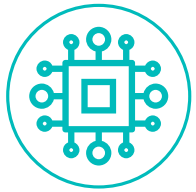
# Bank Note Authentication- Approach

**Nityanandan P**

# THE OBJECTIVE

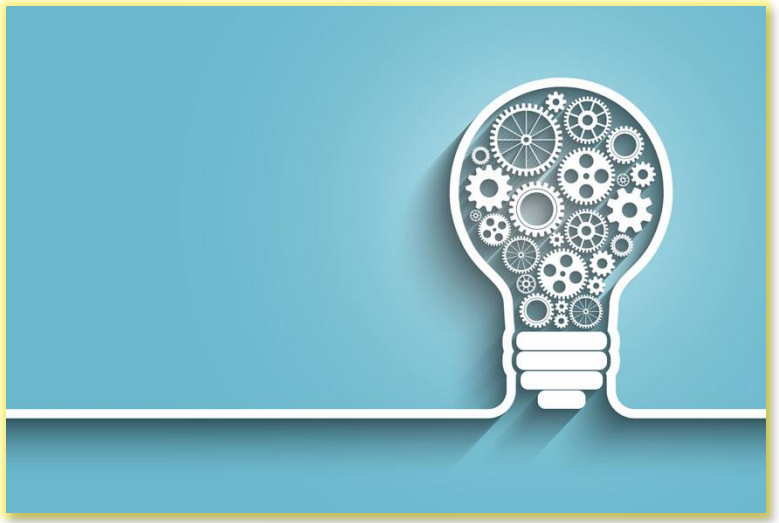
Develop a classification model to detect if a bank note is a fake one or a legitimate one using CART.



Challenge: The data contains only 4 attributes and all are numerical. There are no missing values also.

I have handled the data here using multiple python libraries.

# APPROACH



Problem Definition:

- Predicting whether a bank note is real or fake using CART.

Data Exploration:

- There are no missing data and I have just visualized the data.

Data Preparation:

- Feature scaling done to normalize the data

Splitting the data:

- The data is split into train and test sets

Modelling:

- Logistic Regression
- Decision Tree Classifier
- Decision Tree using Gini
- Decision Tree using Entropy
- Random Forest Classifier

Validation:

- Accuracy
- Precision
- Recall
- F1 score
- 5 fold cross validation
- Logloss

# DATA EXPLORATION

- There are no missing data. The dataset is clean and balanced. The one thing that I had to do was to add column names as the data was a .txt file without any column names.

| | variance | skewness | kurtosis | entropy | class |
|---|---|---|---|---|---|
| **0** | 3.62160 | 8.66610 | -2.8073 | -0.44699 | 0 |
| **1** | 4.54590 | 8.16740 | -2.4586 | -1.46210 | 0 |
| **2** | 3.86600 | -2.63830 | 1.9242 | 0.10645 | 0 |
| **3** | 3.45660 | 9.52280 | -4.0112 | -3.59440 | 0 |
| **4** | 0.32924 | -4.45520 | 4.5718 | -0.98880 | 0 |
| **...** | ... | ... | ... | ... | ... |
| **1367** | 0.40614 | 1.34920 | -1.4501 | -0.55949 | 1 |
| **1368** | -1.38870 | -4.87730 | 6.4774 | 0.34179 | 1 |
| **1369** | -3.75030 | -13.45860 | 17.5932 | -2.77710 | 1 |
| **1370** | -3.56370 | -8.38270 | 12.3930 | -1.28230 | 1 |
| **1371** | -2.54190 | -0.65804 | 2.6842 | 1.19520 | 1 |

No missing values:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   variance  1372 non-null   float64
 1   skewness  1372 non-null   float64
 2   kurtosis  1372 non-null   float64
 3   entropy   1372 non-null   float64
 4   class     1372 non-null   int64  
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

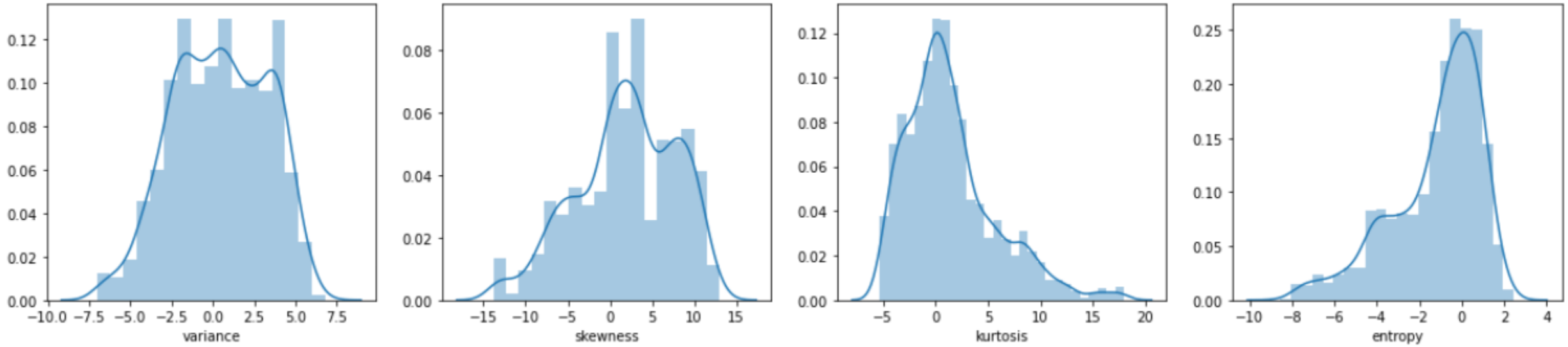
# DATA EXPLORATION

- Since there are no missing data and the dataset is a balanced one, I just went on with Exploratory data Analysis and visualization.

Balanced data:

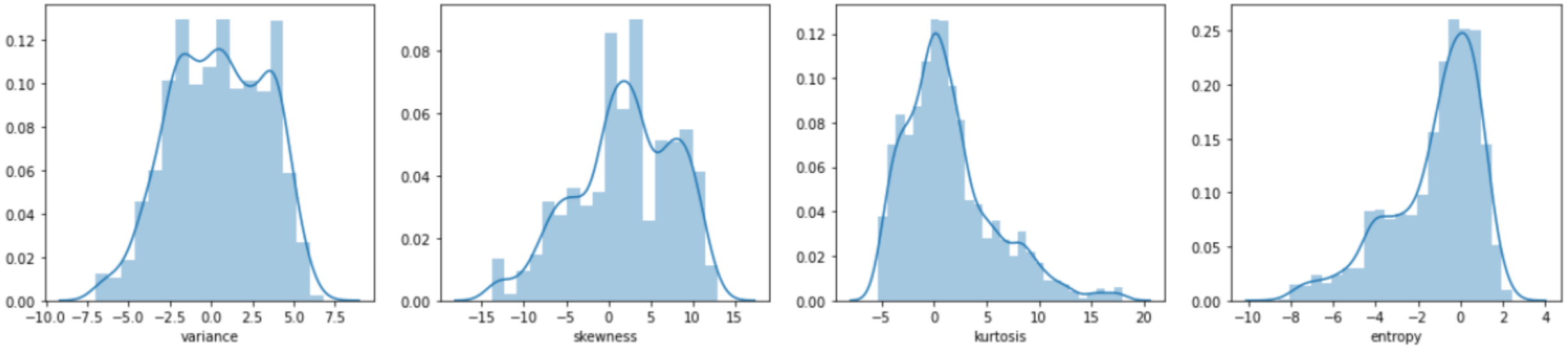
```
0    762
1    610
Name: class, dtype: int64
```

Distribution of the attributes:



# DATA EXPLORATION

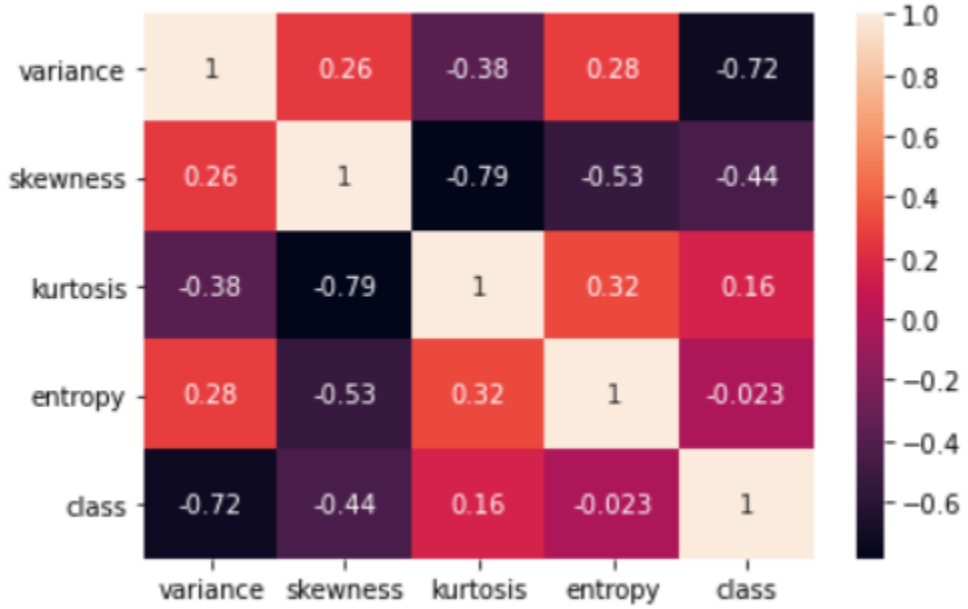
- From the graphs the following inferences were made:




  - Variance: Normally distributed – No skew
  - Skewness: Normally distributed – No skew
  - Kurtosis: Positive skew
  - Entropy: Negative skew

# DATA EXPLORATION

- Following is a heatmap showing how the independent variables are related to the target variable:




- This shows that only kurtosis is positively related to class, meaning as kurtosis increases, there is a higher possibility of a bank note being genuine.

# DATA PREPARATION

- The data was normalized using StandardScaler from Sci-kit Learn because the range of the attributes were different. This would help fit the data into a model easily.
- When we look at the graphs in the previous slide, we can see that the variance ranges from -10 to 7.5, skewness from -15 to 15, kurtosis from -5 to 20 and entropy from -10 to 4.
- Since the data has a lot of negative values and some attributes are skewed StandardScaler plays an important role by scaling the data such that it is centred around 0.

# MODELLING AND VALIDATION

- Logistic Regression:

```
[[150   9]
 [  0 116]]
              precision    recall  f1-score   support

           0       1.00      0.94      0.97       159
           1       0.93      1.00      0.96       116

    accuracy                           0.97       275
   macro avg       0.96      0.97      0.97       275
weighted avg       0.97      0.97      0.97       275
```

- Cross Validation:

```
[0.98636364 0.99090909 0.98173516 0.99543379 0.97260274]
```

# MODELLING AND VALIDATION

- Decision Tree Classifier:

```
[[157   2]
 [  3 113]]
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       159
           1       0.98      0.97      0.98       116

    accuracy                           0.98       275
   macro avg       0.98      0.98      0.98       275
weighted avg       0.98      0.98      0.98       275
```

- Cross Validation:

```
[0.99090909 0.98636364 0.98173516 0.98630137 0.98173516]
```

# MODELLING AND VALIDATION

- Decision Tree Classifier Using Gini:

```
[[149  10]
 [ 13 103]]
              precision    recall  f1-score   support

           0       0.92      0.94      0.93       159
           1       0.91      0.89      0.90       116

    accuracy                           0.92       275
   macro avg       0.92      0.91      0.91       275
weighted avg       0.92      0.92      0.92       275
```

- Log Loss:

```
The log loss score is: 2.888726738389992
```

# MODELLING AND VALIDATION

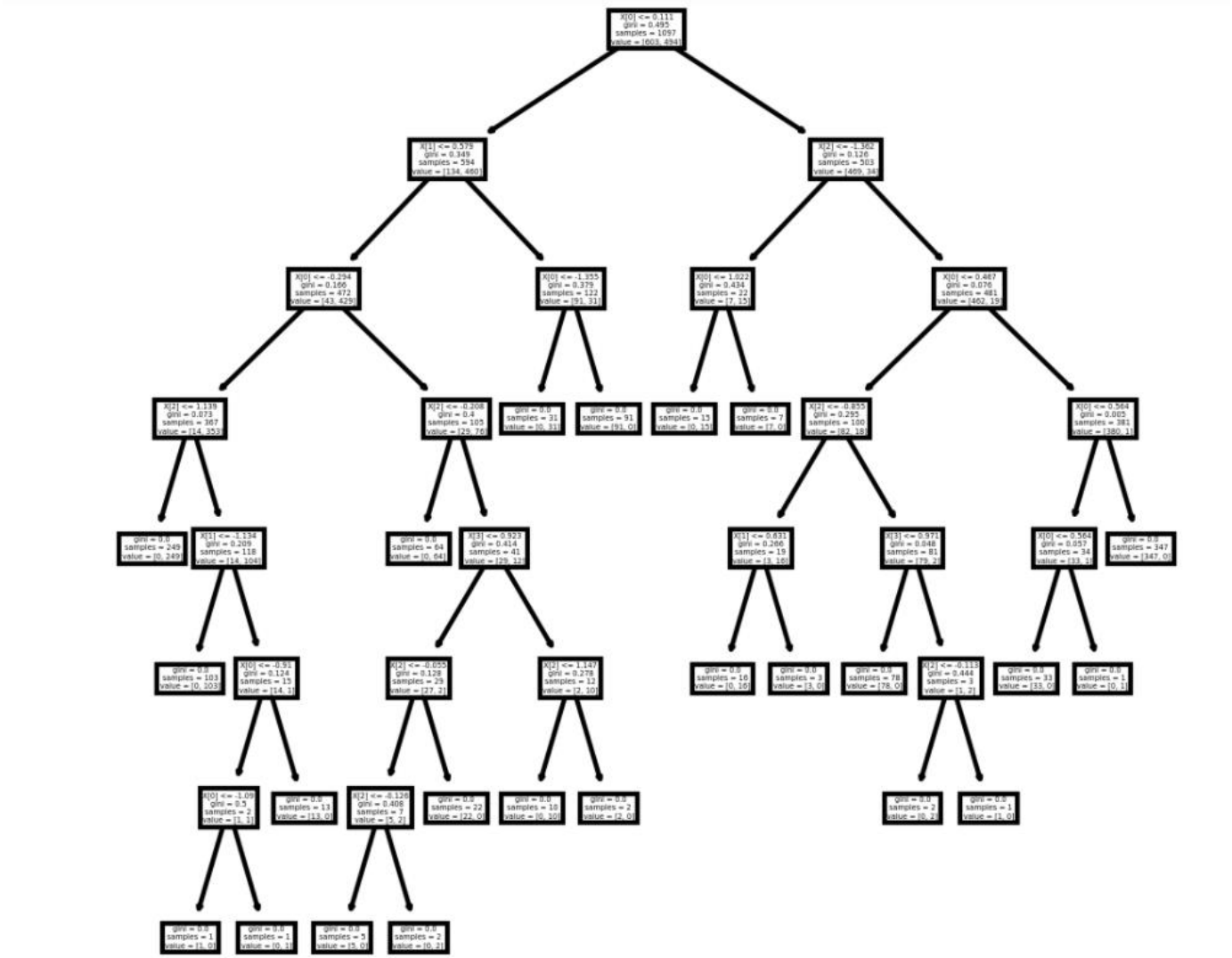
- Decision Tree Classifier Using Entropy:

```
[[154   5]
 [  1 115]]
              precision    recall  f1-score   support

           0       0.99      0.97      0.98       159
           1       0.96      0.99      0.97       116

    accuracy                           0.98       275
   macro avg       0.98      0.98      0.98       275
weighted avg       0.98      0.98      0.98       275
```

- Log Loss:

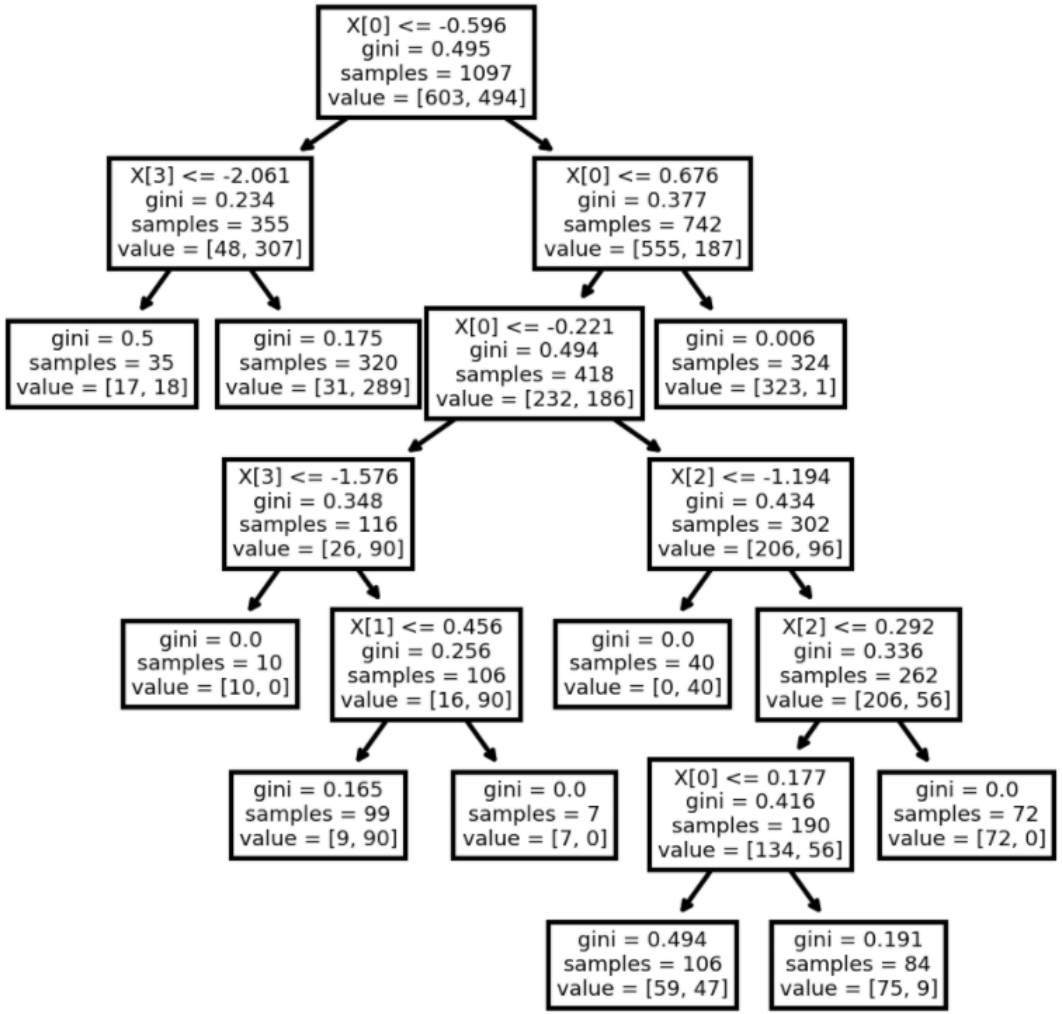
```
The log loss score is: 0.753587841296783
```

# DECISION TREE – BEFORE PRUNING



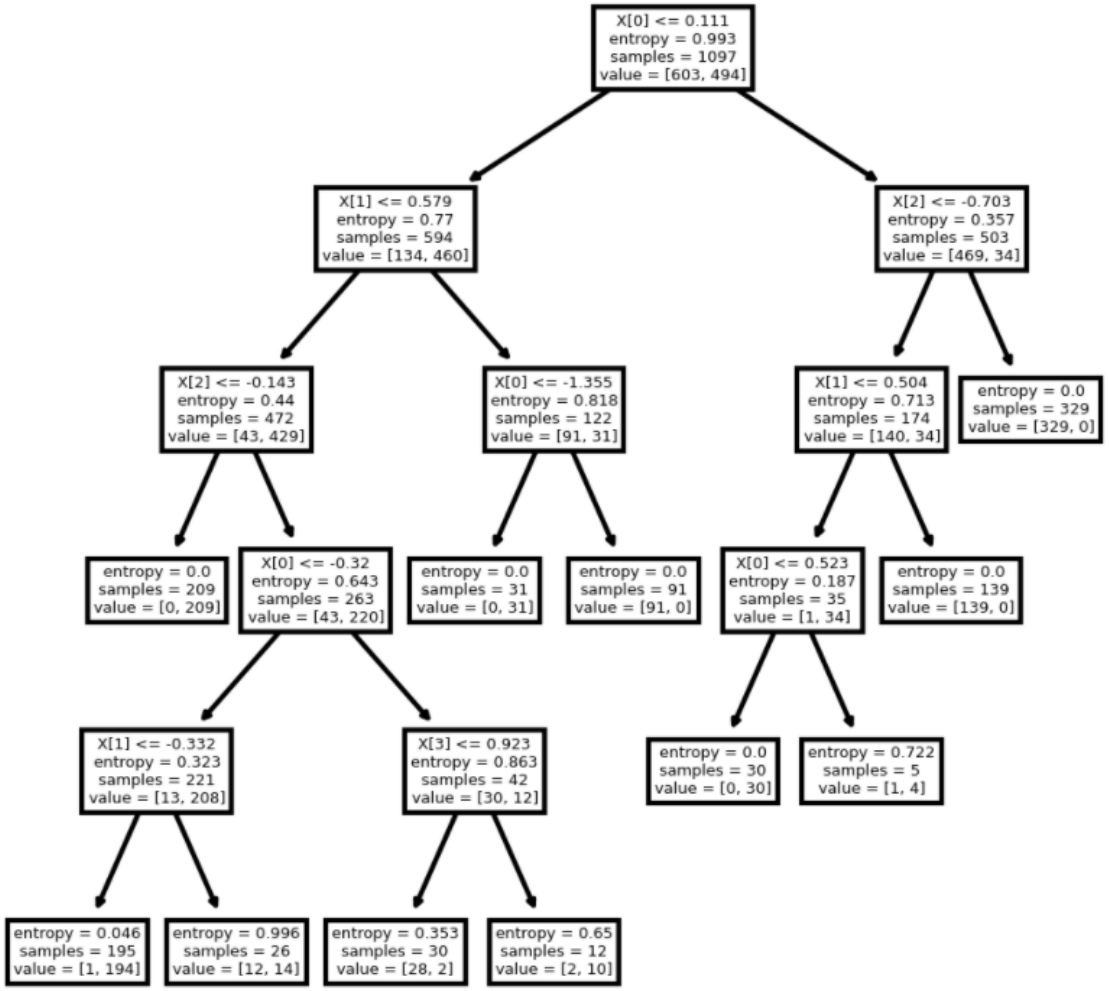
# DECISION TREE – AFTER PRUNING - GINI

We can see that the nodes have been reduced after pruning.



# DECISION TREE – AFTER PRUNING - ENTROPY

We can see that the nodes have been reduced after pruning.



# MODELLING AND VALIDATION

- Random Forest Classifier:

```
[[158   1]
 [  1 115]]
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       159
           1       0.99      0.99      0.99       116

    accuracy                           0.99       275
   macro avg       0.99      0.99      0.99       275
weighted avg       0.99      0.99      0.99       275
```

- Cross Validation:

```
[1.         0.99545455 1.         0.98630137 0.99543379]
```

# SORTING MODELS AS PER THE ACCURACY

| | Model | Score |
|---|---|---|
| 4 | Random Forest | 99.272727 |
| 1 | Decision Tree | 98.181818 |
| 3 | Decision Tree Entropy | 97.818182 |
| 0 | Logistic Regression | 96.727273 |
| 2 | Decision Tree Gini | 91.636364 |

- We can see that the accuracy of Random Forest Classifier is better than Decision Tree Classifier and Logistic Regression. Random Forest Classifier is a collection of decision trees and they limit overfitting, so they are typically more accurate than single decision trees.

- Decision Tree Classifier is better than Logistic Regression here because, the latter just fits a best line to divide the space into two, but the decision tree can bisect into multiple smaller regions.

- Here the Decision Tree with Entropy criterion performs well because of the short depth and Decision Tree with Gini criterion performs well because the depth is more.

# REFERENCES


- https://towardsdatascience.com/decision-tree-build-prune-and-visualize-it-using-python-12ceee9af752
- https://stackoverflow.com/questions/59365994/how-to-create-column-names-from-txt-files-in-pandas-dataframe
- https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

# THANK YOU