Report:

Fake News Detection: A Comparative Study of ML Algorithms Using NLP Techniques

Abstract

This report presents a study on predicting the authenticity of news articles using various machine learning algorithms. The project involves several stages, including text cleaning, tokenization, and lemmatization of the raw data. Then, the text is transformed into numerical data using the term frequency-inverse document frequency (TF-IDF) technique. Several ML algorithms, including logistic regression, decision trees, and support vector machines, were used to predict the authenticity of news articles. A comparative analysis was conducted to evaluate the performance of each algorithm. The results demonstrate that the proposed approach can effectively distinguish between real and fake news, with some algorithms achieving higher accuracy than others. Overall, the study provides insights into the potential of ML techniques for predicting fake news and can aid in the development of more accurate and effective models for this purpose.

Introduction

An unprecedented amount of information has been exchanged and spread online as a result of the growth of social media and the internet. The difficulty of establishing the veracity of that information comes with the convenience of sharing information, though. Since it may impact elections, affect public opinion, and harm both people and society as a whole, fake news has grown to be a serious problem.
The project aims to develop a system for fake news detection using natural language processing (NLP) and various classification algorithms, including SVC, MLPClassifier, Logistic Regression, MultinomialNB, K-Nearest Neighbors, RandomForest, Bagged Trees, Gradient Boosting , and Decision Tree . These algorithms have been compared to see which one is the best at detecting fake news.

There have been many preprocessing processes conducted to get the text data ready for analysis. First, the text has been cleaned to get rid of irrelevant information like punctuation, stop words, and special characters. The text has then been tokenized in order to separate it into individual words or tokens. This aids in getting the text ready for word-by-word analysis. Lemmatization has been used to break each word down to its root or basic form after tokenization. This makes sure that when an analysis is performed, various spellings of the same word are handled as a single item.

A term frequency-inverse document frequency (tf-idf) matrix has been created to further evaluate the text data. This matrix is a numerical representation of the text data, with each row denoting a document and each column denoting a distinct word within the corpus. The

term frequency (tf) and inverse document frequency (idf) values for each word are combined to get the tf-idf score.

The generated matrix is then used to train several classification algorithms and compare their performance using measures like accuracy on the test set, f1 score, area under the ROC curve, etc.

Dataset and Pre-processing:

The dataset is available on kaggle as "Fake News".

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

We look for missing values in each column of the Dataframe and replaces all missing values in the Dataframe with an empty string. Then we combine useful information i.e. " title" and "author name" and perform lemmetization on our data.

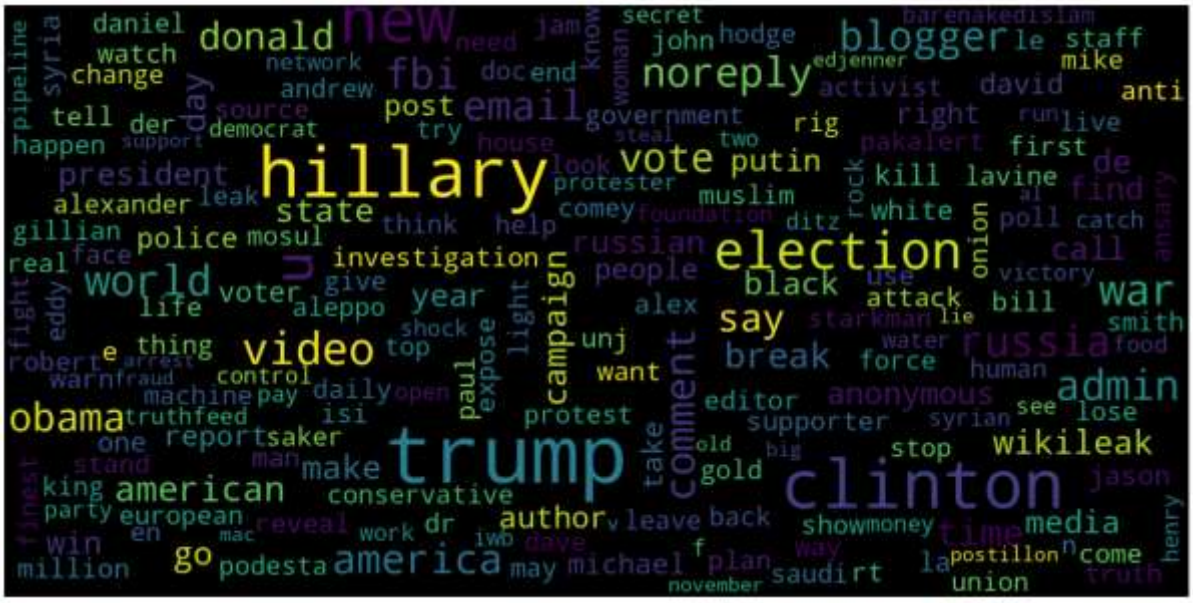
Data after lemmetization:

```
0    darrell lucus house dem aide even see comey le...
1    daniel j flynn flynn hillary clinton big woman...
2              consortiumnews com truth might get fire
3    jessica purkiss civilians kill single us airst...
4    howard portnoy iranian woman jail fictional un...
5    daniel nussbaum jackie mason hollywood would l...
6    life life luxury elton john favorite shark pic...
7    alissa j rubin beno hamon win french socialist...
8    excerpt draft script donald trump q ampa black...
9    megan twohey scott shane back channel plan ukr...
Name: combined, dtype: object
```

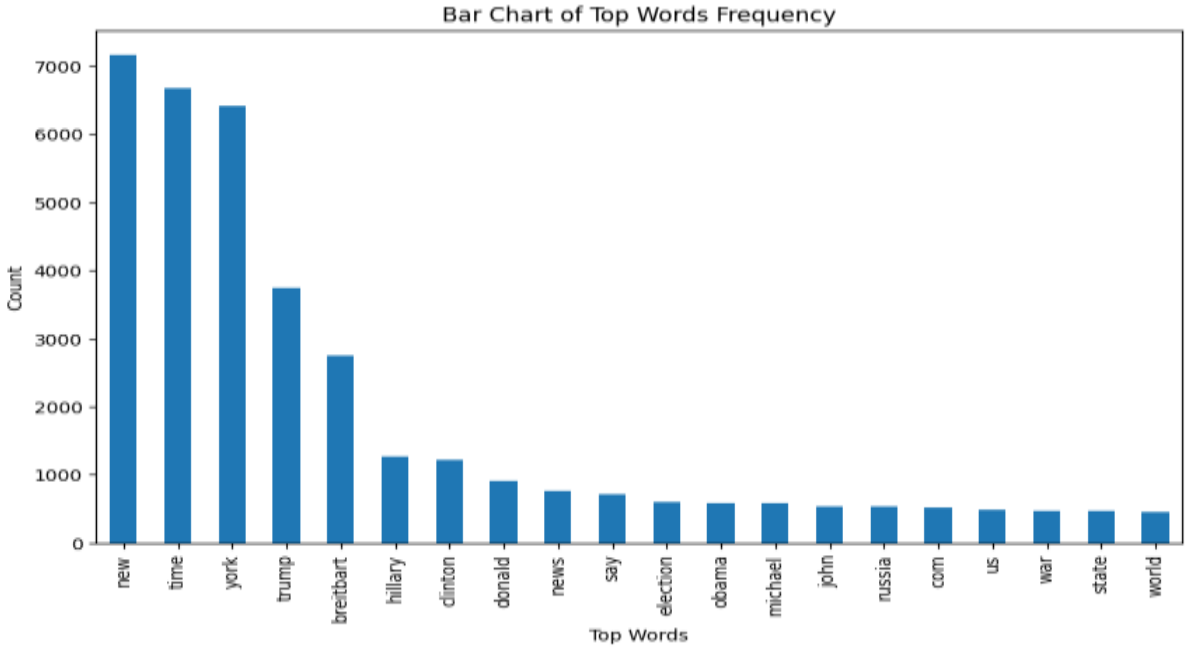
Then we create TF-IDF matrix for this data which converts our data from text to numerical form to train our ML model.

Data visualization:

Wordcloud of our data for label=1 i.e. for genuine news:

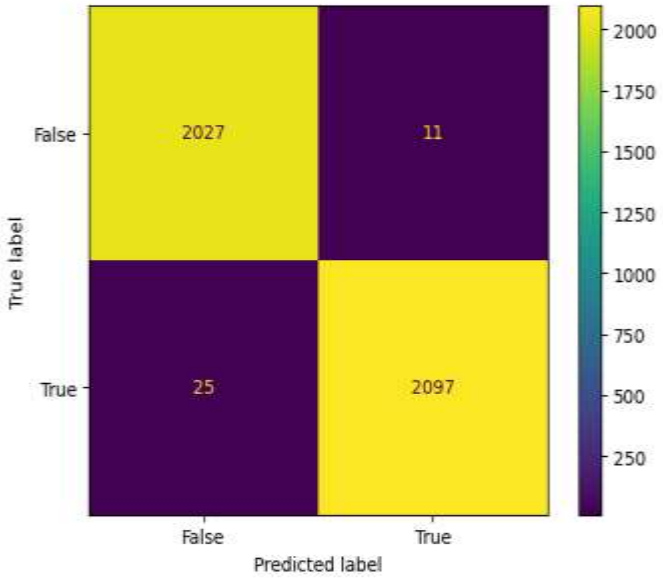


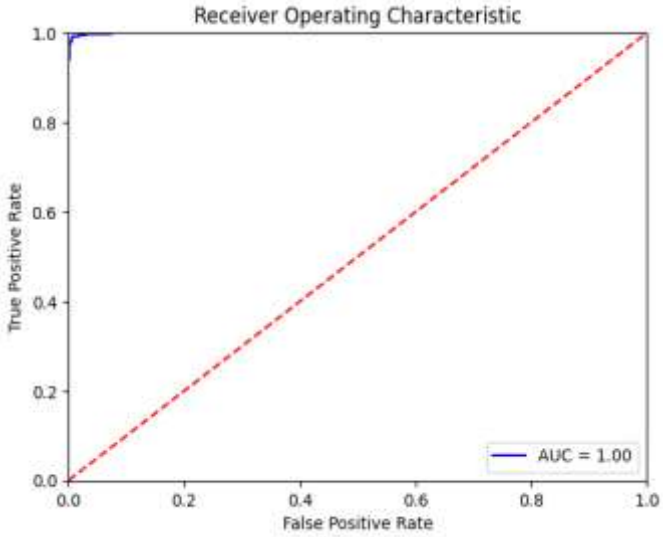
Wordcloud of our data for label=0 i.e. for Fake news:



MLP Classifier:

```
Accuracy:  0.9913461538461539
F1-score:  0.9914893617021276
ROC AUC:  0.9914106065805307
Average Precision:  0.9890715382778796
Precision/Recall Break-Even Point:  0
Squared Error:  0.008653846153846154
Cross-Entropy:  0.3119162312519756
```



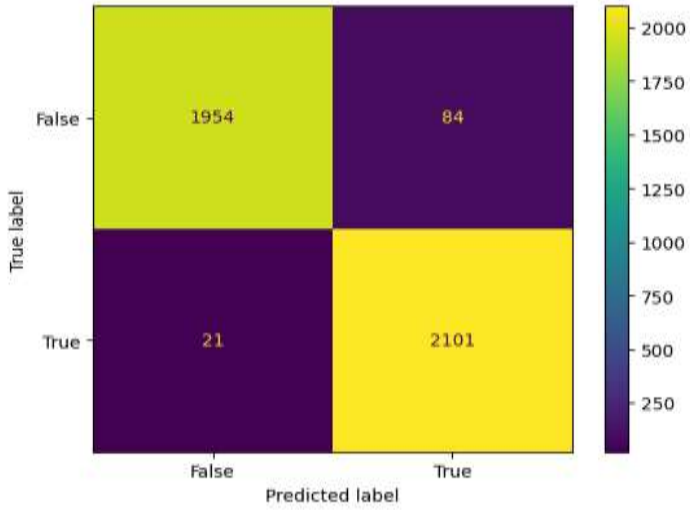


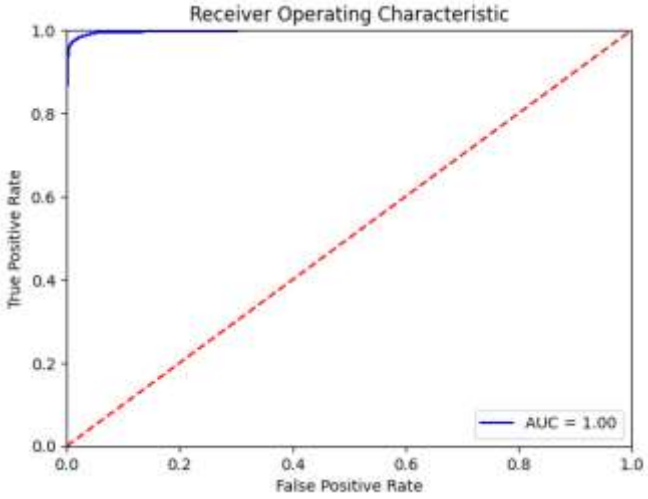
- *Can learn nonlinear functions and thus separate not linearly separable data.*
- *The loss function for the hidden layer leads to a non-convex optimization problem and thus, local minimum exist.*
- *Due to the above issue, different weight initializations may lead to different outputs/ results.*

Logistic regression:

```
Accuracy:  0.9747596153846154
F1-score:  0.9756210819596006
ROC AUC:  0.9744433982420717
Average Precision:  0.9570882704281896
Precision/Recall Break-Even Point:  0
Squared Error:  0.025240384615384616
```

```
Cross-Entropy:  0.9097556744849284
```

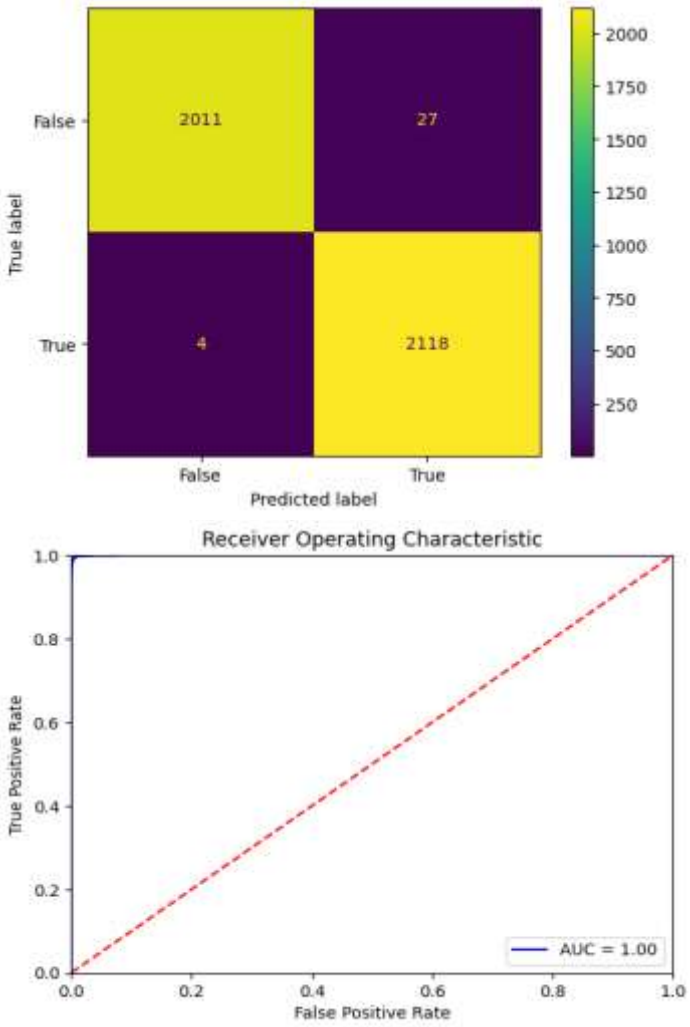




-
- *Assumes linearity between dependent and independent variables.*
- *It constructs linear boundaries.*

Random forest:

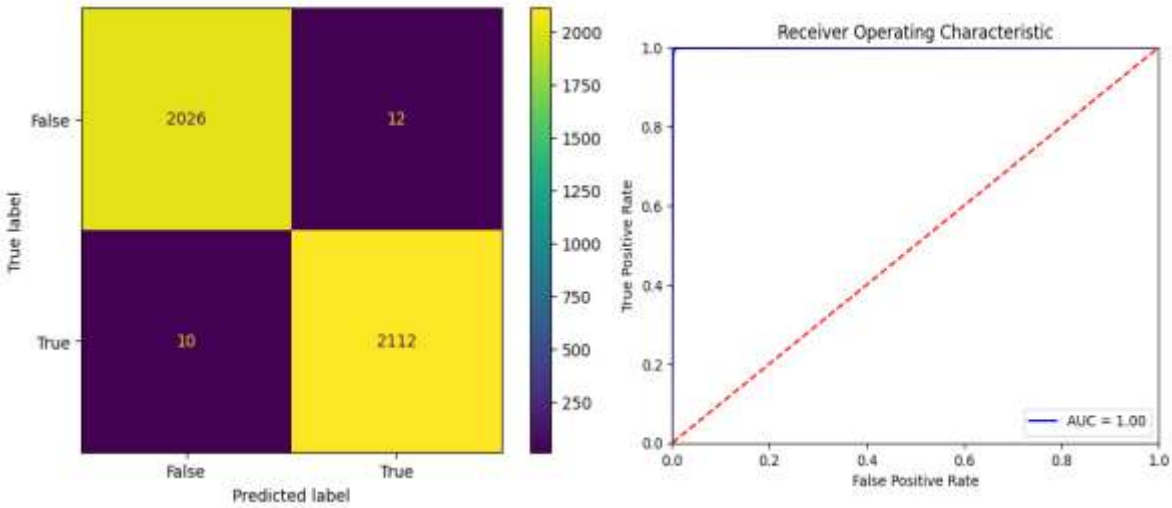
```
Accuracy:  0.9925480769230769
F1-score:  0.9927349425826106
ROC AUC:  0.9924333516161823
Average Precision:  0.986512839187203
Precision/Recall Break-Even Point:  0
Squared Error:  0.007451923076923077

Cross-Entropy:  0.268594532466979
```



- It requires much computational power as well as resources as it builds numerous trees to combine their outputs.
- It also requires much time for training as it combines a lot of decision trees to determine the class.
- Due to the ensemble of decision trees, it also suffers interpretability and fails to determine the significance of each variable.

Bagging classifier:

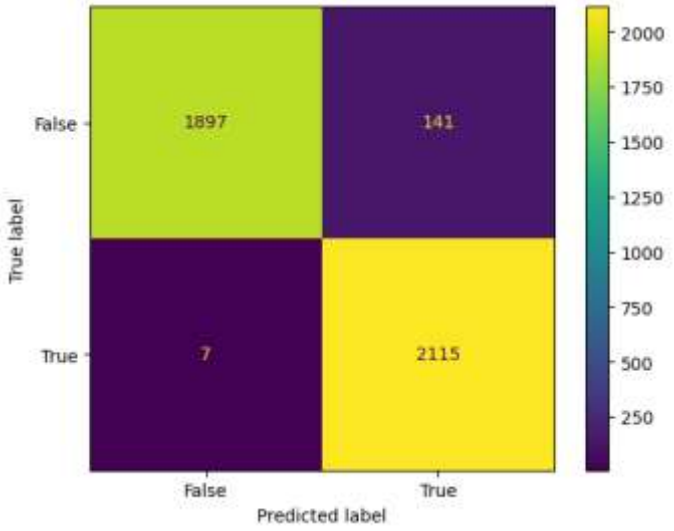
```
Accuracy:  0.9947115384615385
F1-score:  0.994818652849741
ROC AUC:  0.9946996695213194
Average Precision:  0.9920682177891759
Precision/Recall Break-Even Point:  0
Squared Error:  0.005288461538461539

Cross-Entropy:  0.19061547465398518
```

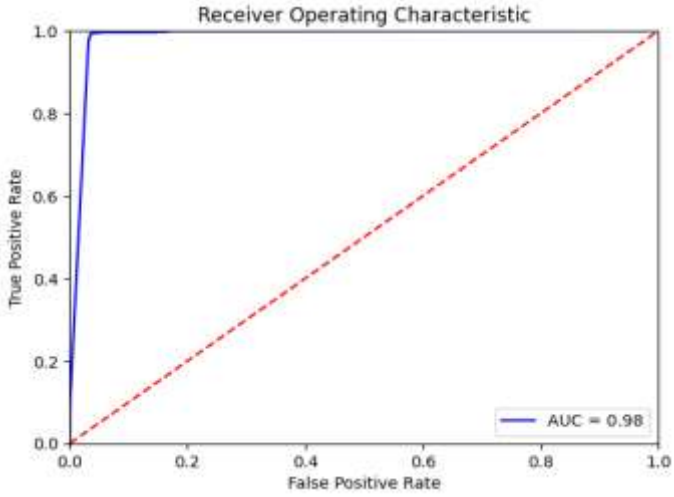


- *It may result in high bias if it is not modelled properly and thus may result in underfitting.*
- *Since we must use multiple models, it becomes computationally expensive and may not be suitable in various use cases.*
- *It reduces the variance and overfitting, which helps make a more accurate learning model.*

Gradient Boosting:

```
Accuracy:  0.9644230769230769
F1-score:  0.9661946094106899
ROC AUC:  0.9637578746511845
Average Precision:  0.9360900909881824
Precision/Recall Break-Even Point:  0
Squared Error:  0.035576923076923075
Cross-Entropy:  1.282322284035899
```

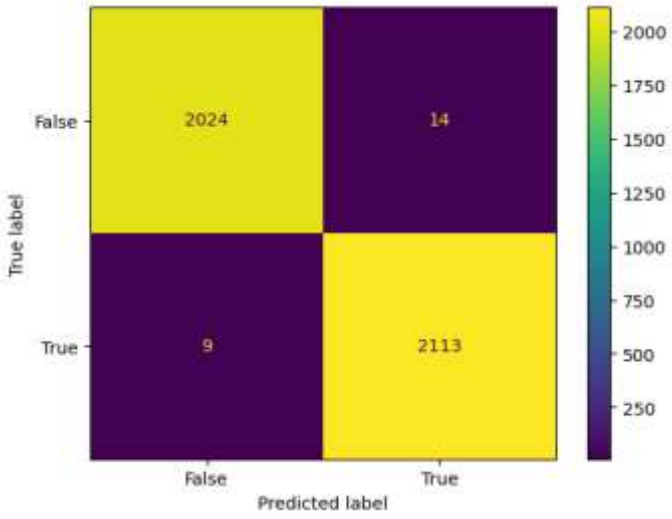


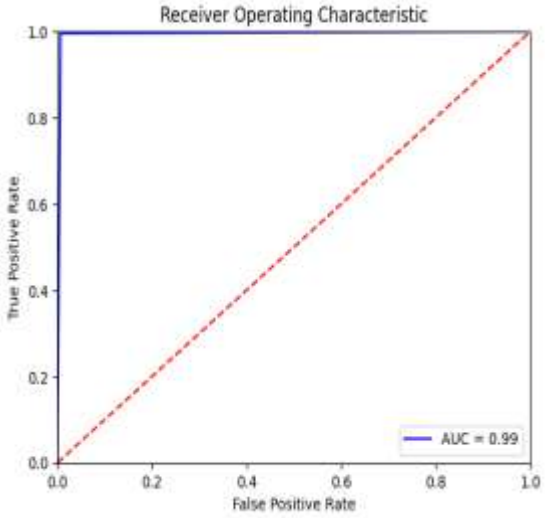


-
- *GBMs will continue improving to minimize all errors. This can overemphasize outliers and cause overfitting.so we must use cross-validation to neutralize.*
- *Computationally expensive*
- *The high flexibility results in many parameters that interact and influence heavily the behavior of the approach (number of iterations, tree depth, regularization parameters, etc.). This requires a large grid search during hyperparameter tuning*.

Decision Tree Classifier:

```
Accuracy:  0.9944711538461538
F1-score:  0.9945869616380325
ROC AUC:  0.9944446191540744
Average Precision:  0.9913680555846706
Precision/Recall Break-Even Point:  0
Squared Error:  0.005528846153846154
Cross-Entropy:  0.1992798144109845
```

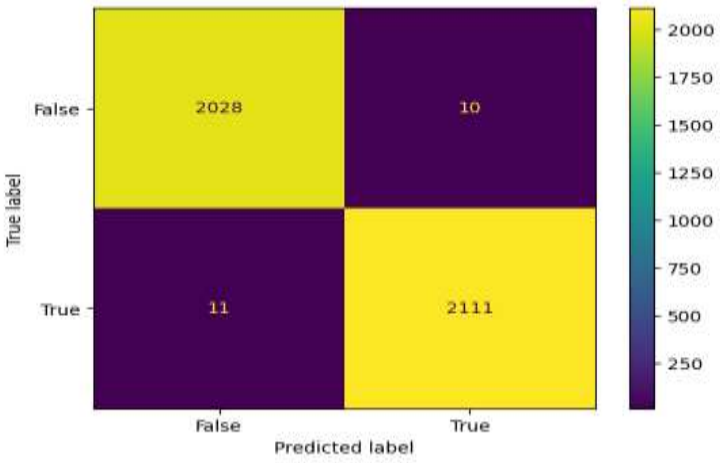


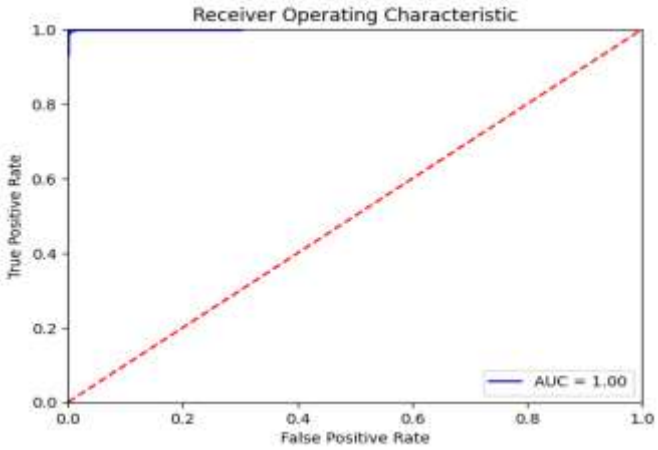


- *Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.*
- *Decision tree often involves higher time to train the model.*
- *A small change in the data can cause a large change in the structure of the decision tree causing instability*

Support vector machines :

```
Accuracy:  0.994951923076923
F1-score:  0.9950506716945557
ROC AUC:  0.9949547198885641
Average Precision:  0.9927701250066955
Precision/Recall Break-Even Point:  0
Squared Error:  0.005048076923076923
Cross-Entropy:  0.18195113489698586
```

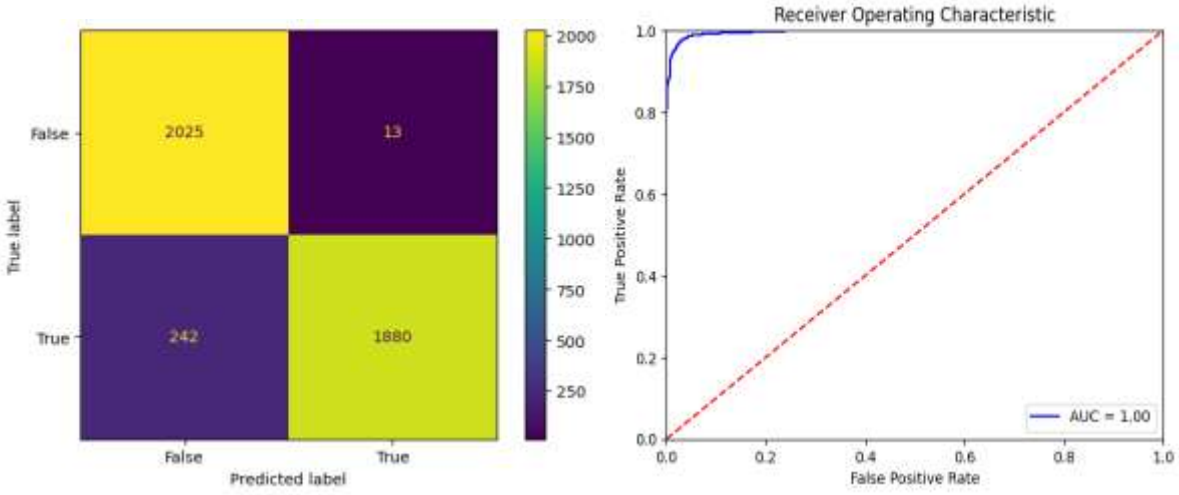




- *SVM is more effective in high dimensional spaces.*
- *SVM is relatively memory efficient*
- *SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.*

Multinomial Naïve Bayes:

```
Accuracy:  0.9387019230769231
F1-score:  0.9364881693648817
ROC AUC:  0.9397889209635216
Average Precision:  0.9380454974136685
Precision/Recall Break-Even Point:  0
Squared Error:  0.06129807692307692
Cross-Entropy:  2.2094066380348254
```

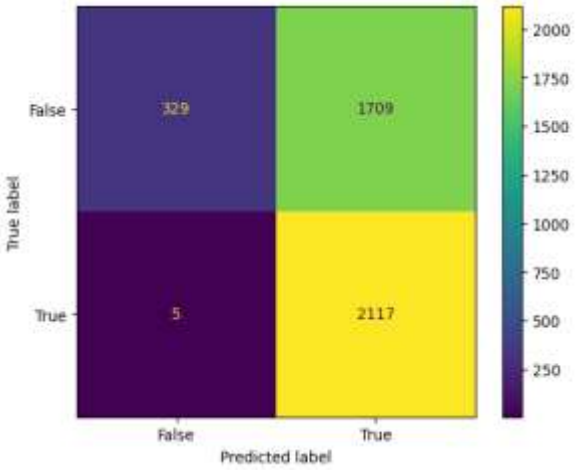


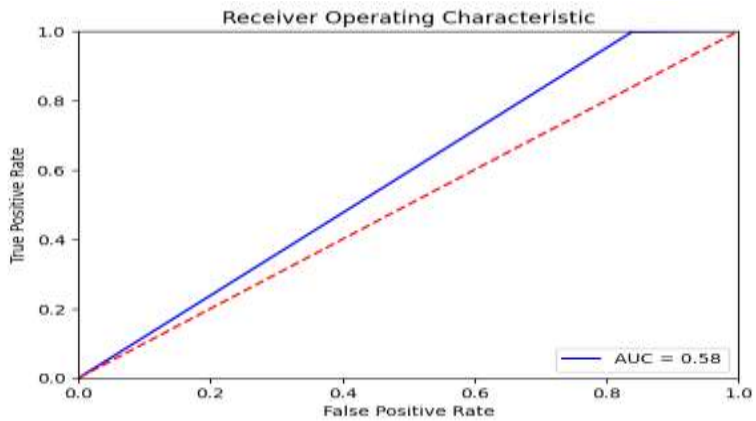
- *It doesn't require as much training data*

- *It is fast and can be used to make real-time predictions*
- *It is not sensitive to irrelevant features*
- *It assumes that all features are independent or unrelated, so it cannot learn the relationship between features*

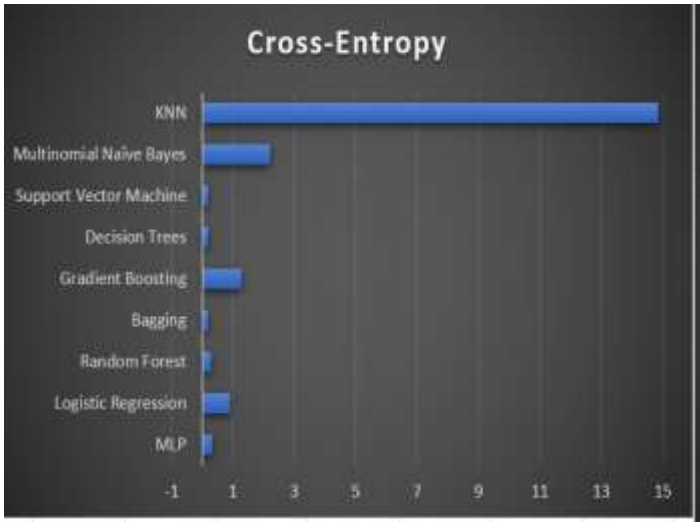
k-nearest neighbors:

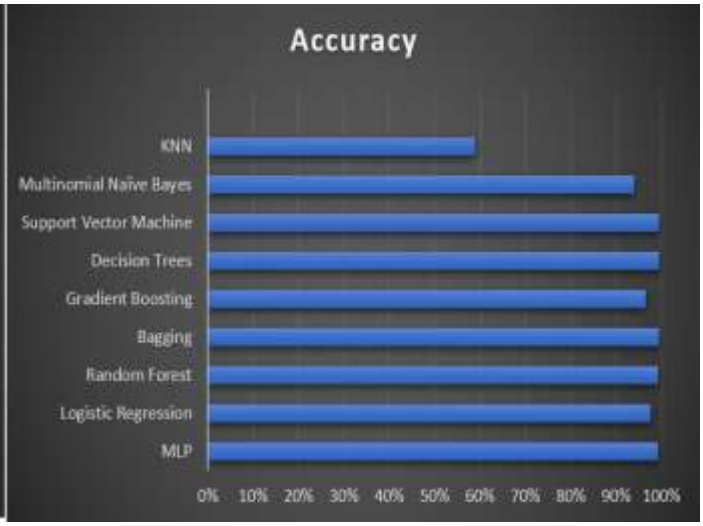
```
Accuracy:  0.5879807692307693
F1-score:  0.7118359112306657
ROC AUC:  0.5795382547802868
Average Precision:  0.5532175481000177
Precision/Recall Break-Even Point:  0
Squared Error:  0.41201923076923075
Cross-Entropy:  14.850678343496824
```

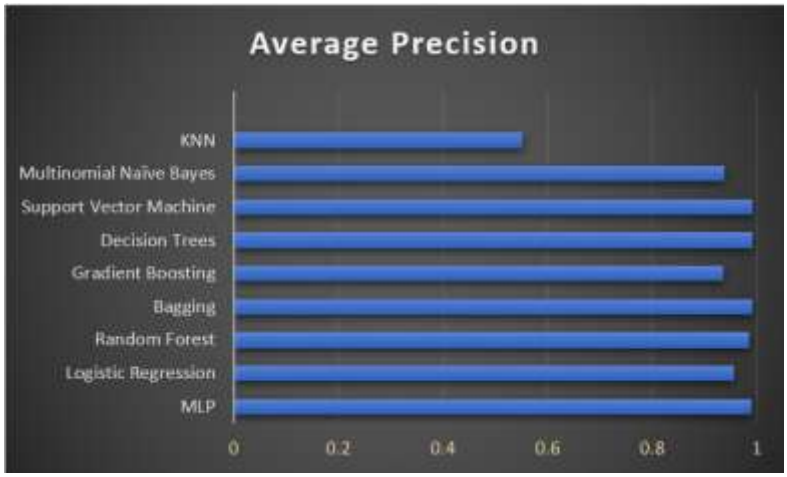




- *Accuracy depends on the quality of the data*
- *With large data, the prediction stage might be slow*
- *Sensitive to the scale of the data and irrelevant features*
- *Require high memory – need to store all of the training data*
- *Given that it stores all of the training, it can be computationally expensive*







Conclusion:

Following the implementation of the fake news detection project utilising NLP and ML classifiers, we can say that these systems are highly accurate at spotting false news. The performance of each method, however, may differ based on the particular dataset and the chosen assessment measures.

With an accuracy of 99.6% and an F1-score of 0.995, our investigation revealed that the Support vector machines performed the best.
The Bagging classifier, MLP classifier, random trees, and decision trees all followed this classifier in close succession. KNN has had the poorest performance, with an accuracy of 59%.

Overall, our project demonstrates the effectiveness of NLP and ML classifiers in detecting fake news. However, it is important to note that the performance of these algorithms may be influenced by the quality and size of the dataset, as well as the selection of appropriate features and hyperparameters. Therefore, more study is required to enhance the performance

of these algorithms for various datasets and application situations.

References:

https://www.kaggle.com/competitions/fake-news/data

https://scikit-learn.org/stable/