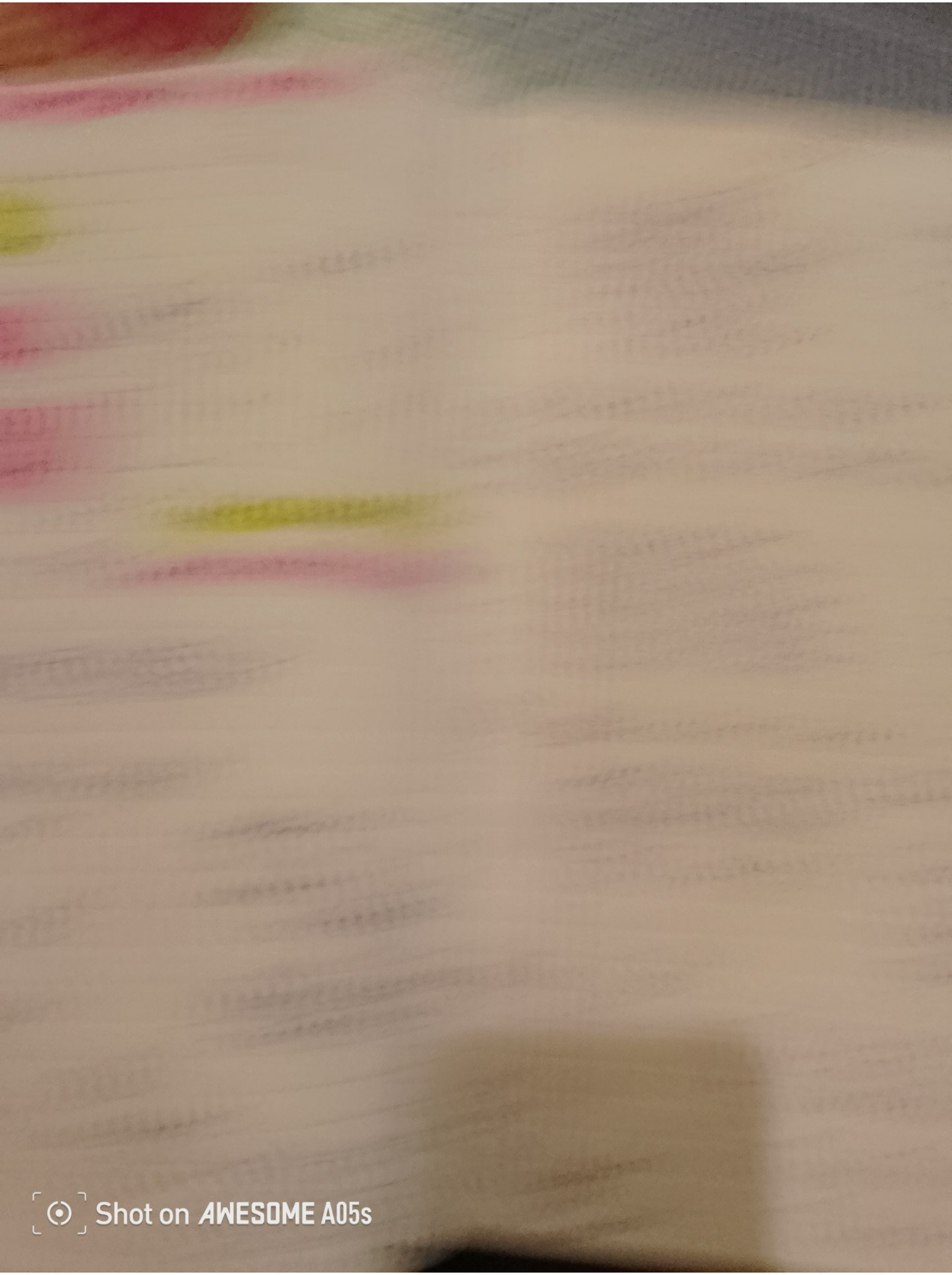
classes

No covid — covid

B SMOTE 95 — 95 [ ]

A SMOTE 95

SMOTE ... It will ... the minority class.


Shot on AWESOME A05s

(22) Variable

Numerical	Categorical
— Euclidean	— Cosine
— Squared Euclidean	— Hamming
— Manhattan	— Mahalanobis
— Chebychev	
— Minkowski	

(23) DISTANCE MEASURES

Depending on the type of data we have different distance measures

Numerical Data	String Data
→ Euclidean distance	→ Cosine distance
→ Manhattan distance	→ Edit distance
→ Minkowski distance	→ Longest Common Sequence
→ Chebyshev's distance	→ Hamming distance

(24) Euclidean distance – numeric data

Consider two points (3,6) and (1,3). Obtain the Euclidean distance

$$\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} = \sqrt{(3-1)^2 + (6-3)^2}$$

$$\sqrt{4+9}$$

$$\sqrt{25} = 5$$

(25)

$$\sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + (x_3-y_3)^2}$$

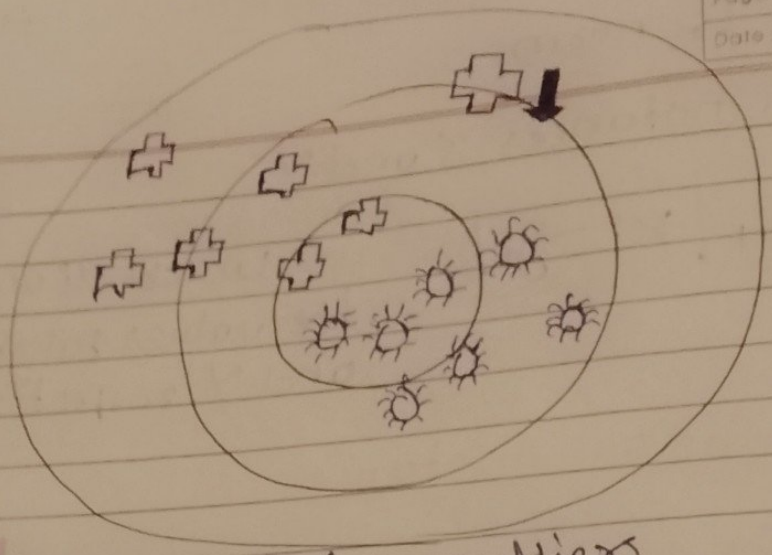
$$\sqrt{(1-3)^2 + (2-4)^2 + (3-5)^2}$$

$$\sqrt{(-2)^2 + (-2)^2 + (-2)^2}$$

$$\sqrt{4+4+4} \quad \sqrt{12} \quad 2\sqrt{3}$$

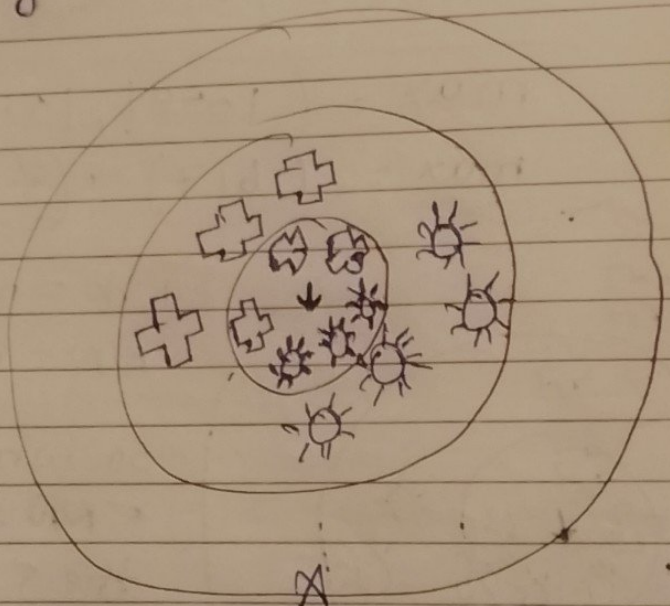
Euclidean method

34



(1) KNN is impacted by outliers
(2) K=1 is very small to decide the nearest neighbour

35



K → Avoid having K value as even as there is a possibility of tie.

36

Observation	Humidity (X)	Temperature (Y)	Rain
1	58	19	0
2	62	26	0
3	40	30	0
4	36	35	0
5	87	19	1
6	93	18	1
7	79	16	1
8	69	17	1
9	62	33	0
10	71	15	1

4	55	33	0
12	78	19	1
13	60	20	1
14	58	35	0
15	35	39	0

A → 1 → D_1

A → 2 → D_2

A → 3 → D_3

A → 4

A → 5

⋮

A → 15

$\sqrt{(84-58)^2 + (37-14)^2}$

$\sqrt{(84-62)^2 + (37-26)^2}$

$\sqrt{(84-40)^2 + (37-30)^2}$

(H) (T)

(84, 37)

37

observation	Euclidean distance (sorted)	class label (rainfall)
5	18.25	1
12	18.97	1
6	21.02	1
7	21.59	0
9	22.36	0
2	24.6	1
8	25.00	1
10	25.53	0
14	26.08	0
11	29.27	1
13	29.41	0
1	31.62	0
3	44-55	0
4	48.04	0
15	49.04	0

It uses voting principal to decide the class.

KNN → Hence this is called lazy learner

Training → It is only memorizing the coordinates of data points

Testing → All calculations are done → It is calculating the euclidean distance

(15) Calculations are done in testing phase Hence. It acts lazily to do all calculation at the end.

(16) **KNN**

Advantages	Disadvantages
① Easy to implement	① To chose apt value for K
② No training required	② Computational expensive
③ New data can be added at any time	③ Can not tell which features gives the best result
④ Effective if training data is large	

(17) **KNN Applications**

→ Image Classification

→ Handwriting recognition

→ Predict credit rating of customers

→ Replace missing values } → We saw one application

Shot on AWESOME A05s

Session 2
Day 6

Navie bayes → It is totally probability based

1) Probability based,

① Probability is how likely an event is to occur

$$P = \frac{\text{No. of ways an event can occur}}{\text{Total possible events.}}$$

② The probability of an event always lies in between 0 and 1

③ 0 indicates impossibility of the event and 1 indicates a certain event.

There are 40 candidates in a team with equal calibre. Out of which 25 are men and 15 are women. A person is randomly chosen to be team leader. What is the probability that the person is a woman?

No of ways event can occur = 15
Total no. of outcomes = 40
So the probability : 15/40 = 0.375 ✓

② bayes theorem → Conditional Probability

Cond Prob = P(A/B)

↓

finding the probability of event A given event B has already happened.

cond Prob = P(B/A)

finding the probability of B (event) given A has already happened

Note

$$P(A/B) = \frac{P(A \cap B)}{P(B)} \quad \text{— eq (1)}$$

$$P(B/A) = \frac{P(A \cap B)}{P(A)} \quad \text{— eq (2)}$$

By eq (1) & eq (2)

$$P(A \cap B) = P(A/B) \cdot P(B) \quad \text{— (3)}$$

$$P(A \cap B) = P(B/A) \cdot P(A) \quad \text{— (4)}$$

By eq (3) & (4)

$$P(A/B) \cdot P(B) = P(B/A) \cdot P(A)$$

?

given information

$$P(A/B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$ — Bayes Theorem

$$P(A/B) \cdot P(B) = P(B/A) \cdot P(A)$$

given information

?

$$P(B/A) = \frac{P(A/B) \cdot P(B)}{P(A)}$$ — Bayes Theorem

Note: Bayes theorem uses conditional probability

conditional prob. ques

→ A pair of fair dice is rolled. If the sum of numbers that appear is 6. Find the probability that one of the dice shows 2 ?

$$P\left(\frac{\text{loan} = NA}{CS = \text{high}}\right) = (1/2) \cdot (2/3) / (3/5) = 1/3$$

$$P\left(\frac{\text{loan} = \text{approved}}{CS = \text{high}}\right) = (2/3)$$

$$P\left(\frac{\text{loan} = NA}{CS = \text{high}}\right) = (1/3)$$

$$P\left(\frac{\text{loan} = A}{CS = \text{high}}\right) > P\left(\frac{\text{loan} = NA}{CS = \text{high}}\right)$$

→ Hence the person will be given a loan.

BAYES THEOREM – FORMULA

conditional prob. of t given that predictor x i.e posterior probability

Probability of variable it is prior probability

$$P(t/x) = \frac{P(t) \cdot P(x/t)}{P(x)}$$

Probability of value taken by the predictor variable i.e. evidence.

conditional Prob. of x given that its class label is t, i.e likelihood

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

$P(A/B)$ → Posterior probability.

$P(A)$ → prior probability

$P(B/A)$ → likelihood

$P(B)$ → Evidence

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

$$P(t/x) = \frac{P(x/t) \cdot P(t)}{P(x)}$$

where
t = target variable
x = Independent variable

$$P\left\{\frac{loan=1}{cs=high}\right\} = \frac{P\left\{\frac{cs=high}{loan=1}\right\} \cdot P(loan=1)}{P(cs=high)}$$

$$P(t/x) = \frac{P(x/t) \cdot P(t)}{P(x)}$$

$$P(t/x_1, x_2) = \frac{P(x_1 x_2/t) \cdot P(t)}{P(x_1 \cdot x_2)}$$

loan
Salary, experience

This where Bayes theorem fails.

For classification
Uses Bayes Theorem
Assumes Independence between the variables – Naive

Because this we failing we applied Assumption

Conditional Probability → Bayes Theorem → Naive Assumption → Naiv

$P(A/B) = P(A \cap B)/P(B)$

Prob. of A given B
$P(A/B) = P(A \cap B)/P(B)$
Prob. of B given A
$P(B/A) = P(A \cap B)/P(A)$

Variables (features) are independent of each other

when
'indepe
mult

P[Diabetic
A

lets sa
– Glu
– Blo
– In
– BM

If B dep
real wor
paramet
diabetic

P(Diabeti

Shot on AWESOME A05s

$$P(\underset{A}{\text{Diabetic}} / \underset{B}{\text{glucose}}) = \frac{P(\text{glucose}/\text{Diabetic}) * P(\text{Diabetic})}{P(\text{glucose})}$$

lets say that parameters are.
- Glucose
- Blood Pressure
- Insulin
- BMI

If B represents condition leading to diabetic. In real world, there might be several conditions or parameters that might lead a patient to be diabetic

$$P(\text{Diabetic} / \text{glucose}, BP, \text{Insulin}, BMI) =$$

$$\frac{P(\text{glucose}, BP, \text{Insulin}, BMI / \text{Diabetic}) * P(\text{Diabetic})}{P(\text{glucose}, BP, \text{Insulin}, BMI)}$$

→ **Naive Bayes** when events are independent multiplied

Naive means basic assumption that variabl are independent of eac other.

$$P(t/x_1 x_2 x_3) = \frac{P(x_1 x_2 x_3/t) \cdot P(t)}{P(x_1 x_2 x_3)}$$

$$P(t/x_1 x_2 x_3) = \frac{P(x_1 \cap x_2 \cap x_3/t) \cdot P(t)}{P(x_1 \cap x_2 \cap x_3)}$$

$P(A \cap B \cap C) = P(A) \cdot P(B) \cdot P(C)$

↳ if A, B, C are independent of each other then

$$= \frac{P(x_1/t) \cdot P(x_2/t) \cdot P(x_3/t) \cdot P(t)}{P(x_1) \cdot P(x_2) \cdot P(x_3)}$$

Lecture 2

→ Naive Bayes is used only for classification

→ It is probability based. This is one reason it can only be used for classification

→ Wide application of NB → Mail Box
↳ Predict (Mail)

Ham or Spam

→ High priority tickets

→ All classification examples

Lonely	2	1
Horoscope	20	5
work	5	12
Snacks	0	5
Money	21	7

Email (Good, work) → Spam / Ham

$$P\left(\frac{t}{x_1, x_2}\right) = \frac{P(x_1 \cap x_2 / t) \cdot P(t)}{P(x_1 \cap x_2)}$$

t = Ham / Spam

Good and work
↓ ↓
x_1 x_2

Assumption → Good is independent of work

$P\left(\frac{SPAM}{Good, work}\right)$ → (1)

$P\left(\frac{HAM}{Good, work}\right)$ → (2)

$P(SPAM / Good, work) >$

$P(HAM / Good, work)$ then

this will be classified as SPAM
else HAM

Day 6 (Regression)

Classification

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier, plot_tree
```

Decision Tree

INFORMATION THEORY

Information theory is based on the intuition that:

Event	Information Gain	Example
Most likely event	No Information	The sun rose this morning
Likely event	Little Information	The sun rose at 6:30 am this morning
Unlikely event	Maximum Information	There was a solar eclipse this morning

- Let $I(x)$ denote the information of an event X
- It is the self Information of an event X at X

$$I(x) = -\ln P(x)$$

- Since we have considered natural log its unit is nat
- For log with base 2, we use units called bits or Shannons.

Shannons Entropy

- Entropy is the measure of information for classification problem, i.e. it measures the heterogeneity of a feature

The entropy of a feature is calculated as

$$E = -\sum_{i=1}^{c} P_c \log_2 P_c$$

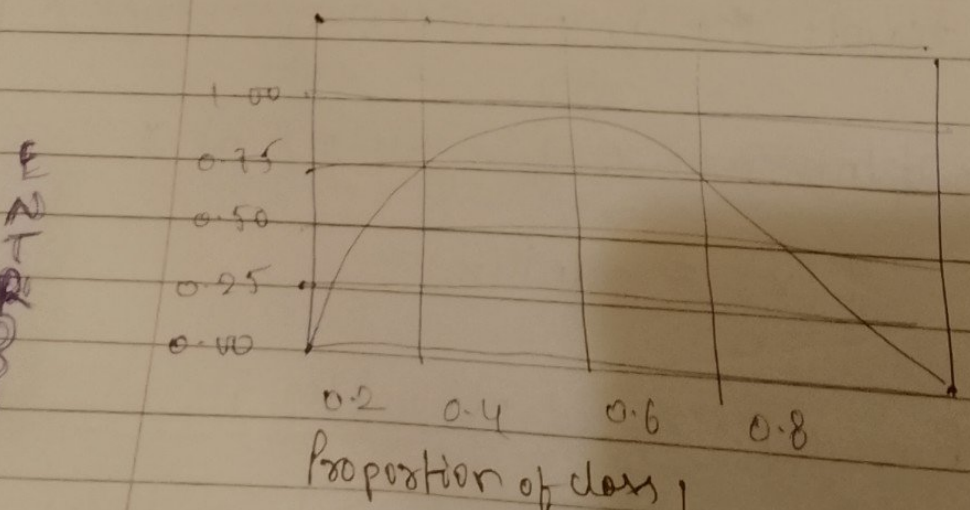
where P_c is the probability of occurrence of the class.

A lower entropy is always prefered

Entropy is always non-negative

Consider a feature with two class the entropy for various proportion of classes is given below

class 1	0	0.1	0.3	0.5	0.7	0.9	1
class 2	1	0.9	0.7	0.5	0.3	0.1	0
Entropy	0	0.46	0.88	1	0.88	0.46	0



① p_default = 6/20
p_non_default = 14/20

Entropy = -(p_default * np.log2(p_default) + p_non_default
* np.log2(p_non_default

Entropy
0.881290899230627

② p_default = 10/20
p_non_default = 10/20

Entropy = -(p_default * np.log2(p_default) + p_non_default
* np.log2(p_non_default)

Entropy
1.0

Conditional Entropy

The conditional entropy of one feature given other is calculated as from the contingency table of the two features

$$E(T/X) = \sum_{x \in X} P(c) E(c)$$

It is the sum of the of the product of probability of occurence of the each class and the entropy of it

Q Obtain the condition entropy of loan given the income of the person

Income	loan: Reject	loan: Approved
Low	8	1
High	2	4

Entropy of loan? E(loan)
entropy (10/15, 5/15)
ent_loan
0.9182958340544896

Entropy E(loan/income)

ent_loan_given_income = (9/15) * entropy(8/9, 1/9) + (6/15) * entropy(2/6, 4/6)

ent_loan_given_income
0.669273334487 1833

INFORMATION GAIN

- Information gain is the decrease in entropy at a node
- Information Gain(T, X) = Entropy(T) - Entropy(T/X)
- To construct the decision Tree, the feature with highest information gain is chosen
- Information gain is always positive

Q Obtain the information gain in the feature loan due to income

IG_loan_income = ent_loan - ent_loan_given_income

IG_loan_income
0.24902249951730627

Decision Tree Algorithms

Date /
Page No.

Hunts Algorithm

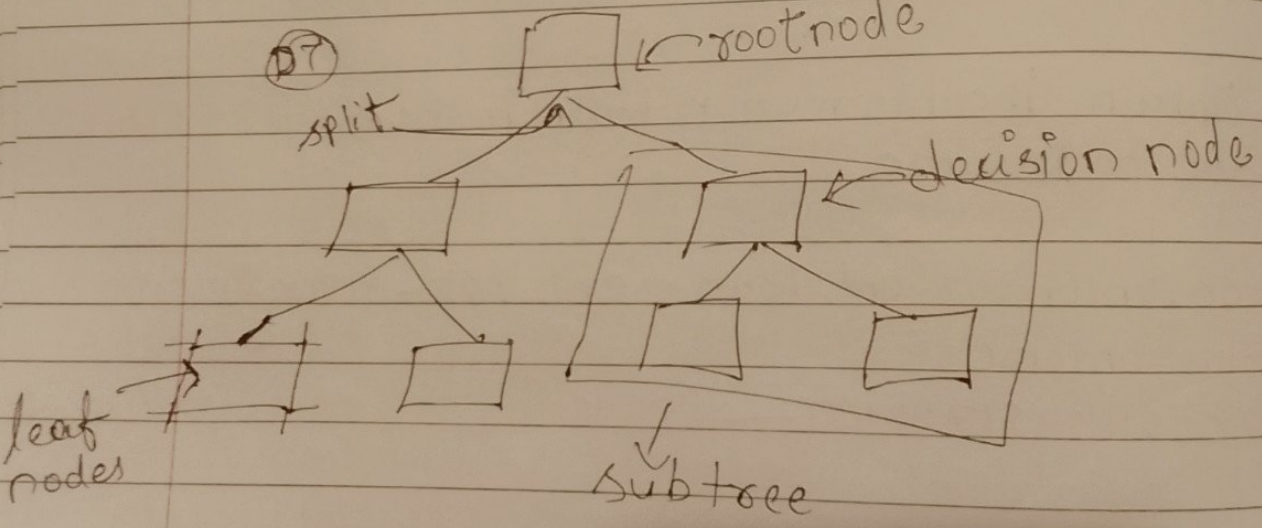
Let S_n to be the training samples associated with node n and y_c be the class labels. The algorithm is as follows

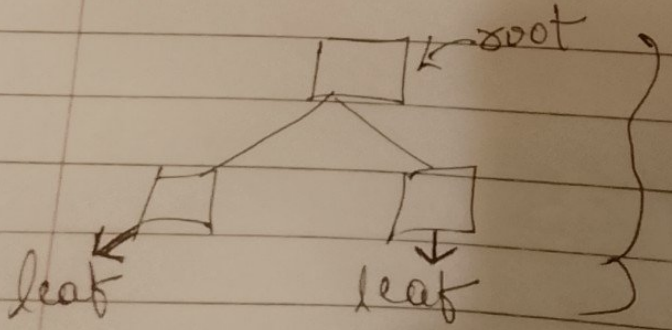
① If all samples belong to the same class y_c, then node n is a leaf node with label y_c

② If S_n has samples with more than one class, an attribute value is selected to partition the samples into smaller subsets such that the samples in the subsets belong to same class

where is the information gain and that feature will be your node

Hunts Algorithm





what is the purity of windy = False (Gini Index)

pd.crosstab(data['Windy'], data['Play'])

Play	no	yes
windy		
False	2	6
True	3	3

gini_index = 1-((2/8)**2+(6/8)**2)

gini_index

0.375

gini_index = 1-((3/6)**2+(3/6)**2)

gini_index

0.5

Yes overcast class in outlook.

gini_index = 1-((0/4)**2+(4/4)**2)

gini_index

0.0

Classification Error

- The classification error of a variable is calculated as

$$\text{Error} = 1 - \max(pc)^2$$

where pc : probability of occurrence of the class

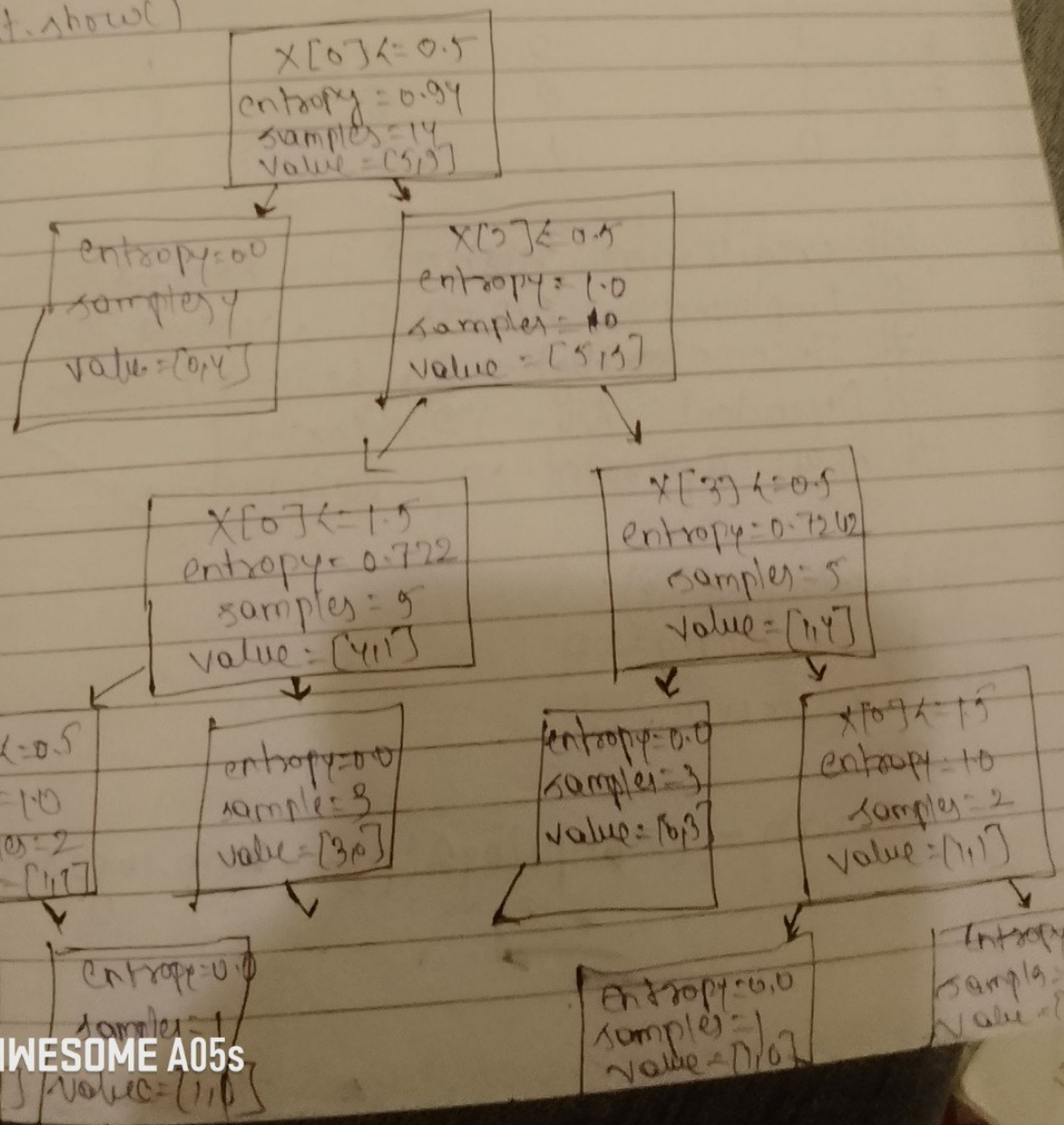
- For samples belonging to one class, the classification error is 0 and for equally distributed samples, the classification error is 0.5

Decision Tree in sklearn()

Date / /
Page No.

data

```
x = data.drop(columns = ['Play'])
y = data['Play']
x.dtypes
dt = DecisionTreeClassifier(criterion='entropy')

le = LabelEncoder()
for i in data.columns:
    data[i] = le.fit_transform(data[i])

plt.figure(figsize=(7,5))
plot_tree(dt)
plt.show()
```



ENSEMBLE LEARNING

Date / /
Page No

→ Ensemble learning algorithms combine multiple models into one predictive model

→ Decisions from several weak learners are combined to increase the model performance.

ENSEMBLE LEARNING MODELS

Bagging	Boosting	Stacking
Homogeneous models can be built independently and their outputs are aggregated at the end.	Homogeneous models can be built sequentially previous model dictates the features the succeeding model will focus on	Heterogeneous base models can be built, outputs from base model are used as inputs to the meta model.
Example Random Forest	Example Ada Boost	Example Voting classifier

RANDOM FOREST CLASSIFIER

Random — The subsets of variables are selected at random to build a decision tree

Forest — Forest of "Decision" trees.

→ Random forest consists of several independent decision trees that operates as an ensemble.

→ It's an ensemble learning algorithm based on bagging

→ Train decision tree models on bootstrap samples where variables are selected at random. The aggregate output from these trees is considered as the final output.

BOOTSTRAP

- Bagging is composed of two parts
  1. Bootstrapping
  2. Aggregation

- Random sampling with replacement
- For a data with i observations, a random sample with replacement of size i, is a bootstrap sample
- The observations n_3 and n_6 are not included in the bootstrap sample; they are the out-of-bag (OOB) samples

Original Data	Bootstrap Sample
n_1	n_4
n_2	n_5
n_3	n_4
n_4	n_5
n_5	n_4
n_6	n_2

...(with replacement)

did you know?

Out-of-bag sample

Date / /
Page No

* Almost 36.8% of the training data has the potential to be out-of-bag sample

How?

* Since, we consider the bootstrap sample i.e. a random sample with replacement. The probability of not selecting a sample is $(1-1/N)$. For large N, the probability $(1-1/N)^N$ is approximately $1/e = 0.368$

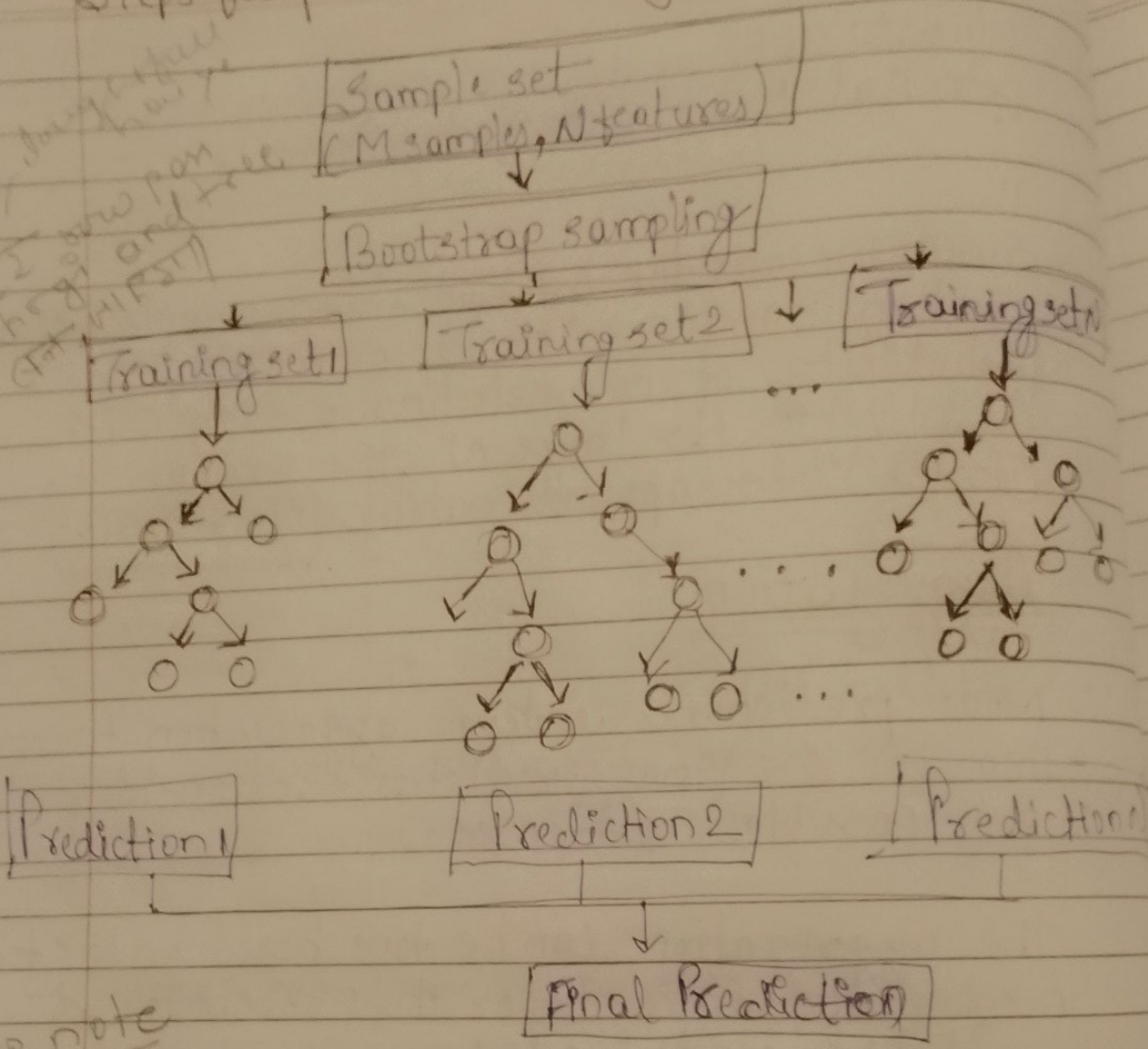
Aggregation

Model predictions undergo aggregation to combine them for the final prediction to consider all the possible outcomes.

The aggregation can be done based on the total number of outcomes, or the probability of predictions derived from the bootstrapping of every model in the procedure.

Date / /

Steps for prediction using random forest



Imp note

For classification, the final prediction considers the mode of the predicted labels and for regression, it considers the average of the predicted values.

Advantages of Random Forest

→ Improved Accuracy — Random Forest often achieve higher accuracy compared to individual models

→ Increased Robustness — Random Forest more robust to outliers and noisy data

→ Reduced Variance — Random forests reduce variance by taking many trees from n bootstrapped samples and averaging their prediction leading to lower variance.

→ Reduced Bias

- Random Forests also reduce bias in the dataset by decorrelating the trees which is a problem with imbalanced datasets Random forests overcome this problem by forcing each split to consider only a subset of predictors
- Therefore, on average $(p-m)/p$ of the splits will not even consider the strong predictor and so other predictors will have more of a chance.
- We can think of this process as decorrelating the trees, thereby making the average of resulting trees less variable and hence more reliable.

→ Reduced Overfitting — It helps in reducing overfitting especially when individual models tend to overfit the training data.

Disadvantages of Random Forest

→ Increased Complexity
Ensembles introduce additional complexity to the modeling process

→ Computational Overhead
Ensemble methods require training and maintaining multiple models, which can be computationally expensive especially when dealing with large datasets or complex models.

→ Interpretability
Ensemble models are typically less interpretable compared to individual models.

Ensemble Technique (Bagging)
Random Forest

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=5, max_features=None)
rfc.fit(x_train, y_train)
```

RandomForestClassifier

```
RandomForestClassifier(max_features=None, n_estimators
```

Ada Boost

Boosting

Step 1

Credit_Score	Employed	Income	Dependent	Loan
1	1	1	0	1
1	1	1	0	1
1	1	1	1	0
1	1	1	1	0
1	1	1	1	0
1	1	1	1	0
1	0	1	1	0
1	0	0	0	1
0	1	1	1	1
0	0	1	0	1
0	1	0	1	1
0	1	1	0	1
0	0	0	0	1
0	1	0	0	1
0	0	1	0	1

Excel data Hai? Excel pe Ada Boost Boosting example Hai

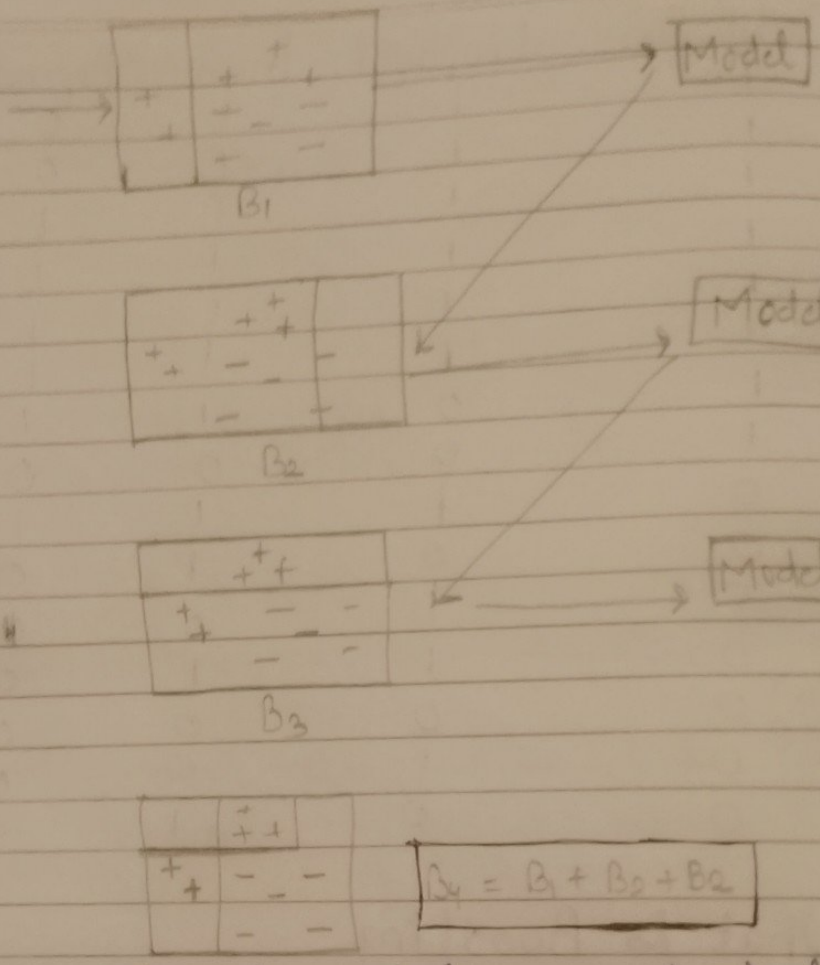
What is Boosting

→ The ensemble of weak learners that learn sequentially

→ In each iteration weights of the samples are adjusted, such that the misclassified samples have a higher weight, therefore higher chance of getting selected to train the next classifier

→ Boosting reduces bias and variance

→ Here B_1, B_2 and B_3 are base learners and B_4 is boosting ensemble of weak learners



→ The ensemble of weak learners that learn sequentially

→ In each iteration weights of the samples are adjusted, such that misclassified samples have a higher weight, therefore higher chance of getting selected to train the next classifier

→ Boosting reduces bias and variance

→ Here B_1, B_2 and B_3 are base learners and B_4 is the boosting ensembler of weak learners.

Boosting = advantages

→ Enhances the efficiency of weak classifiers
→ Both precision and recall can be enhanced through boosting algorithms

Boosting = Disadvantages

→ loss of simplicity and explanatory power
→ Increased computational complexity

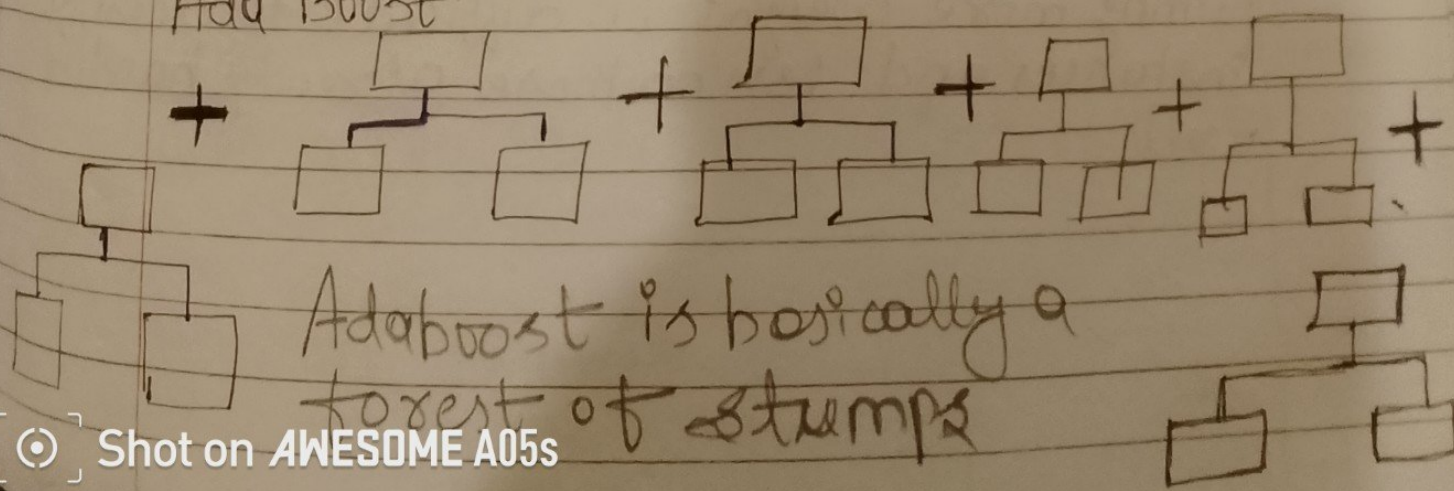
How boosting differs from bagging?

BAGGING	BOOSTING
Base learners learn is parallel	Baselearners learn sequentially
Random sampling	Non-Random Sampling
Reduces variance	Reduces bias and variance

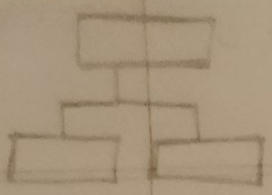
BOOSTING ALGORITHMS ARE :-

→ Ada Boost
→ Gradient Boosting
→ XGBoost

Ada Boost



Adaboost is basically a forest of stumps



Stump is a tree with just one parent node and two leaves

→ A single algorithm or a stump may classify the objects poorly

→ But if we combine multiple classifiers with a selection of training set at every iteration and assigning the right amount of weight in the final voting, we can have good accuracy score for overall classifier

Ada Boost

→ Ada Boost is short for Adaptive Boosting

→ It was formulated by Yoav Freund and Robert Schapire

→ It is sensitive to noisy data and outliers

→ Usually, decision trees are used for modelling

→ Multiple sequential models are created, each correcting the errors from last model

→ Ada Boost works by weighing the observations, putting more weight on difficult to classify instances and less on those already handled well

Step ② Initialize 0.067 to all rows
↓
Initial weight

Date / /
Page No

	Credit_Score	Employed	Income	Dependent	Loan	Initial Weight
1	1	1	1	0	1	1/15 0.067
2	1	1	1	0	1	0.067
3	1	1	1	0	0	0.067
4	1	1	1	1	0	0.067
5	1	1	1	1	0	0.067
6	1	1	1	1	0	0.067
7	1	0	1	1	0	0.067
8	1	0	0	0	1	0.067
9	0	1	1	1	1	0.067
10	0	0	1	0	1	0.067
11	0	1	0	1	1	0.067
12	0	1	1	0	1	0.067
13	0	0	0	0	1	0.067
14	0	1	0	0	1	0.067
15	0	0	1	0	1	0.067

Step ③ calculate entropy

Count of Credit_Score	Column Labels	
Row Labels	0	1	Grand Total
0	0	7	7
1	5	3	8
Grand Total	5	10	15

Credit_Score

0 7

5 3

(7/15)*entropy(0/7, 7/7) + (8/15)*entropy(5/8, 3/8)

0.509031468226648

#14 Employed

#46

entropy(5/15, 4/5) + (10/15)*entropy(4/10, 6/10)

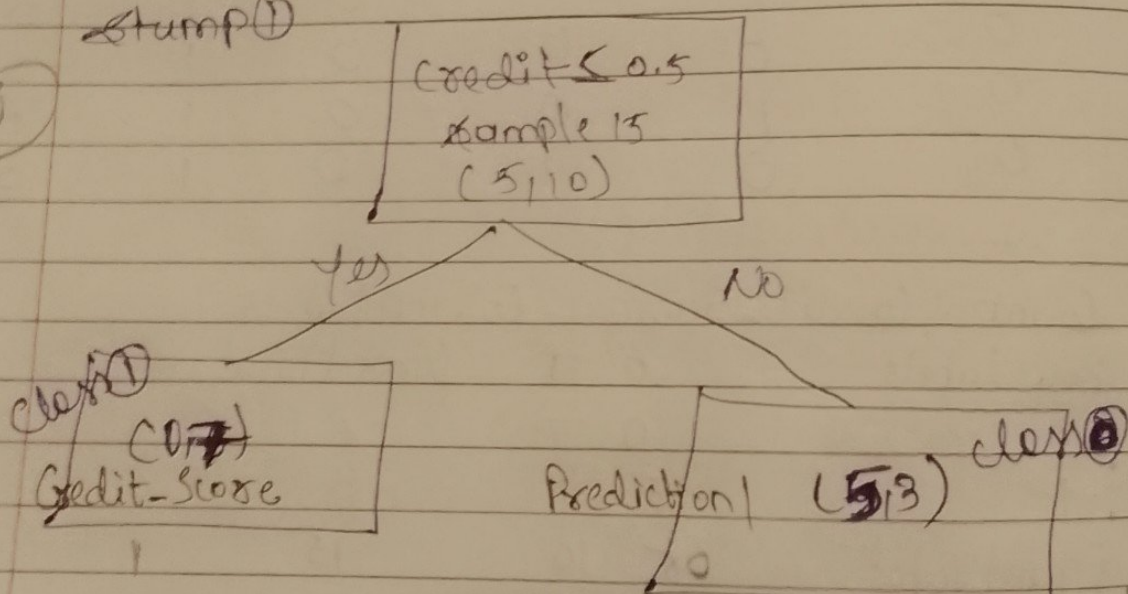
0 4   Income
5 6
(4/15)*entropy(0/4,4/4)+(11/15)*entropy(5/11,6/11)
0.7289354884164346

1 8   Dependent
4 2
(9/15)*entropy(1/9,8/9)+(6/15)*entropy(4/6,2/6)
0.66.

Step① Stump① बनाएंगे and stump① बनाने k liye जिसकी entropy सब कम आई है, उसमें Info. gain सबसे ज्यादा Hai, and. उसी से stump① बनेगा।

Stump①

(i)



(ii) अब एक prediction column बना and जहाँ credit score 1 Hai वहाँ 0 आ जाएगा and जहाँ credit score 0 Hai वहाँ 1 आ जाएगा

	Credit_Score	Prediction1
1	1	0
2	1	0
3	1	0
4	1	0
5	1	0
6	1	0
7	1	0
8	1	0
9	0	0
0	0	0
1	0	0

13	0	1
14	0	1
15		

(iii) अब prediction ① को column [illegible] usase Loan Mein compare करेंगे and misclassification data find करेंगे।

	Credit-Score	Loan	Initial Weight	prediction1
1	1	1	0.067	0
2	1	1	0.067	0
8	1	1	0.067	0

ये Misclassification data hai

(iv) Total error find करने k liye no. of misclassification data ÷ by Total no. of data.

$$\text{Total error} = 3/15 = 0.2$$

(v) Amount of say = [(1/2)* log[(1−TE)/TE] formula

$$= 0.69315$$

Amount of Say and Total Error

→ when stump does a good job then Total Error is small Amount of Say will be a large positive value

⇒ when stump classifies only half of the samples correctly and half incorrectly, then Total Error is 0.5. Amount of say will be zero

→ When stump does a very poor job i.e. the stump gives an opposite classification, then the total error would be close to −1 and the Amount of say will be a large neg. value

Shot on AWESOME A05s

Decision Tree

① It deals with categorical entity where relationship b/w entities of target variable is not linear

② Can be used for both regression & classification

③ It is like a flowchart that makes decision based on a series of questions and answers. It starts of a "root Node" with question of "leaf node" (represents decision)

④ It divides data into tree like structure (root node of leaf node)

⊙ This distribution is based on Gini index & Entropy

⊙ This process combines until we reach the stopping criteria, such as max. depth or a min. no. of samples at node.

Ex:- Bank wants to decide if person is eligible for a loan based on credit score.

← entropy :- Measure of Information
- Is always +Ve
- Lower entropy is prefered.

$$\left[-\sum P_c \log_2(P_c)\right] \therefore P_c = \text{Prob of occurence}$$

Here 2 in log represents no. of classes in target of dtss

← Conditional entropy

helps to measure how much uncertainity is there in one thing when we know the information about another thing

$$\left\{ E(T|X) = \sum_{x \in X} P(c) E(c) \right\}$$

← Information gain = Decrease in Entropy at a node.

Shot on AWESOME A05s

① To make decision (tree), highest Information Gain is used
② Always +ve

→ Measure of Impurity
④ Entropy ⑤ Gini Index ⑥ classification error

⑤ Gini Index $(1 - P(Yes)^2 - P(No)^2)$ ✓
① Samples belonging to same class, GI = 0
② For equally distributed sample, GI = 0
$GI(T/X) = \text{entropy}(T) - \text{entropy}(T/X)$
Info. Gain $(T/X) = GI(T) - GI(T/X)$ } T = Target, X = features

⑥ classification error
① Samples of same class, CE = 0
② Samples of equally dist. CE = 0.5

⑦ Pruning Reduce complexity of tree.
① Pre Pruning It stops tree before it fully grown
② Post Pruning Tree is allowed to grow completely then prune

To reduce Overfitting, we use pruning by passing Hyperparameters
→ max_depth ✓
→ max_sample_split
→ max_sample_leaf
→ max_leaf_nodes
→ min_sample_leaf
→ min_sample_split

⑧ Importance of Node
[% of data in parent * Ent. of parent −
% of data in left child * Ent. of left child −
% of data in right child * Ent. of Right child]

feature imp. of any class = imp. of node X / Imp. of all Nodes

Xgr Boost

- Xgr Boost combines the predictors of multiple weak learners to create a single, strong predictive model.
- It involves building models sequentially, where each new model focused on correcting the errors made by previous ones
- The no. of boosting rounds (trees) is a tunable hyperparameter
- "Xgr Boost Assign weights" to data points based on errors made by previous trees (Data points that are more difficult to predict require weights)
- Xgr Boost provides a measure of "feature Importance" that helps us to identify which feature have the most influence on models predictions
- Xgr Boost has built in handling of missing values.
- Xgr Boost has 3 components :-

1. Objective function
   This defines loss function to be minimized during training
   - Mean Squared error → Regression
   - log loss → classification
2. Weak learners (Base learners)
   Xgr Boost typically uses decision tree as weak learners (stumps)
3. Regularization
   Xgr Boost includes "L_1 (Lasso)" & "L_2 (Ridge)" regularization terms to control the complexity of model & reduce overfitting.

Gradient Boosting

* Gradient Boosting can use a wide range of base learners such as decision trees and linear models.
* It updates weights based on the gradients, which are less sensitive to outliers.

Step① The boosting process begins with weak learners (base learners)

Step② This base learner is trained on the dataset and predictions are made.

Step③ The difference b/w predicted values and the actual values is calculated and this difference is also k/s residual error, becomes target for next weak learners

Step④ Weak learners are then trained to predict the residual error of previous learners. each new learners focuses on minimizing the error made by the ensemble upto that point.

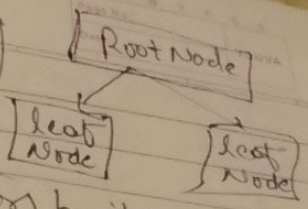
Step⑤ By combining predictions from multiple weak learners Gradient Boosting is able to capture complex relationship in data and make accurate predictions

* Gradient Boosting works well on numerical & categorical data and can handle large data sets.

* If the problem has a pattern of wise classifications or errors that can be fixed iteratively. Gradient Boosting can be interesting approach.

ADA Boost

- It is basically a forest of stumps
- Ada boost assigns weights to the predictions made by each weak learners by iterations
- Predictions of more accurate weak learners hold great weight in the final ensemble
- Ada Boost learns from its past mistakes & adapts its learning process accordingly



→ Weights :—
Ada boost progresses through its iterations, it adopts & Adjusts these weights based on the performance of the weak learners

→ Error minimization :- Once the weights are appropriate updated, Ada Boost proceeds to train a weak learners on the modified training data.
{ The aim is to minimize the error made by models during iteration }

Step 1: Assign weights to each sample

- In begining each sample will have same weights
- { Sample weight = 1 / No. of samples }

Step 2: Build the stumps

- Build stumps with each variable, using Gini Index or entropy for each variable.
- Smallest Gini Index = Base learning Model

Naive Bayes

- Based on Probability
- It will be helpful if an algorithm can label required received emails as important ( ) emails or junk (spam) emails for a user
- Probability :- It is how likely an event is to occur

$$P \Rightarrow \frac{\text{No. of ways an event can occur}}{\text{Total possible events}}$$

- Prob of an event always lies in b/w 0 & 1
- 0 indicates impossibility of event and 1 indicates a certain events

- Bayes Theorem :- → Conditional Probability
  - P(A/B) → finding the prob. of event A given event B has already happened
  - P(B/A) → finding the prob. of B given event A has already happened.

$$P(A/B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$ BAYES THEOREM

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

Posterior prob. → P(B/A) → Prior

P(A/B) → Likelihood

P(A) → Evidence

Naive Assumptions

Variable / features are independent of each other

Note: When we apply Bayes theorem with a Naive assumption the outcome we get is "Naive Bayes" when events are independent then they can be Naive

- Naive Bayes is used only for classification
- It is prob based. This is the reason it can only be used for classification
- Application of NB → Mail Box
  - Ham
  - Spam

Note:
- Naive Bayes fails when frequency of occurance of word = 0
- To overcome this problem we introduce Laplace Theorem
  - Increase the frequency to avoid 0

Binary Cross Entropy

OOB

Entropy Node

Y | N A Y | N