

The Model which is already fitted:

		Predicted Exit	
		0	1
Actual Exit	0	^{TN} 1,338	^{FP} 2,270
	1	^{FN} 175	^{TP} 2,495

TN - True negatives: we predicted a costumer would not churn and they did not churn.

FP - False positives: We predicted a costumer would churn, but they did not.

FN - False negatives: we predicted a costumer would not churn and they churn.

TP - True positives: we predicted a costumer churn, and they actually churn.

Precision

Another way to analyze a classification algorithm is by calculating precision, which is basically obtained by dividing true positives by the sum of true positive and false positive, as shown below:

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall

Recall is calculated by dividing true positives by the sum of the true positive and false negative, as shown below:

$$\text{Recall} = \frac{tp}{tp + fn}$$

F1 Measure

F1 measure is simply the harmonic mean of precision and recall and is calculated as follows:

$$F_1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Accuracy

Accuracy refers to the number of correctly predicted labels divided by the total number of observations in a dataset.

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$\text{Precision} = 2495 / (2495 + 2270) = 0.52$$

$$\text{Recall/Sensitivity} = 2495 / (2495 + 175) = 0.93$$

$$\text{F1 Score} = 0.67$$

$$\text{Accuracy} = (2495 + 1338) / (2495 + 1338 + 2270 + 175) = 0.61$$

$$\text{Specificity} = 1338 / (1338 + 2270) = 0.37$$

The existing model is pretty good because of the following reason

1. The model will catch 93% of the customers who will actually churn.
2. Overall all accuracy is 61%

But there is still scope to achieved better accuracy with other ML deep learning models.

I have tested:

- Primal logistic regression
- Logistic Regression with pol 2 Kernel
- SVM with RBF Kernel
- Random Forest classifier with GINI index
- Extreme Gradient Boost

With Random Forest I got better accuracy and F1 Score:

```
: print(confusion_matrix(df_train.Exited, RF.predict(df_train.loc[:, df_train.columns != 'Exited'])))  
print(classification_report(df_train.Exited, RF.predict(df_train.loc[:, df_train.columns != 'Exited'])))
```

```
[[5849 896]  
 [1242 3730]]  
      precision    recall  f1-score   support  
  
     0       0.82      0.87      0.85       6745  
     1       0.81      0.75      0.78       4972  
  
 accuracy          0.82  
 macro avg       0.82      0.81      0.81  
 weighted avg    0.82      0.82      0.82
```

The Confusion Matrix shows that for 82 percent of the records in the test set, Random Forest correctly predicted whether a customer will leave the bank.