

# **DESIGN AND DEVELOPMENT OF FAKE NEWS DETECTION SYSTEM USING MACHINE LEARNING**

by

Mannat Kaur Nagpal (1601410064)

Prachi Gupta (1601410078)

Saumya Agarwal (1601410094)

Submitted to the Department of Computer Science & Engineering  
in partial fulfillment of the requirements  
for the degree of  
Bachelor of Technology  
in  
Computer Science



*Department of Computer Science & Engineering*

**Shri Ram Murti Smarak College of Engineering & Technology, Bareilly**

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

<July, 2020>


# TABLE OF CONTENTS


<b>DECLARATION</b> .....	iv
<b>CERTIFICATE</b> .....	v
<b>ACKNOWLEDGMENT</b> .....	vi
<b>ABSTRACT</b> .....	vii
<b>LIST OF SYMBOLS</b> .....	viii
<b>LIST OF ABBREVIATIONS</b> .....	ix
<b>CHAPTER 1 (Introduction)</b> .....	1
1.1    Introduction.....	1
1.2    Motivation.....	3
1.3    Problem Definition.....	3
1.4    Objective .....	4
<b>CHAPTER 2 (Literature Review)</b> .....	5
<b>CHAPTER 3 (Characteristics &amp; Datasets)</b> .....	7
3.1    Fake News Characterization .....	7
3.2    Datasets .....	7
<b>CHAPTER 4 (Important Terms)</b> .....	9
<b>CHAPTER 5 (Methodology)</b> .....	10
5.1    Data Preprocessing.....	10
5.2    Generating News Feature Vector .....	11
5.2.1    Bag of Words .....	11
5.2.2    TF-IDF .....	12
5.2.3    Count Vectorizer .....	13
5.2.4    Shallow & Deep Syntactical Analysis .....	14
5.2.4    Semantic Analysis.....	15
5.2.5    Combining features to form final news vector .....	15
<b>CHAPTER 6 (Existing &amp; Proposed Systems)</b> .....	18
6.1    Existing Systems.....	18
6.2    Proposed Systems .....	18
<b>CHAPTER 7 (Classification)</b> .....	20

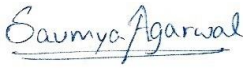
7.1	Naïve Bayes .....	20
7.2	Random Forests .....	21
7.3	Gradient Boosting .....	21
<b>CHAPTER 8 (Building the detector) .....</b>		<b>22</b>
<b>CHAPTER 9 (Result) .....</b>		<b>23</b>
<b>CHAPTER 10 (Conclusion &amp; Future Work) .....</b>		<b>33</b>
10.1	Conclusion .....	33
10.2	Future Work .....	34
<b>REFERENCES.....</b>		<b>35</b>

## ***DECLARATION***

*I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.*

*Signature*   
*Name* Mannat Kaur Nagpal  
*Roll No* 1601410064  
*Date* 30/07/2020

*Signature*   
*Name* Prachi Gupta  
*Roll No* 1601410078  
*Date* 30/07/2020

*Signature*   
*Name* Saumya Agarwal  
*Roll No* 1601410094  
*Date* 30/07/2020

## ***CERTIFICATE***

*This is to certify that the Project Report entitled Design and Development of Fake News Detection System using ML which is submitted by Mannat Kaur Nagpal (1601410064), Prachi Gupta (1601410078.), Saumya Agarwal (1601410094), and is a record of the candidates own work carried out by them under my supervision. The matter embodied in this work is original and has not been submitted for the award of any other work or degree.*

**Dr. L. S. Maurya**  
**HOD (CSE/IT)**

**Ms. Neha Sharma**  
**Project Incharge**

**Mr. Ashish Agarwal**  
**Supervisor**

## ***ACKNOWLEDGEMENT***

*It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to Assistant Professor Mr. Ashish Agarwal, Computer Science, S.R.M.S.C.E.T, Bareilly for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.*

*We also take the opportunity to acknowledge the contribution of Dr. L. S. Maurya, Head, Department of Computer Science & Engineering/Information Technology, S.R.M.S.C.E.T, Bareilly for his full support and assistance during the development of the project.*

*We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.*

Signature 

Name **Mannat Kaur Nagpal**

Roll No 1601410064

Date 30/07/2020

Signature 

Name **Prachi Gupta**

Roll No 16014100xx

Date 30/07/2020

Signature 

Name **Saumya Agarwal**

Roll No 1601410094

Date 30/07/2020

## **ABSTRACT**

*Fake news and hoaxes have been there since before the advent of the Internet. The widely accepted definition of Internet fake news is: fictitious articles deliberately fabricated to deceive readers”. Social media and news outlets publish fake news to increase readership or as part of psychological warfare. In general, the goal is profiting through clickbaits. Clickbaits lure users and entice curiosity with flashy headlines or designs to click links to increase advertisements revenues. This exposition analyzes the prevalence of fake news in light of the advances in communication made possible by the emergence of social networking sites. The purpose of the work is to come up with a solution that can be utilized by users to detect and filter out sites containing false and misleading information. We use simple and carefully selected features of the title and post to accurately identify fake posts. The experimental results show a 99.4% accuracy using logistic classifier.*

## LIST OF SYMBOLS

$[x]$	Integer value of $x$ .
$\neq$	Not Equal
$\square$	Belongs to
$\text{€}$	Euro- A Currency
$-$	Optical distance
$-o$	Optical thickness or optical half thicken



## **LIST OF ABBREVIATIONS**

AAM	Active Appearance Model
ICA	Independent Component Analysis
ISC	Increment Sign Correlation
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristics

# CHAPTER 1

## 1.1 Introduction

Fake news becomes a major issue for the public and government. Fake news can take advantage of multimedia content to mislead readers and get published, which can lead to negative effects or even manipulation of public events. One of the unique challenges of detecting fake news on social media is how to identify fake news about recent events. The task of detecting fake news has tested a variety of labels, from misinformation to rumours; to spam. There has been a large body of work surrounding text analysis of fake news and similar topics such as rumours or spam.

The large use of social media has tremendous impact on our society, culture, business with potentially positive and negative effects. Now-a-days, due to the increase in use of online social networks, the fake news for various commercial and political purposes has been emerging in large numbers and widely spread in the online world. The existing systems are not efficient in giving a precise statistical rating for any given news. Also, the restrictions on input and category of news make it less varied. This project develops a method for automating fake news detection for various events. We are building a classifier that can predict whether a piece of news is fake based on data sources, thereby approaching the problem from a purely Machine Learning perspective.

**“The challenge of misinformation has been prevalent for years now and we still haven’t got our arms around it as a society.”**

For instance, Conroy, Rubin, and Chen [1] have mentioned three types of fake news:

- Serious Fabrications (Type A),
- Large-Scale Hoaxes (Type B),
- Humorous Fakes (Type C)

In simpler words, Fake news is a news article that is intentionally and verifiably false and could mislead readers [2]. This narrow definition is useful in the sense that it is able to eliminate the ambiguity between fake news and other related concepts e.g., hoaxes, and satires. Most of the existing researches have been focused on classifying online news and

social media posts. Different methods have been proposed by different researches for deception detection.

In [3], Mykhailo Granik proposed simple technique for fake news detection the usage of naive Bayes classifier. They used BuzzFeed news for getting to know and trying out the Naïve Bayes classifier. The dataset is taken from facebook news publish and completed accuracy upto seventy four% on test set.

In [4], Cody Buntain advanced a method for automating fake news detection on Twitter. They applied this method to Twitter content sourced from BuzzFeed's fake news dataset. Furthermore, leveraging non-professional, crowdsourced people instead of journalists presents a beneficial and much less costly way to classify proper and fake memories on Twitter rapidly.

Marco L. Della [5] offered a paper which allows us to recognize how social networks and gadget studying (ML) strategies may be used for faux news detection. They have used novel ML fake news detection method and carried out this approach inside a Facebook Messenger chatbot and established it with a actual-world application, acquiring a fake information detection accuracy of eighty one.7%.

In [6], Rishabh Kaushal carried out 3 getting to know algorithms specifically Naive Bayes, Clustering and Decision bushes on some of features such astweet-degree and consumer-level like Followers/ Followees, URLs, SpamWords, Replies and HashTags. Improvement of unsolicited mail detection is measured on the premise of general Accuracy, Spammers Detection Accuracy and Non Spammers Detection Accuracy.

We observe that the performance of models is not dataset invariant and so it is quite difficult to obtain a unique superior model for all datasets. **We have also found that traditional machine learning architecture like Naive Bayes can achieve very high accuracy with proper feature selection.** On a small dataset with less than 100k news articles, Naïve Bayes(with n-gram) can be a primary choice as it achieves similar performance compared to neural network-based high overhead models. The proposed methodology in this project uses the publicly available Liar dataset obtained from the Kaggle.

The most common algorithms used by fake news detection systems includes machine learning algorithms such as Support Vectors Machines, Random Forests, Decision Trees,

Stochastic Gradient Descent, Logistic Regression and so on. In this project we have attempted to implement these algorithms to test and train our results.

**The main emphasis is to obtain maximum accuracy by implementing the most suitable machine learning classification algorithm.**

## **1.2 Motivation**

The extensive spread of fake news can have a serious negative impact on individuals and society. It has brought down the authenticity of news ecosystem as it is even more widely spread on social media than most popular authentic news. It is one of the biggest problems which has the ability to change opinions and influence decisions and interrupts the way in which people responds to real news. It have political influence, can encourages mistrust in legitimate media outlet, influence financial markets, damage to individual's reputation.

During the American presidential elections of 2016, a survey revealed that many young men and teens in Veles were running hundreds of websites which published many false viral stories that supported Trump. These fake news influenced many people which affected the election results. This is just a small instance of how the spread of fake news can influence people.

Many organizations have come forward to stop the spread of fake news.

Eg. Google app uses Artificial Intelligence to select stories and stop fake news.

We aim to develop a model using machine learning and NLP techniques to determine whether a news is fake or real.

## **1.3 Problem Definition**

Fake news is a real menace as it can quickly spread panic among the public. It can also affect major world events, as was seen in the US Presidential Elections. With the flood of news arising from online content generators, as well as various formats and genres, it is

impossible to verify news using traditional fact checkers and vetting. To tackle this problem of quick and accurate classification of news as fake or authentic, we provide a computational tool.

## **1.4 Objectives**

The concept of fake news is not something new. It has been around for ages. But during the 2016 US presidential election, journalists began noticing a rash of viral made-up stories online. Later the way we look at fake news changed! So we decided to build a model to detect fake from factual news stories published around that event. Our goal is to build a model that can classify fake and real news from the subtle yet consistent differences in the language of the two classes of articles.

The major objectives of this project are:

1. Taking the help of linguistic cues to develop a machine learning based model for accurately determining whether the given news is fake or authentic.
2. To get high accuracy to determine a news is fake or true.

## **CHAPTER 2**

### **LITERATURE REVIEW**

In the research paper[8] "Syntactic Stylometry for Deception Detection" by Feng, Song and Banerjee, Ritwik and Choi, Yejin, they focus on the language patterns followed by deceptors in their language. Most liars use their language strategically to avoid being caught, in spite of their attempt to control what they are saying, language “leakage” occurs with certain verbal aspects that are hard to monitor manually such as frequencies and patterns of pronoun, conjunction, and negative emotion word usage. The research focuses on user reviews and essays, but is equally applicable in fake news detection as well, where the author tries to deceive the readers. For detecting these linguistic patterns, the research uses shallow and deep syntax analysis which use POS(parts-of-speech) tags and Probabilistic Context Free Grammars(PCFG) respectively.

In their research paper [9] "Automatic deception detection for detecting fake news" by Conroy, Niall J. and Rubin, Victoria L. and Chen, Yimin, they discussed two methods for Fake News detection. The first one is linguistic approach, this discusses the various syntactical and semantical features that are useful in deception detection. It uses deep syntax analysis with context free grammar generation using stanford parser. The basis of semantic analysis provided in this research is that, the author may use contradictions and omit facts while writing, whereas a writer of a truthful review is more likely to make similar comments about aspects of the product as other truthful reviewers. Network approach depends on querying existing knowledge networks, or publicly available structured data, such as DBpedia ontology, or the Google Relation Extraction Corpus (GREC). It also takes into account that on social media, the authentication of identity of the user posting an article is paramount for the notion of trust.

1. In their research paper [10] by Veronica Pérez-Rosas , Bennett Kleinberg , Alexandra Lefevre Rada Mihalcea<sup>1</sup>, that focused on the automatic identification of fake content in the online news. They introduced two novel datasets for the task of fake news detection, covering seven different news domains. They build the fake news detection models by extracting several linguistic features like n-grams,

punctuations, psycholinguistic features, readability, syntax etc. They used LIWC to generate these features. Their best models achieved accuracies which are similar to the human ability to spot fake news.

2. In their research paper [11] "Evaluating Machine Learning Algorithms for Fake News Detection" by Shlok Gilda, he explored various Natural Language Processing techniques for the detection of fake news detection. He applied techniques like TF-IDF of bi-grams and probabilistic context free grammar (PCFG) detection on a data corpus of about 11,000 articles. He tested the dataset on multiple classification algorithms like SVM, Stochastic Gradient Descent, Gradient Boosting, Bounded Decision Trees, and Random Forests. He found that the Stochastic Gradient Descent model gives an accuracy of 77.2%.
3. In their research paper [7] "Fake News Detection on Social Media : A Data Mining Perspective" by Kai Shu, Amy Sliva, Suhan Wang, Jiliang Tang, Huan Liu, they present a comprehensive review of detecting fake news on social media, including fake news characterizations on psychology and social theories, existing algorithms from a data mining perspective, evaluation metrics and representative datasets. For feature extraction they used : News Content Features - Linguist and Visual based. Social Context Features - It include features from users, posts and network. and for Model Constructions they used: News Content Models - knowledge based and style based. Social Context Models - Stance and propagation based.
4. In their research paper [12] "Automatic detection of deception in child-produced speech using syntactic complexity features" by Maria Yancheva, Frank Rudzicz, the evaluations are done using a novel set of syntactic features, including measures of complexity. Their results show that sentence length, the mean number of clauses per utterance, and the Stajner Mitkov measure of complexity are highly informative syntactic features.

## CHAPTER 3

### 3.1 Fake News Characterization

Fake news definition is made of two parts: authenticity and intent. Authenticity means that fake news content false information that can be verified as such, which means that conspiracy theory is not included in fake news as there are difficult to be proven true or false in most cases. The second part, intent, means that the false information has been written with the goal of misleading the reader.

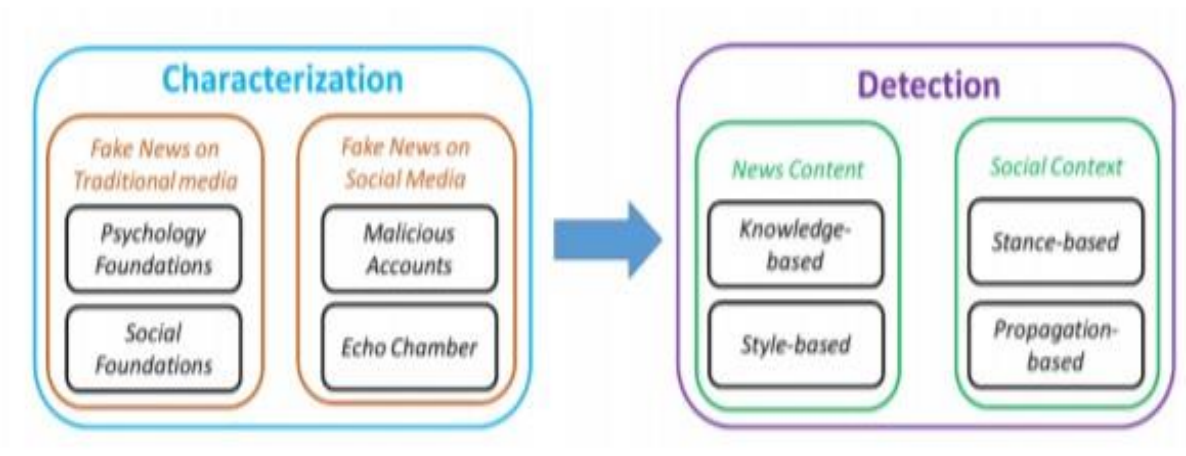


Figure 1.1: Fake news on social media: from characterization to detection.

### 3.2 Datasets

We have 3 datasets for fake news detection, we will use the required attributes of these datasets to train our model.

- Getting Real about Fake News Dataset- Kaggle
- Fake News Detection Dataset- Kaggle



- Fake News Dataset - Kaggle

The total datasets have about 37,808 data points. But after preprocessing the data, (i.e) removing duplicates and removing data points which are NULL, there are 27865 data points out of which 15343 articles are REAL and 12522 articles are FAKE.

The distribution of data points in the dataset is as follows:

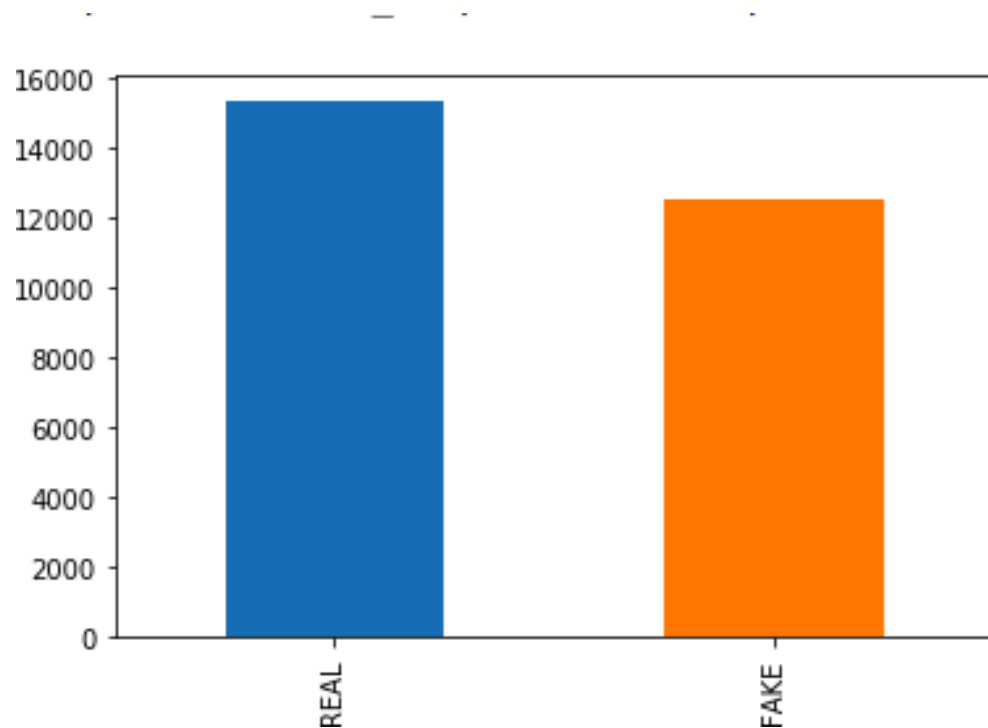


Figure 1: Distribution of datasets

## **CHAPTER 4**

### **IMPORTANT TERMS**

#### **1. Pickle**

“*Pickling*” is the process whereby a Python object hierarchy is converted into a byte stream, and “*unpickling*” is the inverse operation, whereby a byte stream is converted back into an object hierarchy.

#### **2. Flask App**

It is a Web Server Gateway Interface (WSGI) web application framework designed to create web apps. It is super convenient and the code is simple.

#### **3. Newspaper3k Library**

Newspaper is an amazing python library for extracting & curating articles.

#### **4. TF-IDF**

Term Frequency-Inverse Document Frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

#### **5. Multinomial Naive Bayes**

It relies on Relies on very simple representation of document as a ‘Bag of words’. It estimates the conditional probability of a particular word given a class as the relative frequency of term  $t$  in documents belonging to class( $c$ ), in our case Fake or Real.

## **CHAPTER 5**

# METHODOLOGY

The approach proposed for this project is:

1. Data Preprocessing
2. Generating News Feature Vector
3. Classification

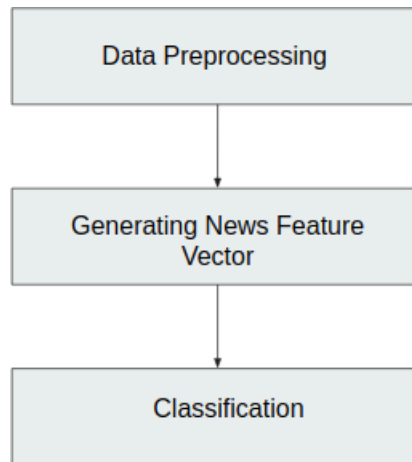


Figure 2: Flowchart

## 5.1 Data Preprocessing

Data preprocessing is done to convert the raw data into a required format. Data preprocessing can be done by various methods like data cleaning, data reduction, data integration etc. In this project, the datasets are collected from different resources which have different formats and attributes. Hence, the data can be duplicate and they may contain some attributes which are not useful. So, we convert the data into our required format with required attributes which are used to train our model.

## **5.2 Generating News Feature Vector**

The most important part of detecting if a given news is fake or not is to convert the news article into a news vector which contains the important features which are used to determine the nature of the news.

There are several ways to generate this feature vector [9] [10] . We tried different approaches for the same to determine which method gives the best accuracy. Some of the methods are:

### **5.2.1 Bag of Words**

Bag of words is a way of representing text in a format which can be easily processed by the machine learning algorithms. BoW is one of the ways of extracting features from text. In this type of text representation, majorly 2 things are involved:

1. Vocabulary of known words.
2. Measure of the presence of known words

In BoW, representation, the order of words or structure of the sentence is not considered, it is only concerned with whether the word is present in the document or not. The BoW is carried out as follows:

1. Collect all the unique words in all the documents.
2. Now, represent the documents in a vector of all the unique words by counting the number of times each word appears in that document.

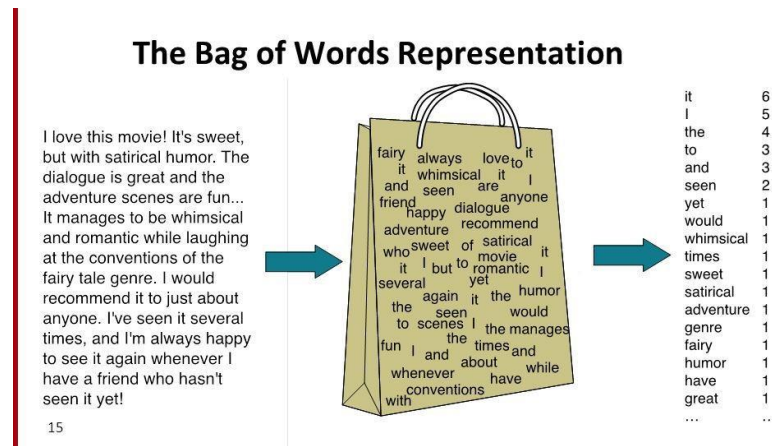


Figure 3: Example of Bag of Words

There is another type of BoW named binary BoW, in which instead of count of the count of each word, it just checks whether a word is in that document and then represent with a 0 or 1.

## 5.2.2 TF-IDF

TF-IDF stands for term frequency-inverse document frequency. TF-IDF is a method used to represent text in a format which can be easily processed by the machine learning algorithms. It is a numerical statistic that shows how important a word is to a document in a word corpus. The importance of a word is proportional to the number of times the word appears in the document but inversely proportional to the the number of times the word appears in the corpus. The TF-IDF weight is composed of 2 terms:

1. **TF (Term Frequency):** Term frequency is defined as the frequency of a word in the document. TF is calculated as:

$$TF(w) = (\text{Number of times word 'w' appears in the document}) / (\text{Total number of words in the document}).$$

2. **IDF (Inverse Document Frequency):** This measures how important a word is in the document. For example, words like and, of, the, a appears lot of times but they are less important. Thus most repeated terms are given less weights and less frequent terms are given more weights. IDF is calculated as:  $IDF(w) = \log_e(\text{Total number of$

documents / Number of documents with word 'w' in it).

The TF-IDF weight is given to each word by calculating TF\*IDF values.

$$w_{i,j} = tf_{i,j} \times \log \left( \frac{N}{df_i} \right)$$

$tf_{i,j}$  = number of occurrences of  $i$  in  $j$   
 $df_i$  = number of documents containing  $i$   
 $N$  = total number of documents

Figure 4: Formula for TF-IDF

For generating the news vector, we calculate the tf-idf values of the bigrams and represent the tf-idf vector of that bigrams. We choose bigrams over unigrams because it gives the context.

**n-grams** : n-gram is a contiguous sequence of n terms from a given text. An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram", size 3 as a "trigram" and so on with larger n, a model can store more context.

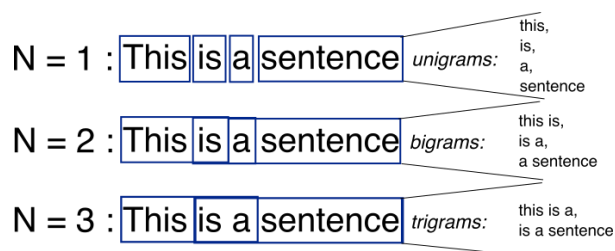


Figure 5: Example of n-grams

### 5.2.3 Count Vectorizer

The CountVectorizer provides a simple way to both tokenize a collection of text

documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. You can use it as follows: Create an instance of the `CountVectorizer` class.

We can use it as follows:

1. Create an instance of the *CountVectorizer* class.
2. Call the *fit()* function in order to learn a vocabulary from one or more documents.
3. Call the *transform()* function on one or more documents as needed to encode each as a vector.

An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.

Because these vectors will contain a lot of zeros, we call them sparse. Python provides an efficient way of handling sparse vectors in the `scipy.sparse` package. The vectors returned from a call to `transform()` will be sparse vectors, and you can transform them back to numpy arrays to look and better understand what is going on by calling the `toarray()` function.

#### **5.2.4 Shallow and Deep syntactical Analysis**

We generated POS (part-of-speech) tags using the Spacy library. Our POS features will be encoded as tf-idf values for each for these tags. Research paper by Feng et al [8] states that, even though POS tags are effective in detecting fake product reviews, they are not as effective as words. Therefore, we strengthen POS features with unigram/bigram features.

For deep syntactical analysis we used the Stanford/Berkeley parser to generate CFG rules for the sentences and we encoded these rules with tf-idf values for each production rule.

### 5.2.5 Semantic Analysis

A widely used open-source resource for incorporating semantic information is Empath (developed by Stanford). Empath is a lexicon of words grouped into semantic categories relevant to psychological processes. Several research works have relied on semantic analysis to build deception models using machine learning approaches and showed that the use of semantic information is helpful for the automatic identification of deceit. Empath has 194 semantic categories, some of these semantic classes are emotional tone (positive or negative), anger, nervousness.

We get a score between 0-100 for each semantic class. The lexicon we get is converted to a TF-IDF vector by taking the score for a semantic class (like nervousness) as its frequency.

### 5.2.6 Combining features to form final news vector

We considered 3 methods for generating feature vectors:

1. TF-IDF bigram vector of the news article.
2. Feature Vector generated by Syntax Analysis of the news article.
3. Feature Vector generated by the semantic analysis of the news article.

After generating these features and generating their individual feature vector, we have to combine these features to form the final news vector on which classification is performed.

The method we approached for combining the feature vectors is:

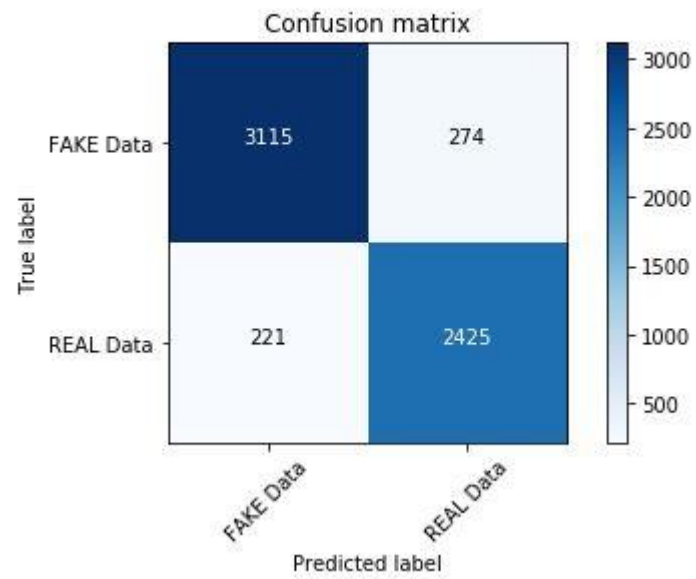
1. Take the most important features for the 3 feature vectors.
2. Assign weights to each vector and then take the weighted combination of the 3 feature vectors to generate the final feature vector. If  $x$  is the weight corresponding to the first feature vector,  $y$  for the second, and  $1-x-y$  for the third. The final feature vector will be the linear combination of these feature vectors multiplied by their corresponding weights.



## Confusion Matrix for TF-IDF

accuracy: 0.918

Confusion matrix, without normalization



## Performance Metrics

### Performance Metrics And Accuracy

```
In [107]: y_pred=model.predict_classes(X_test)
```

```
In [108]: from sklearn.metrics import confusion_matrix
```

```
In [109]: confusion_matrix(y_test,y_pred)
```

```
Out[109]: array([[3140, 279],
                 [ 278, 2338]], dtype=int64)
```

```
In [110]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

```
Out[110]: 0.907705053852527
```

```
In [ ]: from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import re
ps = PorterStemmer()
corpus = []
for i in range(0, len(messages)):
    review = re.sub('[^a-zA-Z]', ' ', messages['text'][i])
    review = review.lower()
    review = review.split()

    review = [ps.stem(word) for word in review if not word in stopwords.words('english')]
    review = ' '.join(review)
    corpus.append(review)
```

```
In [24]: corpus[3]
```

```
Out[24]: 'civilian kill singl us airstrik identifi'
```

```
In [ ]:
```

```
In [17]: ## TFidf Vectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_v=TfidfVectorizer(max_features=5000,ngram_range=(1,3))
X=tfidf_v.fit_transform(corpus).toarray()
```

# CHAPTER 6

## 6.1 Existing System

There exists a large body of research on the topic of machine learning methods for deception detection, most of it has been focusing on classifying online reviews and publicly available social media posts. Particularly since late 2016 during the American Presidential election, the question of determining 'fake news' has also been the subject of particular attention within the literature.

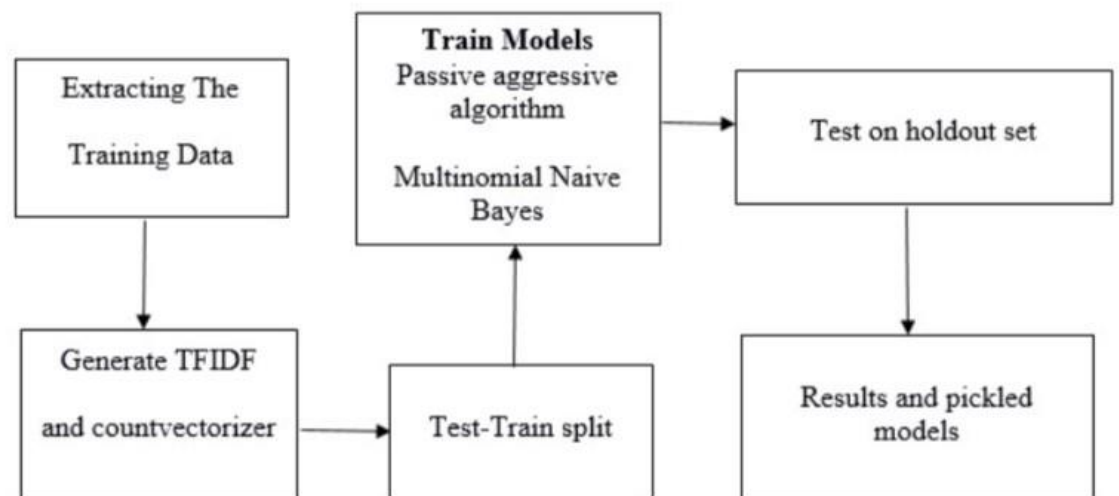
Conroy, Rubin, and Chen <sup>[1]</sup> outlines several approaches that seem promising towards the aim of perfectly classify the misleading articles. They note that simple content-related n-grams and shallow parts-of-speech (POS) tagging have proven insufficient for the classification task, often failing to account for important context information. Rather, these methods have been shown useful only in tandem with more complex methods of analysis. Deep Syntax analysis using Probabilistic Context Free Grammars (PCFG) have been shown to be particularly valuable in combination with n-gram methods. Feng, Banerjee, and Choi <sup>[2]</sup> are able to achieve 85%-91% accuracy in deception related classification tasks using online review corpora.

Feng and Hirst implemented a semantic analysis looking at 'object:descriptor' pairs for contradictions with the text on top of Feng's initial deep syntax model for additional improvement. Rubin, Lukoianova and Tatiana analyze rhetorical structure using a vector space model with similar success. Ciampaglia et al. employ language pattern similarity networks requiring a pre-existing knowledge base.

## 6.2 Proposed System

In this paper a model is build based on the count vectorizer or a tfidf matrix ( i.e ) word tallies relatives to how often they are used in other artices in your dataset ) can help . Since this problem is a kind of text classification, Implementing a Naive Bayes classifier will be best as this is standard for text-based processing. The actual goal is in developing a model which was the text transformation (count vectorizer vs tfidf vectorizer) and

choosing which type of text to use (headlines vs full text). Now the next step is to extract the most optimal features for countvectorizer or tfidf-vectorizer, this is done by using a n-number of the most used words, and/or phrases, lower casing or not, mainly removing the stop words which are common words such as “the”, “when”, and “there” and only using those words that appear at least a given number of times in a given text dataset.



# CHAPTER 7

## CLASSIFICATION

After generating the news feature vector, now we classify the vector to whether it is fake or real. We aim to use the following classification algorithms [5] for the purpose of classification:

### 7.1 Naive Bayes

Naive bayes is a supervised learning algorithm which is used for classification. It is based on bayes theorem assuming that features are independent of each other. It calculates the probability of every class, the class with maximum probability is chosen as the output.

The basic idea of Naive-Bayes model is that all features are independent of each other. This is a particularly strong hypothesis in the case of text classification because it supposes that words are not related to each other. But it knows to work well given this hypothesis. Given an element of class  $y$  and vector of features  $X = (x_1, \dots, x_n)$ . The probability of the class given that vector is defined as

$$P(y|X) = \frac{P(y) * P(X|y)}{P(X)}$$

Thanks to the assumption of conditional independence, we have that

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y)$$

Using Bayes rules we have that

$$P(y|x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^n P(x_i|y)}{P(x_1, \dots, x_n)}$$

Because  $P(x_1, \dots, x_n)$  is constant, we have the classification rule

$$\hat{y} = \underset{y}{argmax} P(y) \prod_{i=1}^n P(x_i|y)$$

## 7.2 Random Forests

Random Forests is a bagging type of ensemble model in which the base model for bagging is decision tree. In addition to bagging (i.e taking random samples of total data and train those samples independently and then take the majority voting of numerous samples of the total dataset to find the output of the total dataset), there is feature bagging (i.e column sampling in which not all the columns/features are taken into consideration while training but rather random samples of features are considered while training different samples).

Random Forest = Bagging with decision tree as base model + feature bagging

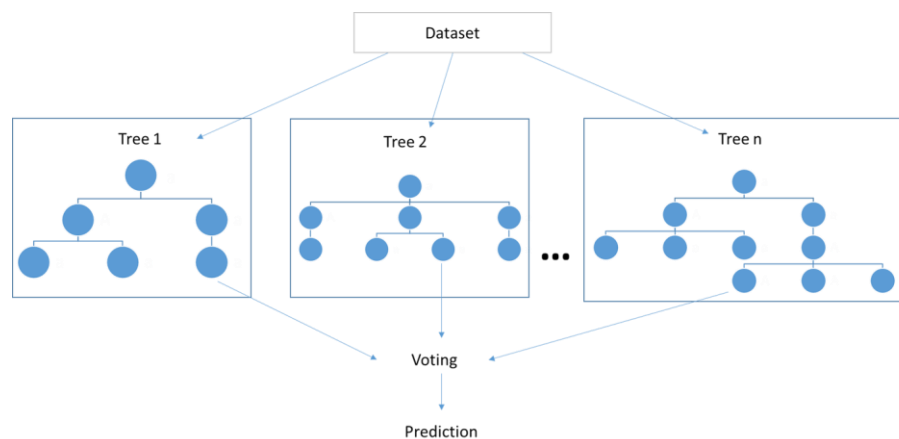
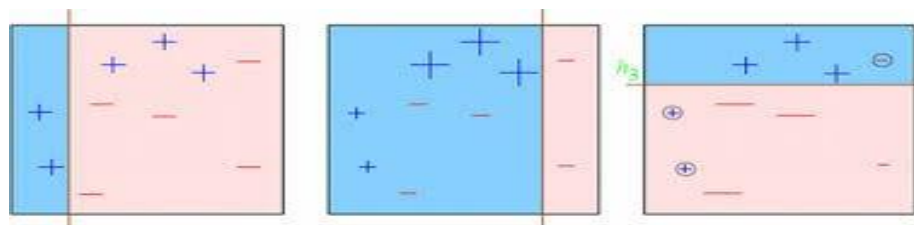


Figure 6: Example of Random Forests

## 7.3 Gradient Boosting

Gradient boosting is an example of boosting ensemble model. Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. Boosting tries to convert a weak learner to become better. It learns from its mistakes/error and



tries to reduce the error.

Figure 7: Example of Gradient Boosti

# CHAPTER 8

## BUILDING A DETECTOR

The dataset we chose had real news articles from center, right and left affiliations and fake news articles flagged as fake by [PolitiFact](#), a fact-checking website. It included 6K+ news articles with equal number of real and fake news. Having a balanced dataset, I chose accuracy as my performance metric.

Working with text data is extremely challenging as it is complex, mysterious, does not follow consistent rules, yet, it is very intriguing in its own way! To enable computer to interact with the news articles involved multiple steps.

Like any data science project, the first step was preprocessing and scrubbing the data as clean as possible. This included filtering out non English words and stop-words. Stop-words are the most common words in English language which don't necessarily contribute much to the context of the text, like propositions, for instance. From a huge chunk of text data, now the dataset is tokens aka words that contribute to the meaning of that document.

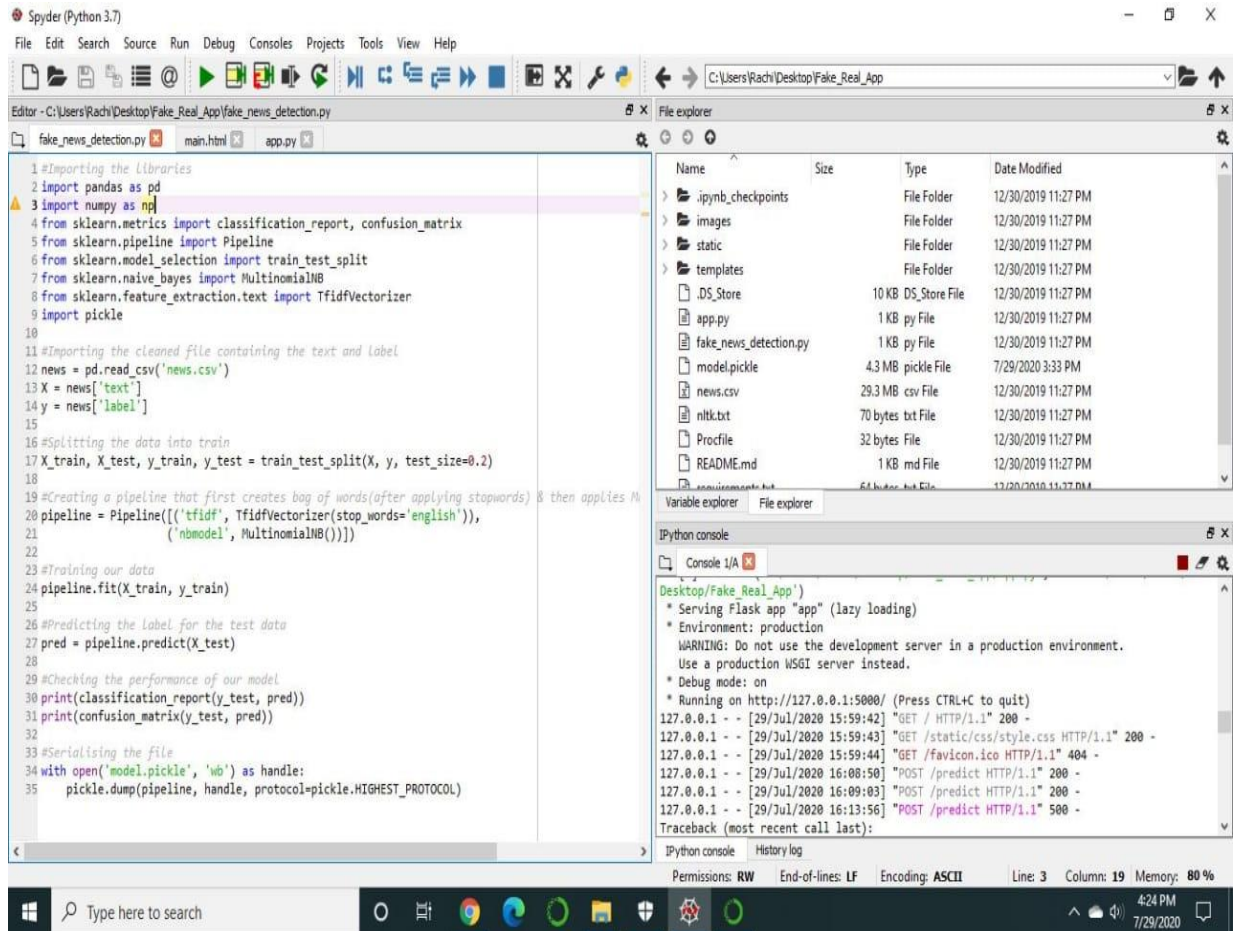
In order to transform these tokens into numbers, I used count vectorizer. This means that the words got allotted a number according to their occurrence in the entire corpus — 1 when it appears in an article and 0 otherwise.

We fed these numbers into various models and evaluated their performance based on their accuracy.

# CHAPTER 9

## RESULTS

### ML Pipelining Code



The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `fake_news_detection.py` with the following code:

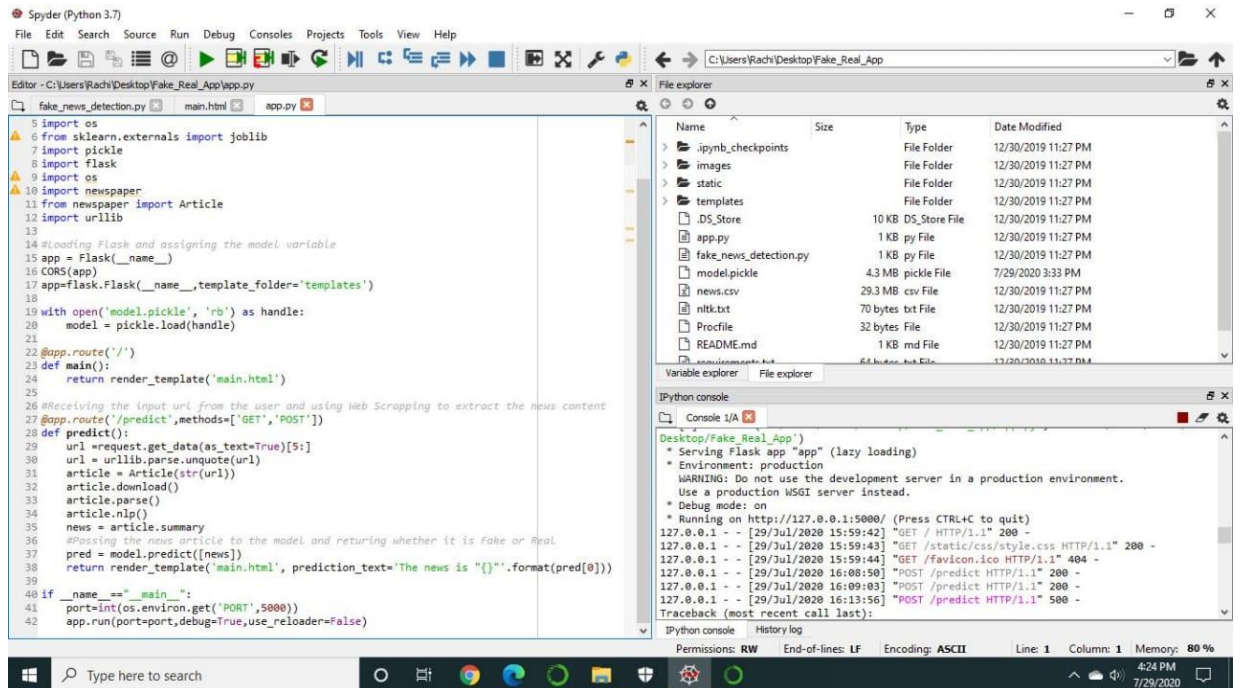
```
1 #Importing the Libraries
2 import pandas as pd
3 import numpy as np
4 from sklearn.metrics import classification_report, confusion_matrix
5 from sklearn.pipeline import Pipeline
6 from sklearn.model_selection import train_test_split
7 from sklearn.naive_bayes import MultinomialNB
8 from sklearn.feature_extraction.text import TfidfVectorizer
9 import pickle
10
11 #Importing the cleaned file containing the text and label
12 news = pd.read_csv('news.csv')
13 X = news['text']
14 y = news['label']
15
16 #Splitting the data into train
17 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
18
19 #Creating a pipeline that first creates bag of words(after applying stopwords) & then applies NB
20 pipeline = Pipeline([('tfidf', TfidfVectorizer(stop_words='english')),
21                      ('nbmodel', MultinomialNB())])
22
23 #Training our data
24 pipeline.fit(X_train, y_train)
25
26 #Predicting the Label for the test data
27 pred = pipeline.predict(X_test)
28
29 #Checking the performance of our model
30 print(classification_report(y_test, pred))
31 print(confusion_matrix(y_test, pred))
32
33 #Serialising the file
34 with open('model.pickle', 'wb') as handle:
35     pickle.dump(pipeline, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

The right-hand pane of the IDE is divided into two sections. The top section, labeled "File explorer", shows a file tree for the project located at `C:\Users\Rach\Desktop\Fake_Real_App`. It lists various files and folders, including `.ipynb_checkpoints`, `images`, `static`, `templates`, `.DS_Store`, `app.py`, `fake_news_detection.py`, `model.pickle`, `news.csv`, `nlkt.txt`, `Profile`, and `README.md`. The bottom section, labeled "Python console", shows the output of a Flask application running on `http://127.0.0.1:5000/`. The console output includes a warning about using a development server in a production environment and a list of HTTP requests and responses:

```
Desktop/Fake_Real_App')
* Serving Flask app "app" (lazy loading)
* Environment: production
WARNING: Do not use the development server in a production environment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [29/Jul/2020 15:59:42] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [29/Jul/2020 15:59:43] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [29/Jul/2020 15:59:44] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [29/Jul/2020 16:00:50] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [29/Jul/2020 16:09:03] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [29/Jul/2020 16:13:56] "POST /predict HTTP/1.1" 500 -
Traceback (most recent call last):
```

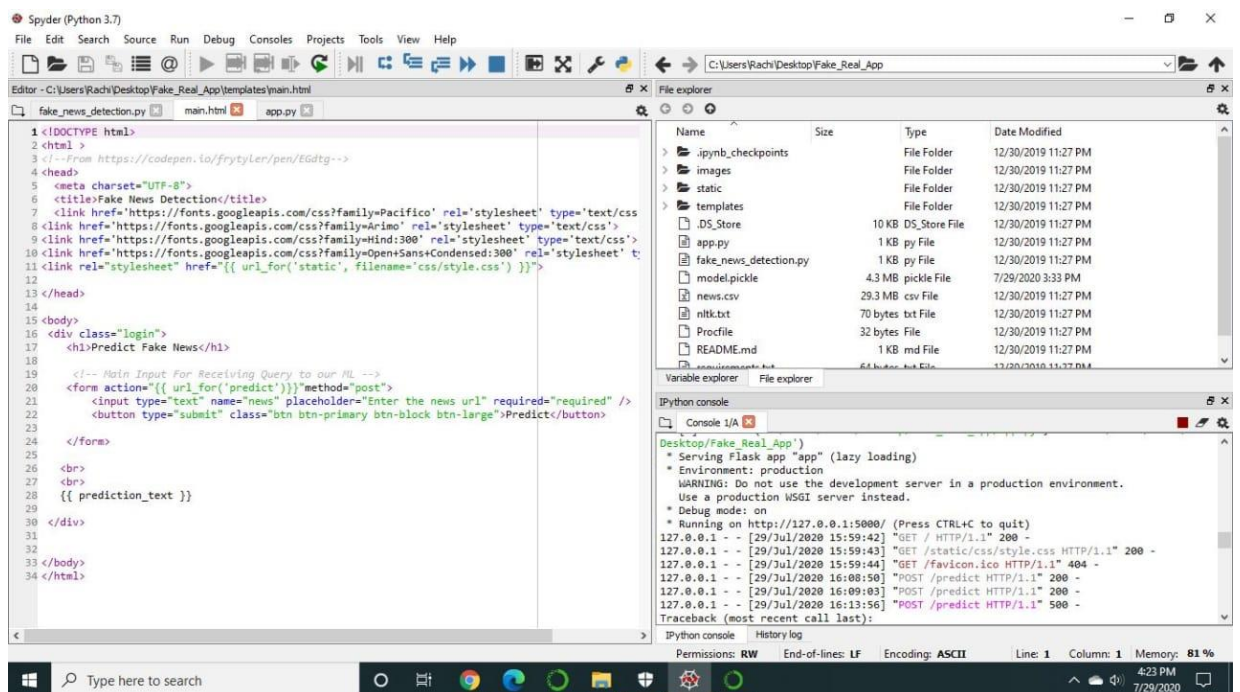


# Flask Code



```
5 import os
6 from sklearn.externals import joblib
7 import pickle
8 import Flask
9 import os
10 import newspaper
11 from newspaper import Article
12 import urllib
13
14 #Loading Flask and assigning the model variable
15 app = Flask(__name__)
16 CORS(app)
17 app = flask.Flask(__name__, template_folder='templates')
18
19 with open('model.pickle', 'rb') as handle:
20     model = pickle.load(handle)
21
22 @app.route('/')
23 def main():
24     return render_template('main.html')
25
26 #Receiving the input url from the user and using Web Scrapping to extract the news content
27 @app.route('/predict', methods=['GET', 'POST'])
28 def predict():
29     url = request.get_data(as_text=True)[5:]
30     url = urllib.parse.unquote(url)
31     article = Article(str(url))
32     article.download()
33     article.parse()
34     article.nlp()
35     news = article.summary
36     #Passing the news article to the model and returning whether it is Fake or Real
37     pred = model.predict([news])
38     return render_template('main.html', prediction_text='The news is {}'.format(pred[0]))
39
40 if __name__ == '__main__':
41     port = int(os.environ.get('PORT', 5000))
42     app.run(port=port, debug=True, use_reloader=False)
```

# HTML Interface Code



```
1 <!DOCTYPE html>
2 <html>
3 <!-- from https://codepen.io/frytyler/pen/EGdtg-->
4 <head>
5 <meta charset="UTF-8">
6 <title>Fake News Detection</title>
7 <link href="https://fonts.googleapis.com/css?family=Pacifico" rel="stylesheet" type="text/css">
8 <link href="https://fonts.googleapis.com/css?family=Arimo" rel="stylesheet" type="text/css">
9 <link href="https://fonts.googleapis.com/css?family=Indie" rel="stylesheet" type="text/css">
10 <link href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" rel="stylesheet" type="text/css">
11 <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
12 </head>
13
14
15 <body>
16 <div class="login">
17 <h1>Predict Fake News</h1>
18
19 <!-- Main Input For Receiving Query to our ML -->
20 <form action="{{ url_for('predict') }}" method="post">
21 <input type="text" name="news" placeholder="Enter the news url" required="required" />
22 <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
23
24 </form>
25
26 <br>
27 <br>
28 {{ prediction_text }}
29 </div>
30
31 </body>
32 </html>
```

## Count Vectorizer

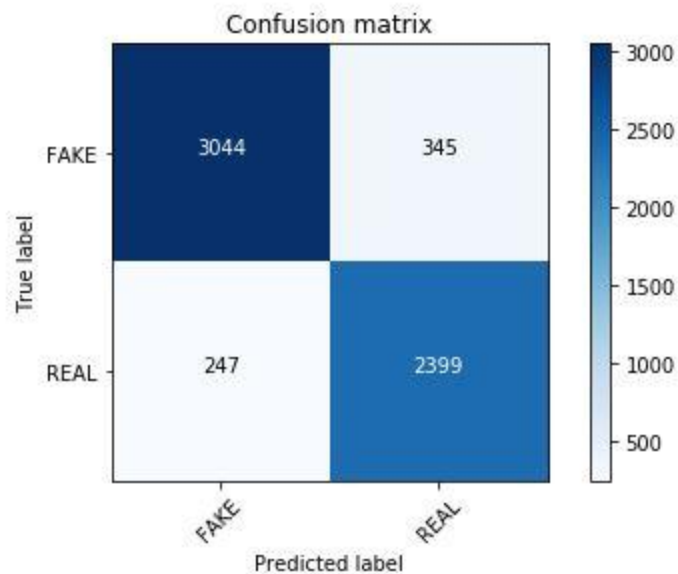
```
In [79]: from sklearn.naive_bayes import MultinomialNB  
classifier=MultinomialNB()
```

```
In [96]: from sklearn import metrics  
import numpy as np  
import itertools
```

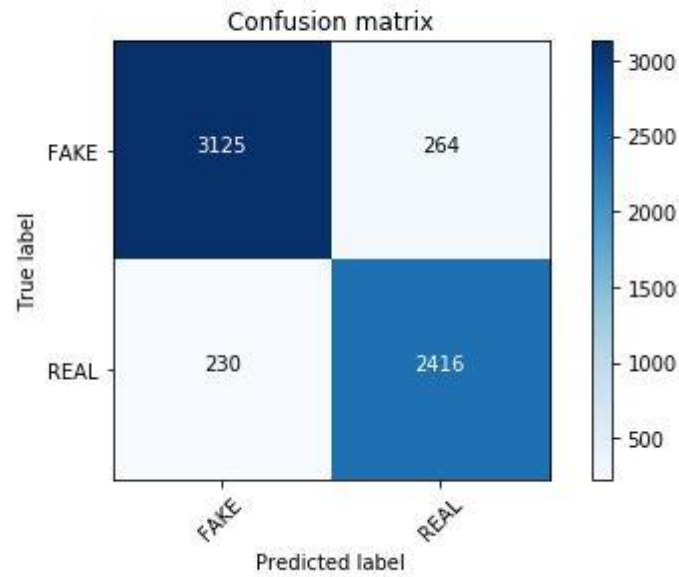
```
In [97]: classifier.fit(X_train, y_train)  
pred = classifier.predict(X_test)  
score = metrics.accuracy_score(y_test, pred)  
print("accuracy:  %0.3f" % score)  
cm = metrics.confusion_matrix(y_test, pred)  
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
```

accuracy: 0.902

Confusion matrix, without normalization



accuracy: 0.918  
Confusion matrix, without normalization



## Bidirectional Results

In [1]: `import pandas as pd`

In [2]: `df=pd.read_csv('train/train.csv')`

In [3]: `df.head()`

Out[3]:

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucas	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 Aistr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

```
In [10]: import tensorflow as tf
```

```
In [11]: tf.__version__
```

```
Out[11]: '2.2.0'
```

```
In [61]: from tensorflow.keras.layers import Embedding
         from tensorflow.keras.preprocessing.sequence import pad_sequences
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.preprocessing.text import one_hot
         from tensorflow.keras.layers import LSTM
         from tensorflow.keras.layers import Dense
         from tensorflow.keras.layers import Bidirectional
         from tensorflow.keras.layers import Dropout
```

```
In [13]: ### Vocabulary size
         voc_size=5000
```

## Model Training

```
In [66]: ### Finally Training
         model1.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=10,batch_size=64)
```

```
Epoch 1/10
192/192 [=====] - 7s 35ms/step - loss: 0.3051 - accuracy: 0.8561 - val_loss: 0.2011 - val_accuracy: 0.9167
Epoch 2/10
192/192 [=====] - 5s 26ms/step - loss: 0.1404 - accuracy: 0.9451 - val_loss: 0.2111 - val_accuracy: 0.9180
Epoch 3/10
192/192 [=====] - 5s 26ms/step - loss: 0.0987 - accuracy: 0.9635 - val_loss: 0.2341 - val_accuracy: 0.9171
Epoch 4/10
192/192 [=====] - 6s 30ms/step - loss: 0.0755 - accuracy: 0.9731 - val_loss: 0.2541 - val_accuracy: 0.9095
Epoch 5/10
192/192 [=====] - 7s 35ms/step - loss: 0.0536 - accuracy: 0.9816 - val_loss: 0.3352 - val_accuracy: 0.9147
Epoch 6/10
192/192 [=====] - 6s 33ms/step - loss: 0.0392 - accuracy: 0.9869 - val_loss: 0.3757 - val_accuracy: 0.9125
Epoch 7/10
192/192 [=====] - 7s 34ms/step - loss: 0.0323 - accuracy: 0.9895 - val_loss: 0.3839 - val_accuracy: 0.9082
Epoch 8/10
192/192 [=====] - 6s 32ms/step - loss: 0.0175 - accuracy: 0.9947 - val_loss: 0.5377 - val_accuracy: 0.9044
Epoch 9/10
192/192 [=====] - 7s 34ms/step - loss: 0.0138 - accuracy: 0.9960 - val_loss: 0.4772 - val_accuracy: 0.9171
Epoch 10/10
192/192 [=====] - 6s 34ms/step - loss: 0.0203 - accuracy: 0.9935 - val_loss: 0.4786 - val_accuracy: 0.9065
```

```
Out[66]: <tensorflow.python.keras.callbacks.History at 0x1d50e386f60>
```



## Performance Metrics And Accuracy

```
In [67]: y_pred1=model1.predict_classes(X_test)
```

```
In [ ]:
```

```
In [68]: from sklearn.metrics import confusion_matrix
```

```
In [69]: confusion_matrix(y_test,y_pred1)
```

```
Out[69]: array([[3093,  326],
                [ 238, 2378]], dtype=int64)
```

```
In [70]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred1)
```

```
Out[70]: 0.9065451532725767
```

```
In [71]: from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred1))
```

```

                precision    recall  f1-score   support

     0           0.93       0.90      0.92       3419
     1           0.88       0.91      0.89       2616

 accuracy                   0.91       0.91       0.91       6035
 macro avg           0.90      0.91      0.91       6035
 weighted avg        0.91      0.91      0.91       6035
```

## LTSM

```
In [3]: df=pd.read_csv('train/train.csv')
```

```
In [4]: df.head()
```

```
Out[4]:
```

	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 Aistr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

## Performance Metrics And Accuracy

```
In [107]: y_pred=model.predict_classes(X_test)
```

```
In [108]: from sklearn.metrics import confusion_matrix
```

```
In [109]: confusion_matrix(y_test,y_pred)
```

```
Out[109]: array([[3140,  279],
                 [ 278, 2338]], dtype=int64)
```

```
In [110]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

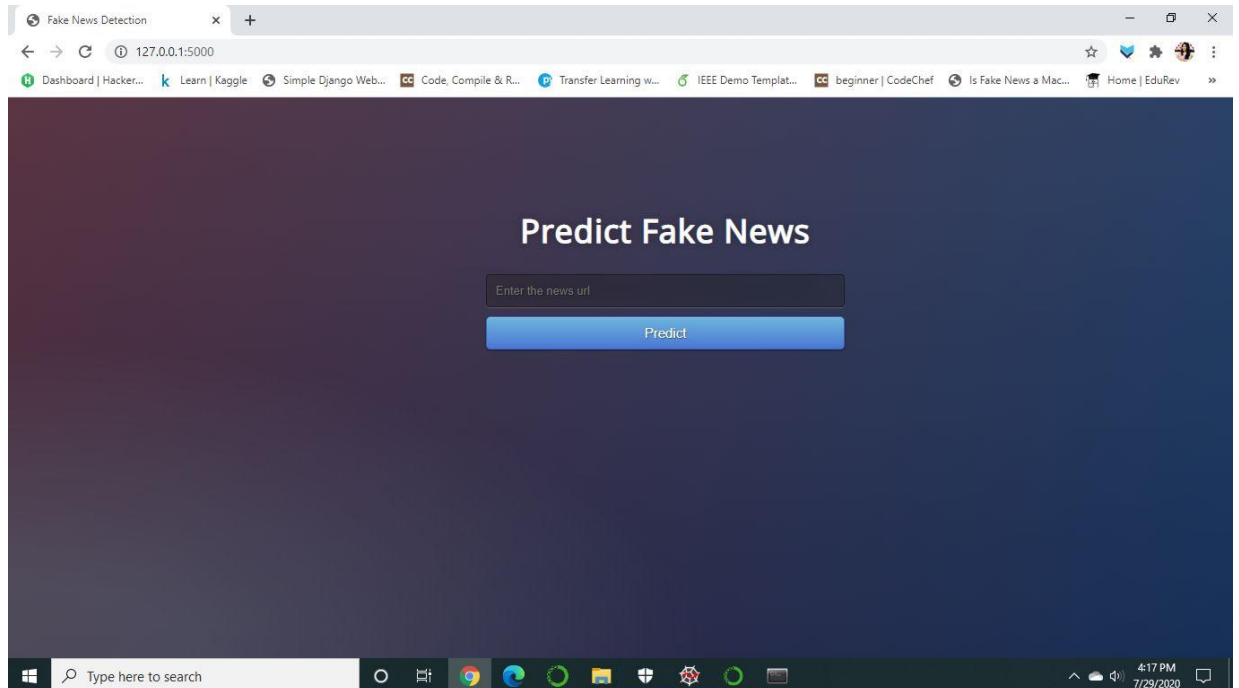
```
Out[110]: 0.907705053852527
```

```
In [ ]:
```

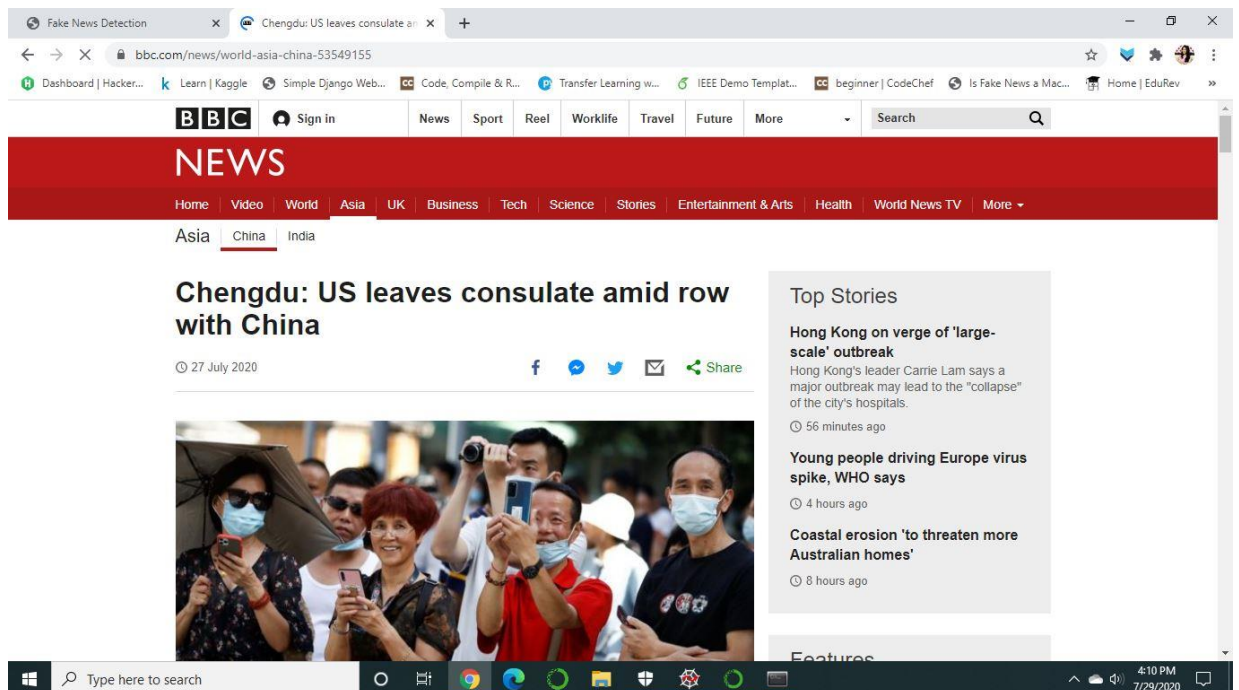
## Adding Dropout

```
In [ ]: from tensorflow.keras.layers import Dropout
        ## Creating model
        embedding_vector_features=40
        model=Sequential()
        model.add(Embedding(voc_size,embedding_vector_features,input_length=sent_length))
        model.add(Dropout(0.3))
        model.add(LSTM(100))
        model.add(Dropout(0.3))
        model.add(Dense(1,activation='sigmoid'))
        model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
```

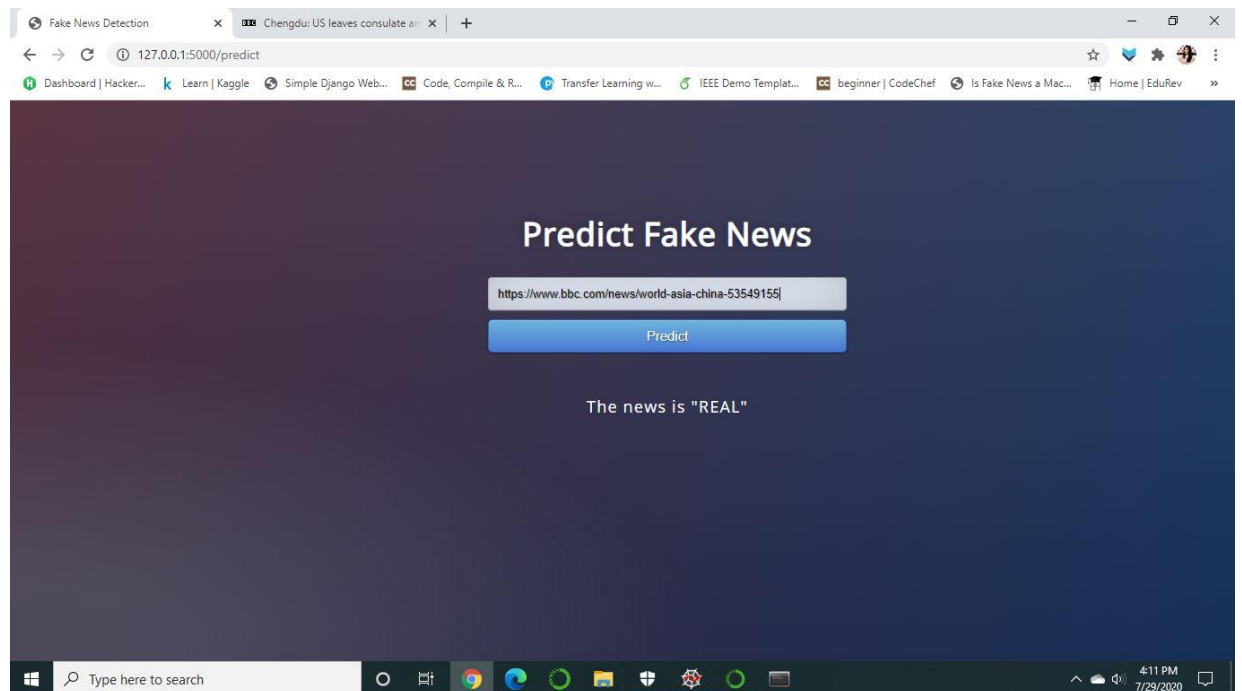
# Web App Interface Using Flask



## Input



# Output

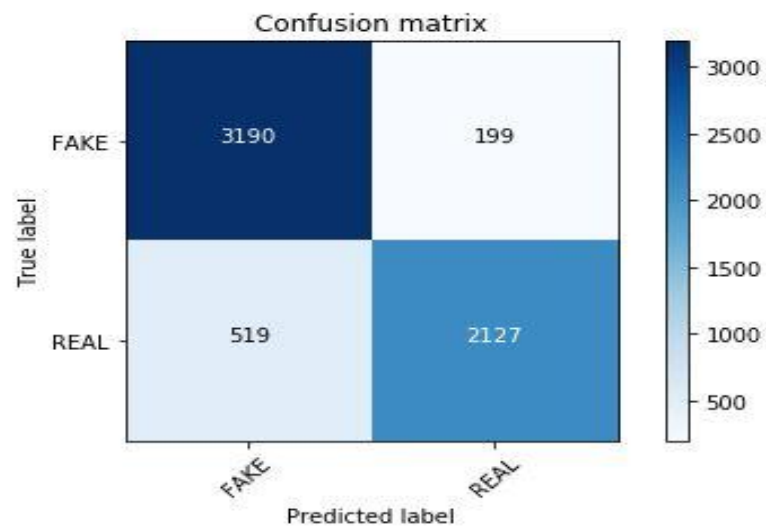




## Confusion Matrix

```
In [32]: classifier.fit(X_train, y_train)
pred = classifier.predict(X_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
```

accuracy: 0.881  
Confusion matrix, without normalization



# CHAPTER 10

## 10.1 Conclusion

We achieved the best result with an accuracy of 88.1% for feature vectors derived by bigrams, syntax and semantic analysis. Thus we conclude that linguistic features are pivotal in detecting whether an article is real or fake.

Data exploration has shown that there is no real statistical differences between text metadata for fake and reliable news, and thus make it not interesting for using it for classifying new texts. The main contribution of this project is support for the idea that machine learning could be useful in a novel way for the task of classifying fake news.

This could also be useful in researchers trying to develop improved models through the use of improved and enlarged datasets, different parameters, etc. The application also provides a way to see manually how changes in the body text affect the classification.

This application could be a tool for humans trying to classify fake news, to get indications of which words might cue them into the correct classification. It could also be useful in researchers trying to develop improved models through the use of improved and enlarged datasets, different parameters, etc. The application also provides a way to see manually how changes in the body text affect the classification.

## 10.2 Future Work

Through the work done in this project, we have shown that machine learning certainly does have the capacity to pick up on sometimes subtle language patterns that may be difficult for humans to pick up on. The next steps involved in this project come in three different aspects. The first of aspect that could be improved in this project is augmenting and increasing the size of the dataset. We feel that more data would be beneficial in ridding the model of any bias based on specific patterns in the source. There is also question as to whether or not the size of our dataset is sufficient.

Our research focuses on daily news articles which have on average around 1000 words. It is difficult to detect linguistic cues in single (or few) statement news. Some other method can be researched upon for these cases.

For designing a fake news detector for social media like Facebook or twitter, we can take into account the user information, user authenticity and origin of the news.

Another aspect in which this project could be expanded is by comparing it to humans performing the same task. Comparing the accuracies would be beneficial in deciding whether or not the dataset is representative of how difficult the task of separating fake from real news is. If humans are more accurate than the model, it may mean that we need to choose more deceptive fake news examples. Because we acknowledge that this is only one tool in a toolbox that would really be required for an end-to-end system for classifying fake news, we expect that its accuracy will never reach perfect.

# REFERENCES

- [1]Victoria L Rubin, Yimin Chen, and Niall J Conroy. Deception detection for news: three types of fakes. In Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community, page 83. American Society for Information Science, 2015
- [2]Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–36, 2017.
- [3]Mykhailo Granik, Volodymyr Mesyura, “Fake News Detection Using Naive Bayes Classifier”, 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)
- [4]Cody Buntain, Jennifer Golbeck, “Automatically Identifying Fake News in PopularTwitter Threads”, 2017 IEEE International Conference on Smart Cloud.
- [5]Marco L. Della Vedova, Eugenio Tacchini, Stefano Moret, Gabriele Ballarin, Massimo DiPierro, Luca de Alfaro, “Automatic Online Fake News Detection Combining Content and Social Signals”, ISSN 2305-7254,2017.
- [6]Arushi Gupta, Rishabh Kaushal, “Improving Spam Detection in Online Social Networks”, 978-1-4799-7171-8/15/\$31.00 ©2015 IEEE.
- [7]K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, “Fake news detection on social media: A data mining perspective,” *CoRR*, vol. abs/1708.01967, 2017.
- [8]S. Feng, R. Banerjee, and Y. Choi, “Syntactic stylometry for deception detection,” in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL ’12, (Stroudsburg, PA, USA), pp. 171–175, Association for Computational Linguistics, 2012.

- [9]N. J. Conroy, V. L. Rubin, and Y. Chen, “Automatic deception detection: Methods for finding fake news,” in *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*, ASIST ’15, (Silver Springs, MD, USA), pp. 82:1–82:4, American Society for Information Science, 2015.
- [10]V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, “Automatic detection of fake news,” *CoRR*, vol. abs/1708.07104, 2017.
- [11]S. Gilda, “Evaluating machine learning algorithms for fake news detection,” in *2017 IEEE 15th Student Conference on Research and Development (SCORED)*, pp. 110–115, Dec 2017.
- [12]M. Yancheva and F. Rudzicz, “Automatic detection of deception in child-produced speech using syntactic complexity features,” in *ACL*, 2013.