

Fake news analysis using machine learning



Contents

[**Introduction: -** 2](#_Toc21302720)

[**Machine learning: -** 3](#_Toc21302721)

[**Why Use Machine Learning?** 3](#_Toc21302722)

[**Some machine learning methods: -** 3](#_Toc21302723)

[**Supervised learning** 3](#_Toc21302724)

[**Unsupervised learning** 4](#_Toc21302725)

[**Semi supervised learning** 4](#_Toc21302726)

[**Reinforcement Learning** 5](#_Toc21302727)

[**Proposed approach and Models: -** 5](#_Toc21302728)

[**Data Collection: -** 5](#_Toc21302729)

[**Data Pre-processing: -** 5](#_Toc21302730)

[**Stop Word Removal: -** 6](#_Toc21302731)

[**Features Extraction: -** 6](#_Toc21302732)

[**Classification Process: -** 7](#_Toc21302733)

[**Evaluation Metrics: -** 8](#_Toc21302734)

[**Experiments: -** 9](#_Toc21302735)

[**Tools and Language: -** 9](#_Toc21302736)

[**Dataset: -** 9](#_Toc21302737)

[**Experiments Procedure: -** 11](#_Toc21302738)

[**Implementation of this experiment procedure in python: -** 11](#_Toc21302739)

[**Conclusion: -** 21](#_Toc21302740)

[**References: -** 22](#_Toc21302741)

# 

# **Introduction: -**

In the recent years, online content has been playing a significant role in swaying user’s decisions and opinions. Opinions such as online reviews are the main source of information for e-commerce customers to help with gaining insight into the products they are planning to buy.

Recently it has become apparent that opinion spam does not only exist in product reviews and customers’ feedback. In fact, fake news and misleading articles is another form of opinion spam, which has gained traction. Some of the biggest sources of spreading fake news or rumors are social media websites such as Google Plus, Facebook, Twitters, and other social media outlet.

Even though the problem of fake news is not a new issue, detecting fake news is believed to be a complex task given that humans tend to believe misleading information and the lack of control of the spread of fake content. Fake news has been getting more attention in the last couple of years, especially since the US election in 2016. It is tough for humans to detect fake news. It can be argued that the only way for a person to manually identify fake news is to have a vast knowledge of the covered topic. Even with the knowledge, it is considerably hard to successfully identify if the information in the article is real or fake. The open nature of the web and social media in addition to the recent advance in computer science simplify the process of creating and spreading fake news. While it is easier to understand and trace the intention and the impact of fake reviews, the intention, and the impact of creating propaganda by spreading fake news cannot be measured or understood easily. For instance, it is clear that fake review affects the product owner, customer and online stores; on the other hand, it is not easy to identify the entities affected by the fake news. This is because identifying these entities require measuring the news propagation, which has shown to be complex and resource intensive. Trend Micro, a cyber security company, analyzed hundreds of fake news services provider around the globe. They reported that it is effortless to purchase one of those services. In fact, according to the report, it is much cheaper for politicians and political parties to use those services to manipulate election outcomes and people opinions about certain topics. Detecting fake news is believed to be a complex task and much harder than detecting fake product reviews

given that they spread easily using social media and word of mouth.

We present in this paper an n-gram features-based approach to detect fake news, which consists of using text analysis based on n-gram features and machine learning classification techniques. We study and compare six different supervised classification techniques, namely, K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Logistic Regression (LR), Linear Support Vector Machine (LSVM), Decision tree (DT) and Stochastic Gradient Descent (SGD). Experimental evaluation is conducted using a dataset compiled from real and fake news websites, yielding very encouraging results.

# **Machine learning: -**

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

# **Why Use Machine Learning?**

* Problems for which existing solutions require a lot of hand-tuning or long lists of rules: one Machine Learning algorithm can often simplify code and perform better.
* Complex problems for which there is no good solution at all using a traditional approach: the best Machine Learning techniques can find a solution.
* Fluctuating environments: A Machine Learning system can adapt to new data.
* Getting insights about complex problems and large amounts of data.

# **Some machine learning methods: -**

Machine Learning systems can be classified according to the amount and type of supervision they get

during training. There are four major categories: supervised learning, unsupervised learning,

semi supervised learning, and Reinforcement Learning.

## **Supervised learning**

In supervised learning, the training data we feed to the algorithm includes the desired solutions, called labels. A typical supervised learning task is classification. The spam filter is a good example of this: it is trained with many examples’ emails along with their class (spam or ham), and it must learn how to classify new emails.

Another typical task is to predict a target numeric value, such as the price of a car, given a set of features (mileage, age, brand, etc.) called predictors. This sort of task is called regression. To train

the system, we need to give it many examples of cars, including both their predictors and their labels (i.e., their prices).

Some regression algorithms can be used for classification as well, and vice versa. For example,

Logistic Regression is commonly used for classification, as it can output a value that corresponds to the probability of belonging to a given class (e.g., 20% chance of being spam).

Here are some of the most important supervised learning algorithms:

* k-Nearest Neighbors
* Linear Regression
* Logistic Regression
* Support Vector Machines (SVMs)
* Decision Trees and Random Forests
* Neural networks2

## **Unsupervised learning**

In unsupervised learning, as we might guess, the training data is unlabeled. The system tries

to learn without a teacher.

Here are some of the most important unsupervised learning algorithms:

* Clustering
* k-Means
* Hierarchical Cluster Analysis (HCA)
* Expectation Maximization
* Visualization and dimensionality reduction
* Principal Component Analysis (PCA)
* Kernel PCA
* Locally-Linear Embedding (LLE)
* t-distributed Stochastic Neighbor Embedding (t-SNE)
* Association rule learning

## **Semi supervised learning**

Some algorithms can deal with partially labeled training data, usually a lot of unlabeled data and a little bit of labeled data. This is called semi supervised learning.

Most semi supervised learning algorithms are combinations of unsupervised and supervised algorithms. For example, deep belief networks (DBNs) are based on unsupervised components called restricted Boltzmann machines (RBMs) stacked on top of one another. RBMs are trained sequentially in an unsupervised manner, and then the whole system is fine-tuned using supervised learning techniques.

## **Reinforcement Learning**

Reinforcement Learning is a very different beast. The learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards). It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation.

# **Proposed approach and Models: -**

## **Data Collection: -**

Fake news is broadly spread crosswise over different online stages. A large portion of the phony news datasets we see online depend on specific occasions, similar to the 2016 U.S. races. That is a quite certain example and can prompt genuine inclination in the model whenever utilized only. It is imperative to broaden utilizing various areas and timeframes.

We can utilize a portion of the certainties checking sites like PolitiFact or BuzzFeed as a hotspot for gathering fake news data. In these actualities checking destinations, fake news data is given by believed creators and significant cases are made by creators on why the referenced news is phony. We propose a system for gathering counterfeit news information in an intermittent way to refresh the store. To start with, we gather the confirmed fake and genuine news from realities checking sites on consistent schedule. The utilizing the twitter propelled search API, we accumulate the tweets identified with that genuine/fake news that spread them in twitter. Additionally, we assemble social commitment of clients, for example, answers of tweets, retweet and top choices through twitter APIs. In online life, clients from social gatherings will likewise be influenced by phony news in view of reverberation chamber impact. In this way, we additionally gather the devotees and followings of the clients who draw in with phony news help portray client includes in the location task.

## **Data Pre-processing: -**

Before representing the data using vector-based model, the data need to be subjected to certain refinements like stop-word removal, word-tokenization, a lower casing, sentence segmentation, and punctuation removal. This will help us reduce the size of actual data by removing the irrelevant information that exists in the data. We created a generic processing function to remove punctuation and non-letter characters for each document; then we lowered the letter case in the document. In addition, an n-gram word-based tokenizer was created to slice the text based on the length of n.

### 

### **Stop Word Removal: -**

Stop words are insignificant words in a language that will create noise when used as features in text classification. These are words commonly used a lot in sentences to help connect thought or to assist in the sentence structure. Articles, prepositions and conjunctions and some pronouns are considered stop words. We removed common words such as, a, about, an, are, as, at, be, by, for, from, how, in, is, of, on, or, that, the, these, this, too, was, what, when, where, who, will, etc. Those words were removed from each document, and the processed documents were stored and passed on to the next step

### **Features Extraction: -**

One of the challenges of text categorization is learning from high dimensional data. There is a large number of terms, words, and phrases in documents that lead to a high computational burden for the learning process. Furthermore, irrelevant and redundant features can hurt the accuracy and performance of the classifiers. Thus, it is best to perform feature reduction to reduce the text feature size and avoid large feature space dimension. We studied in this research two different features selection methods, namely, count vectorizer and Term Frequency-Inverted Document Frequency (TF-IDF). These methods are described in the following.

#### **TF-IDF: -**

The Term Frequency-Inverted Document Frequency (TF-IDF) is a weighting metric often used in information retrieval and natural language processing. It is a statistical metric used to measure how important a term is to a document in a dataset. A term importance increases with the number of times a word appears in the document; however, this is counteracted by the frequency of the word in the corpus.

Term Frequency is an approach that utilizes the counts of words appearing in the documents to figure out the similarity between documents. Each document is represented by an equal length vector that contains the words counts. Next, each vector is normalized in a way that the sum of its elements will add to one. Each word count is then converted into the probability of such word existing in the documents. For example, if a word is in a certain document it will be represented as one, and if it is not in the document, it will be set to zero. Thus, each document is represented by groups of words.

**TF(t)= Number of times term t appears in a document /Total numbers of terms in the document**

One of the main characteristics of IDF is it weights down the term frequency while scaling up the rare ones. For example, words such as “the” and “then” often appear in the text, and if we only use TF, terms such as these will dominate the frequency count. However, using IDF scales down the impact of these terms.

**IDF(t) = loge (Total number of documents/Number of documents with term t in it)**

**TF-IDF Score = TF \*IDF**

#### **Count Vectorizer: -**

We will be creating vectors that have a dimensionality equal to the size of our vocabulary, and if the text data features that vocab word, we will put a one in that dimension. Every time we encounter that word again, we will increase the count, leaving 0s everywhere we did not find the word even once.

The result of this will be very large vectors, if we use them on real text data, however, we will get very accurate counts of the word content of our text data. Unfortunately, this won’t provide use with any semantic or relational information, but that’s okay since that’s not the point of using this technique.

## **Classification Process: -**

Figure 1 is a diagrammatic representation of the classification process. It starts with preprocessing the data set, by removing unnecessary characters and words from the data. N-gram features are extracted, and a features matrix is formed representing the documents involved. The last step in the classification process is to train the classifier. We investigated different classifiers to predict the class of the documents. We investigated specifically, six different machine learning algorithms, namely, Support Vector Machines (SVM), Naïve Bayes classification, Random forest, Decision trees and Logistic regression. We used implementations of these classifiers from the Python Machine learning library (sklearn).

**Feature extractions**

**(TFIDF and count vectorizer)**

**Dataset**

**Pre-processing data (stop words removal word-tokenizer, punctation removal)**

rrwr removal)

**Training Classifier**

**(Multinomial Naïve-Bayes, SVM)**

**News Classification**

**Real News Fake News**

**Fig 1: Classification process**

We split the dataset into training and testing sets. So, for this classification process we take around 70% of the dataset for training and 30% for testing.

## **Evaluation Metrics: -**

To evaluate the performance of algorithms for fake news detection problem, various evaluation metrics have been used. In this subsection, we review the most widely used metrics for fake news detection. Most existing approaches consider the fake news problem as a classification problem that predicts whether a news article is fake or not:

True Positive (TP): when predicted fake news pieces are actually annotated as fake news;

True Negative (TN): when predicted true news pieces are actually annotated as true news;

False Negative (FN): when predicted true news pieces are actually annotated as fake news;

False Positive (FP): when predicted fake news pieces are actually annotated as true news.

By formulating this as a classification problem, we can define following metrics,

* **Precision = (TP/ (TP + FP))**
* **Recall = (TP/ (TP + FN))**
* **F1 = 2 \*(Precision\* Recall/ (Precision + Recall))**
* **Accuracy = (TP + TN/ (TP + TN + FP+ FN))**

These metrics are commonly used in the machine learning community and enable us to evaluate the performance of classifiers from different perspectives. Specially, accuracy measures the similarity between predicted fake news and real fake news. Precision measures the fraction of all detected fake news that are annotated as fake news, addressing the important problem of identifying which news is fake. However, because fake news datasets are often skewed, a high precision can be easily achieved by making fewer positive predictions. Thus, recall is used to measure the sensitivity, or the fraction of annotated fake news articles that are predicted to be fake news. F1 is used to combine precision and recall, which can provide an overall prediction performance for fake news detection. Note that for Precision, Recall, F1, and Accuracy, the higher the value, the better the performance.

# **Experiments: -**

## **Tools and Language: -**

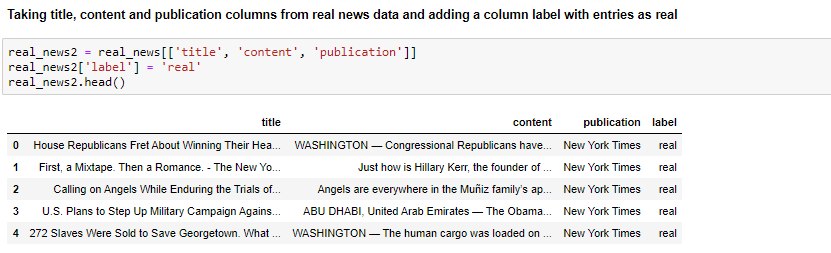
In this research python is used for the machine learning part. Python is one of the most popular

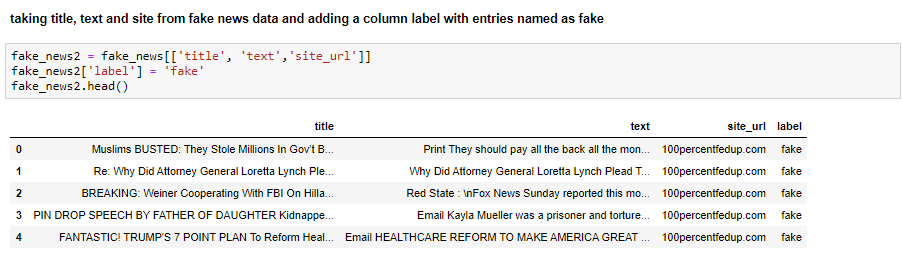
language scientific areas and ML. For running python Jupyter notebook is used.

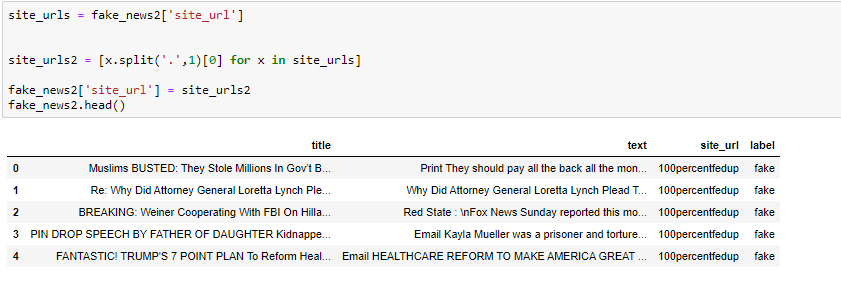
## **Dataset: -**

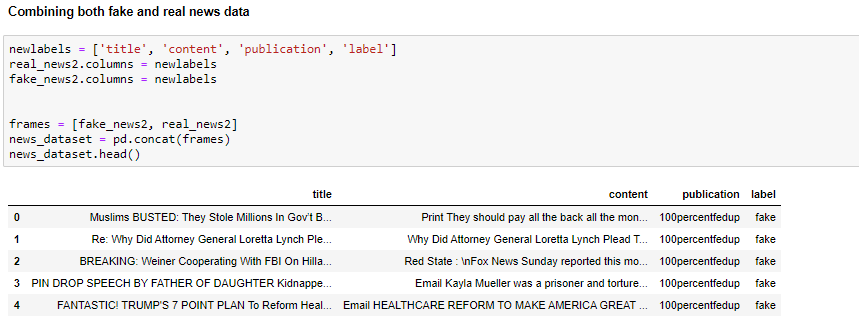
The field of fake news detection is a relatively new area of research. Hence, few public datasets are available. We used in this work primarily a dataset from Kaggle which has two datasets one for real news and other one for fake news separately. Real news dataset has 15712 rows and 11 columns. Fake news data set has 12999 rows and 20 columns. For combining both the dataset we took some variables from real news dataset and same columns from fake news dataset and combined both to obtain a single dataset.











Now, we are left with 4 columns in the final dataset: Title, content, publication and label. Title is the title of the news and content is the complete information od the news. Publication is the publication in which news was published and label is representing whether the news is true or false.

## **Experiments Procedure: -**

We run the aforementioned machine learning algorithms on the dataset, with the goal of predicting whether the articles are truthful or fake. The experiments started by Data prepossessing like labeling the label column as 1 and 0. Then checking the null value. If there is any null value in data drop that entry from the dataset. After doing these two things we do text prepossessing like convert into lower case, word tokenizer, punctuation removal, stop words removal. Then the dataset is divided into 70% for training and 30% for testing.

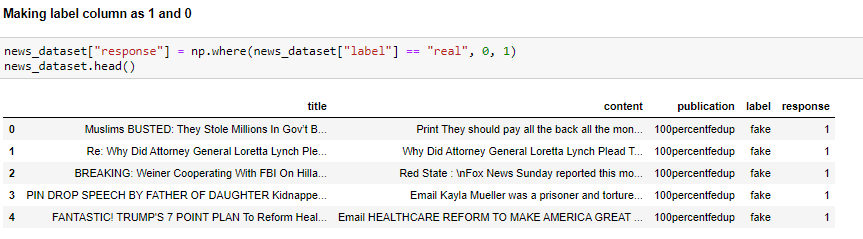
After splitting the data, we did feature selection. We applied two techniques for feature selection one is TF-IDF and second one is Count vectorizer which has been explained before. Then applied these feature selection methods on both train and test data. The algorithms were used to create learning models, and then the learned models were used to predict the labels assigned to the testing data. Experiment results were then presented, analyzed and interpreted. Based on the evaluation metrics we select the best method for fake news detection.

### **Implementation of this experiment procedure in python: -**

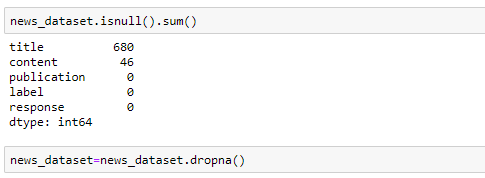
#### **Data Prepossessing**

**Labeling label column as 1 and 0**

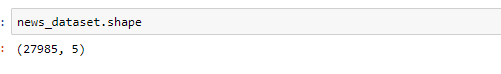
**Here 1 means fake news and 0 means real news.**



**Checking missing value in data**

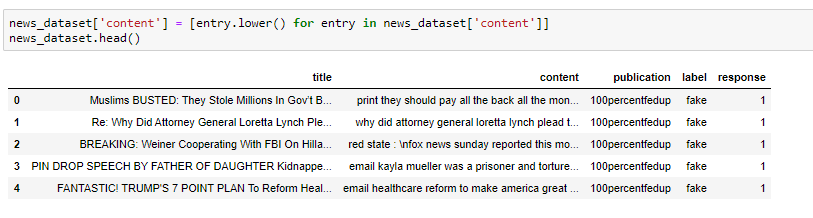


From here we can see that there are some variables in the data with null values. So, drop these null values.

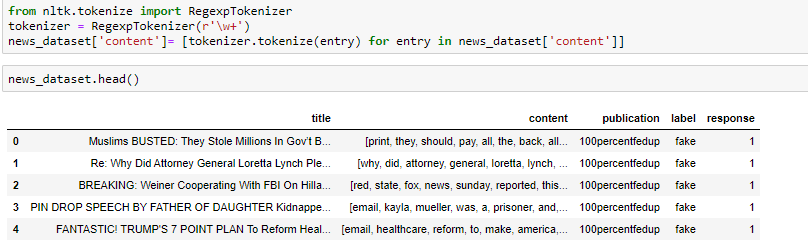


After removing null values there are 27985 rows and 5 columns left in data.

**Changing the text in content variable to lower case**

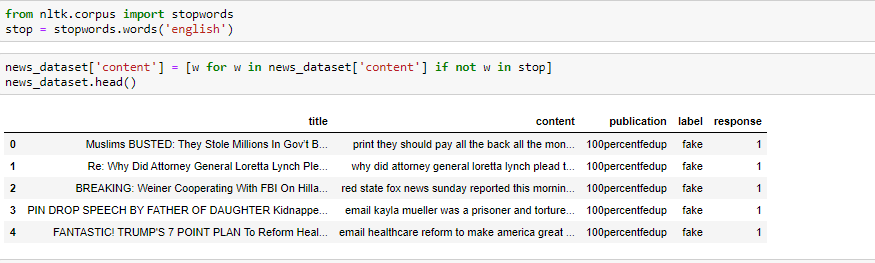


**Applying word tokenizer and remove punctuation from content column**

****

The function used here RegexpTokenizer is available in nltk library in python. This function remove punctuation and do world tokenizer of content variable at a time.

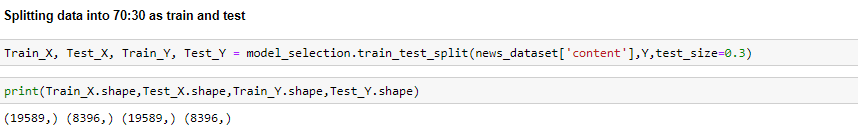
**Removing stop words from content column**



Taking the response variable into Y.

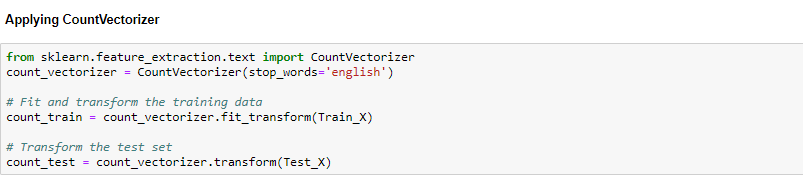
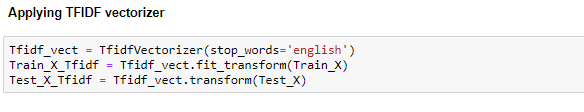


**Splitting the final data into 70:30 for training and testing respectively.**



Here content variable of final dataset work as X.

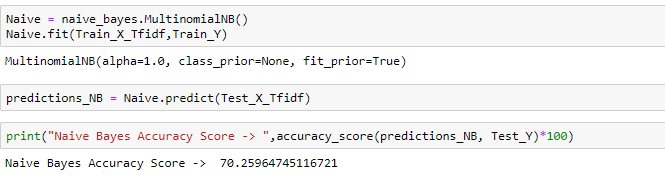
**Feature Selection: -**



#### **Machine learning models: -**

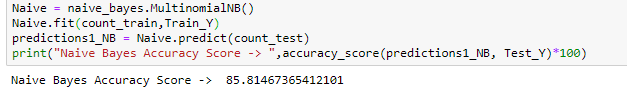
**Naïve Bayes Classifier**

* **Naïve Bayes classifier on TFIDF vectorizer data**



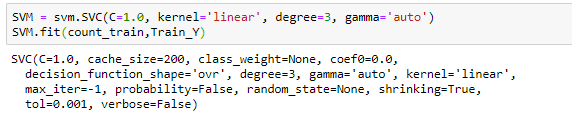
On applying Naïve Bayes classifier on TFIDF vectorizer data we obtained 70.25% accuracy.

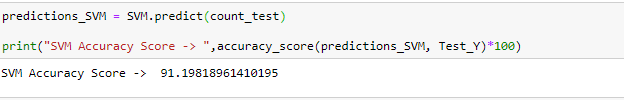
* **Naïve Bayes classifier on count vectorizer data**

****

On applying Naïve Bayes classifier on count vectorizer data we obtained 85.81% accuracy.

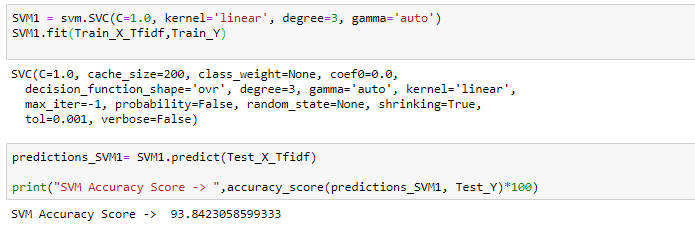
* **SVM on count vectorizer data**





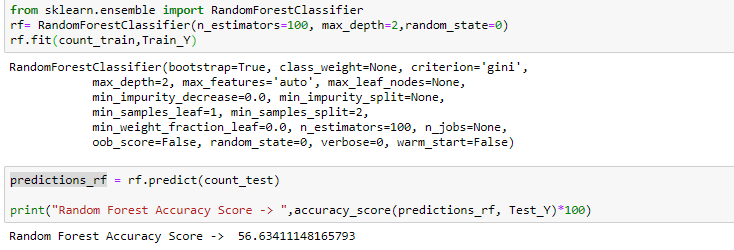
On applying SVM classifier on count vectorizer data we obtained 91.19% accuracy.

* **SVM on TFIDF vectorizer data**

****

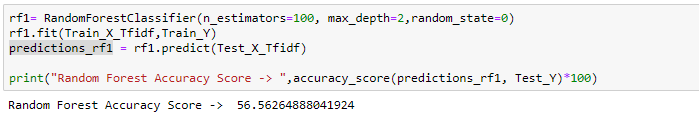
On applying SVM classifier on count TFIDF data we obtained 93.84% accuracy.

* **Random Forest on count vectorizer data**

****

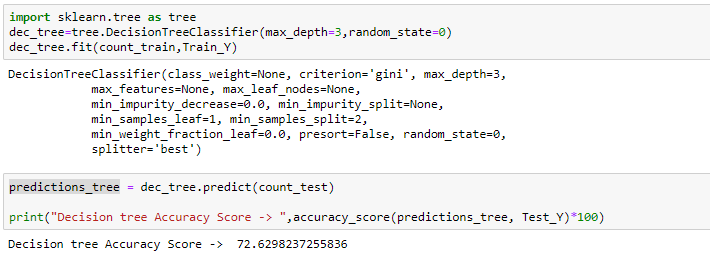
On applying Random forest classifier on count vectorizer data we obtained only 56.63% accuracy.

* **Random Forest on TFIDF vectorizer data**

****

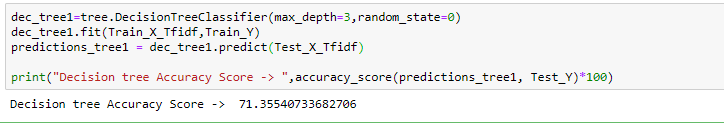
On applying Random forest classifier on TFIDF vectorizer data we obtained only 56.56% accuracy.

* **Decision tree on count vectorizer data**

****

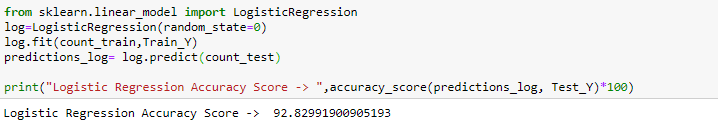
On applying Decision Tree classifier on count vectorizer data we obtained 72.63% accuracy.

* **Decision tree on TFIDF vectorizer data**

****

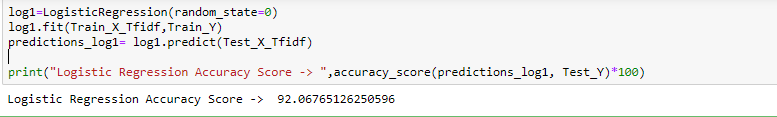
On applying Decision Tree classifier on TFIDF vectorizer data we obtained 71.35% accuracy.

* **Logistic Regression on count vectorizer data**

****

On applying Logistic Regression on count vectorizer data we obtained 92.83% accuracy.

* **Logistic Regression on TFIDF vectorizer data**

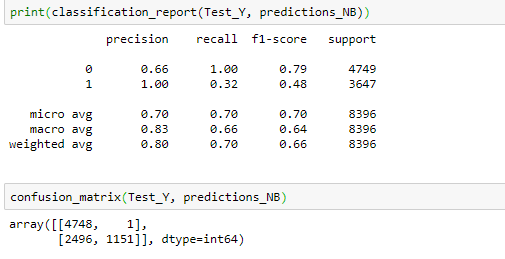
****

On applying Logistic Regression on TFIDF vectorizer data we obtained 92.07% accuracy.

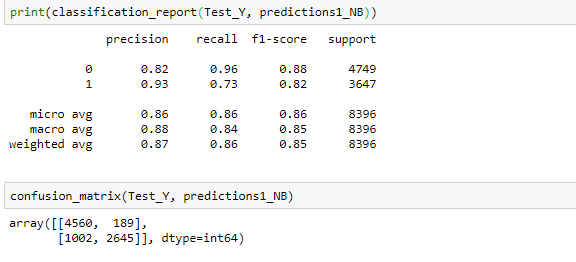
#### **Evaluation metrices for all models: -**

**Naïve Bayes Classifier**

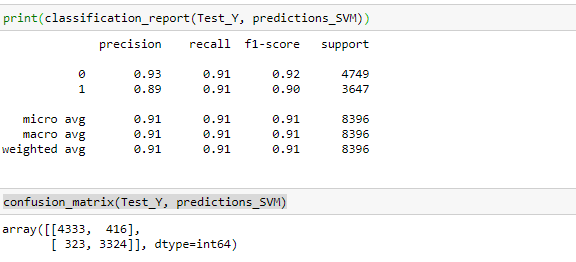
* **Naïve Bayes classifier on TFIDF vectorizer data**



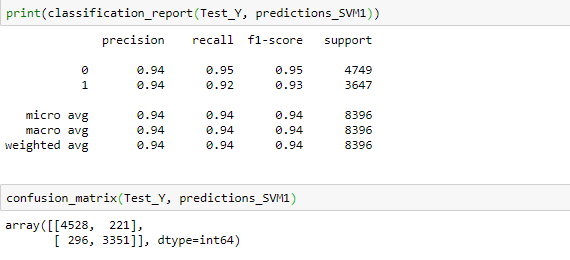
* **Naïve Bayes classifier on count vectorizer data**

****

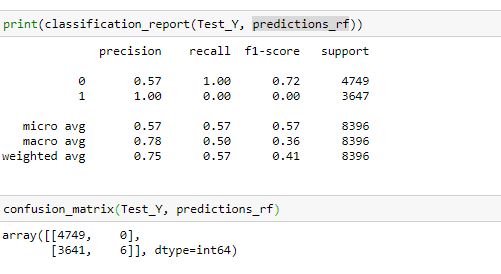
* **SVM on count vectorizer data**



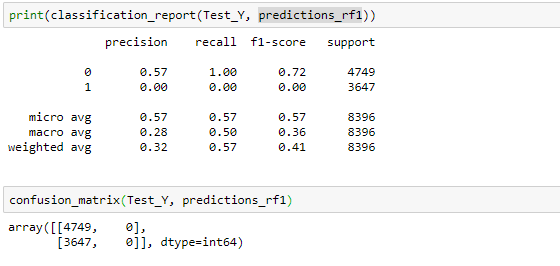
* **SVM on TFIDF vectorizer data**

****

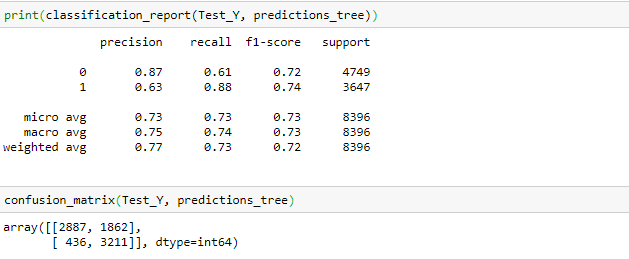
* **Random Forest on count vectorizer data**

****

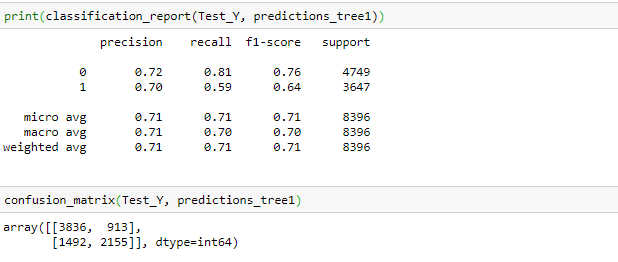
* **Random Forest on TFIDF vectorizer data**

****

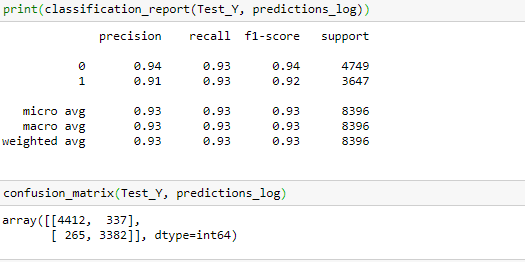
* **Decision tree on count vectorizer data**

****

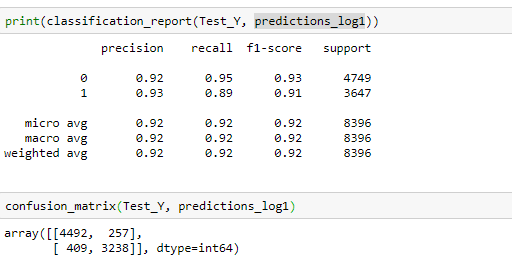
* **Decision tree on TFIDF vectorizer data**

****

* **Logistic Regression on count vectorizer data**

****

* **Logistic Regression on TFIDF vectorizer data**

****

Based on all evaluation metrices obtained above like accuracy, recall score, precision and f1-score we can say that SVM and Logistic Regression are the best performing algorithms for fake news analysis for our dataset on both type of feature selection techniques.

# **Conclusion: -**

The problem of fake news has gained attention in 2016, especially in the aftermath of the last US presidential elections. Recent statistics and research show that 62% of US adults get news on social media [12, 13]. Most of the popular fake news stories were more widely shared on Facebook than the most popular mainstream news stories [14]. A sizable number of people who read fake news stories have reported that they believe them more than news from mainstream media. Dewey [15] claimed that fake news played a huge role in the 2016 US election and that they continue to affect people opinions and decisions.

In this paper, we have presented a detection model for fake news using Machine Learning through the lenses of different features extraction techniques. Furthermore, we investigated two different features extraction techniques and five different machine learning techniques. The proposed model achieves its highest accuracy when using Linear SVM classifier. The highest accuracy score is around 94%. Fake news detection is an emerging research area with few public datasets. We run our model on an existing dataset, showing that our model outperforms the original approach published by the authors of the dataset. In our future work, we will run our model on the few other publicly available datasets, such as the LIAR dataset which was released only recently, after we completed the current phase of our research.

# **References: -**

* Detection of Online Fake News Using Machine Learning Techniques Research gate Conference Paper · October 2017 by Sherif Saad
* Fake News Detection on Social Media: A Data Mining Perspective by Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu; Computer Science & Engineering, Arizona State University, Tempe, AZ, USA Charles River Analytics, Cambridge, MA, USA
* Hands-On Machine Learning with Scikit-Learn and TensorFlow by Aurélien Géron
* Lemann, N.: Solving the Problem of Fake News. The New Yorker (2017). <http://www.newyorker.com/news/news-desk/solving-the-problem-of-fake-news>
* Trends in the Diffusion of Misinformation on Social Media Hunt Allcott, New York University, Microsoft Research, and NBER; Matthew Gentzkow, Stanford University and NBER; Chuan Yu, Stanford University October 2018
* Silverman, C., Singer-Vine, J.: Most americans who see fake news believe it, new survey says. BuzzFeed News (2016)