Main Report

The scope of this project is to classify if an email is a spam or not using ML techniques. A data cleaning, preparation and exploration is conducted before conducting several machine learning methods such as Naïve-Bayes Text Classification, KNN and Random Forest Classifications are used to make classifications. Parameter optimizations, confusion matrixes, and classification reports are presented for each method utilized. For each method used, a detailed explanation is provided in the following sections.

Data & Data Exploration

I. Data:

Project Dataset Link: https://www.kaggle.com/datasets/imdeepmind/preprocessed-trec-2007-public-corpus-dataset

The dataset used in this project is titled "TREC 2007 Public Corpus Dataset" and it is an email spam detection dataset. It contains 50199 spam emails and 25220 ham (not spam) emails.

Dataset is in CSV file. Email Dataset has 5 columns including:

1. label: This is the label for the email if it is 1 then spam else ham
2. subject: Subject of the email
3. email_to: Receiver of the email
4. email_from: Sender of the email
5. message: Email body

The size of the dataset is 75419 rows and 5 columns. A glimpse at the dataset can be found in the Appendix 1.

Throughout this project, SPAM emails will be referred as label 1 or SPAM and HAM emails will be referred as label 0 or HAM.

II. Data Exploration:

In addition to the following machine learning models, the following python packages are used to help with the analysis, data transformation, data exploration and visualization: numpy, pandas, matplotlib, sklearn, seaborn. The code segment for modules used can be found in Appendix 21.

To start data exploration and preparation, first the number of NA values in the "message" column are detected. Appendix 2 shows that there are 1487 rows of messages that have a NA value. These rows are omitted from the dataset since the number is low and in addition, these rows won't add any value to SPAM detection purpose.

Next, a histogram of email message lengths by labels are created. Most of the lengths of the emails are less than 10,000 words in label 0 (NOT SPAM (HAM)). The same is also valid for the label 1 emails (SPAM). (See Appendix 3)

In addition, the number of SPAM emails is 48714, and HAM emails is 25218; and a bar plot is demonstrated to show the balance. (See Appendix 5)

The final part of data exploration is to split the data into training and test sections. Using the sklearn's train_test_split module, data is split with test data size to be 20% of the data and random state is set to 111 so that when the model is reused, it creates the same test and train instances.

Data Analysis code segments can be found in Appendix 17.

Design Decisions & KDD process

I. Data Preparation & KDD Process:

The most critical part of the data preparation is to separate the message and label data from the original dataset and convert the message data to TFIDF values. This is a critical step since the TFIDF values will be used classification methods.

To turn the data to TFIDF matrix, sklearn's TfidVectorizer is used. Both the training and test texts are converted to TFDIF matrixes to be later used in the analysis. As a result of this transformation, train TFDIF is created. It is a <59145x2540134 sparse matrix of type '<class 'numpy.float64'>' with 11662811 stored elements in Compressed Sparse Row format>. The shape of the train text data is 59145 rows and 2540134 columns.

II. Design Decisions:

Using the TFDIF matrix, now the data is ready to be used in the models. Next part is to use this data to be used in models to train, predict and test. For the machine learning models, 3 unique tools are used:

1. Naïve Bayes Classification for Text
2. KNN
3. Random Forest

These methods are selected since their efficiency in text classification and proven technology.

1. Naïve Bayes Classification, specifically Multinomial Naïve Bayes (MNB) is used in this project. Naïve Bayes classification utilizes probability to determine the most likely classification of a given input. This method is generally used when the features of a data represent the frequency like the

TFIDF matrix and is proven to work well with text classification problems. One of the great features of MNB method is that it ignores the non-occurrences of the features, meaning that the features with 0 frequency are ignored and not used in the probability calculations.

2. KNN is a supervised machine learning algorithm that classifies a given input using the k closest neighbors which is calculated using distance metrics such as Euclidean, Manhattan, etc. KNN is a type of lazy learning, allowing KNN to be fast in training but slower and more expensive in testing.

3. Random Forest is another machine learning algorithm to accurately classify text since it supports large datasets. It utilizes multiple decision tree from randomly selected parts of the training data and combines the results of these decision trees to decide the outcome of a classification. It is a very effectively tool to use in this case.

Next, I will explain how the methods mentioned above are utilized, along with the evaluations and comparisons.

Methods & Results

I. Naïve Bayes Classification for Text

For the Naïve Bayes Classification, sklearn's MultinominalNB module is utilized. Alpha value of 1.5 is used. Alpha value is added to the denominator and numerator during the probability calculations to make everything non-zero. Alpha value results a smoothing.

Appendix 8 shows the Classification Report of the Naïve Bayes. The reported accuracy is **0.9864069791032664**, which is very high. After this analysis is completed and an overall accuracy of 0.99 is achieved, it was decided not to pursue any parameter tuning or optimizations.

In addition, Appendix 7 shows the Naïve Bayes Confusion Matrix. False Positive is 1.14%, False Negative is 0.22%, True Positive 33.05%, True Negative 65.59%.

Code segment for Naïve Bayes can be found in Appendix 18.

II. KNN

In order to use the best model for KNN, first the distance metric (Euclidean or Manhattan) must be chosen, in addition to the right value of K (the number of neighbors to classify the text object). To achieve this, two separate Error Rate vs K value plots are plotted.

Appendix 9 shows the Error Rate vs K value plot using the Euclidean as a distance metric. The plot shows that the minimum error rate is achieved with K values of 1 and 2. Appendix 10 shows the Error Rate vs K value plot using the Euclidean as a distance metric. The plot shows that the minimum error rate is achieved with K values of 1 and 2. These two plots show that the Euclidean

Error Rates are lower than the Manhattan distance error rate values. Therefore, Euclidean distance is selected to move forward with the best model. In addition, the error rates are the same in the cases of 1 and 2, K of 2 is chosen to use for the best KNN model. The code segment for KNN optimization can be found in Appendix 19.

Using K value of 2 and a distance metric of Euclidean, best KNN model is constructed, trained, and tested. The confusion matrix shows that the accuracy is **0.9502265503482789**. This proves that the model was a success.

Appendix 11 shows the confusion matrix of the model. True Positive is 29.31%, False Positive is 4.88%, False Negative is 0.09%, and True Negative is 65.71%.

III. Random Forest

Before constructing the random forest model, it was decided to choose the best parameters to tune the model. To achieve that the criterion and number of estimators were used to create different models and test their accuracies to optimize the parameters that will create a higher accuracy model. For the number of estimators parameter: 100, 200 and 500 values; for the criterion parameter: Gini and Entropy indexes were used. Code segment for parameter optimization can be found in Appendix 20.

As a result of this optimization experiment, the best optimization parameters for the random forest model are determined, as such:
1. n_estimators: 200
2. criterion: Gini

Using the optimized parameters, a Random Forest model is constructed, trained, and tested. Appendix 15 shows the Random Forest Classification Report. The reported accuracy is **0.9940488266720768**, which is the highest accuracy between the models used in this project. The high accuracy can be linked to the methodology of the Random Forest algorithm. Since it combines different decision tree results to result the best classification, it is proven to be very effective in classifying text.

Appendix 14 shows the Random Forest Confusion Matrix. True Positive is 33.9%, False Positive is 0.29%, False Negative 0.3%, and True Negative 65.5%.

Conclusion

In this project, a spam detection model is constructed using 3 different classification algorithms including Naïve-Bayes Text Classification, KNN and Random Forest Classifications. Using parameter optimization techniques, confusion matrixes and classification reports, the most optimal machine learning technique is identified.

In conclusion, Random Forest algorithm is found to be the most accurate technique to be used for text classification purpose of the project. The accuracy reported by the Random Forest technique is 0.994. The high accuracy can be linked to the methodology of the Random Forest algorithm. Since it combines different decision tree results to result the best classification, it is proven to be very effective in classifying text.

Code, Code segments with Appendices is submitted in separate files. For further information, please check the files submitted.