Text

Description automatically generated

**Informatics Institute of Technology Department of Computing**

**Bsc(Hons) Artificial Intelligence**

**and Data Science**

**Module: CM2604 Machine Learning**

**Module Coordinator:  Mr. Prasan Yapa**

**Coursework Report**

Vinuwara Ronath Jayasuriya

RGU ID – 2119942

IIT Student No. – 20210167

Spam dataset is done using the Decision Trees Classification and K Nearest Neighbors Classification (KNN)

Preprocessing techniques

**Corpus Preparation**

**Data Cleaning**: This was done by removing duplicates and null values from the dataset.

Removing duplicate in dataset.

|  |
| --- |
| Getting the duplicates in the dataset |
| Data\_set.duplicated() |
| 0 False  1 False  2 False  3 False  4 False  ...  4596 False  4597 False  4598 False  4599 False  4600 False  Length: 4601, dtype: bool |
| removing the duplicate values |
| Data\_set.drop\_duplicates(inplace=True) |

Removing null values

|  |
| --- |
| Data\_set.isna().sum() |
| word\_freq\_make 0  word\_freq\_address 0  word\_freq\_all 0  word\_freq\_3d 0  word\_freq\_our 0  word\_freq\_over 0  word\_freq\_remove 0  word\_freq\_internet 0  word\_freq\_order 0  word\_freq\_mail 0  word\_freq\_receive 0  word\_freq\_will 0  word\_freq\_people 0  word\_freq\_report 0  word\_freq\_addresses 0  word\_freq\_free 0  word\_freq\_business 0  word\_freq\_email 0  word\_freq\_you 0  word\_freq\_credit 0  word\_freq\_your 0  word\_freq\_font 0  word\_freq\_000 0  word\_freq\_money 0  word\_freq\_hp 0  word\_freq\_hpl 0  word\_freq\_george 0  word\_freq\_650 0  word\_freq\_lab 0  word\_freq\_labs 0  word\_freq\_telnet 0  word\_freq\_857 0  word\_freq\_data 0  word\_freq\_415 0  word\_freq\_85 0  word\_freq\_technology 0  word\_freq\_1999 0  word\_freq\_parts 0  word\_freq\_pm 0  word\_freq\_direct 0  word\_freq\_cs 0  word\_freq\_meeting 0  word\_freq\_original 0  word\_freq\_project 0  word\_freq\_re 0  word\_freq\_edu 0  word\_freq\_table 0  word\_freq\_conference 0  char\_freq\_; 0  char\_freq\_( 0  char\_freq\_[ 0  char\_freq\_! 0  char\_freq\_$ 0  char\_freq\_# 0  capital\_run\_length\_average 0  capital\_run\_length\_longest 0  capital\_run\_length\_total 0  spam 0  dtype: int64 |
| Removing the target column |
| data=Data\_set.drop(labels=['spam'], axis=1)  data.head() |

**Data Transformation:** outliers were removed using standard scaling.

Diagram

Description automatically generated

Since capital\_run\_lenghth\_avarage,capital\_run\_length\_longest and Capital\_run\_lenghth\_total are outliers

Chart, box and whisker chart

Description automatically generated

Remove the outliers using the IQR method

code

|  |
| --- |
| import numpy as np  # Calculate the interquartile range for each column  Q1 = Data\_set.quantile(0.25)  Q3 = Data\_set.quantile(0.75)  IQR = Q3 - Q1  # Remove the outliers using the IQR method  Data\_set\_outliers\_removed = Data\_set[~((Data\_set < (Q1 - 1.5 \* IQR)) | (Data\_set > (Q3 + 1.5 \* IQR))).any(axis=1)]  # Create individual box plots for the columns with outliers removed  fig, axs = plt.subplots(1, 3, figsize=(15, 5))  axs[0].boxplot(Data\_set\_outliers\_removed['capital\_run\_length\_total'], vert=False)  axs[0].set\_title('Capital Run Length Total (outliers removed)')  axs[0].set\_xlabel('')  axs[0].set\_xlim([0, 600])  axs[1].boxplot(Data\_set\_outliers\_removed['capital\_run\_length\_average'], vert=False)  axs[1].set\_title('Capital Run Length Average (outliers removed)')  axs[1].set\_xlabel('')  axs[1].set\_xlim([0, 10])  axs[2].boxplot(Data\_set\_outliers\_removed['capital\_run\_length\_longest'], vert=False)  axs[2].set\_title('Capital Run Length Longest (outliers removed)')  axs[2].set\_xlabel('')  axs[2].set\_xlim([0, 50])  plt.show() |

Chart, box and whisker chart

Description automatically generated

Used Standard Scaler on the dataset

Before standard scaler

Diagram

Description automatically generated

A picture containing diagram

Description automatically generatedMean

Code – performing standard scaler on the dataset

|  |
| --- |
| # Perform Standard Scaling on the data  scaler = StandardScaler()  scaled\_data = scaler.fit\_transform(Data\_set.iloc[:, :-1])  Data\_scaled = pd.DataFrame(data=scaled\_data, columns=Data\_set.columns[:-1]) |

Boxplot after performing the standard scaler on the dataset

Chart, scatter chart

Description automatically generated

Chart

Description automatically generatedMean

Variance graph

Chart, line chart

Description automatically generated

Performing PCA to the Dataset

Code

|  |
| --- |
| pca = PCA(n\_components=44)  new\_data = pca.fit\_transform(Data\_set)  # The new data fed to the algorithm.  principal\_Df = pd.DataFrame(data = new\_data, columns = [  'PC1', 'PC2','PC3','PC4','PC5','PC6','PC7','PC8','PC9','PC10',  'PC11','PC12','PC13','PC14','PC15','PC16','PC17','PC18','PC19','PC20',  'PC21', 'PC22','PC23','PC24','PC25','PC26','PC27','PC28','PC29','PC30',  'PC31','PC32','PC33','PC34','PC35','PC36','PC37','PC38','PC39','PC40',  'PC41','PC42','PC43','PC44'  ]) |

Splitting the dataset into the Training set and Test set

* 20 percent of the dataset was used to test the dataset.
* 80 percent of the dataset was used to train the dataset

Code

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.20, random\_state = 0) |

Decision Tree Classification

**Accuracy of testing dataset** : 0.9560570071258907

Summary of the dataset

code

|  |
| --- |
| from sklearn.metrics import classification\_report  predicted\_labels = clf.predict(X\_test)  classification\_report = classification\_report(y\_test, predicted\_labels)  print(classification\_report) |

Classification report

Table

Description automatically generated with low confidence

**Accuracy of the training dataset of the decision tree**: 1.0

Decision tree

Chart

Description automatically generated

Confusion Matrix before pruning

Equations

True positive rate: measures the proportion of positive instances correctly identified as positive.

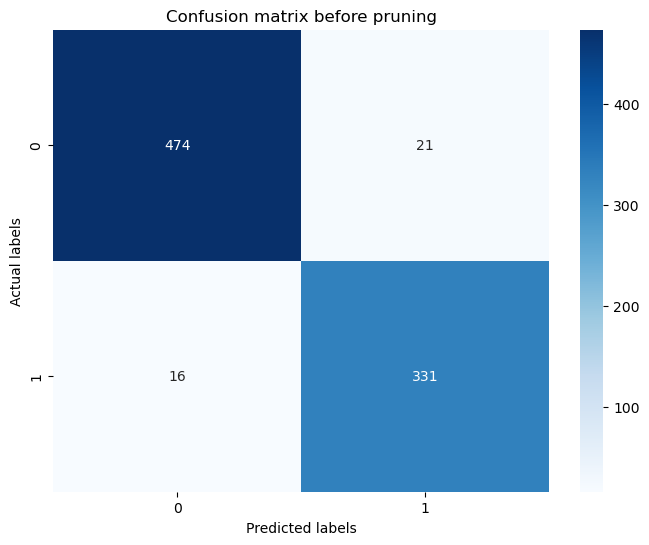
* FN = False Negative : Number of positive instances that are incorrectly classified as positive.

TPR = TP / (TP + FN)

False positive rate : measures the proportion of negative instances incorrectly identified as positive.

* TN = true negative : The number of negative instances that are correctly classified as negative

FPR = FP / (FP + TN

**Experiments**

* TP :474 mails are correctly identified as not spam.
* FP : 21 mails are incorrectly identified as spam as not spam.
* TN :331 mails are correctly identified as spam.
* FN : 16 mails are incorrectly identified as not spam as spam.

**Receiver Operating Characteristics (ROC)** : A plot to test sensitivity as the y coordinate versus its 1-specificity or false positive rate (FPR) as the x coordinate, is an effective method of evaluating the performance of diagnostic tests.

Chart, line chart

Description automatically generated

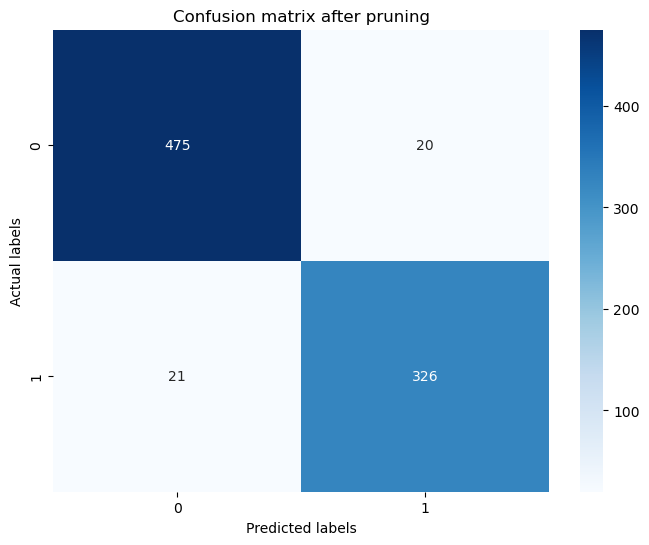
AUC score for the code: 0.96

**Pruning the Decision tree** : Pruning is a technique that removes the parts of the Decision Tree which prevent it from growing to its full depth. The parts that it removes from the tree are the parts that do not provide the power to classify instances.

A graph is plotted between ‘Total impurity of leaves’ as Y axis and ‘Effective alpha’ as the X axis. Using the graph, we can find the optimal ‘ccp alpha’ value required for pruning.The regularization parameter ccp alpha balances the accuracy and model complexity in decision trees.

Chart

Description automatically generated



* TP :475 mails are correctly identified as not spam.
* FP : 20 mails are incorrectly identified as spam as not spam.
* TN :326 mails are correctly identified as spam.
* FN : 21 mails are incorrectly identified as not spam as spam.

Table

Description automatically generated

Decision Tree obtaining after pruning.

Diagram

Description automatically generated

K Nearest Neighbors (KNN) Classification

Classification Report of KNN

Table

Description automatically generated

Confusion matrix of KNN

Chart

Description automatically generated

* TP: 480 mails are correctly identified as not spam.
* FP: 51 mails are incorrectly identified as spam as not spam.
* TN: 296 mails are correctly identified as spam.
* FN: 35 mails are incorrectly identified as spam as spam.

KNN Accuracy: 0.8729216152019003

**Limitations & Enhancements**

Limitations

* Due to the dataset being small the accuracy of the model would be effected.
* Feature engineering step may not have included all the relevant information, making the model performance suboptimal.

Future Enhancements

* Training the model with a larger dataset would increase the accuracy of it
* Utilizing other machine learning algorithms
  + Support vector machines - SVM
  + Random Forest
* Experiment with different feature engineering techniques