



PG Diploma in Data science and Machine Learning

From

Cranes Varsity

Fake News Detection

Project by

Souvik Dey

**FAKE
NEWS**

**REAL
NEWS**



INDEX

1.	Abstract	-----	1
2.	Introduction	-----	2
3.	Methods	-----	4
4.	Algorithms	-----	6
	1) Logistic Regression	-----	6
	2) Multinomial Naive Bayes	-----	7
	3) Random Forest	-----	7
	4) Support Vector Classifier	-----	8
	5) Passive Aggressive Classifier	-----	9
	6) Decision Tree Classifier	-----	10
5.	Procedure	-----	11
6.	Results and Discussion	-----	12
7.	Sample Code	-----	13
8.	Conclusion	-----	21

9.	Future Enhancement	22
10.	Bibliography	23

ABSTRACT

Recently, fake news has been incurring many problems to our society. As a result, many researchers have been working on identifying fake news. Most of the fake news detection systems utilize the linguistic feature of the news. However, they have difficulty in sensing highly ambiguous fake news which can be detected only after identifying meaning and latest related information. In this paper, to resolve this problem, we shall present a new fake news detection system using fact DB which is built and updated by human's direct judgement after collecting obvious facts. Our system receives a proposition, and search the semantically related articles from Fact DB in order to verify whether the given proposition is true or not by comparing the proposition with the related articles in fact DB. To achieve this, we utilize a deep learning model, Bidirectional Multi-Perspective Matching for Natural Language Sentence(BiMPPM), which has demonstrated a good performance for the sentence matching task. However, BiMPPM has some limitations in that the longer the length of the input sentence is, the lower its performance is, and it has difficulty in making an accurate judgement when an unlearned word or relation between words appear. In order to overcome the limitations, we shall propose a new matching technique which exploits article abstraction as well as entity matching set in addition to BiMPPM. In our experiment, we shall show that our system improves the whole performance for fake news detection.

INTRODUCTION

The advent of the World Wide Web and the rapid adoption of social media platforms (such as Facebook and Twitter) paved the way for information dissemination that has never been witnessed in the human history before. Besides other use cases, news outlets benefitted from the widespread use of social media platforms by providing updated news in near real time to its subscribers. The news media evolved from newspapers, tabloids, and magazines to a digital form such as online news platforms, blogs, social media feeds, and other digital media formats. It became easier for consumers to acquire the latest news at their fingertips. Facebook referrals account for 70% of traffic to news websites. These social media platforms in their current state are extremely powerful and useful for their ability to allow users to discuss and share ideas and debate over issues such as democracy, education, and health. However, such platforms are also used with a negative perspective by certain entities commonly for monetary gain and in other cases for creating biased opinions, manipulating mindsets, and spreading satire or absurdity. The phenomenon is commonly known as fake news.

There has been a rapid increase in the spread of fake news in the last decade. Such proliferation of sharing articles online that do not confirm to facts has led to many problems not just limited to politics but covering various other domains such as sports, health, and also science. One such area affected by fake news is the financial markets, where a rumour can have disastrous consequences and may bring the market to a halt.

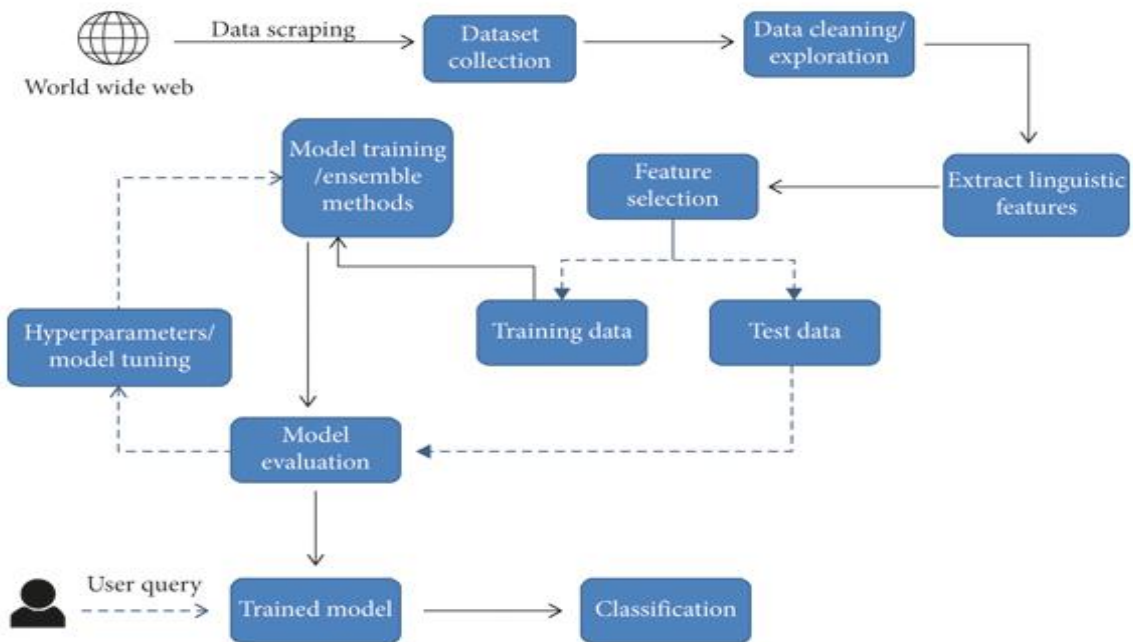
Our ability to take a decision relies mostly on the type of information we consume; our world view is shaped on the basis of information we digest. There is increasing evidence that consumers have reacted absurdly to news that later proved to be fake. One recent case is the spread of novel corona virus, where fake reports spread over the Internet about the origin, nature, and behaviour of the virus. The situation worsened as more people read about the fake contents online. Identifying such news online is a daunting task.

The World Wide Web contains data in diverse formats such as documents, videos, and audios. News published online in an unstructured format (such as news, articles, videos, and audios) is relatively difficult to detect and classify as this strictly requires human expertise. However, computational techniques such as natural language processing (NLP) can be used to detect anomalies that separate a text article that is deceptive in nature from articles that are based on facts. Other techniques involve

the analysis of propagation of fake news in contrast with real news. More specifically, the approach analyses how a fake news article propagates differently on a network relative to a true article. The response that an article gets can be differentiated at a theoretical level to classify the article as real or fake. A more hybrid approach can also be used to analyse the social response of an article along with exploring the textual features to examine whether an article is deceptive in nature or not.

A number of studies have primarily focused on detection and classification of fake news on social media platforms such as Facebook and Twitter. At conceptual level, fake news has been classified into different types; the knowledge is then expanded to generalize machine learning (ML) models for multiple domain. The study by Ahmed et al. included extracting linguistic features such as n-grams from textual articles and training multiple ML models including K-nearest neighbor (KNN), support vector machine (SVM), logistic regression (LR), linear support vector machine (LSVM), decision tree (DT), and stochastic gradient descent (SGD), achieving the highest accuracy (92%) with SVM and logistic regression. According to the research, as the number of increase n -grams calculated for a particular article, the overall accuracy decreased. The phenomenon has been observed for learning models that are used for classification. Shu et al. achieved better accuracies with different models by combining textual features with auxiliary information such as user social engagements on social media. The authors also discussed the social and psychological theories and how they can be used to detect false information online. Further, the authors discussed different data mining algorithms for model constructions and techniques shared for features extraction. These models are based on knowledge such as writing style, and social context such as stance and propagation.

METHODS



The corpus collected from the World Wide Web is preprocessed before being used as an input for training the models. The articles' unwanted variables such as authors, date posted, URL, and category are filtered out. Articles with no body text or having less than 20 words in the article body are also removed. Multicolumn articles are transformed into single column articles for uniformity of format and structure. These operations are performed on all the datasets to achieve consistency of format and structure.

Once the relevant attributes are selected after the data cleaning and exploration phase, the next step involves extraction of the linguistic features. Linguistic features involved certain textual characteristics converted into a numerical form such that they can be used as an input for the training models. These features include percentage of words implying positive or negative emotions; percentage of stop words; punctuation; function words; informal language; and percentage of certain grammar used in sentences such as adjectives, preposition, and verbs. To accomplish the extraction of features from the corpus, we used the LIWC2015 tool which classifies the text into different discrete and continuous variables, some of which are mentioned above. LIWC tool extracts 93 different features from any given text. As all of the features

extracted using the tool are numerical values, no encoding is required for categorical variables. However, scaling is employed to ensure that various feature's values lie in the range of (0, 1) This is necessary as some values are in the range of 0 to 100 (such as percentage values), whereas other values have arbitrary range (such as word counts). The input features are then used to train the different machine learning models. Each dataset is divided into training and testing set with a 70/30 split, respectively. The articles are shuffled to ensure a fair allocation of fake and true articles in training and tests instances.

The learning algorithms are trained with different hyperparameters to achieve maximum accuracy for a given dataset, with an optimal balance between variance and bias. Each model is trained multiple times with a set of different parameters using a grid search to optimize the model for the best outcome. Using a grid search to find the best parameters is computationally expensive; however, the measure is taken to ensure the models do not overfit or underfit the data.

Novel to this research, various ensemble techniques such as bagging, boosting, and voting classifier are explored to evaluate the performance over the multiple datasets. We used two different voting classifiers composed of three learning models: the first voting classifier is an ensemble of logistic regression, random forest, and KNN, whereas the second voting classifier consists of logistic regression, linear SVM, and classification and regression trees (CART). The criteria used for training the voting classifiers is to train individual models with the best parameters and then test the model based on the selection of the output label on the basis of major votes by all three models. We have trained a bagging ensemble consisting of 100 decision trees, whereas two boosting ensemble algorithms are used, XGBoost and AdaBoost. A k -fold ($k = 10$) cross validation model is employed for all ensemble learners.

ALGORITHMS

We used the following learning algorithms in conjunction with our proposed methodology to evaluate the performance of fake news detection classifiers.

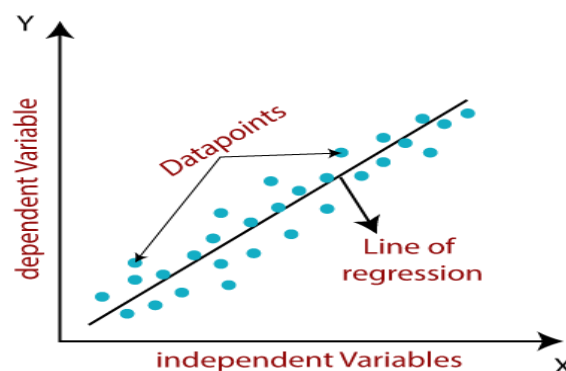
1) Logistic Regression:

As we are classifying text on the basis of a wide feature set, with a binary output (true/false or true article/fake article), a logistic regression (LR) model is used, since it provides the intuitive equation to classify problems into binary or multiple classes. We performed hyper parameters tuning to get the best result for all individual datasets, while multiple parameters are tested before acquiring the maximum accuracies from LR model. Mathematically, the logistic regression hypothesis function can be defined as follows:

$$h_{\theta}(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Logistic regression uses a sigmoid function to transform the output to a probability value; the objective is to minimize the cost function to achieve an optimal probability. The cost function is calculated as shown in

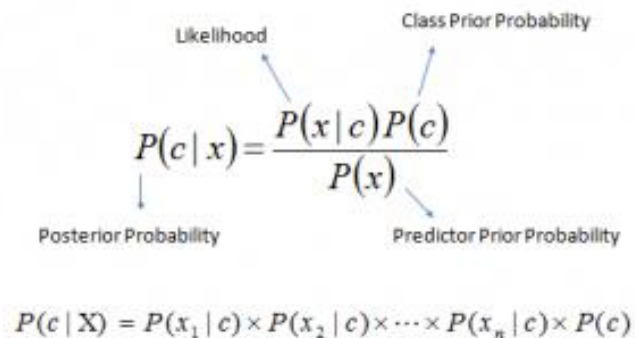
$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$



2) Multinomial Naive Bayes:

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature. The presence or absence of a feature does not affect the presence or absence of the other feature.



The diagram shows the Naive Bayes formula with labels for its components:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Labels and arrows:

- Likelihood** points to $P(x|c)$.
- Class Prior Probability** points to $P(c)$.
- Posterior Probability** points to $P(c|x)$.
- Predictor Prior Probability** points to $P(x)$.

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

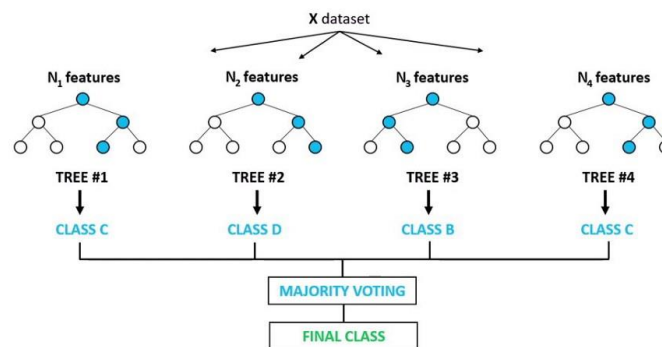
3) Random Forest (RF) :

Random forest (RF) is an advanced form of decision trees (DT) which is also a supervised learning model. RF consists of large number of decision trees working individually to predict an outcome of a class where the final prediction is based on a class that received majority votes. The error rate is low in random forest as compared to other models, due to low correlation among trees [33]. Our random forest model was trained using different parameters; i.e., different numbers of estimators were used in a grid search to produce the best model that can predict the outcome with high accuracy. There are multiple algorithms to decide a split in a decision tree based on the problem of regression or classification. For the classification problem, we have used the Gini index as a cost function to estimate a split in the dataset. The Gini index is calculated by subtracting the sum of the squared probabilities of

each class from one. The mathematical formula to calculate the Gini index is as follows:

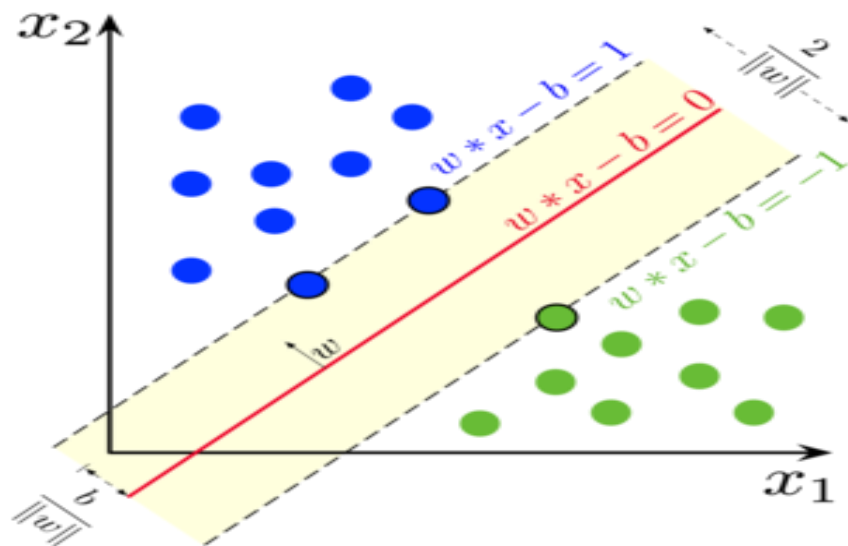
$$G_{\text{ind}} = 1 - \sum_{i=1}^c (P_i)^2,$$

Random Forest Classifier



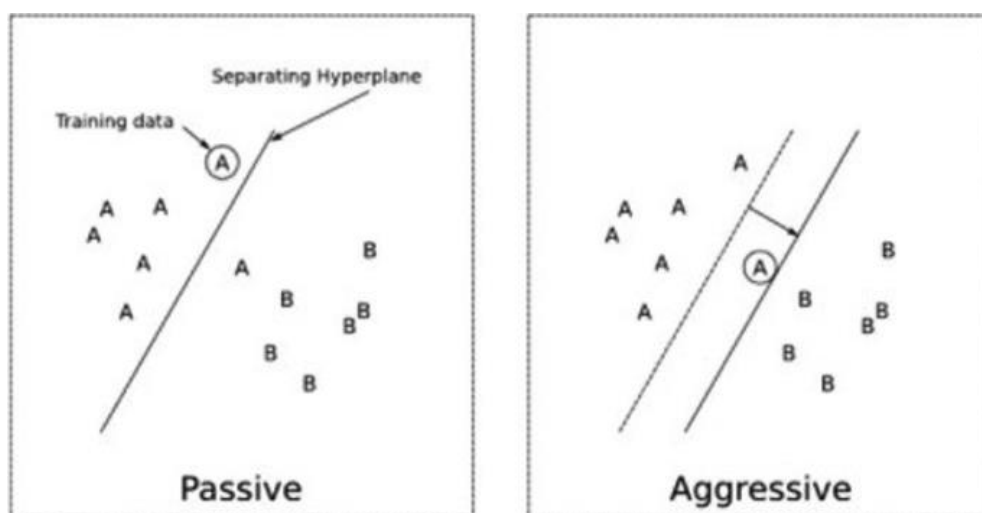
4) Support Vector Classifier(SVC) :

SVC, or Support Vector Classifier, is a supervised machine learning algorithm typically used for classification tasks. SVC works by mapping data points to a high-dimensional space and then finding the optimal hyperplane that divides the data into two classes. The Linear Support Vector Classifier (SVC) method applies a linear kernel function (Kernel Function is a method used to take data as input and transform it into the required form of processing data.) to perform classification and it performs well with a large number of samples. If we compare it with the SVC model, the Linear SVC has additional parameters such as penalty normalization which applies 'L1' or 'L2' and loss function. SVC is class capable of performing binary and multi-class classification on a dataset.



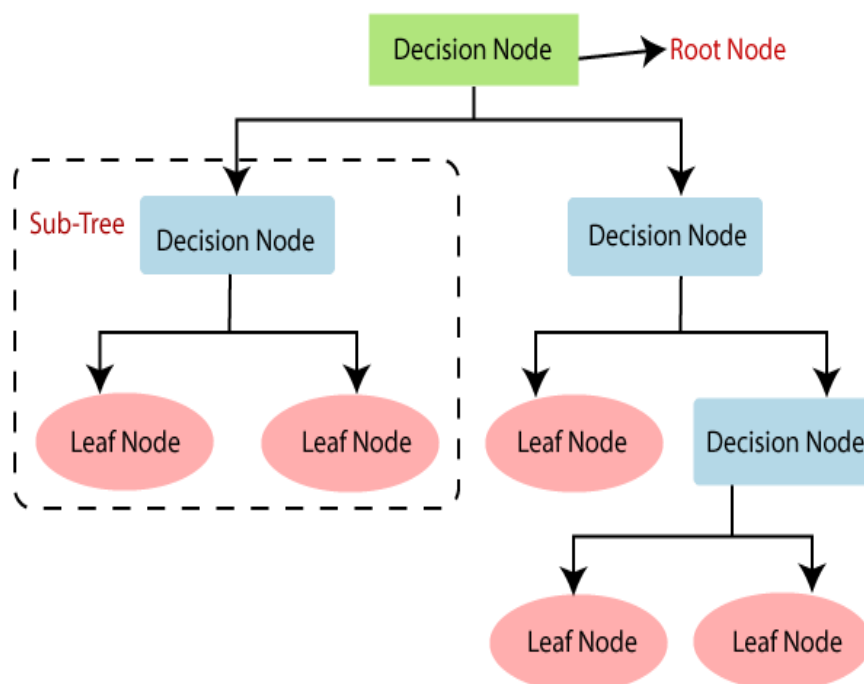
5) Passive Aggressive Classifier:

The passive aggressive classifier is a machine learning algorithm that is used for classification tasks. This algorithm is a modification of the standard Perceptron algorithm. The passive aggressive classifier was first proposed in 2006 by Crammer et al. as a way to improve the performance of the Perceptron algorithm on linearly separable data sets.



6) Decision Tree Classifier:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.



PROCEDURE

1. Importing all the Libraries
2. Reading the Dataset of csv format
3. Checking for Null Values
4. Splitting the data into Train Test
5. Transforming a given data into vector based on frequency of each
6. (Model1-MultinomialNB) - Creating and Training MultinomialNB Model, checking accuracy, predicting, confusion_matrix and classification report.
7. (Model2-SVC) - Creating and Training SVC Model, checking accuracy, predicting, confusion_matrix and classification report.
8. (Model3-LogisticRegression)- Creating and Training LogisticRegression Model, checking accuracy, predicting, confusion_matrix and classification report.
9. (Model4-RandomForestClassifier)-Creating and Training RandomForestClassifier Model, checking accuracy, predicting, confusion_matrix and classification report.
10. (Model5-PassiveAggressiveClassifier) - Creating and Training PassiveAggressiveClassifier Model, checking accuracy, predicting, confusion_matrix and classification report.
11. (Model6-DecisionTreeClassifier) - Creating and Training DecisionTreeClassifier Model, checking accuracy, predicting, confusion_matrix and classification report.
12. Comparing Accuracy of All ML Models.

RESULTS AND DISCUSSION

Internet is one of the great sources of information for its users (Donepudi, 2020). There are different social media platforms that includes Facebook or Twitter that helps the people to connect with other people. Different kind of news are also shared on these platforms. People nowadays prefer to access the news from these platforms because these are easy to use and easy to access platforms. Another advantage to the people is that these platforms provide options of comments, reacts etc. These advantages attract people to use these platforms. But as like their advantages, these platforms are also used as the best source by the cyber criminals. These persons can spread the fake news through these platforms. There is also a feature of sharing the post or news on these platforms and this feature also proves helpful for spreading such fake news. People start believing in such news as well as shares the news with other peoples.

Anyone can be registered on these platforms and can start spreading news. A person can create a page as a source of news and can spread the fake news. These platforms do not verify the person whether he is really reputable publisher. In this way, anyone can spread news against a person or an organization. These fake news can also harm a society or a political party. The report shows that it is easy to change people opinions by spreading fake news (Levin, 2017). Therefore, there is a need for detecting these fake news from spreading so that the reputation of a person, political party or an organization can be saved.

Through the use of machine learning these fake news can be detected easily and automatically. Once someone will post the fake news, machine learning algorithms will check the contents of the post and will detect it as a fake news. Different researchers are trying to find the best machine learning classifier to detect the fake news. Accuracy of the classifier must be considered because if it failed in detecting the fake news then it can be harmful to different persons. The accuracy of the classifier depends on the training of this classifier. A model that is trained in a good way can give more accuracy.

A classifier must have to be trained in a proper way with proper data set. Different researchers have trained the machine learning classifiers to detect the fake news. The main problem that occurs while training these classifiers is that mostly the training data set in an imbalanced form. Researchers have used the supervised machine learning classifiers for fake news detection. To train these classifiers they have used the three different models for feature extraction. Actually, these features are used to train the classifiers. These models are the TF-IDF Model, N-Gram Model, Bag of Words Model. These models extract the features from the training data set and then the classifier is trained through these features.

SAMPLE CODE

```
Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
1. Importing the necessary dependencies.

[1] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from collections import Counter
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer

#All using Algorithm
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix, classification_report

import warnings
warnings.filterwarnings('ignore')
```

```
Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
2. Now, let's read the data into a DataFrame.

[2] df = pd.read_csv('/content/news.csv')

3. Now we will get the shape of the data and the first 5 records.

[3] df.shape
(7795, 141)

df.head()

```

	Unnamed: 0	title	text	label	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 131	Unnamed: 132	Unnamed: 133	Unnamed: 134	Unnamed: 135	Unnamed: 136	Unnamed: 137
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fellow...	FAKE	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1
2
3
4

```
Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
4. Checking for Null Values.

[15] df.isnull().sum()

```

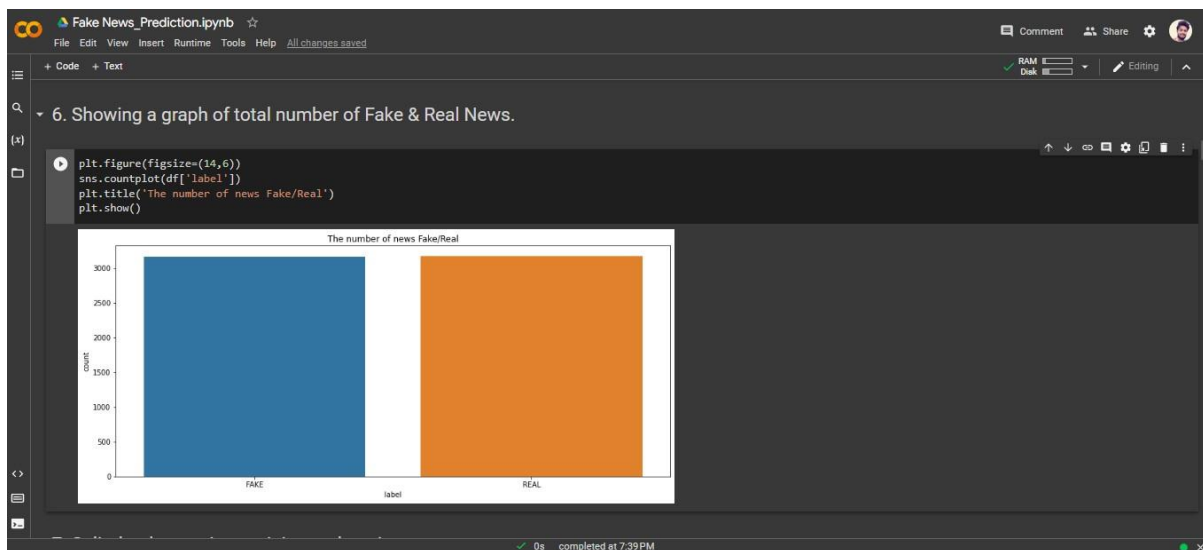
	Unnamed: 0	title	text	label	Unnamed: 4
0	219	610	866	1040	7477
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141

```
5. And get the labels from the DataFrame.

[17] df['label']

```

	label
0	FAKE
1	FAKE
2	REAL
3	FAKE
4	REAL



Fake News_Prediction.ipynb

7. Split the dataset into training and testing sets.

```
[ ] X_train,X_test,y_train,y_test=train_test_split(df['text'],df['label'],test_size=0.2,random_state=7)
len(X_train), len(X_test)

(5068, 1267)
```

8. Transforming a given data into vector based on frequency of each.

```
v = CountVectorizer()
X_train_count = v.fit_transform(X_train)
X_train_count.toarray()[:5]

array([[0, 1, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 3, 0, ..., 0, 0, 0],
       [0, 1, 0, ..., 0, 0, 0],
       [0, 1, 0, ..., 0, 0, 0]])

[ ] X_test_count = v.transform(X_test)
X_test_count.toarray()[:5]
```

14s completed at 7:42 PM

Fake News_Prediction.ipynb

```
[ ] X_test_count = v.transform(X_test)
X_test_count.toarray()[:5]

array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 1, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

9. (Model1 -MultinomialNB) - Creating and Training MultinomialNB Model, checking accuracy, predicting, confusion_matrix and classification report.

```
[ ] model1 = MultinomialNB()
model1.fit(X_train_count,y_train)

MultinomialNB()

[ ] s1 = model1.score(X_train_count,y_train)
s1

0.9293606945540647
```

```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help

+ Code + Text
[ ] y_pred1 = model1.predict(X_test_count)
y_pred1[:10]

array(['REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL',
      'REAL', 'FAKE'], dtype='<U4')

[ ] y_test[:10]

3534 REAL
6265 FAKE
3123 REAL
3940 REAL
2856 REAL
3831 REAL
4854 REAL
5861 REAL
387 REAL
2956 FAKE
Name: label, dtype: object

```



```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help

+ Code + Text
print(classification_report(y_pred1,y_test))

```

	precision	recall	f1-score	support
FAKE	0.84	0.92	0.88	589
REAL	0.93	0.85	0.89	687
accuracy			0.88	1267
macro avg	0.88	0.89	0.88	1267
weighted avg	0.89	0.88	0.88	1267

10. (Model2-SVC) - Creating and Training SVC Model, checking accuracy, predicting, confusion_matrix and classification report.

```

[ ] model2 = SVC()
model2.fit(X_train_count,y_train)

SVC()

[ ] s2 = model2.score(X_train_count,y_train)
s2
0.877663772691397
14s completed at 7:42PM

```

```

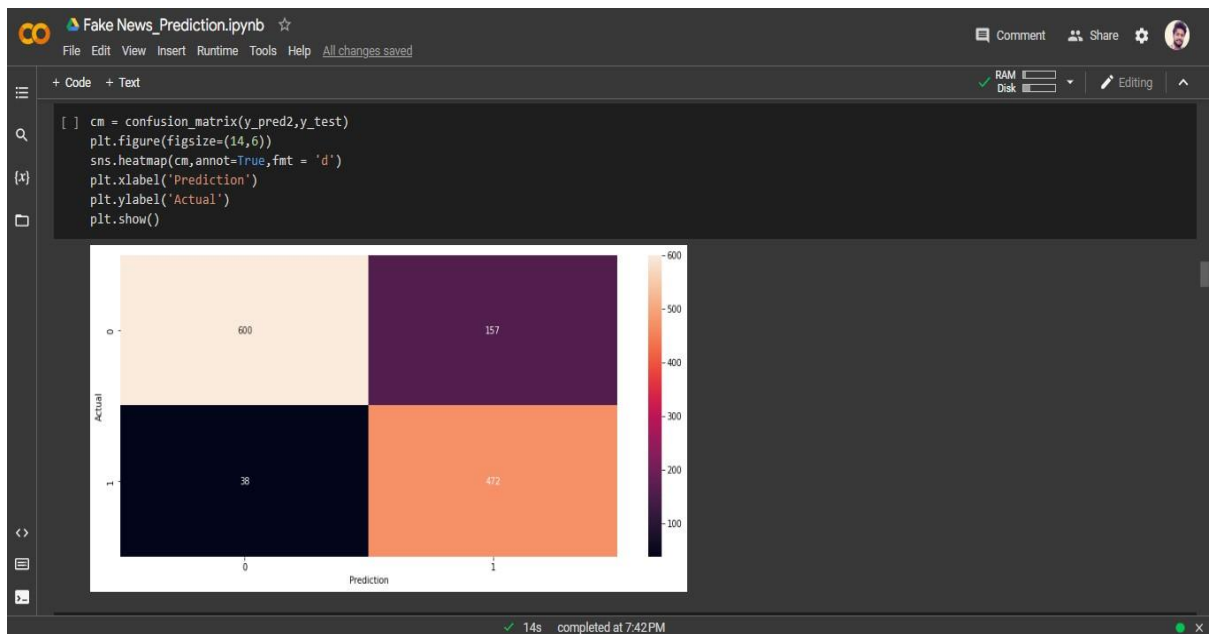
Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[ ] y_pred2 = model2.predict(X_test_count)
y_pred2[:10]

array(['FAKE', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'FAKE',
      'REAL', 'FAKE'], dtype=object)

[ ] y_test[:10]

3534 REAL
6265 FAKE
3123 REAL
3940 REAL
2856 REAL
3031 REAL
4854 REAL
5861 REAL
307 REAL
2956 FAKE
Name: label, dtype: object

```



```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[ ] print(classification_report(y_pred2,y_test))

          precision    recall  f1-score   support

     FAKE       0.94       0.79       0.86         757
     REAL       0.75       0.93       0.83         510

 accuracy          0.85          0.85          0.85        1267
 macro avg          0.85          0.86          0.84        1267
 weighted avg          0.86          0.85          0.85        1267

11. (Model3-LogisticRegression)- Creating and Training LogisticRegression Model,
checking accuracy, predicting, confusion_matrix and classification report.

[ ] model3 = LogisticRegression()
model3.fit(X_train_count,y_train)

LogisticRegression()

[ ] s3 = model3.score(X_train_count,y_train)
s3

0.9992107340173638

```

14s completed at 7:42 PM

```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved

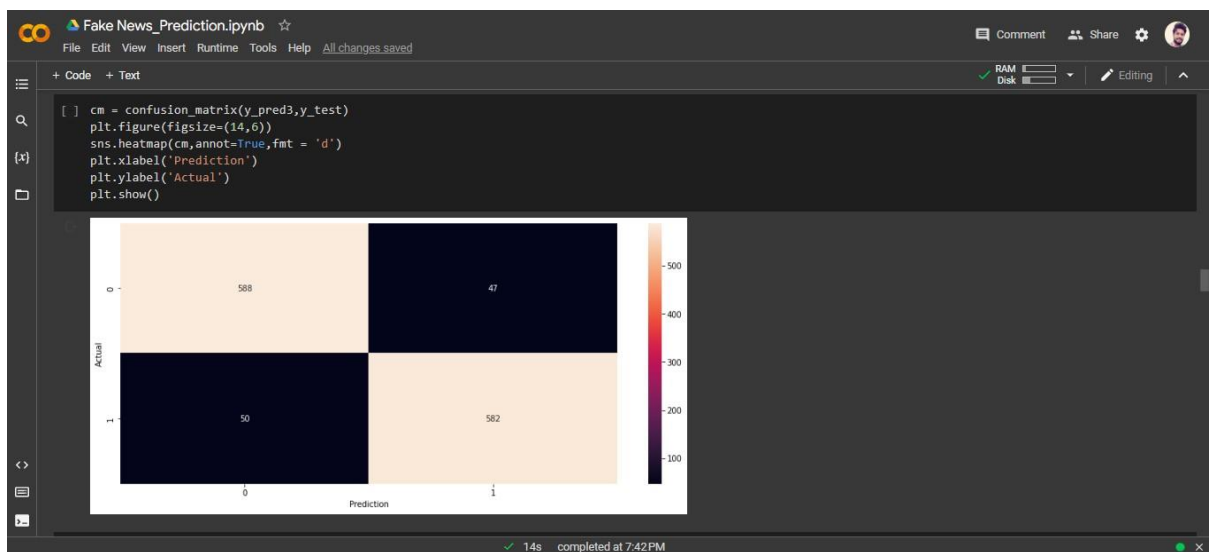
+ Code + Text
[ ] y_pred3 = model3.predict(X_test_count)
y_pred3[:10]

array(['REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL',
      'REAL', 'FAKE'], dtype=object)

[ ] y_test[:10]

3534 REAL
6265 FAKE
3123 REAL
3940 REAL
2856 REAL
3031 REAL
4854 REAL
5861 REAL
307 REAL
2956 FAKE
Name: label, dtype: object

```



```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text
[ ] print(classification_report(y_pred3,y_test))

```

	precision	recall	f1-score	support
FAKE	0.92	0.93	0.92	635
REAL	0.93	0.92	0.92	632
accuracy			0.92	1267
macro avg	0.92	0.92	0.92	1267
weighted avg	0.92	0.92	0.92	1267

12. (Model4-RandomForestClassifier) - Creating and Training RandomForestClassifier Model, checking accuracy, predicting, confusion_matrix and classification report.

```

[ ] model4 = RandomForestClassifier(n_estimators=45)
model4.fit(X_train_count, y_train)

RandomForestClassifier(n_estimators=45)

[ ] s4 = model4.score(X_train_count,y_train)
s4

1.0

```

14s completed at 7:42PM

```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved

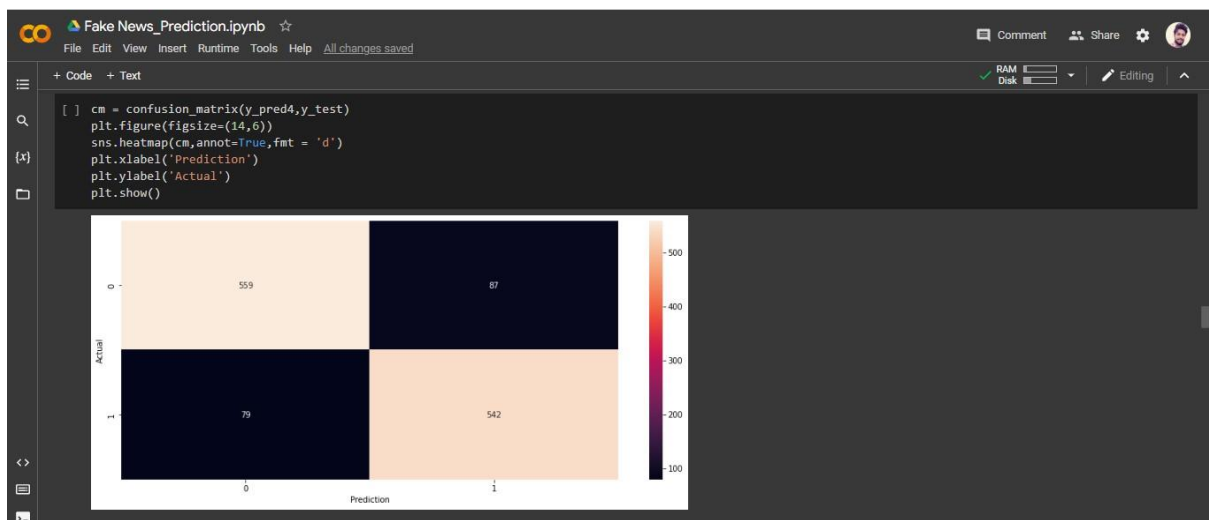
+ Code + Text
[ ] y_pred4 = model4.predict(X_test_count)
y_pred4[:10]

array(['REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL',
       'REAL', 'REAL'], dtype=object)

[ ] y_test[:10]

3534 REAL
6265 FAKE
3123 REAL
3940 REAL
2856 REAL
3031 REAL
4854 REAL
5861 REAL
307 REAL
2956 FAKE
Name: label, dtype: object

```



```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text
[ ] print(classification_report(y_pred4,y_test))

precision    recall  f1-score   support

   FAKE       0.88     0.87     0.87        646
   REAL       0.86     0.87     0.87        621

 accuracy          0.87       1267
 macro avg         0.87     0.87     0.87       1267
 weighted avg      0.87     0.87     0.87       1267

13. (Model5-PassiveAggressiveClassifier) - Creating and Training
PassiveAggressiveClassifier Model, checking accuracy, predicting, confusion_matrix
and classification report.

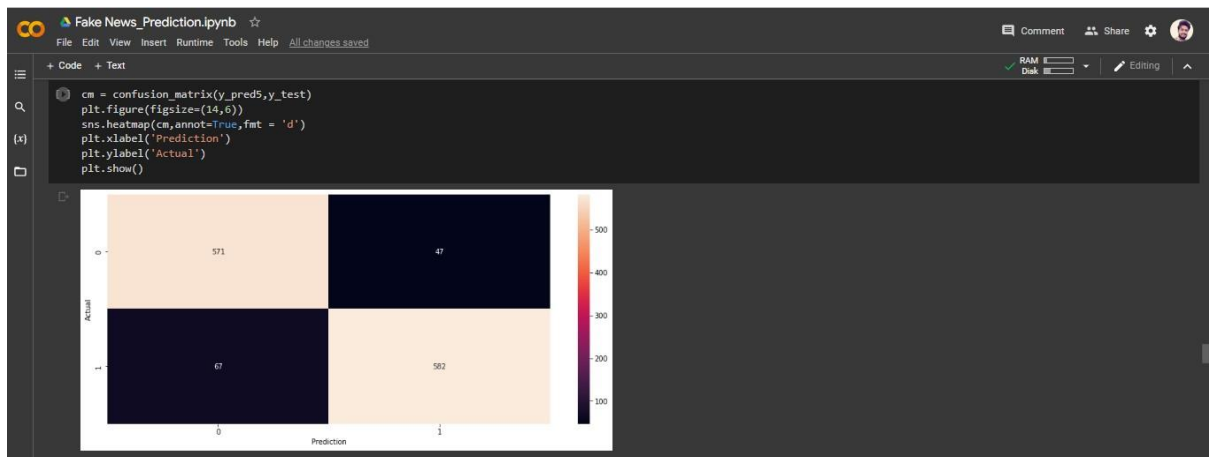
[ ] model5 = PassiveAggressiveClassifier(max_iter=50)
model5.fit(X_train_count,y_train)

PassiveAggressiveClassifier(max_iter=50)

[ ] s5 = model5.score(X_train_count,y_train)
s5

0.99822415151398686

```



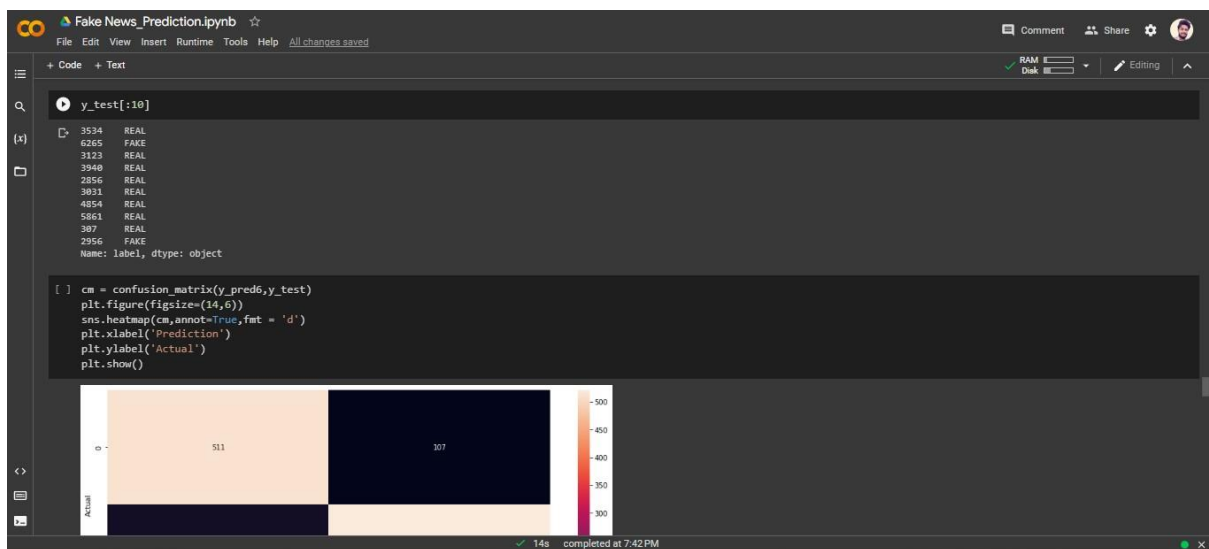
Fake News_Prediction.ipynb

```
y_pred5=model5.predict(X_test_count)
y_pred5[:10]
```

```
array(['REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL',
       'REAL', 'FAKE'], dtype='<U4')

[ ] y_test[:10]
```

```
3534 REAL
6265 FAKE
3123 REAL
3940 REAL
2856 REAL
3031 REAL
4854 REAL
5861 REAL
307 REAL
2956 FAKE
Name: label, dtype: object
```



```

Fake News_Prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
print(classification_report(y_pred5,y_test))

precision    recall  f1-score   support

   FAKE       0.89       0.92       0.91         618
   REAL       0.93       0.90       0.91         649

 accuracy          0.91          0.91          0.91        1267
 macro avg          0.91          0.91          0.91        1267
 weighted avg          0.91          0.91          0.91        1267

14. (Model6-DecisionTreeClassifier) - Creating and Training DecisionTreeClassifier
Model, checking accuracy, predicting, confusion_matrix and classification report.

model6 = DecisionTreeClassifier()
model6.fit(X_train_count,y_train)

DecisionTreeClassifier()

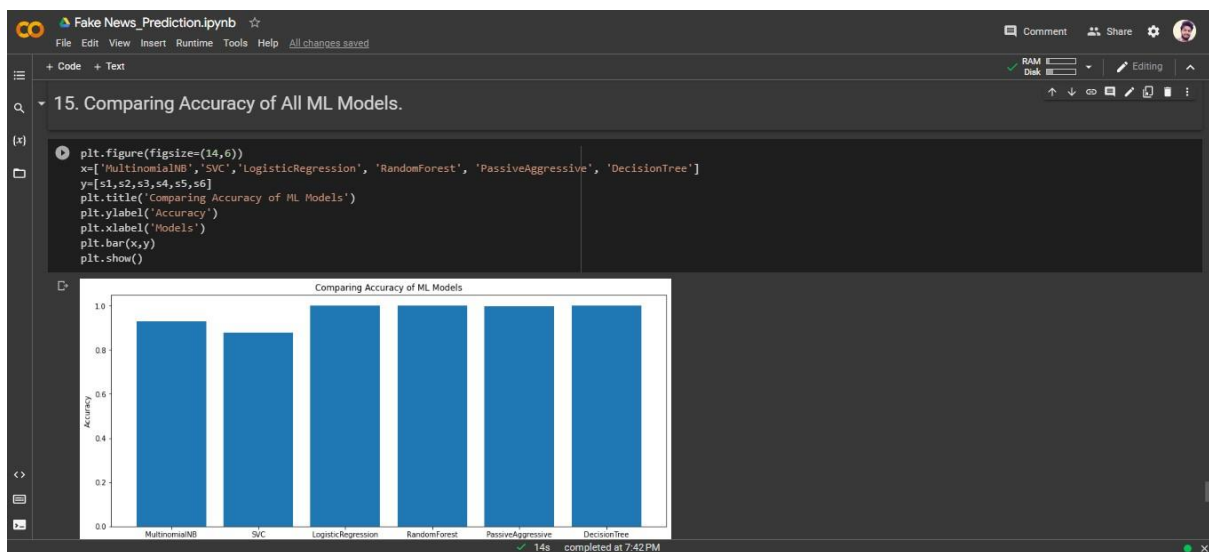
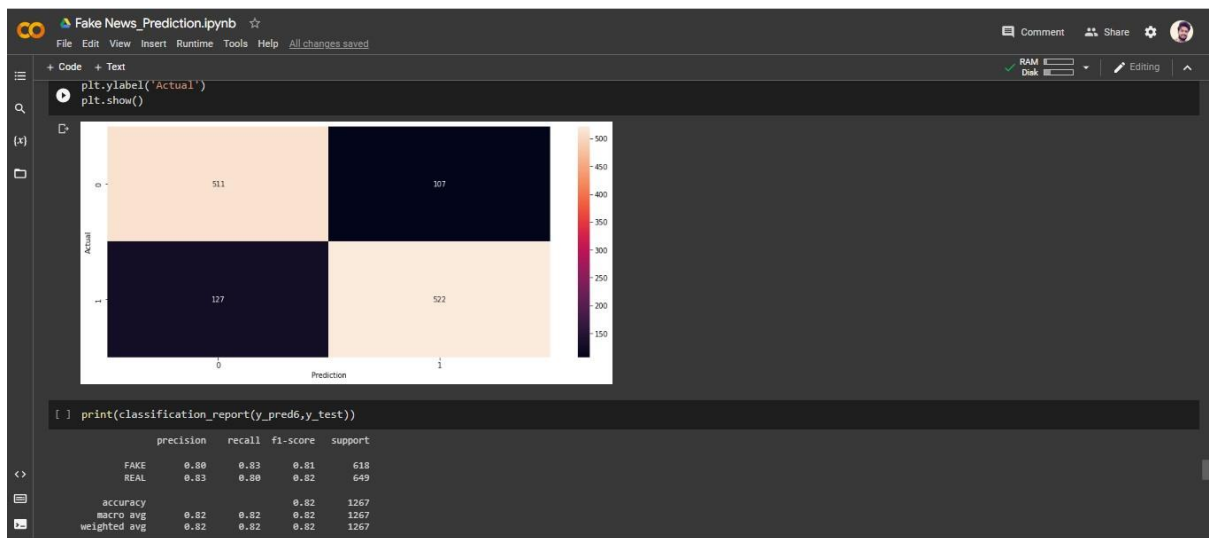
s6 = model6.score(X_train_count,y_train)
s6

1.0

y_pred6 = model6.predict(X_test_count)
y_pred6[:10]

array(['REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL', 'REAL'],
      dtype=object)
14s completed at 7:42 PM

```



CONCLUSION

The task of classifying news manually requires in-depth knowledge of the domain and expertise to identify anomalies in the text. In this research, we discussed the problem of classifying fake news articles using machine learning models and ensemble techniques. The data we used in our work is collected from the World Wide Web and contains news articles from various domains to cover most of the news rather than specifically classifying political news. The primary aim of the research is to identify patterns in text that differentiate fake articles from true news. We extracted different textual features from the articles using an LIWC tool and used the feature set as an input to the models. The learning models were trained and parameter-tuned to obtain optimal accuracy. Some models have achieved comparatively higher accuracy than others. We used multiple performance metrics to compare the results for each algorithm. The ensemble learners have shown an overall better score on all performance metrics as compared to the individual learners.

Fake news detection has many open issues that require attention of researchers. For instance, in order to reduce the spread of fake news, identifying key elements involved in the spread of news is an important step. Graph theory and machine learning techniques can be employed to identify the key sources involved in spread of fake news. Likewise, real time fake news identification in videos can be another possible future direction.

FUTURE ENHANCEMENT

There are many future improvement aspects of this project. Introducing a cross checking feature on the machine learning model so it compares the news inputs with the reputable news sources is one way to go. It has to be online and done in real time, which will be very challenging. Improving the model accuracy using bigger and better datasets, integrating different machine learning algorithms is also something we hope to do in the future.

BIBLIOGRAPHY

1. A. Douglas, "News consumption and the new electronic media," *The International Journal of Press/Politics*, vol. 11, no. 1, pp. 29–52, 2006.
View at: [Publisher Site](#) | [Google Scholar](#)
2. J. Wong, "Almost all the traffic to fake news sites is from facebook, new data show," 2016.
View at: [Google Scholar](#)
3. D. M. J. Lazer, M. A. Baum, Y. Benkler et al., "The science of fake news," *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.
View at: [Publisher Site](#) | [Google Scholar](#)
4. S. A. García, G. G. García, M. S. Prieto, A. J. M. Guerrero, and C. R. Jiménez, "The impact of term fake news on the scientific community scientific performance and mapping in web of science," *Social Sciences*, vol. 9, no. 5, 2020.
View at: [Google Scholar](#)
5. A. D. Holan, *2016 Lie of the Year: Fake News*, Politifact, Washington, DC, USA, 2016.
6. S. Kogan, T. J. Moskowitz, and M. Niessner, "Fake News: Evidence from Financial Markets," 2019, <https://ssrn.com/abstract=3237763>.
View at: [Google Scholar](#)
7. A. Robb, "Anatomy of a fake news scandal," *Rolling Stone*, vol. 1301, pp. 28–33, 2017.
View at: [Google Scholar](#)
8. J. Soll, "The long and brutal history of fake news," *Politico Magazine*, vol. 18, no. 12, 2016.
View at: [Google Scholar](#)
9. J. Hua and R. Shaw, "Corona virus (covid-19) "infodemic" and emerging issues through a data lens: the case of China," *International Journal of Environmental Research and Public Health*, vol. 17, no. 7, p. 2309, 2020.
View at: [Publisher Site](#) | [Google Scholar](#)

10. N. K. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015.
View at: [Publisher Site](#) | [Google Scholar](#)
11. F. T. Asr and M. Taboada, "Misinfotext: a collection of news articles, with false and true labels," 2019.
View at: [Google Scholar](#)
12. K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.
View at: [Publisher Site](#) | [Google Scholar](#)
13. S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.
View at: [Publisher Site](#) | [Google Scholar](#)
14. H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 211–236, 2017.
View at: [Publisher Site](#) | [Google Scholar](#)
15. V. L. Rubin, N. Conroy, Y. Chen, and S. Cornwell, "Fake news or truth? using satirical cues to detect potentially misleading news," in *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, pp. 7–17, San Diego, CA, USA, 2016.
View at: [Google Scholar](#)
16. H. Jwa, D. Oh, K. Park, J. M. Kang, and H. Lim, "exBAKE: automatic fake news detection model based on bidirectional encoder representations from transformers (bert)," *Applied Sciences*, vol. 9, no. 19, 2019.
View at: [Publisher Site](#) | [Google Scholar](#)
17. H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *Proceedings of the International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, pp. 127–138, Springer, Vancouver, Canada, 2017.
View at: [Publisher Site](#) | [Google Scholar](#)
18. W. Y. Wang, *Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection*, Association for Computational Linguistics, Stroudsburg, PA, USA, 2017.

19. B. Riedel, I. Augenstein, G. P. Spithourakis, and S. Riedel, “A simple but tough-to-beat baseline for the fake news challenge stance detection task,” 2017, <https://arxiv.org/abs/1707.03264>.
View at: [Google Scholar](#)
20. N. Ruchansky, S. Seo, and Y. Liu, “Csi: a hybrid deep model for fake news detection,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 797–806, Singapore, 2017.
View at: [Google Scholar](#)
21. V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, “Automatic detection of fake news,” 2017, <https://arxiv.org/abs/1708.07104>.
View at: [Google Scholar](#)
22. P. Bühlmann, “Bagging, boosting and ensemble methods,” in *Handbook of Computational Statistics*, pp. 985–1022, Springer, Berlin, Germany, 2012.
View at: [Google Scholar](#)
23. H. Ahmed, I. Traore, and S. Saad, “Detecting opinion spams and fake news using text classification,” *Security and Privacy*, vol. 1, no. 1, 2018.
View at: [Publisher Site](#) | [Google Scholar](#)
24. Kaggle, *Fake News*, Kaggle, San Francisco, CA, USA, 2018, <https://www.kaggle.com/c/fake-news>.
25. Kaggle, *Fake News Detection*, Kaggle, San Francisco, CA, USA, 2018, <https://www.kaggle.com/jruvika/fake-news-detection>.
26. J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.
View at: [Google Scholar](#)
27. T. M. Mitchell, *The Discipline of Machine Learning*, Carnegie Mellon University, Pittsburgh, PA, USA, 2006.
28. N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, UK, 2000.
29. T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” *The Annals of Statistics*, vol. 36, no. 3, pp. 1171–1220, 2008.

View at: [Publisher Site](#) | [Google Scholar](#)

30. V. Kecman, *Support Vector Machines-An Introduction in "Support Vector Machines: Theory and Applications"*, Springer, New York City, NY, USA, 2005.

31. S. Akhtar, F. Hussain, F. R. Raja et al., "Improving mispronunciation detection of arabic words for non-native learners using deep convolutional neural network features," *Electronics*, vol. 9, no. 6, 2020.

View at: [Publisher Site](#) | [Google Scholar](#)

32. D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Information Fusion*, vol. 6, no. 1, pp. 63–81, 2005.

View at: [Publisher Site](#) | [Google Scholar](#)

33. B. Gregorutti, B. Michel, and P. Saint-Pierre, "Correlation and variable importance in random forests," *Statistics and Computing*, vol. 27, no. 3, pp. 659–678, 2017.

View at: [Publisher Site](#) | [Google Scholar](#)

34. L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Springer, Berlin, Germany, 1984.

35. R. E. Schapire, "A brief introduction to boosting," *IJCAI*, vol. 99, pp. 1401–1406, 1999.

View at: [Google Scholar](#)

36. E. M. Dos Santos, R. Sabourin, and P. Maupin, "Overfitting cautious selection of classifier ensembles with genetic algorithms," *Information Fusion*, vol. 10, no. 2, pp. 150–162, 2009.

View at: [Publisher Site](#) | [Google Scholar](#)

37. T. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco, CA, USA, 2016.

View at: [Google Scholar](#)

38. T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

View at: [Publisher Site](#) | [Google Scholar](#)

39. L. Lam and S. Y. Suen, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 27, no. 5, pp. 553–568, 1997.

View at: [Publisher Site](#) | [Google Scholar](#)