Logistic Regression

Sample code is in **logistic-example1.r**
Sample code is in logistic-example2.r

- Accuracy evaluation of logistic regression are done by

Confusion Matrix

- A **confusion matrix** is a matrix that summarizes the performance of a machine learning model on a set of test data.
- It is often used to measure the performance of classification models, which aim to predict a categorical label for each input instance.
- The matrix displays the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) produced by the model on the test data.
- For binary classification, the matrix will be of a 2X2 table, For multi-class classification, the matrix shape will be equal to the number of classes i.e for n classes it will be nXn.

A 2X2 Confusion matrix is shown below for the image recognization having a Dog image or Not Dog image.

		Actual	
		Dog	**Not Dog**
Predicted	**Dog**	True Positive (TP)	False Positive (FP)
	Not Dog	False Negative (FN)	True Negative (TN)

- **True Positive (TP):** It is the total counts having both predicted and actual values are Dog.
- **True Negative (TN):** It is the total counts having both predicted and actual values are Not Dog.
- **False Positive (FP):** It is the total counts having prediction is Dog while actually Not Dog.
- **False Negative (FN):** It is the total counts having prediction is Not Dog while actually, it is Dog.

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Actual	Dog	Dog	Dog	Not Dog	Dog	Not Dog	Dog	Dog	Not Dog	Not Dog
Predicted	Dog	Not Dog	Dog	Not Dog	Dog	Dog	Dog	Dog	Not Dog	Not Dog
Result	TP	FN	TP	TN	TP	FP	TP	TP	TN	TN

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- True Positive Counts = 5
- False Positive Counts = 1
- True Negative Counts = 3
- False Negative Counts = 1

Confusion matrix of above data set is

		Actual	
		Dog	**Not Dog**
Predicted	**Dog**	True Positive (TP =5)	False Positive (FP=1)
	Not Dog	False Negative (FN =1)	True Negative (TN=3)

From the confusion matrix, we can find the following metrics

1. **Accuracy:** Accuracy is used to measure the performance of the model. It is the ratio of Total correct instances to the total instances.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

- For the above case:
- Accuracy = (5+3)/(5+3+1+1) = 8/10 = 0.8

2. **Precision:** Precision is a measure of how accurate a model’s positive predictions are. It is defined as the ratio of true positive predictions to the total number of positive predictions made by the model

$$\text{Precision} = \frac{TP}{TP+FP}$$

For the above case:
Precision = 5/(5+1) =5/6 = 0.8333

3. Recall: Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. It is the ratio of the number of true positive (TP) instances to the sum of true positive and false negative (FN) instances.

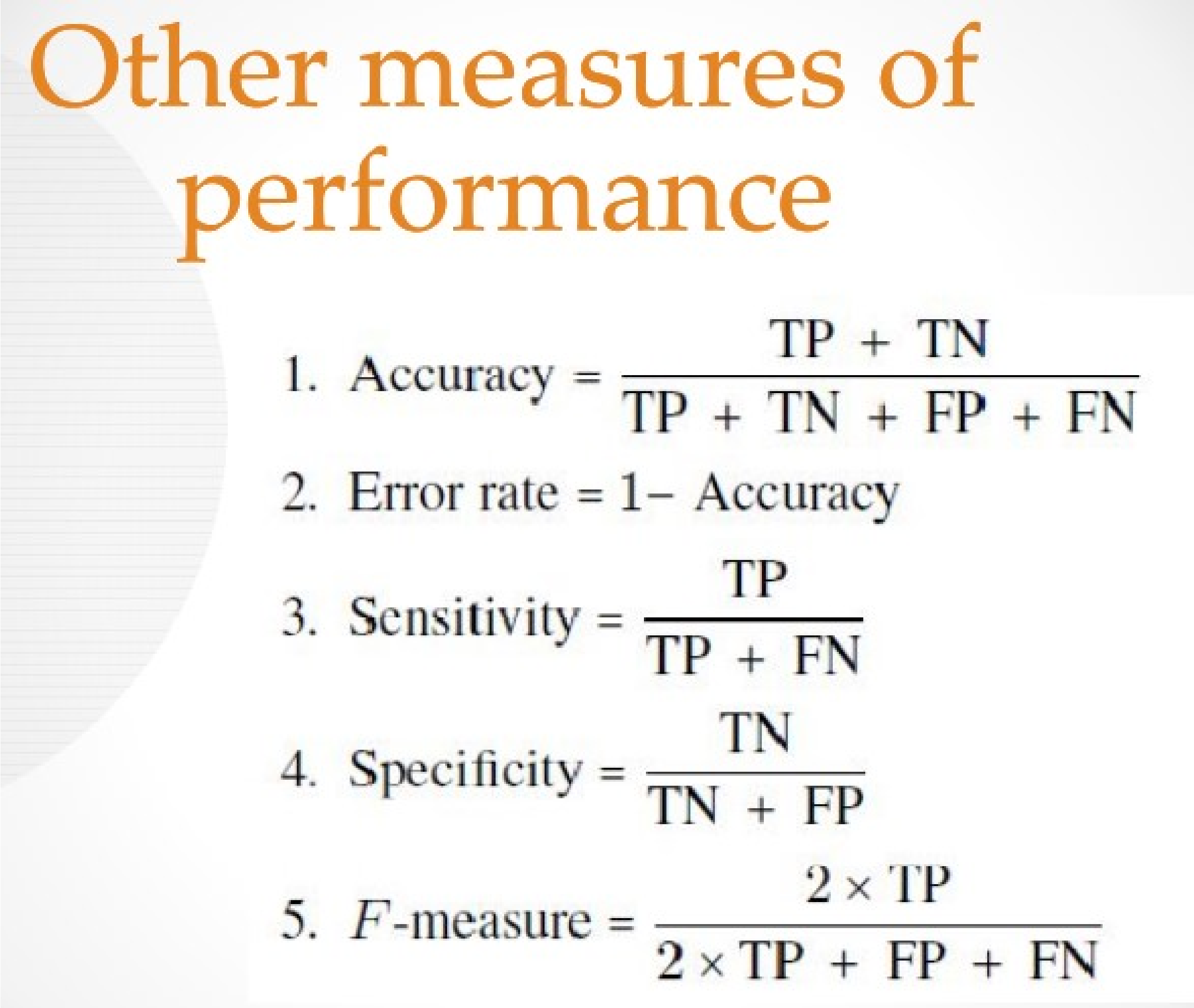
$$\text{Recall} = \frac{TP}{TP+FN}$$

For the above case:
Recall = 5/(5+1) =5/6 = 0.8333

4. **F1-Score:** F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall,

$$\text{F1-Score} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

For the above case:
F1-Score: = (2* 0.8333* 0.8333)/(0.8333+ 0.8333) = 0.8333



Other measures of performance

1. $\text{Accuracy} = \dfrac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$

2. $\text{Error rate} = 1 - \text{Accuracy}$

3. $\text{Sensitivity} = \dfrac{\text{TP}}{\text{TP} + \text{FN}}$

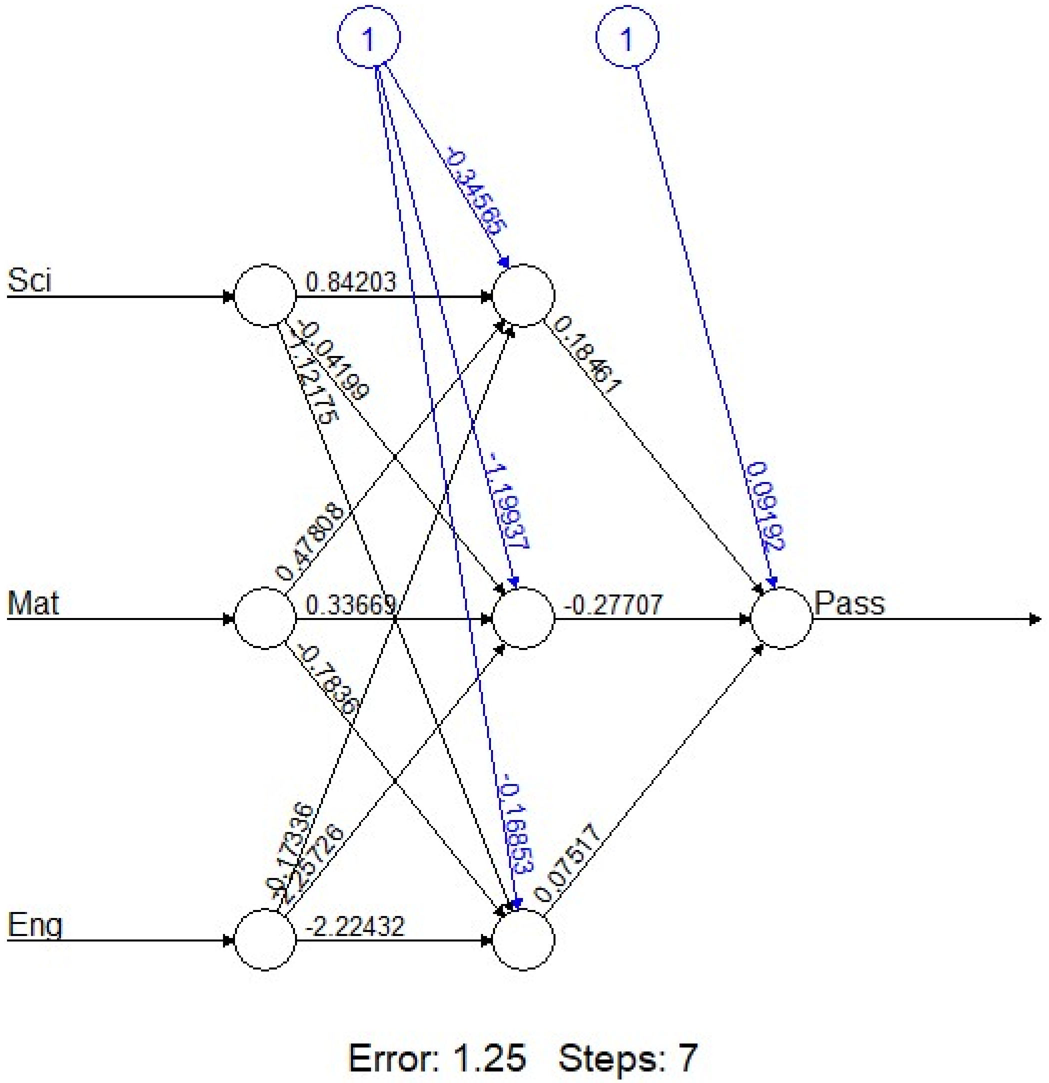
4. $\text{Specificity} = \dfrac{\text{TN}}{\text{TN} + \text{FP}}$

5. $F\text{-measure} = \dfrac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}$

Neural network model

- How to train a neural network in R?
  - Step 1: Define the training set.
  - Step 2: Install and load neuralnet package
  - Step 3: Fit neural network
  - Step 4: Plot the neural network
  - Step 5: Create a test dataset
  - Step 6: Predict results for the test dataset

Sample code is in **nn-example.r**


1
1
Sci
Mat
Eng
Pass
0.84203
-0.04199
-1.12175
0.47808
0.33669
-0.7836
-0.17336
-2.25726
-2.22432
-0.34565
-1.19937
-0.16853
0.18461
-0.27707
0.07517
0.09192
Error: 1.25 Steps: 7