ROC and AUC curve

Confusion Matrix

This is like a report card used to check how well model predicted in test data

		Predicted	
		1	0
Actual	1	True Positive	False Negative
	0	False Positive	True Negative

True Positive Rate (TPR) → Benefit

		Predicted	
		1	0
Actual	1	TP	FN
	0	FP	TN

80	20
20	80

} total 200 test data

Mathematical

$$\boxed{TPR = \frac{TP}{TP + FN}} = \frac{80}{80+20} = \frac{80}{100} = 80\%$$

Eg:- Email spam classification

1 - spam

0 - Not spam

rest 20 not spam

Eg:- Test data 200 emails

100 spam

100 ham

and we predicted 80

Total we have 100 spam mail

TPR means Benefit ⇒ Maximizing ↑ is a good thing
→ True positive rate gives you an intuition of benefit

Another Eg :- Churn rate predict ⇒ Netflix

	1	0
1	TP	FN
0	FP	TN

1 ⇒ leave
0 ⇒ not leave

100 customers — 80 → 20
↓ leave
↓ detect

80	20

TRP = 80 %

model has to face or faced

② False Positive Rate → Cost

100 not spam → 20 spam

$$FPR = \frac{FP}{FP + TN}$$ } formula ↓ Intuition

$\frac{20}{20 + 80} = \frac{20}{100} = 20\%$

→ Email spam classifction

not spam → spam

TPR → Benefit ↑↑
TPR → Cost ⇊

who were not leaving → they will leave the platform

1 ⇒ should always be yes related type
0 ⇒ should always be No related type

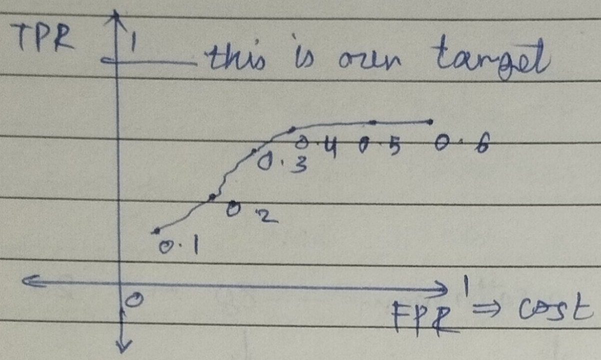
Page No.
Date

ROC curve

- ROC which stands (Receiver operating characteristic) curve } → is a graph between benefit & cost

TPR ⇒ 1
FPR ⇒ 0

TPR ⟷ FPR



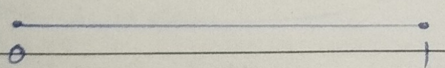
0.1
0.2
⋮
0.6
} Threshold Value

Why do we use binary classification
↳ When a model gives predictions, it usually doesn't just say 0 or 1
- It gives a probablity score between 0 and 1
Eg :-
Model say :- email is spam = 0.97 → high change of spam

Not spam = 0.12 → low change spam

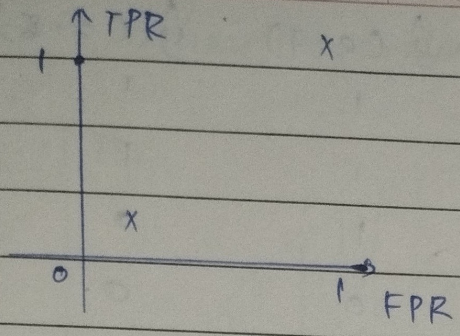
But then we need a threshold
- If prob ≥ threshold → predict 1 (yes)
- If prob < threshold → predict 0 (no)

threshold level



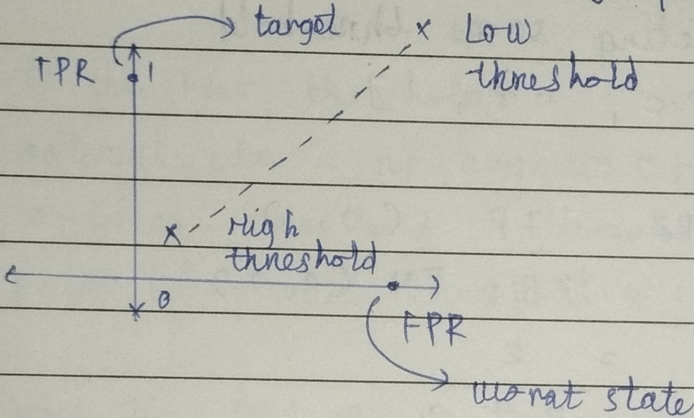
small → 0.1 ⇒ 0
large → 0.99 ⇒ 1

- If threshold value is smaller ⇒ FPR, TPR } ↑
- If threshold value is Larger ⇒ FPR, TPR } ↓



TPR is directly propotional to ~~[illegible]~~ FPR

Case :- TPR is not directly propotional to FPR

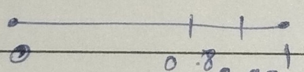


• ⇒ target

	1	0	
1	TP	FN	Actual
0	FP	TN	

Predicted

threshold



High Threshold TPR ↓ FPR ↓
Low Threshold TPR ↑ FPR ↑

from 0.95 → to 0.8

P >= 0.8 is called spam

this will significantly increase TP and reduce FN ↓

negative → not spam — 0
positive → spam — 1

ROC and AUC curve

y	$\hat{y}$	$\hat{y}(0)$	$\hat{y}(0.2)$	$\hat{y}(0.4)$	$\hat{y}(0.8)$	$\hat{y}(0.8)$
1	0.8	1	1	1	1	1
0	0.96	1	1	1	1	1
1	0.4	1	1	1	0	0
1	0.3	1	1	0	0	0
0	0.2	1	0	0	0	0
1	0.7	1	1	1	1	0

→ For constructing ROC and AUC curve
↳ we we'll be selecting some threshold
[0, 0.2, 0.4, 0.6, 0.8, 1]

$TPR = \frac{TP\ (1,1)}{FP + FN\ (1,0)}$ $\quad$ $FPR = \frac{FP\ \ (0,1)}{FP + TN\ (0,0)}$

$= \frac{4}{4+0}$ $\quad$ $= \frac{2}{2+0}$

$TPR = 1$ $\quad$ $FPR = 1$

$\frac{3}{3+1} = \frac{3}{4} = 0.75$

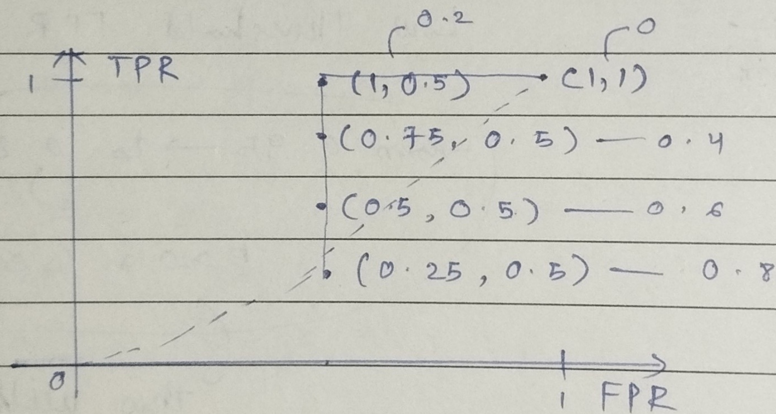
$\frac{1}{1+1} = \frac{1}{2} = 0.5$

$\frac{2}{2+2} = \frac{2}{4} = 0.5$

$\frac{1}{1+1} = \frac{1}{2} = 0.5$

$\frac{1}{1+3} = \frac{1}{4} = 0.25$

$\frac{1}{1+1} = \frac{1}{2} = 0.5$



for $\hat{y}(0.2) \Rightarrow TPR = \frac{4}{4+0} = 4$

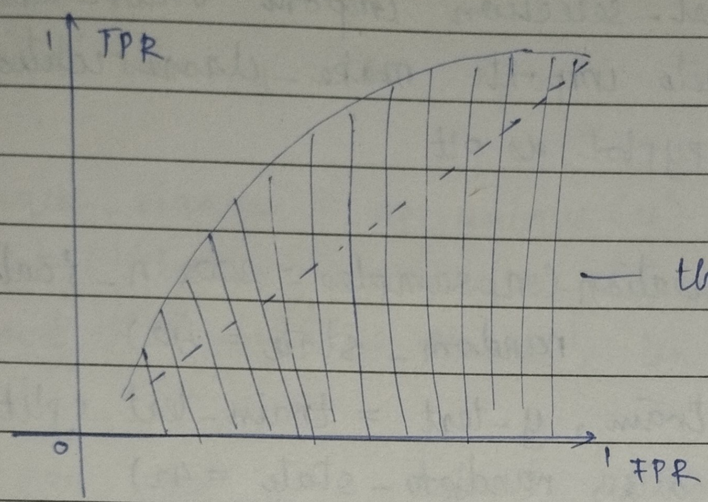
$FPR = \frac{1}{1+1} = \frac{1}{2} = 0.5$

for $\hat{y}(0.4) \Rightarrow TPR = 0.75 \Rightarrow FPR = 0.5$

$\hat{y}(0.6) \Rightarrow TPR = 0.5 \Rightarrow FPR = 0.5$

$\hat{y}(0.8) \Rightarrow TPR = 0.25 \Rightarrow FPR = 0.5$

ROC :- (Receiver operating characteristic)



this is Area Under the Curve

↳ to the best threshold

```
optimal_idx = np.argmax(tpr - fpr)
optimal_threshold = thresholds[optimal_idx]
print('Optimal threshold is :', optimal_threshold)
```

Why Use ROC ⇒ 1.) Threshold manipulation
→ 2.) Model Comparison for better classifier

AUC :- (Area Under the Curve)
↳ Values fall between 0.5 (random) and 1 (perfect) with higher values indicating better classification problem performance

ROC - AUC code

```
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification
import matplotlib.pyplot as plt

X, y = make_classification(n_samples = 200, n_features = 5,
                           random_state = 42)
X_train, X_test, y_train, y_test = train_test_split(X, y,
                           test_size = 0.3, random_state = 42)

model = LogisticRegression()
model.fit(X_train, y_train)

y_pred_proba = model.predict_proba(X_test)[:, 1]

roc_auc = roc_auc_score(y_test, y_pred_proba)

fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
```

[:, 1] ⇒ this 1 indicates how many classes you have since this is a binary classification problem

If you train a model with data containing classes 0, 1, and 2 then after training, using model.classes_ will give you the array

→ [0, 1, 2]
0 1 2

→ [:, 2]

0 1 2 3
if we have [3, 9, 10, 14]

[:, 3]