MEHRAN UNIVERSITY
OF ENGINEERING & TECHNOLOGY
JAMSHORO, PAKISTAN

# Example - Confusion Matrix-based ML Algorithm Evaluation

- Add problem definition
- Define Confusion matrix values, (True / False - predicted and observed values )
- 6 measures derived from the above values and their solution

# Confusion matrix example

following matrix represents the results of a *Logistic Regression model* predicting whether a patient will have heart disease after 10 years or not The training data consists of 751 observations.

- **Predict variable (desired target):**
  - 10 year risk of coronary heart disease CHD (binary: “1”, means “Yes”, “0” means “No”)

## Logistic Regression

is a type of regression analysis in statistics used for prediction of the outcome of a categorical dependent variable from a set of predictor or independent variables. In logistic regression, the dependent variable is always binary. Logistic regression is mainly used for prediction and also for calculating the probability of success.

```
from sklearn.linear_model import LogisticRegression
logreg=LogisticRegression()
logreg.fit(x_train,y_train)
y_pred=logreg.predict(x_test)
```

# Model Evaluation

## Model accuracy

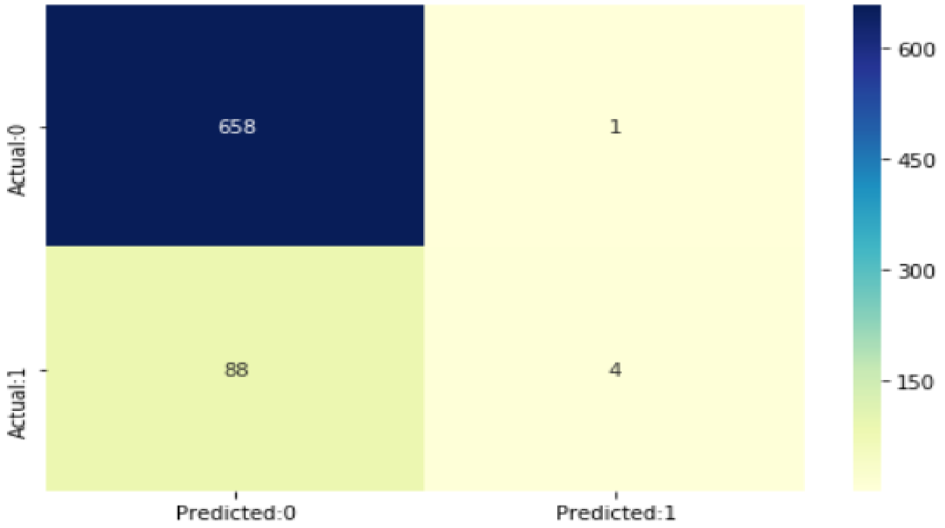
```
sklearn.metrics.accuracy_score(y_test,y_pred)
```

**Accuracy of the model is 0.88**

What if  there is a class imbalance due to which the model predicts the majority class for all input coming in and hence has a high accuracy score? So we evaluate the accuracy of the model using

## Confusion matrix

```
Out[20]:
        <matplotlib.axes._subplots.AxesSubplot at 0x7fe729d357f0>
```



| | N= 751 | **PREDICTED** | | **TOTAL** |
|---|---|---|---|---|
| | | *No* | *Yes* | |
| **ACTUAL** values | *Negative* | TN =658 | FP=1 | 659 |
| | *Positive* | FN=88 | TP=4 | 92 |
| **TOTAL** | | Predicted risk of no heart disease=746 | Predicted risk of heart disease=5 | —------ |

### *CONFUSION MATRIX VALUES*

*The confusion matrix shows 658+4 = 662 correct predictions and 88+1= 89 incorrect ones.*

**True Positives:** 4

the patient has the symptoms and the model classified correctly means it predicted yes they will have the disease

**True Negatives:** 658

Model predicted no, and they don't have the disease.

**False Positives:** 1 (*Type I error*)

predicted yes, but they don't actually have the disease.

**False Negatives:** 88 ( *Type II error*)

predicted no, but they actually have the disease.

Is this a good model? Will figure this out by calculating important classification model evaluation metrics derived from confusion matrix values.

## Model Evaluation - Statistics

```
from sklearn.metrics import classification_report,confusion_matrix
print("Classification Report:\n", classification_report(y_test,rfc_pred))
```

### #1. Recall/Sensitivity

Recall or Sensitivity or True Positive Rate (TPR) or Probability of Detection is the ratio of the correct positive predictions (TP) to the total positives (i.e., TP and FN).

```
Sensitivity or True Positive Rate = TP/(TP+FN) = 4 ÷( 4 + 88 )= 89
```

Use Recall when knowing the false negatives is important. For example, if a person has multiple blockages in the heart and the model shows he is absolutely fine, it could prove to be fatal.

### #2. Precision

is the measure of the correct positive results out of all the positive results predicted, including both true and false positives.

```
Positive Predictive value = TP/(TP+FP) =  0.8
```

### #3. Specificity

or True Negative Rate (TNR)

```
Specificity or True Negative Rate = TN/(TN+FP) =
```

It is a measure of how well your classifier is identifying the negative values.

### #4. Accuracy

Accuracy is the number of correct predictions out of the total number of predictions.

The accuracy of the model = TP+TN/(TP+TN+FP+FN) =

### #5.Error Rate or misclassification rate

The Misclassification = 1-Accuracy =

### #6.False positive rate

FP/(FP+TN)=

# References

https://www.kaggle.com/code/neisha/heart-disease-prediction-using-logistic-regression/notebook
https://www.youtube.com/watch?v=prWyZhcktn4

https://geekflare.com/confusion-matrix-in-machine-learning/