SELECTED TOPICS IN ENGINEERING

INTR. TO PROG. FOR DATA SCIENCE
ENGR 350

Tuesday-Thursday 10:00-12:45
ENG B05

Dr. Banu Yobaş

Binary classification

By definition, entry *i,j* in a confusion matrix is

- the number of observations actually in group *i*,
- but predicted to be in group *j*

In a binary classification task,

- the terms ‘’positive’’ and ‘’negative’’ refer to the classifier’s prediction, and
- the terms ‘’true’’ and ‘’false’’ refer to whether that prediction corresponds to the external judgment (sometimes known as the ‘’observation‘‘).

PREDICTED	ACTUAL	
	YES	NO
YES	61	16
NO	19	64

Confusion Matrix

- Total # of cases
- Predictions
- The real cases

PREDICTED	ACTUAL	
	YES	NO
YES	61	16
NO	19	64

True Positive (TP)
True Negative (TN)
False Positive (FP) – aka Type I error
False Negative (FN) – aka Type II error

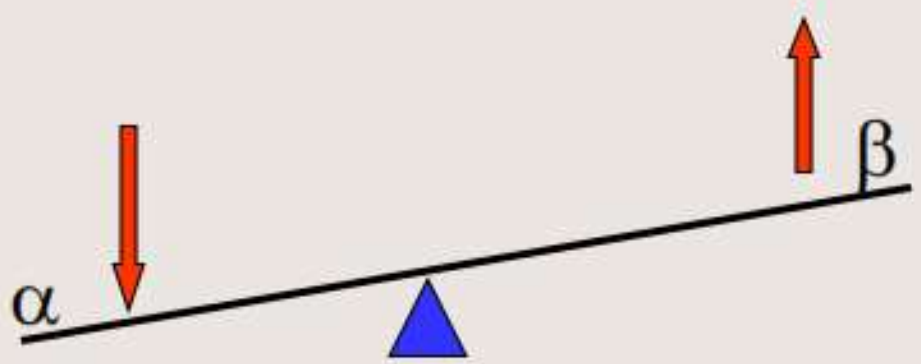
Statistical Significance

- **Type I error** occurs if H_0 is rejected when it's true
- **Type II error** occurs if we do not reject H_0 when it's false

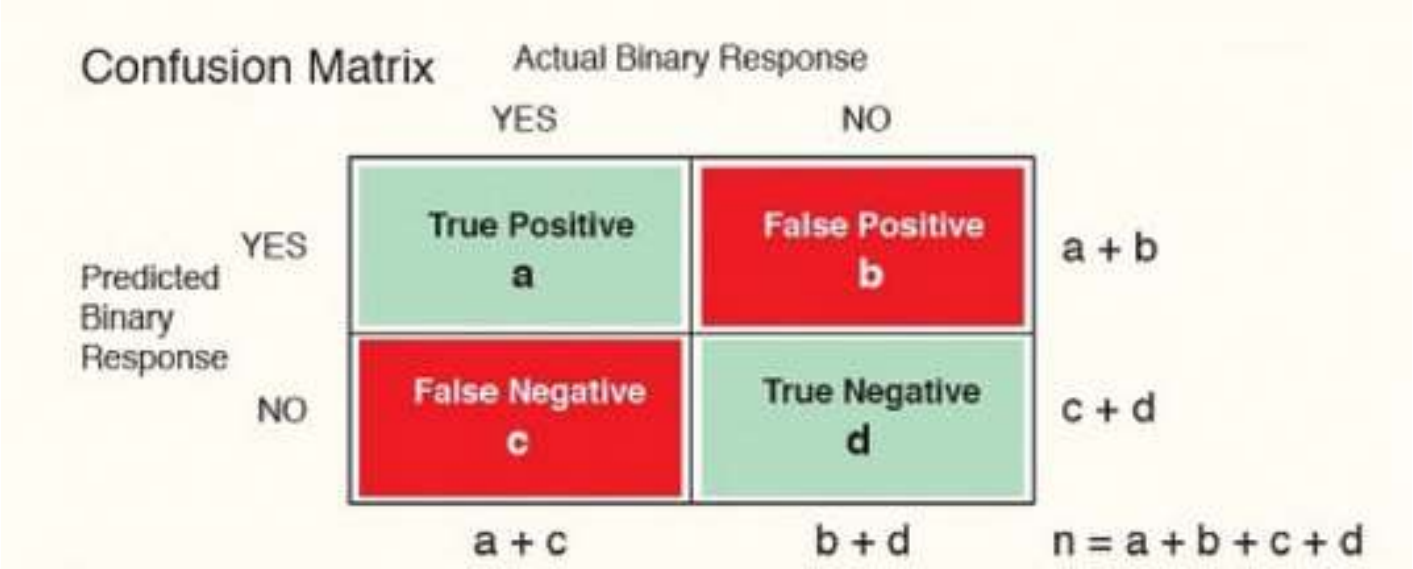
Decision \ Reality	H_0 : No Diff	H_a : Difference
H_0: No Diff	1- α	ß Type II error
H_a : Diff	α Type I error	1- ß Power

Statistical Significance

- We can not minimize α and β together.
  - If we minimize α , β increases
  - If we minimize β , α increases



Evaluating the Predictive Accuracy of a Binary Classifier



Confusion Matrix

- **Accuracy**: Overall, how often is the classifier correct?
  - (TP+TN)/total = (61+64)/160 = 0.78
- **Misclassification Rate**: Overall, how often is it wrong?
  - (FP+FN)/total = (16+19)/160 = 0.22
  - = (1 – Accuracy)
  - also known as "**Error Rate**"
- **True Positive Rate**: When it's actually yes, how often does it predict yes?
  - TP/actual yes = 61/80 = 0.76
  - also known as "**Sensitivity**" or "**Recall**"
- **False Positive Rate**: When it's actually no, how often does it predict yes?
  - FP/actual no = 16/80 = 0.2

PREDICTED	ACTUAL		
N=160	YES	NO	
YES	61	16	77
NO	19	64	83
	80	80	

Confusion Matrix

- **Specificity**: When it's actually no, how often does it predict no?
  - TN/actual no = 64/80 = 0.8
  - = (1 – False Positive Rate)
- **Precision**: When it predicts yes, how often is it correct?
  - TP/predicted yes = 61/77 = 0.79
- **Prevalence**: How often does the yes condition actually occur in our sample?
  - actual yes/total = 80 / 160 = 0.5

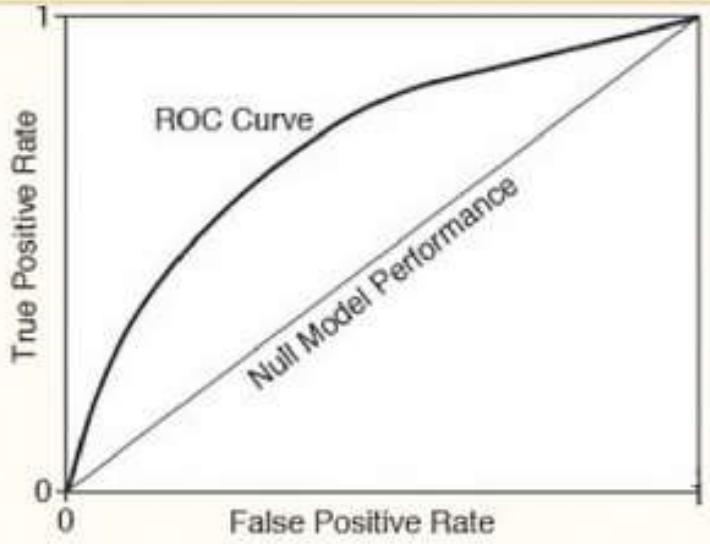
PREDICTED	ACTUAL		
N=160	YES	NO	
YES	61	16	77
NO	19	64	83
	80	80	
	Sensitivity	Specificity	

Evaluation of models

- **Positive predictive value** is the probability that subjects with a positive screening test truly have the disease.
- **Negative predictive value** is the probability that subjects with a negative screening test truly don't have the disease.

PREDICTED	ACTUAL			
N=160	YES	NO		
YES	61	16	77	Positive Predictive Value
NO	19	64	83	Negative Predictive Value
	80	80		
	Sensitivity	Specificity		

Confusion Matrix



$$\text{False Positive Rate} = \frac{b}{b + d}$$

The false positive rate shows the proportion of negatives (NO responses) incorrectly identified as positive. We want this to be low.

$$\text{True Positive Rate} = \frac{a}{a + c}$$

The true positive rate, also called sensitivity, shows the proportion of positives (YES responses) that are correctly identified as positive. We want this to be high.

The receiver operating characteristic (ROC) curve shows the performance of a binary classifier across the full range of decision criteria or cutoffs. Perfect performance would correspond to the point (0,1), that is, a false positive rate of 0 and a true positive rate of 1. The area under the curve provides a general index of classification performance.

Confusion Matrix

Multi output

		Reference Class				Total
		A	B	C	D	p_{i+}
Mapped Class	A	35	14	11	1	61
		0.2147	*0.0859*	*0.0675*	*0.0061*	*0.3742*
	B	4	11	3	0	18
		0.0245	*0.0675*	*0.0184*	*0.0000*	*0.1104*
	C	12	9	38	4	63
		0.0736	*0.0552*	*0.2331*	*0.0245*	*0.3865*
	D	2	5	12	2	21
		0.0123	*0.0307*	*0.0736*	*0.0123*	*0.1288*
Total		53	39	64	7	163
p_{+j}		*0.3252*	*0.2393*	*0.3926*	*0.0429*	*1.0000*

Table 3: Example of a confusion matrix

confusion_matrix

- from sklearn.metrics import confusion_matrix

confusion_matrix(actual, predicted)

```
>>> confusion_matrix(y_test,y_pred)
array([[13,  0,  0],
       [ 0, 15,  1],
       [ 0,  3,  6]], dtype=int64)
```