CAPSTONE I (AIDI 1003-02)

PROJECT REPORT

FAKE NEWS DETECTION

Facilitator:
Dr. Reza Dibaj

Submitted by:
Deval Shah (xxxxxx807)
Neha Joseph (xxxxxx187)
Roshna babu (xxxxxx012)
Riyaben Gajjar (xxxxxx043)
Abraham Mathew (xxxxxx875)

Submission Date: 14th April 2022

Contents

1. **Executive Summary** 3
2. **Introduction** 3
3. **Rationale statement** 5
4. **Problem Statement** 6
5. **Data Requirements** 6
6. **Data** 7
7. **Model/Architecture Approach** 7
8. **Project Work Breakdown** 9
9. **Exploratory Data Analysis (EDA)** 10
10. **Data Pre-processing Pipeline** 10
11. **Algorithm Evaluation** 19
    a. **Passive Aggressive Classifier:** 19
    b. **KNN (K-Nearest Neighbor)** 20
    c. **Decision Tree:** 22
    d. **Naïve Bayes Classifier:** 24
    f. **Logistic Regression:** 27
12. **Candidate Algorithm Selection** 28
13. **Inference** 30
14. **Appendix** 31

1. Executive Summary

The term "fake news" was pretty much unknown and unpopular a few decades ago, but it has emerged as a massive monster in the digital era of social media. Fake news is spreading like wildfire these days, and people are sharing it without confirming it. This is frequently done to promote or enforce specific views, and it is often carried out through political agendas. Fake news refers to news that may or may not be true and is widely disseminated via social media and other internet platforms.

In this digital age, it is very difficult to tackle the spread of fake news, where thousands of information-sharing sites via fake news or misinformation can be shared. It has become a greater issue as AI advances, bringing with it artificial bots that may be used to create and propagate fake news The problem is critical because many individuals believe anything they read on the internet, and those who are inexperienced or new to digital technologies are vulnerable to being misled. Fraud is another issue that can arise as a result of spam or harmful emails and communications.

Fake news has grown in popularity and spread as a result of recent political events. Humans are inconsistent, if not outright terrible detectors of fake news, as evidenced by the pervasive effects of the widespread onset of fake news. As a result, efforts have been made to automate the process of detecting fake news. The most prominent of these attempts are "blacklists" of unreliable sources and authors. While these technologies are useful, we need to account for more complex instances when trusted sources and authors leak fake news in order to provide a more complete end-to-end solution. As a result, the goal of this project was to develop a tool that used machine learning and natural language processing techniques to recognize the language patterns that distinguish fake and true news. The outcomes of this project show that machine learning can be effective in this situation. We developed a model that detects a variety of intuitive indicators of real and fake news, as well as an application to aid in the visual representation of the classification decision. We aim to give users the ability to classify news as fake or real, as well as verify the legitimacy of the website that published it.

2. Introduction

As time passes, the amount of data, particularly text data, grows exponentially. Along with the growth of data, our knowledge in AI also increases and the processing power enables us to train overly complex and large models faster. Fake news has recently gotten a lot of attention throughout the world Fake news, information bubbles, news manipulation, and a lack of faith in the media are all problems that are becoming increasingly prevalent in our culture. As we spend more time connecting online through social media platforms, more people are seeking out and consuming news through social media rather than traditional news organizations. The reasons for this shift in consumer habits are built into the nature of those social media platforms: When compared to traditional journalism, such as newspapers or television, news on social media is frequently more timely and less expensive; and It's easier to share, discuss, and debate the news on social media with friends or other readers.

For example, in 2016, 62 percent of adults in the United States received news on social media, compared to only 49 percent in 2012. It was also discovered that, as a key news source, social media currently exceeds television. Despite the advantages of social media, the quality of social media stories is lower than that of traditional news organizations. Large volumes of fake news, i.e. news pieces with purposely incorrect material, are created online for a variety of reasons, including financial and political gain because it is inexpensive to supply news online and significantly faster and easier to distribute through social media. Over 1 million tweets have been linked to the false news "Pizza-gate" scandal, according to estimates "by the end of the presidential election's first round. Given the prevalence of this new phenomenon, “Fake news" was even named the word of the year by the Macquarie Dictionary in 2016.

The widespread dissemination of fake news has the potential to harm both individuals and society. For starters, fake news has the potential to disrupt the news ecosystem's authenticity equilibrium; for example, during the 2016 presidential election in the United States, the most popular false news was even more widely shared on Facebook than the most widely accepted genuine mainstream news. Second, fake news is designed to induce customers to accept biased or incorrect information. Propagandists use fake news to disseminate political ideas or influence. For example, according to some reports, Russia has developed fake accounts and social bots to spread misleading tales. Third, false news influences how people interpret and respond to actual news. For example, some fake news was generated solely to instill distrust and confusion in people, making it difficult for them to tell what is true from what is not. To help reduce the negative consequences of bogus news (both to profit the general public and therefore the news ecosystem). It's critical that we develop tools for detecting fake news on social media automatically. The internet and social media have made it much easier and more comfortable to get news information.

Often, Internet users may follow up on events that concern them in an online format, and the growing number of mobile devices makes this process much easier. However, with great potential comes enormous responsibility. The mass media has a huge influence on society, and as is often the case, someone wants to take advantage of this. To achieve certain objectives, the media may manipulate knowledge in a variety of ways. As a result, news pieces that aren't entirely true or completely fake are produced. There are even websites dedicated nearly entirely to the dissemination of fake news. They disseminate hoaxes, half-truths, propaganda, and disinformation appearing as actual news, and they frequently use social media to generate traffic and amplify their impact. The most common purpose of fake news websites is to influence public opinion on certain topics (mostly political). Websites of this type can also be found in Ukraine, the United States, Germany, China, and many other nations. As a result, fake news may be both a global issue and a global task. Many scientists believe that artificial intelligence and machine learning could be used to combat fake news.

However, to begin tackling this issue, a thorough understanding of fake news and its origins is essential. Only then one can understand the different techniques and fields of machine learning (ML), natural language processing (NLP), and artificial intelligence (AI) that may be useful in combating this problem. In the previous six months, the term "fake news" has been used in a variety of contexts, with numerous definitions offered. The New York Times, for example, describes it as "a made-up story with the goal to deceive." Measuring false news, or even accurately defining it, may swiftly devolve into a subjective rather than objective exercise. Despite all these shortcomings, several entities have tried to categorize fake news in different manners. This paper discusses the approach of natural language processing and machine learning to solve this fake news problem. We have Used bag-of-words, count vectorizer, TF-IDF, and training the data on three classifiers to investigate which of them works well for this specific dataset. The Accuracy, precision, recall, and f1 scores help us determine which model works best.

3. Rationale statement

In the world of rapidly increasing technology, information sharing has become an easy task. Without a doubt, we can say that the internet has made our lives easier and provided us with access to a wealth of knowledge. This is a development in human history, but it also blurs the distinction between real and intentionally produced information. During the 2016 U.S. Presidential Election, the development of fake news exposed not only the dangers of fake news's consequences, but also the difficulties in separating false news from true news. Although the term "fake news" is new, it is not always a new phenomenon. Fake news has been present since the advent and popularity of one-sided, political publications in the 19th century. However, technological advancements and the dissemination of news through many forms of media have worsened the spreading of fake news today. Anyone can now publish content that can be consumed by the internet, whether it is credible or not. Fake news, unfortunately, attracts a lot of attention on the internet, particularly on social media. People are duped, and they don't think twice about disseminating such false information to the rest of the globe. This type of information fades away, but not without causing the harm is intended. Social media platforms such as Facebook, Twitter, and WhatsApp play a significant role in spreading false information. As a result, the impact of fake news has grown dramatically in recent years, and something must be done to prevent this from happening again. As a means of narrowing the search in a meaningful way, we identified the three most common motivations for generating false news and selected only one as the target for this research. The first motivation for fake news, which stretches back to one-sided party journals in the nineteenth century, is to influence public opinion. The second method, which necessitates more modern technological advancements, is the use of fake headlines as clickbait to raise money.

The lack of manually labelled fake news datasets is undoubtedly a major hurdle in the development of computationally costly, text-based models that cover a broad range of topics. The datasets available on the internet for the fake news challenge is insufficient for our purposes because it only contains ground truth about text relationships, not whether those texts are true or false assertions. We require a set of news stories that are directly categorized into news types (e.g., real vs. fake or true vs. parody vs. clickbait vs. propaganda) for our purposes. There is an abundance of labeled data from a variety of sources, including Twitter, Amazon Reviews, and IMDb Reviews, for easy and frequent NLP classification tasks, such as sentiment analysis. Unfortunately, the same cannot be said for identifying fake and real news pieces. This will be a challenge to researchers and data scientists who want to explore the topic by implementing supervised machine learning techniques.

Many scientists believe that machine learning and artificial intelligence can help solve the problem of fake news. Various models are employed to achieve a 60-75 percent accuracy range. Naive Bayes classifier, decision tree model, SVM, and others are included.
The news content has diverse unstructured format data (such as documents, videos, and audios), here we will concentrate on text format news. With the advancement of Natural language processing, It is possible now that we can identify the deceptive and fake nature of articles or sentences.

4. Problem Statement

Consumption of news is a two-edged sword. On the one hand, the low cost, ease of access, and speed with which information is disseminated encourage people to seek out and consume news. It allows for the widespread dissemination of "fake news," or low-quality news that contains purposely incorrect material. The widespread availability of fake news has the potential to have tremendously negative consequences for both individuals and society. As a result, false news identification is a young field of study that is gaining a lot of attention. First, fake news is prepared with the purpose of deceiving readers into believing false information, making it difficult and time-consuming to detect based on news content.

Text, or natural language, is a tough form to comprehend due to a variety of linguistic traits and styles such as sarcasm, metaphors, and so on. There are also thousands more spoken languages, each with its own grammar, script, and syntax. Natural language processing is a subset of artificial intelligence that includes methods for analyzing text, creating models, and making predictions. Developing algorithms that can distinguish fake articles from actual or human-written information, as well as comparing and assessing several language-producing models with the human style of writing and inter-model styles is a challenging task. To develop a FAKE NEWS DETECTION system using natural language processing and its accuracy will be tested using machine learning algorithms. The algorithm must be able to detect fake news in each scenario. The project should follow the necessary methodologies (data collection, preprocessing, feature selection, model training, algorithms, visualization, etc.) to get a better result. Without following the methodologies, we will not be able to complete the project.

5. Data Requirements

For the fake news detection project, there are a lot of datasets available on the internet. Internet news from multiple data sources (websites), including social networking websites, search engines, news agency homepages, and fact-checking websites are available. Some of the major data constraints or Principal components expected are Source, Headline, Body Text, Classification, Image, or video. We may acquire internet news from multiple sources of sites, including social networking websites, search engines, news agency homepages, and fact-checking websites.

Data requirements are usually the data items which are necessary for the analysis. While considering the data analysis part of fake news detection, there are certain variables which are mandatory to do as part of analysis. We should be having text data which contains articles in it, hence the text data could be vectorized and tokenized and later could be used for analysis. While considering the dataset the other thing to be considered is that the dataset that we chose should contain both real and fake data in it that is, if the dataset only contains fake or real that is just either of the one then the analysis conducted will not have an accurate result or report as it only focuses on one of the types. Another point is that the data assembled should be shuffled well as we do not want any error or difference in evaluation as all the fake news are residing together, as it may affect the accuracy and results of the algorithm. If these things and points are considered while selecting the data then rest of the things related to the data such as date, type of news article and the id and even the source are optional as the analysis could be done just with the text article, but rest of the variables may give additional information for the analysis. Here in this dataset, we could see that the data information available is related to title, text, subject, date, and id number but not all the variables are required for the analysis. We will be seeing in the coming session in detail the variables we have opted out of and the variables we are using for the analysis.

6. Data

After several research and studies, we have chosen the necessary datasets from Kaggle. We have opted for two datasets that are essential for the analysis. As the analysis is related t o fake news detection the dataset contains the information regarding certain articles which a re addressed as Fake or True. The news articles are obtained from certain news reporting me dia. And the news reports were generated between January 2016 and December 2017. Each dataset contains 4 columns in it, which are Tile, Text, Subject, and Date. The two datasets ar e the fake news dataset and the true news dataset. The fake news dataset contains data tha t are fake I.e., the news articles which are fake. (https://www.kaggle.com/datasets/clmentb isaillon/fake-and-real-news-dataset?select=Fake.csv). The total number of columns in the d ataset is 23481. And the True dataset contains the data which are real that is the news articl es that are real. (https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dat aset?select=True.csv). The total number of entries in the dataset is 21417. And all together 4 4689 entries in the dataset once we remove the missing values and the duplicates and comb ine them together.

The link for the dataset is as follows, (Fake and real news dataset | Kaggle)

7. Model/Architecture Approach

Fake news detection is a classification problem. Hence, we have chosen six classification algorithms for the analysis. To do the analysis, we have to do the EDA and data Preprocessing. Prior to the analysis, we have done certain exploratory data analysis and data preprocessing methodologies, that made the results more accurate. The methodologies involved in the analysis are as follows:

i. Data Collection

After certain studies and research related to the fake news project, we have gone through certain datasets which were available on open platforms. We have opted four datasets and out of which we have finalized on Fake and real news dataset | Kaggle . After finalizing the data we have downloaded the data and the exported it to the software i.e., the Jupyter notebook for further cleaning, pre-processing, and analysis.

ii. Data Exploration

The Exploration of the Data contains certain things such as loading the data, checking the data information, and evaluating the data understanding the relationship between each variable, understanding the relationship between the dependent variables and independent variables. Another important part of the project is getting basic insights from the data related to the analysis. And checking for the outliers is another important part of the analysis.

iii. Data Prepossessing

Pre-processing of the data includes certain steps such as checking missing values and duplicates of the data. Removal of the missing values, duplicates and cleaning the data for further analysis. Pre-processing is done to identify the errors and unwanted noises present in the dataset and eliminate those things from the dataset. Removing the unnecessary variables, the variables which are not needed for the analysis.

iv. Algorithm Selection

Once done with the Data exploration and Data pre-processing the very next thing in the analysis process is to do the algorithm selection. To select the algorithm, we must consider certain things such as the type of analysis to be done, whether it is a classification model or a regression model by looking the dependent variables.
For each analysis there are certain different algorithms. For both the classification and regression model we use separate algorithms. Some of the classification algorithms are Logistic Regression, Naïve Bayes, Decision tree, and Random Forest. And some of the regression models are Linear regression, Support vector machine, decision tree, and Random Forest. Meanwhile, there are some models that can be used for both classification and regression analysis such as KNN, Decision Tree, Random Forest algorithms.

v. Model training

Model training is done in data analysis to train the data. The training of the model is done to check the performance of the data. This is to be done before the analysis to have a better performance.

vi. Model Evaluation

The final steps involved are model evaluation and the evaluation of the model explains how good the model is. There are certain techniques to do the model evaluation such as Mean absolute error, R-Squared, confusion matrix, Accuracy Score, Precision and Recall, AUC-ROC curve.

As a part of this project, we have gone through certain classification algorithms. And for the analysis we have chosen six algorithms at first. And the six algorithms we chose are;

i. Passive aggressive classifier

PAC has the concept of using the model while prediction is correct and to make changes to the model while the prediction is incorrect.

ii. K Nearest Neighbour algorithm

Supervised algorithm used for both Classification and regression model. In the given graph the distance between two of the points are measured to consider the similarity. and if the points are closure, they are not considered to be similar. KNN has higher accuracy and are simple to use and implement.

iii. Logistic Regression

Classification algorithm identifies the variables, relationships between the variables, and how they interact with each other. Categorical variables could be predicted using the logistic variables.

iv. Random Forest

Used for both classification and regression models but for classification models. Decision tree are generated on the data to get the predictions and then finalise on the solutions.

v. Decision Tree

Classification and regression model. It’s a classifier which is tree-structured. The data is presented in the internal nodes. The decision and the decision rules are represented by the branches of the tree and outcomes including the final outcomes are represented by the leaf.

vi. Naïve Bayes

Classification algorithm that uses bayes theorem, Real – time prediction problems could be best solved with naïve bayes. Could be specially used for the text classification problems with training datasets with high dimensionality. It looks upon the probability of an object for the prediction.

The software used for the project is Jupyter notebook.

8. Project Work Breakdown

Module	Task	Time given (# of days)	Completion Date (dd/mm/yyyy)
Project Proposal	Project Proposal	10	9/2/2022
Data Collection	Importing relevant data from multiple sources	2	11/2/2022

Data pre-processing	Collating the datasets	2	13/2/2022
	Data pre-processing	2	15/2/2022
	Exploratory Data Analysis	2	17/2/2022
	Data Vectorization	2	19/2/2022
Model Training	Select the models to try	2	21/2/2022
	Training and testing of the models	10	3/3/2022
	Model Evaluation	2	4/3/2022
Algorithm Selection	Compare the model parameters	2	6/3/2022
	Create project status report	2	8/3/2022
	Presentation of model evaluation	1	10/3/2022
Final Model	Test the final model	5	17/3/2022
	Create Project Report	10	30/3/2022
	Project Report presentation	1	31/3/2022

9. Exploratory Data Analysis (EDA)

For the analysis we have imported the necessary libraries and the fake news and true news datasets for the analysis. Once the datasets were imported then we have checked the basic information related to the datasets. The information related to datasets got from the EDA are as follows, the shape of fake news dataset is (23481,4) and for the true dataset the shape is (21417,4). Obtaining the information regarding the fake news dataset and the true news dataset, for both the datasets the variables are, title, text, subject, date. For both the datasets the title variable is heading, or summary of the news articles present in the text variable. Subject states about the type of news extracted. From the dataset we could see that none of the variables are dependent variables. Hence for the datasets we would be adding an additional column named fake and real, respectively. We will be looking into it in the pre-processing of the data. And grouping by the subject we can see that the greatest number of articles in fake news are stated as news (9050) followed by to politics news (6841) and the least is middle east and US_news. For the real news there were only two subjects with 11272 politicsnews and 10145 worldnews.

10. Data Pre-processing Pipeline

.
There are seven significant steps in data pre-processing.

1. Acquire the dataset
2. Import all the crucial libraries
3. Import the dataset
4. Identifying and handling the missing values
5. Encoding the categorical data
6. Splitting the dataset
7. Feature scaling

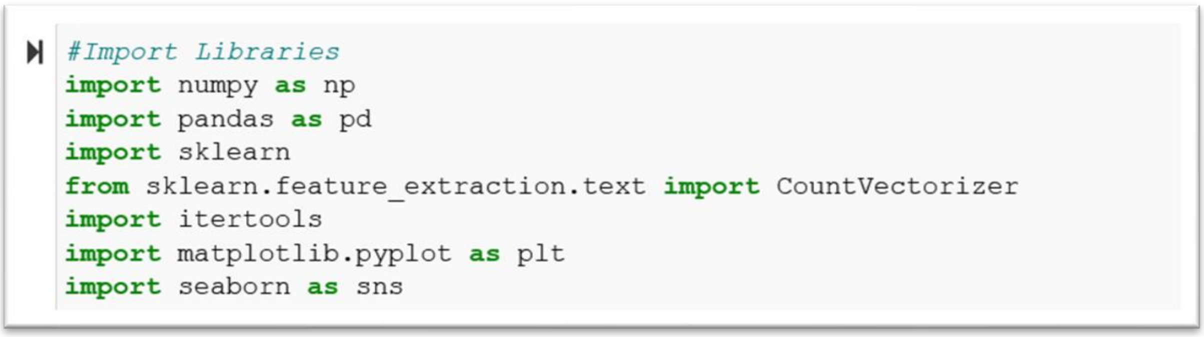
1. Acquire the dataset:

Acquiring the dataset is the first step in data preprocessing. To build and develop models, we must first acquire the relevant dataset. This dataset will be comprised of data gathered from multiple and disparate sources which are then combined in a proper format to form a dataset. For our project we chose fake and true datasets from Kaggle website.

Data source: [Fake and real news dataset | Kaggle]

2. Import all the crucial libraries:

The predefined Python libraries can perform specific data preprocessing jobs. Importing all the crucial libraries is the second step in data preprocessing. The three core Python libraries used for our project are NumPy, pandas, matplotlib. We also use other libraries like sklearn, seaborn.



```
#Import Libraries
import numpy as np
import pandas as pd
import sklearn
from sklearn.feature_extraction.text import CountVectorizer
import itertools
import matplotlib.pyplot as plt
import seaborn as sns
```

3. Import the dataset:

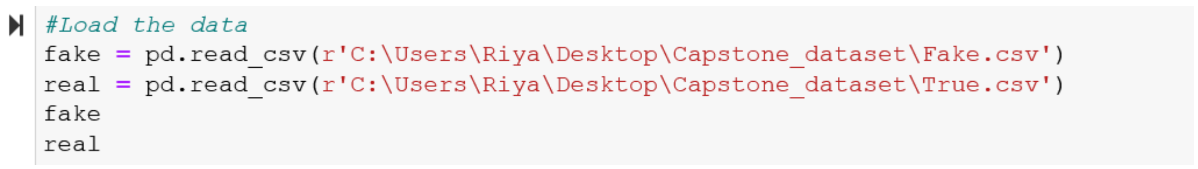
In this step, we need to import the dataset/s that we have gathered for the project at hand. Importing the dataset is one of the important steps in data preprocessing. Save your Python file in the directory containing the dataset.

we can import the dataset using the “read_csv()” function of the Pandas library. This function can read a CSV file (either locally or through a URL) and perform various operations on it. The read_csv() is written as:

data_set= pd.read_csv(‘Dataset.csv’)

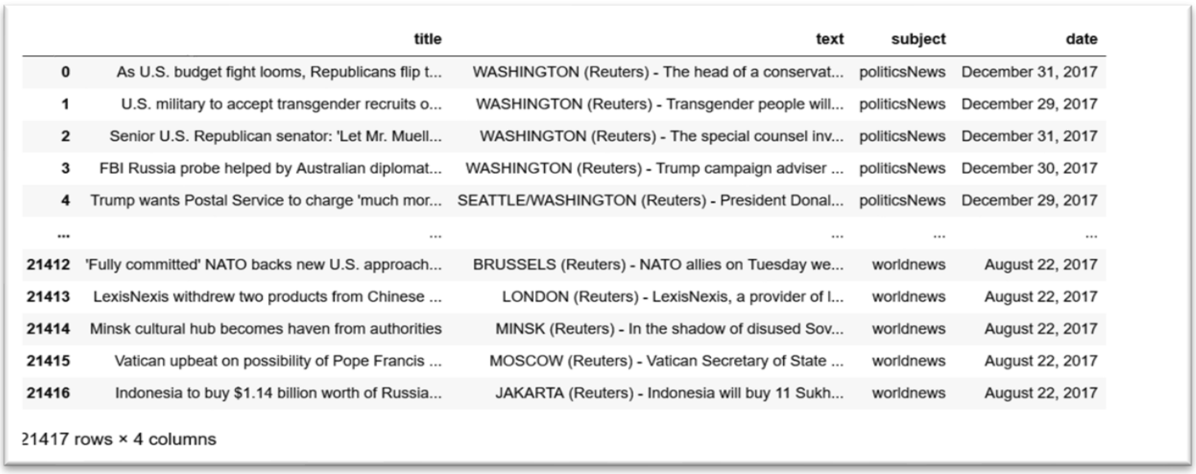
In this line of code, “data_set” denotes the name of the variable wherein we stored the dataset. The function contains the name of the dataset as well. Once we execute this code, the dataset will be successfully imported.

Here we load our dataset fake.csv and read.csv files as a fake and real variable respectively.



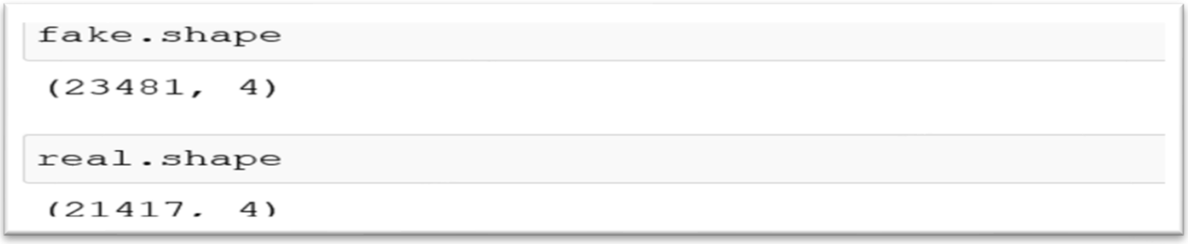
```
#Load the data
fake = pd.read_csv(r'C:\Users\Riya\Desktop\Capstone_dataset\Fake.csv')
real = pd.read_csv(r'C:\Users\Riya\Desktop\Capstone_dataset\True.csv')
fake
real
```

By writing fake and real we get datasets of Fake and True csv files as below. We also get a total number of rows and columns of our dataset (21417 rows x 4 columns).



	title	text	subject	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017
...	...	...	...	...
21412	'Fully committed' NATO backs new U.S. approach...	BRUSSELS (Reuters) - NATO allies on Tuesday we...	worldnews	August 22, 2017
21413	LexisNexis withdrew two products from Chinese ...	LONDON (Reuters) - LexisNexis, a provider of l...	worldnews	August 22, 2017
21414	Minsk cultural hub becomes haven from authorities	MINSK (Reuters) - In the shadow of disused Sov...	worldnews	August 22, 2017
21415	Vatican upbeat on possibility of Pope Francis ...	MOSCOW (Reuters) - Vatican Secretary of State ...	worldnews	August 22, 2017
21416	Indonesia to buy $1.14 billion worth of Russia...	JAKARTA (Reuters) - Indonesia will buy 11 Sukh...	worldnews	August 22, 2017

21417 rows × 4 columns

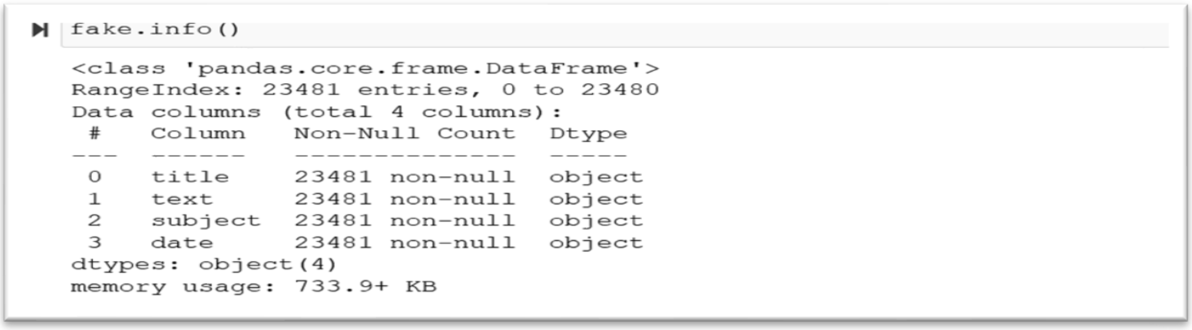


```
fake.shape
(23481, 4)
real.shape
(21417, 4)
```

The separate rows and columns are as above for both the datasets.

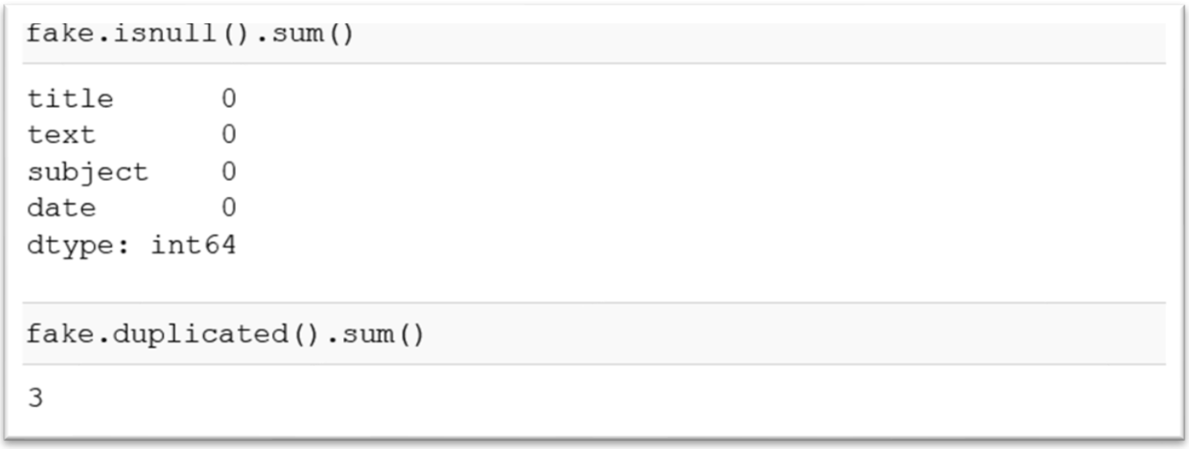
4. Identifying and handling the missing values:

In data preprocessing, it is pivotal to identify and correctly handle the missing values, failing to do this, you might draw inaccurate and faulty conclusions and inferences from the data. This will hamper our project.



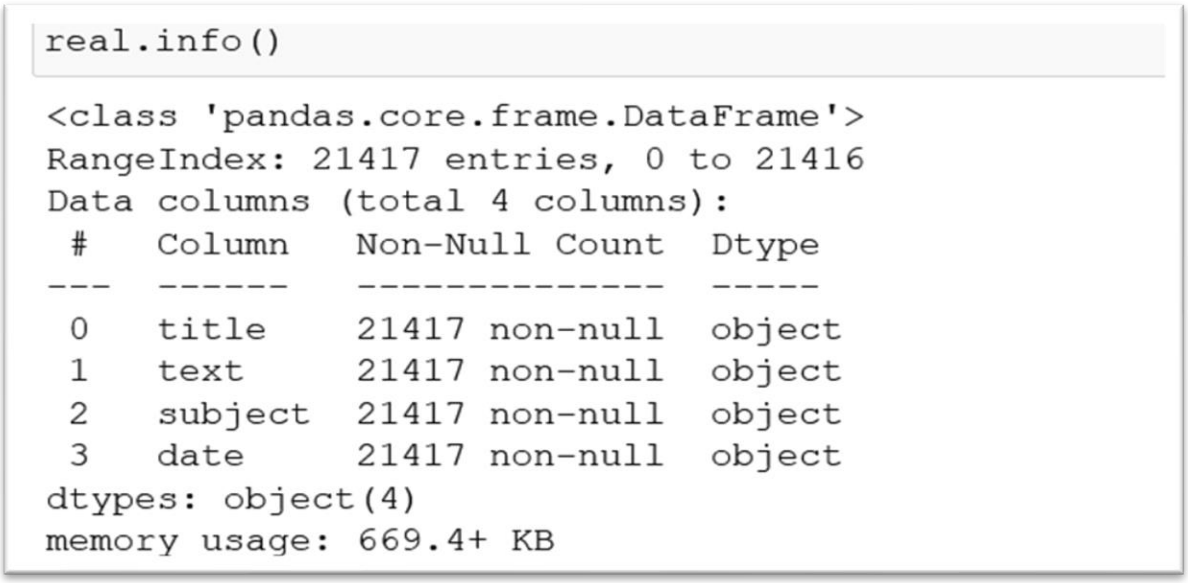
```
fake.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23481 entries, 0 to 23480
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   title    23481 non-null  object
 1   text     23481 non-null  object
 2   subject  23481 non-null  object
 3   date     23481 non-null  object
dtypes: object(4)
memory usage: 733.9+ KB
```

First, we use info function to identify columns and their datatypes and other information like null values.



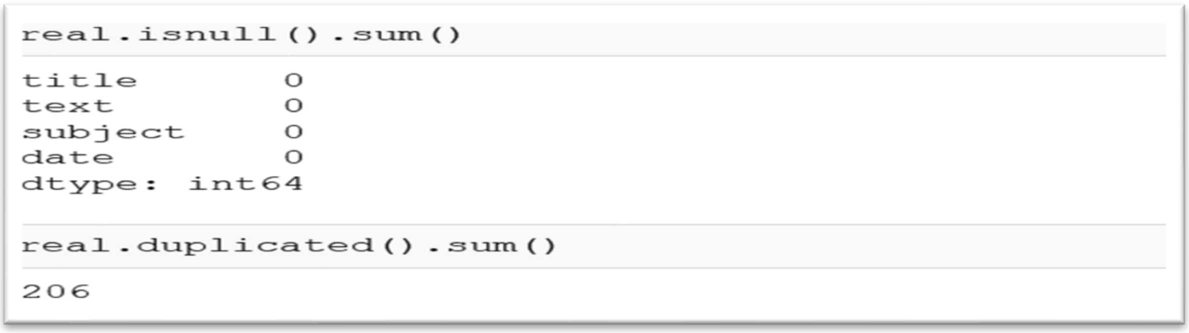
```
fake.isnull().sum()

title      0
text       0
subject    0
date       0
dtype: int64

fake.duplicated().sum()

3
```

We find the null and duplicate values for fake dataset on the above code. Same as fake dataset we also get the information about real dataset all columns and their datatypes and other details like memory usage.



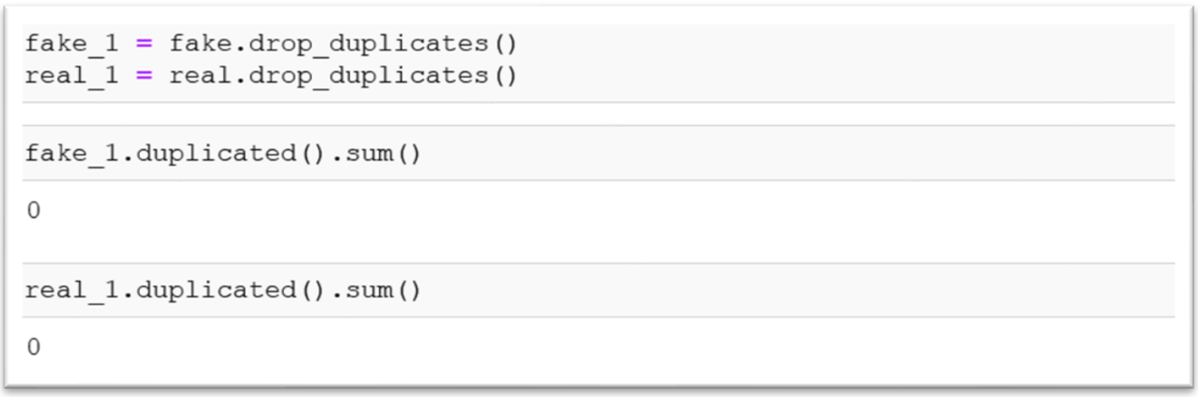
```
real.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21417 entries, 0 to 21416
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   title    21417 non-null  object
 1   text     21417 non-null  object
 2   subject  21417 non-null  object
 3   date     21417 non-null  object
dtypes: object(4)
memory usage: 669.4+ KB
```

Then we check null and duplicate values for real dataset as listed in below code.



```
real.isnull().sum()

title      0
text       0
subject    0
date       0
dtype: int64

real.duplicated().sum()

206
```

Dropping the duplicate values is the necessary step to get accurate dataset. We first drop the duplicate values and then also check if any duplicate values are left in the code below.



```
fake_1 = fake.drop_duplicates()
real_1 = real.drop_duplicates()
```

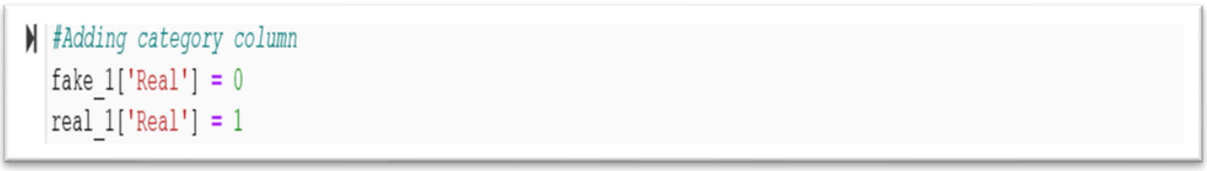
```
fake_1.duplicated().sum()
```

```
0
```

```
real_1.duplicated().sum()
```

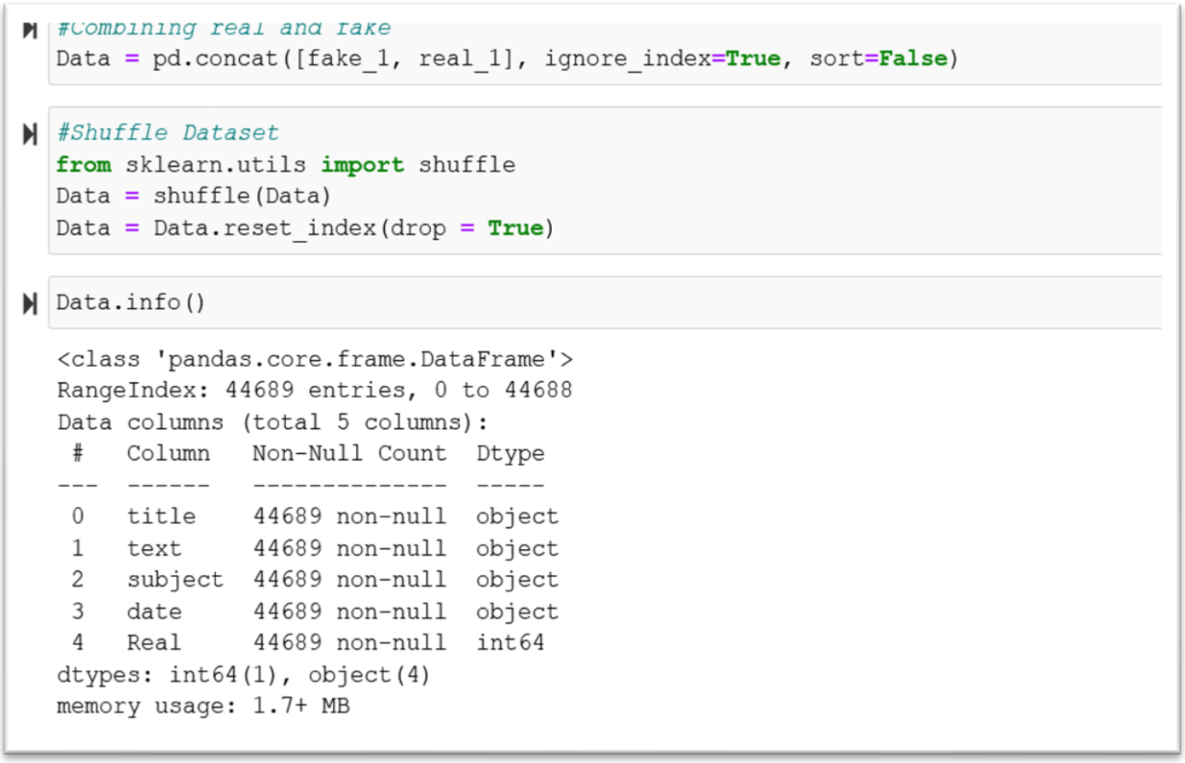
```
0
```

5. Encoding the categorical data:



```
#Adding category column
fake_1['Real'] = 0
real_1['Real'] = 1
```

Here we add the category to column and put values for fake 0 and for real 1.



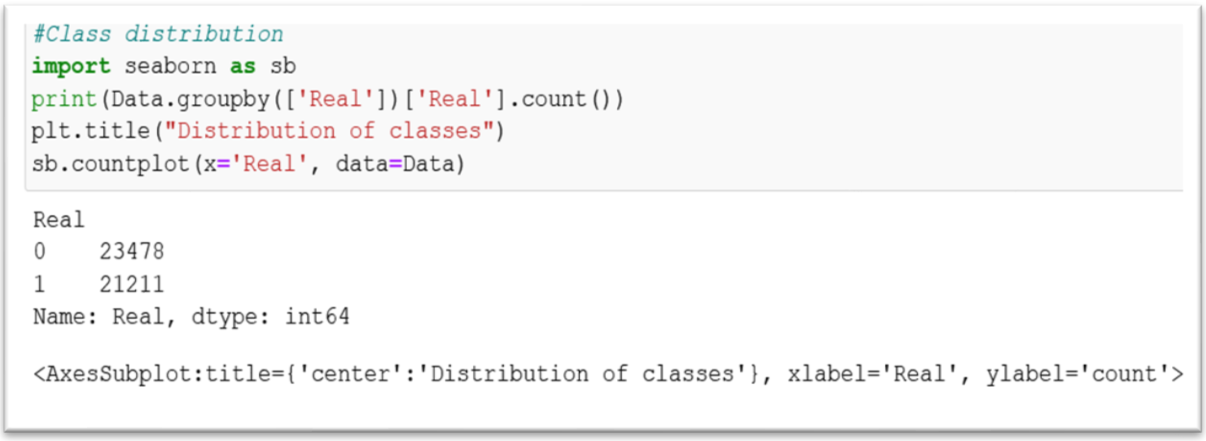
```
#Combining real and fake
Data = pd.concat([fake_1, real_1], ignore_index=True, sort=False)
```

```
#Shuffle Dataset
from sklearn.utils import shuffle
Data = shuffle(Data)
Data = Data.reset_index(drop = True)
```

```
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44689 entries, 0 to 44688
Data columns (total 5 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   title    44689 non-null  object
 1   text     44689 non-null  object
 2   subject  44689 non-null  object
 3   date     44689 non-null  object
 4   Real     44689 non-null  int64
dtypes: int64(1), object(4)
memory usage: 1.7+ MB
```

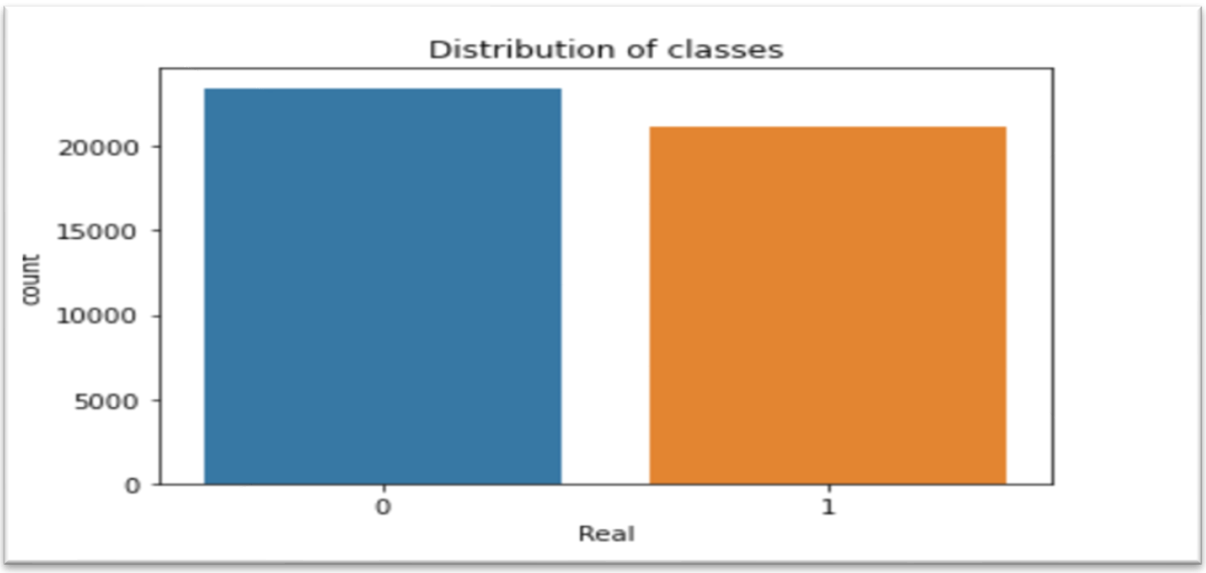
We also combine the fake and real datasets so that we have only one dataset to use. Using the shuffle function from sklearn.utils library we get our dataset in a mix form not first whole fake dataset and then real dataset.



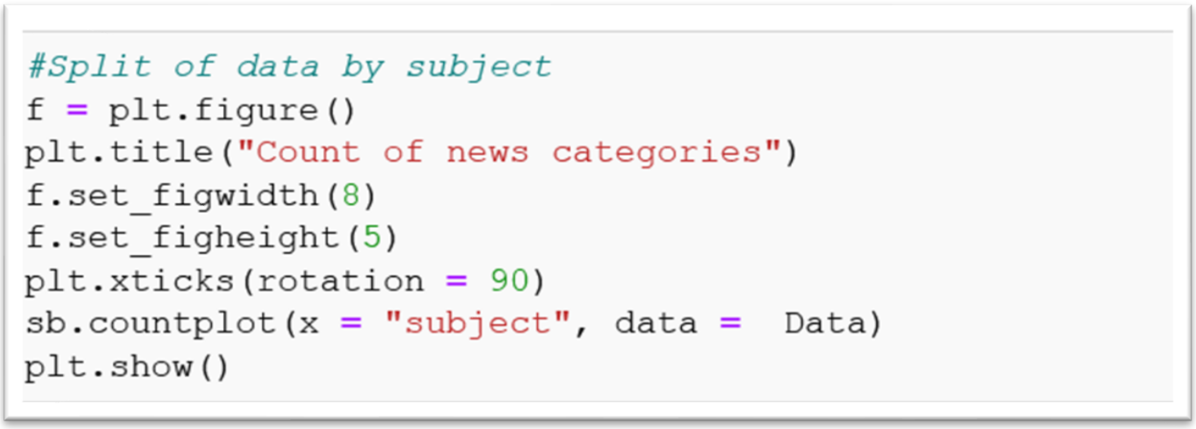
```
#Class distribution
import seaborn as sb
print(Data.groupby(['Real'])['Real'].count())
plt.title("Distribution of classes")
sb.countplot(x='Real', data=Data)
```

```
Real
0    23478
1    21211
Name: Real, dtype: int64

<AxesSubplot:title={'center':'Distribution of classes'}, xlabel='Real', ylabel='count'>
```

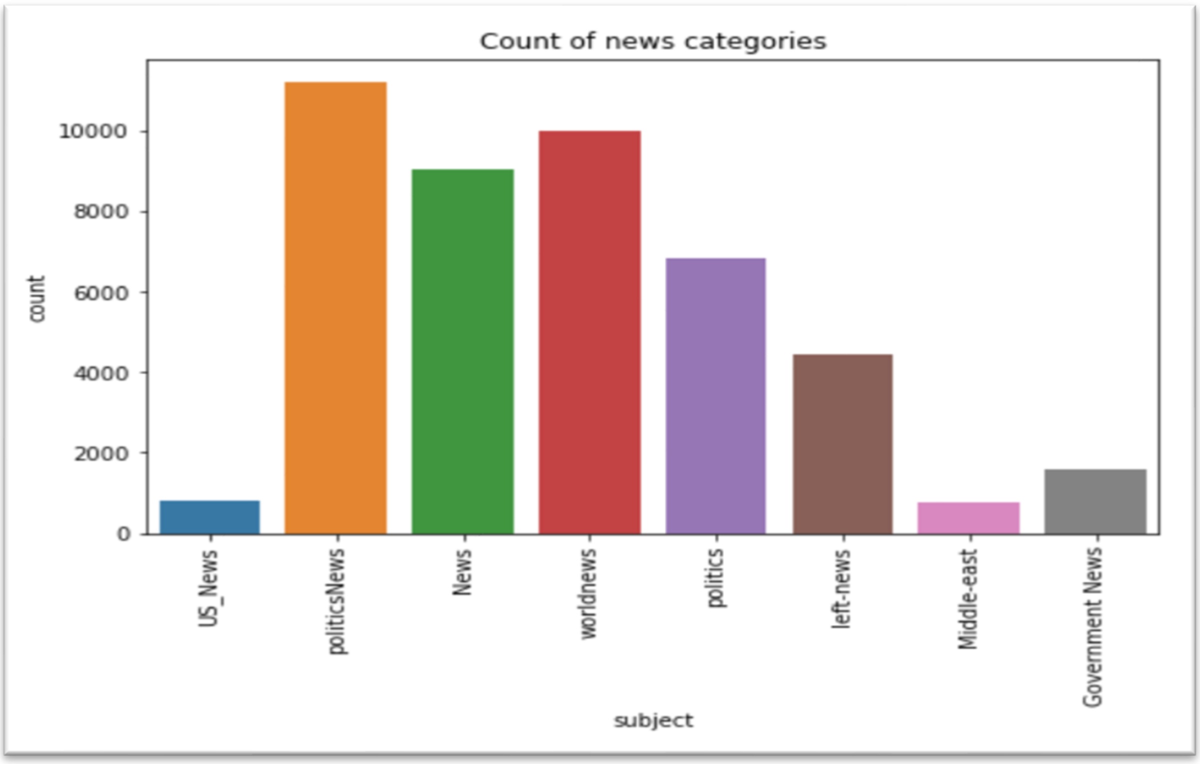
Here we distribute the data as fake and real. The value for fake is 0 and real is 1. So, we can see in the output 0 is 23478 and 1 is 21211. The visualized output in a chart form is below.



In the code below we split the data by categories to the different subjects and their counts. By the below code we plot the graph for the same.

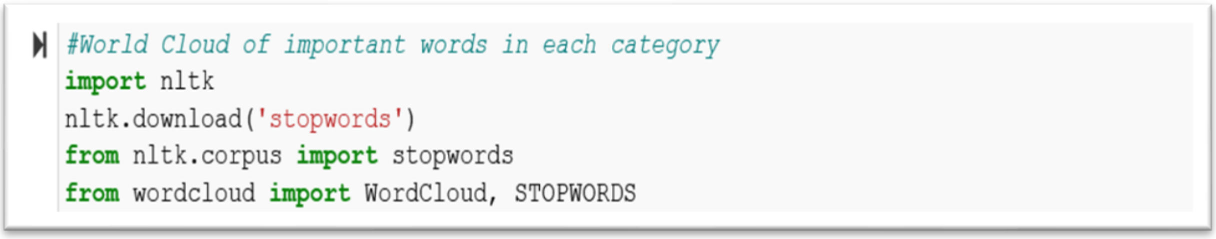


```
#Split of data by subject
f = plt.figure()
plt.title("Count of news categories")
f.set_figwidth(8)
f.set_figheight(5)
plt.xticks(rotation = 90)
sb.countplot(x = "subject", data =  Data)
plt.show()
```

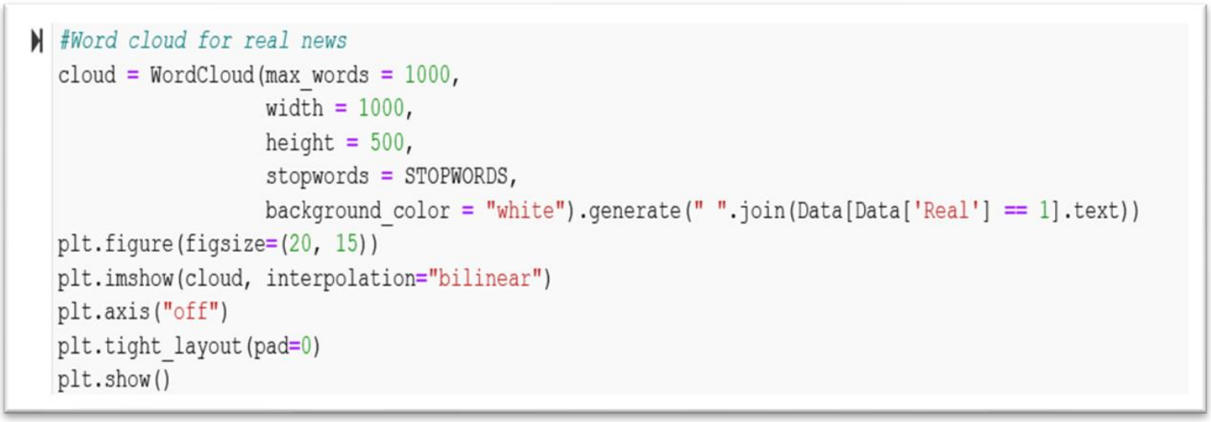


6. Splitting the dataset

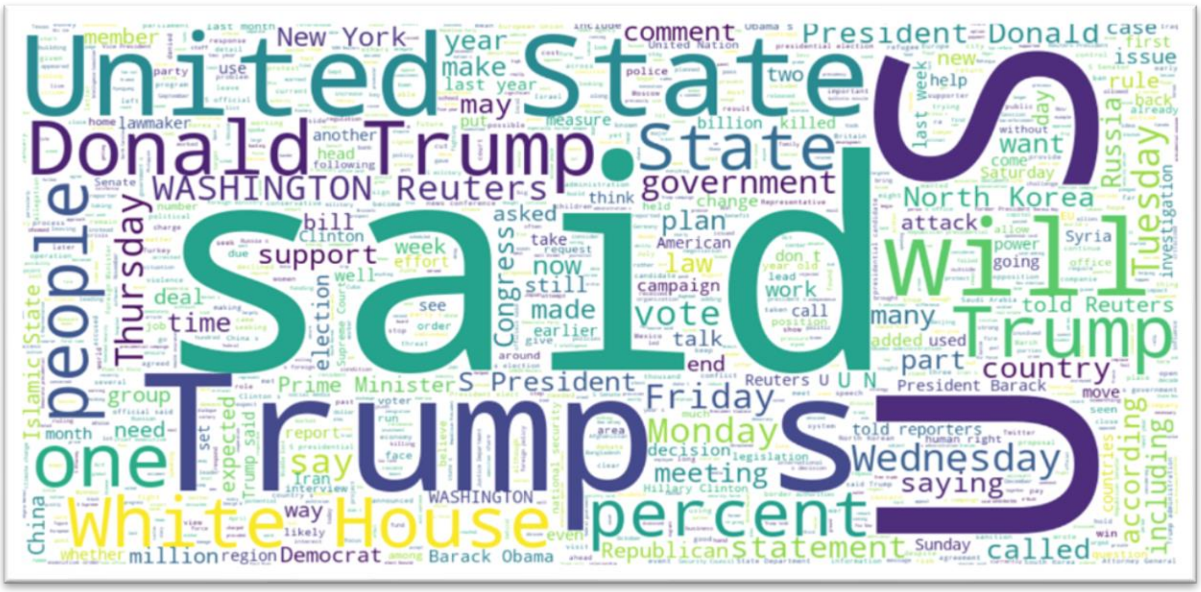
Before splitting the dataset into train and test we did word cloud for both real and fake dataset in the below code.

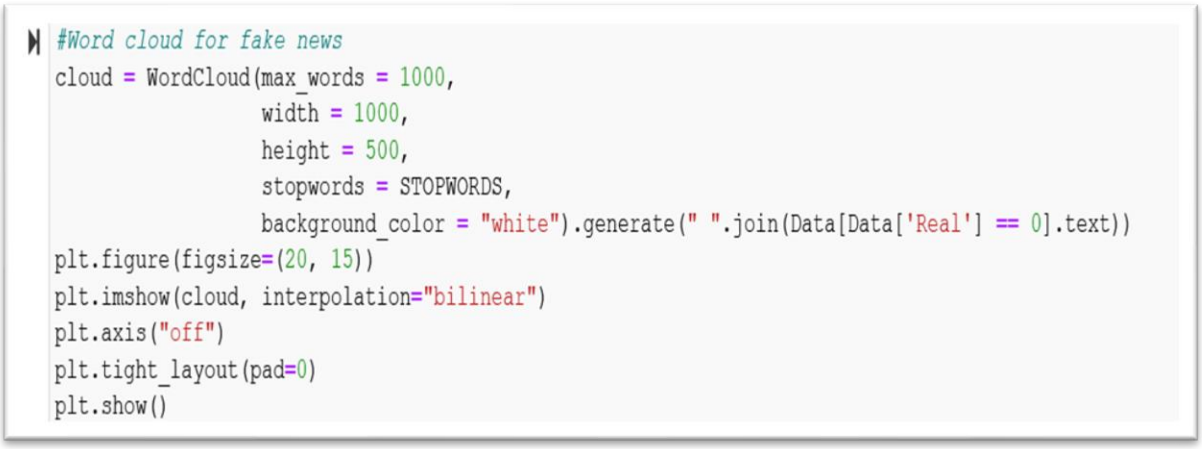


```
#World Cloud of important words in each category
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from wordcloud import WordCloud, STOPWORDS
```

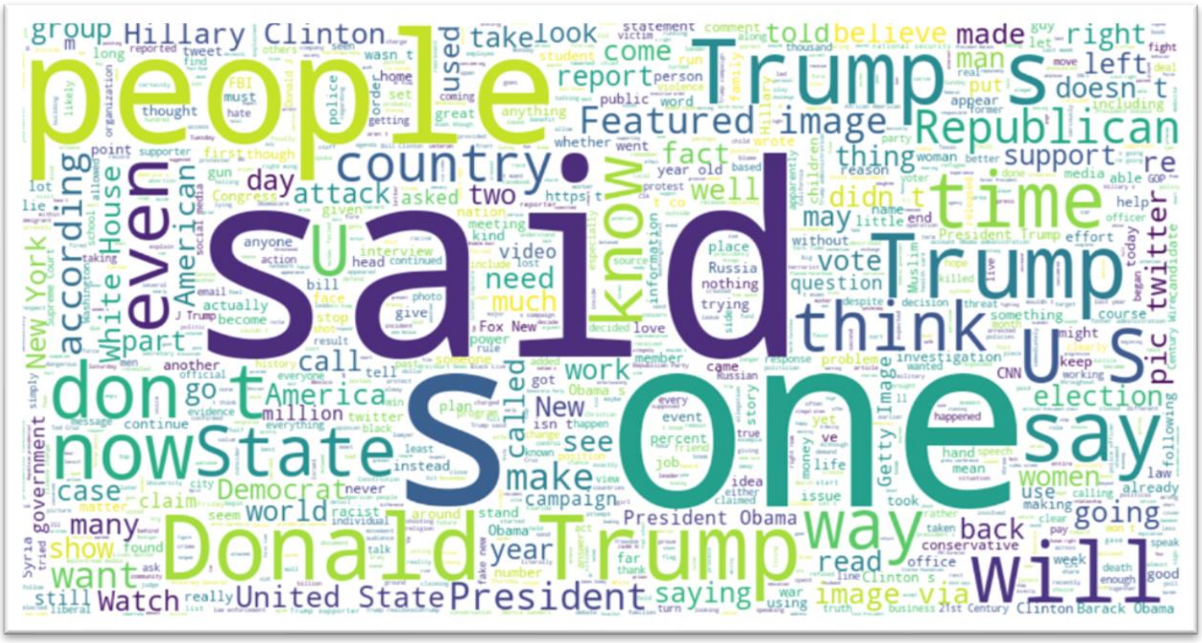


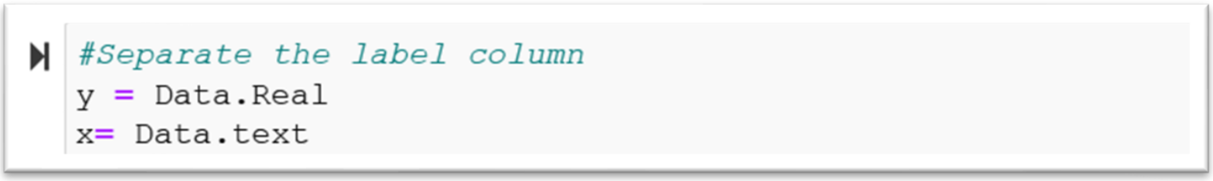
```
#Word cloud for real news
cloud = WordCloud(max_words = 1000,
                  width = 1000,
                  height = 500,
                  stopwords = STOPWORDS,
                  background_color = "white").generate(" ".join(Data[Data['Real'] == 1].text))
plt.figure(figsize=(20, 15))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```





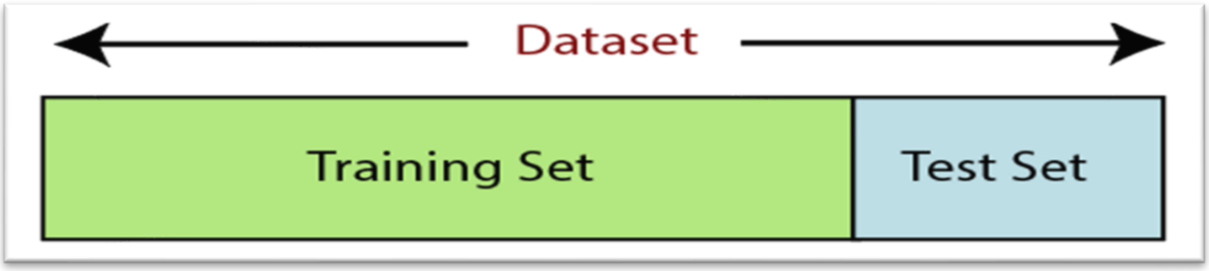
```
#Word cloud for fake news
cloud = WordCloud(max_words = 1000,
                  width = 1000,
                  height = 500,
                  stopwords = STOPWORDS,
                  background_color = "white").generate(" ".join(Data[Data['Real'] == 0].text))
plt.figure(figsize=(20, 15))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```



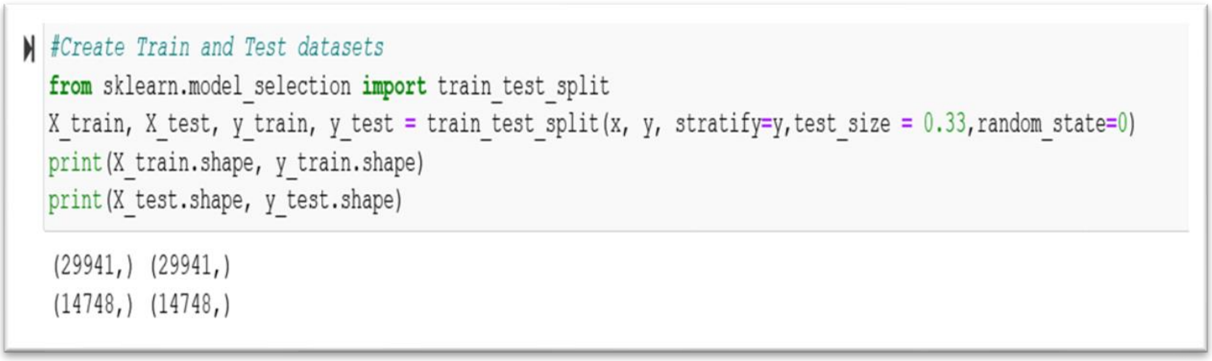


```
#Separate the label column
y = Data.Real
x= Data.text
```

Splitting the dataset is the next step in data preprocessing. Every dataset for Machine Learning model must be split into two separate sets – training set and test set.



Here we split the dataset in train and test with train_df and test_df variable. Using random_state attribute to split the data randomly. We get the output of train datasets as 10248 and 4392 as test datasets.



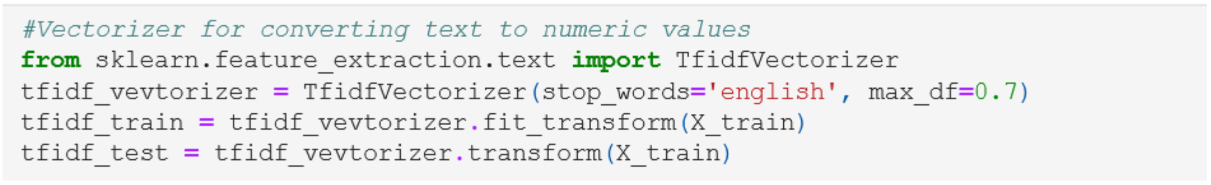
```
#Create Train and Test datasets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, stratify=y,test_size = 0.33,random_state=0)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(29941,) (29941,)
(14748,) (14748,)
```

7. Feature scaling:

Feature scaling marks the end of the **data preprocessing in Machine Learning.** It is a method to standardize the independent variables of a dataset within a specific range. In other words, feature scaling limits the range of variables.

Here we use Tfidvectorizer to ensure that if the words are not necessary then we can remove that. Basically, it creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix.



```
#Vectorizer for converting text to numeric values
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vevtorizer = TfidfVectorizer(stop_words='english', max_df=0.7)
tfidf_train = tfidf_vevtorizer.fit_transform(X_train)
tfidf_test = tfidf_vevtorizer.transform(X_train)
```

11. Algorithm Evaluation

a. Passive Aggressive Classifier:

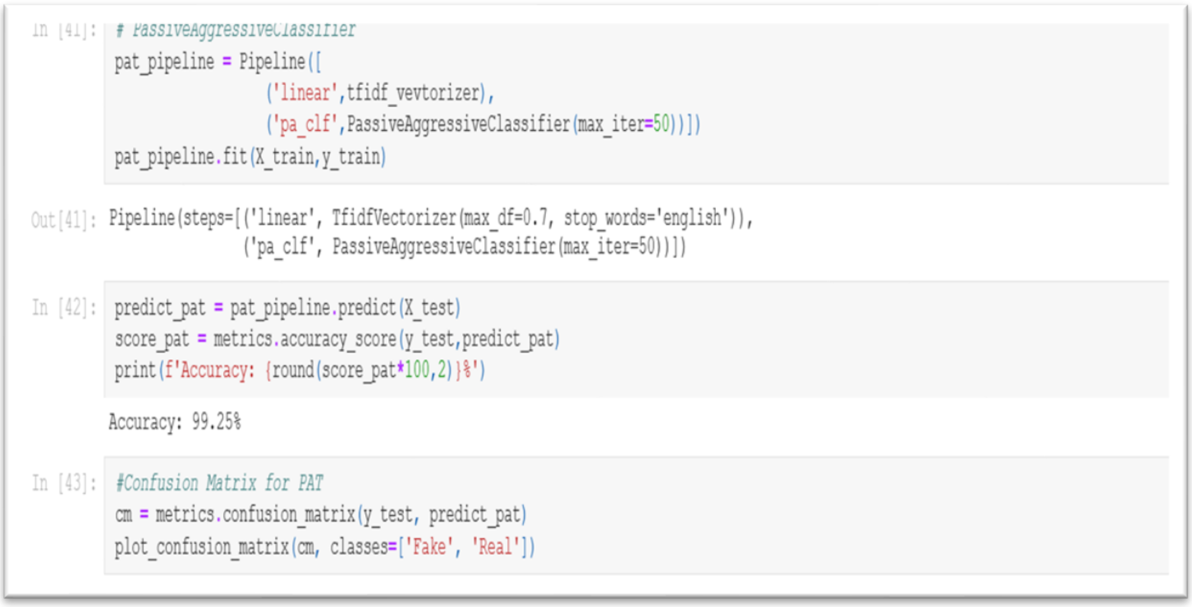
The Passive Aggressive rule is a web algorithm; ideal for classifying huge streams of information (e.g., twitter). It's straightforward to implement and extremely quick. It works by taking Associate in Nursing example, learning from it and so throwing it away. Such Associate in Nursing rule is still passive for an accurate classification outcome, and turns aggressive within the event of a mistake, change and adjusting. in contrast to most alternative algorithms, it doesn't converge. Its purpose is to create updates that correct the loss, inflicting little or no amendment within the norm of the burden vector.

Pros:
It is one of the best classifiers for fake news as it gives the highest accuracy. And it has a tiny memory footprint and they do not require a learning rate.

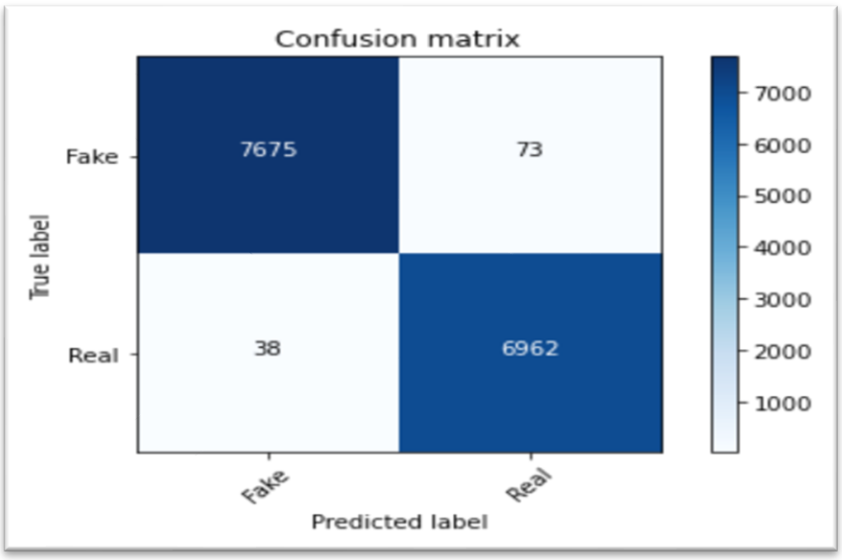
Cons:
It has only one disadvantage that they don't have a big picture of data so the result will be affected.

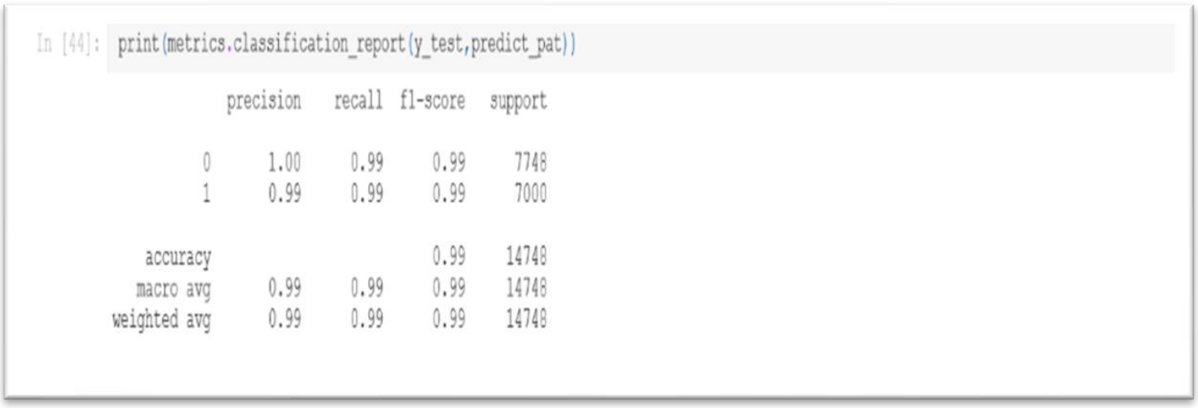
We will initialize a Passive Aggressive Classifier. We will fit this on X_train and y_train. Then, we will predict on the test set from the TfidfVectorizer and calculate the accuracy with accuracy_score () from sklearn.metrics.

We got an accuracy of 99.25% with this model. Finally, let us print out a confusion matrix to gain insight into the number of false and true negatives and positives.



```
In [41]: # PassiveAggressiveClassifier
         pat_pipeline = Pipeline([
                        ('linear',tfidf_vevtorizer),
                        ('pa_clf',PassiveAggressiveClassifier(max_iter=50))])
         pat_pipeline.fit(X_train,y_train)

Out[41]: Pipeline(steps=[('linear', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('pa_clf', PassiveAggressiveClassifier(max_iter=50))])

In [42]: predict_pat = pat_pipeline.predict(X_test)
         score_pat = metrics.accuracy_score(y_test,predict_pat)
         print(f'Accuracy: {round(score_pat*100,2)}%')

         Accuracy: 99.25%

In [43]: #Confusion Matrix for PAT
         cm = metrics.confusion_matrix(y_test, predict_pat)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```





```
In [44]: print(metrics.classification_report(y_test,predict_pat))

              precision    recall  f1-score   support

           0       1.00      0.99      0.99      7748
           1       0.99      0.99      0.99      7000

    accuracy                           0.99     14748
   macro avg       0.99      0.99      0.99     14748
weighted avg       0.99      0.99      0.99     14748
```

Passive Aggressive Classifier is one of the best classifier for the fake news detection as it shows the highest accuracy rate of 99% as per above code and we can recognize that the precision of the fake news is 100% with the support of 7748 rows.

b. KNN (K-Nearest Neighbor)

This is a supervised rule of machine learning that's used for resolution the classification issues. This stores the info concerning all the cases to classify the new case. a number of the constraints of KNN square measure it does not work well with an oversized dataset and is sensitive to outliers and missing values.

Pros:

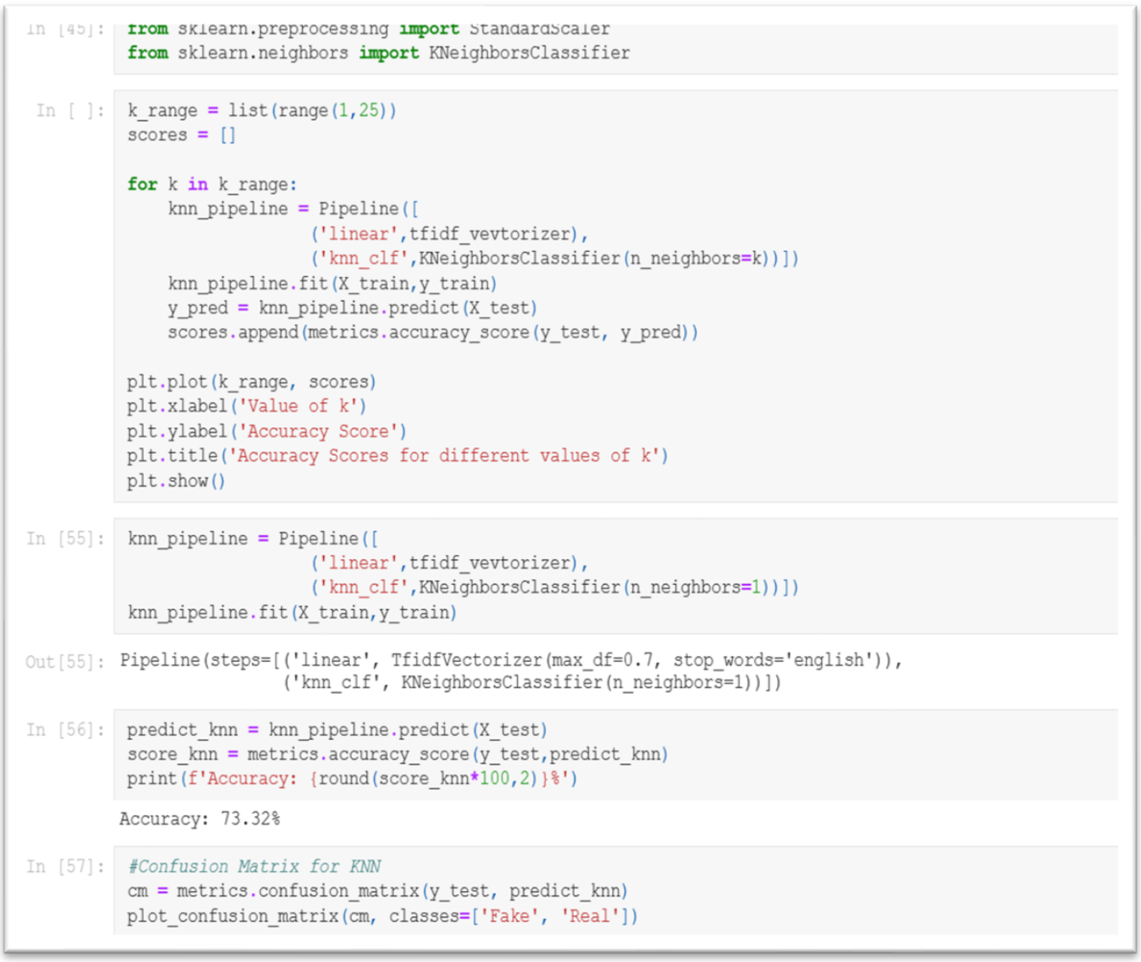
KNN is intuitive and simple. It has no assumptions, no training step, it constantly evolves, it's extremely easy to implement for multi class problems and most importantly it can be used for both classification and regression models.

Cons:

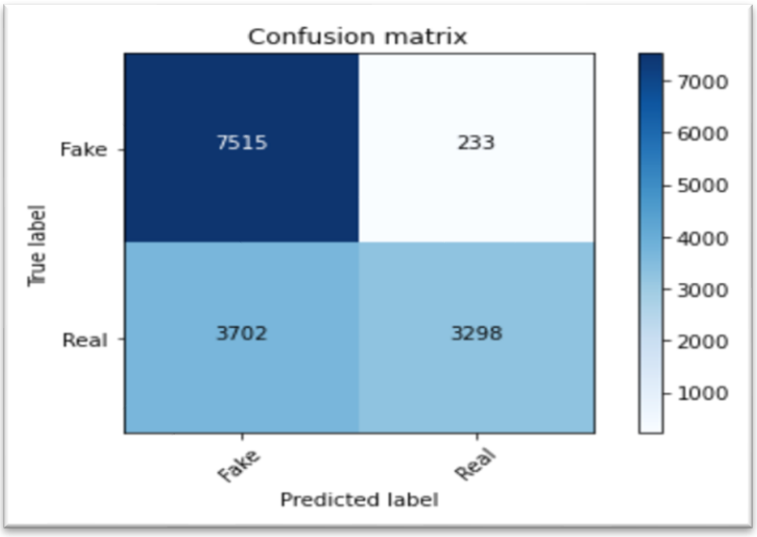
It is a slow algorithm, it needs a homogenous feature, optimal number of neighbors, if imbalanced data occurs it causes a problem, outlier sensitivity.

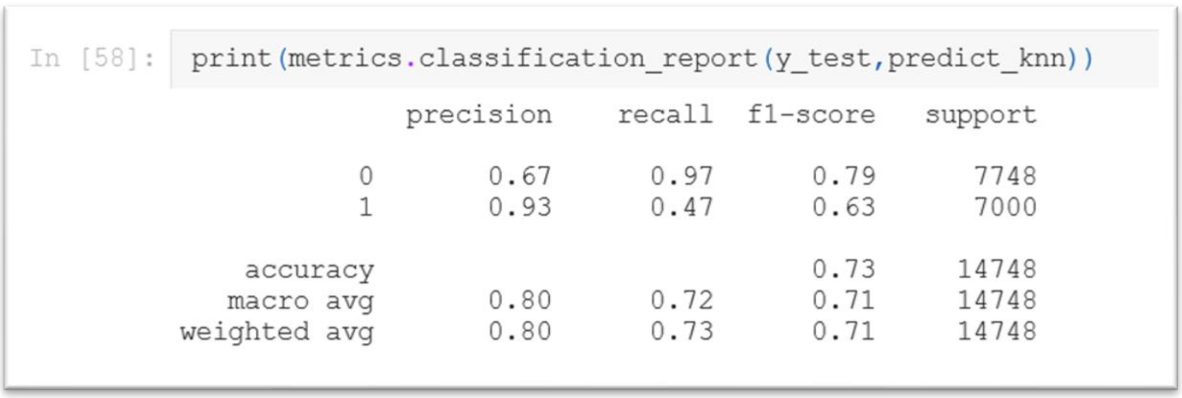
We will initialize a K Nearest Neighbor Classifier. We will fit this on X_train and y_train. Then, we will predict on the test set from the TfidfVectorizer and calculate the accuracy with accuracy_score () from sklearn.metrics.

We got an accuracy of 73% with this model. Finally, let’s print out a confusion matrix to gain insight into the number of false and true negatives and positives.



```
In [45]: from sklearn.preprocessing import StandardScaler
         from sklearn.neighbors import KNeighborsClassifier

In [ ]:  k_range = list(range(1,25))
         scores = []

         for k in k_range:
             knn_pipeline = Pipeline([
                          ('linear',tfidf_vevtorizer),
                          ('knn_clf',KNeighborsClassifier(n_neighbors=k))])
             knn_pipeline.fit(X_train,y_train)
             y_pred = knn_pipeline.predict(X_test)
             scores.append(metrics.accuracy_score(y_test, y_pred))

         plt.plot(k_range, scores)
         plt.xlabel('Value of k')
         plt.ylabel('Accuracy Score')
         plt.title('Accuracy Scores for different values of k')
         plt.show()

In [55]: knn_pipeline = Pipeline([
                          ('linear',tfidf_vevtorizer),
                          ('knn_clf',KNeighborsClassifier(n_neighbors=1))])
         knn_pipeline.fit(X_train,y_train)

Out[55]: Pipeline(steps=[('linear', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('knn_clf', KNeighborsClassifier(n_neighbors=1))])

In [56]: predict_knn = knn_pipeline.predict(X_test)
         score_knn = metrics.accuracy_score(y_test,predict_knn)
         print(f'Accuracy: {round(score_knn*100,2)}%')

         Accuracy: 73.32%

In [57]: #Confusion Matrix for KNN
         cm = metrics.confusion_matrix(y_test, predict_knn)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```





```
In [58]: print(metrics.classification_report(y_test,predict_knn))
              precision    recall  f1-score   support

           0       0.67      0.97      0.79      7748
           1       0.93      0.47      0.63      7000

    accuracy                           0.73     14748
   macro avg       0.80      0.72      0.71     14748
weighted avg       0.80      0.73      0.71     14748
```

K Nearest Neighbor Classifier is less used classifier for the fake news detection as it shows the accuracy rate of 73% as per above code and we can recognize that the precision of the fake news is 67% with the support of 7748 rows. And the precision of real news is 93% with the support of 7000 rows.

c. <u>Decision Tree:</u>

The decision tree could be a tree model; the question method corresponds to a path from the basis to a leaf. In every inner node, one feature price is going to be examined, by examination it with a pre-calculated price it'll decide that subtree to travel. Once it reaches a leaf, the result keeps on going to be the solution. Compared to alternative algorithms call trees need less effort for information preparation throughout pre-processing. It doesn't need social control of information and doesn't need scaling of information also.

Pros:

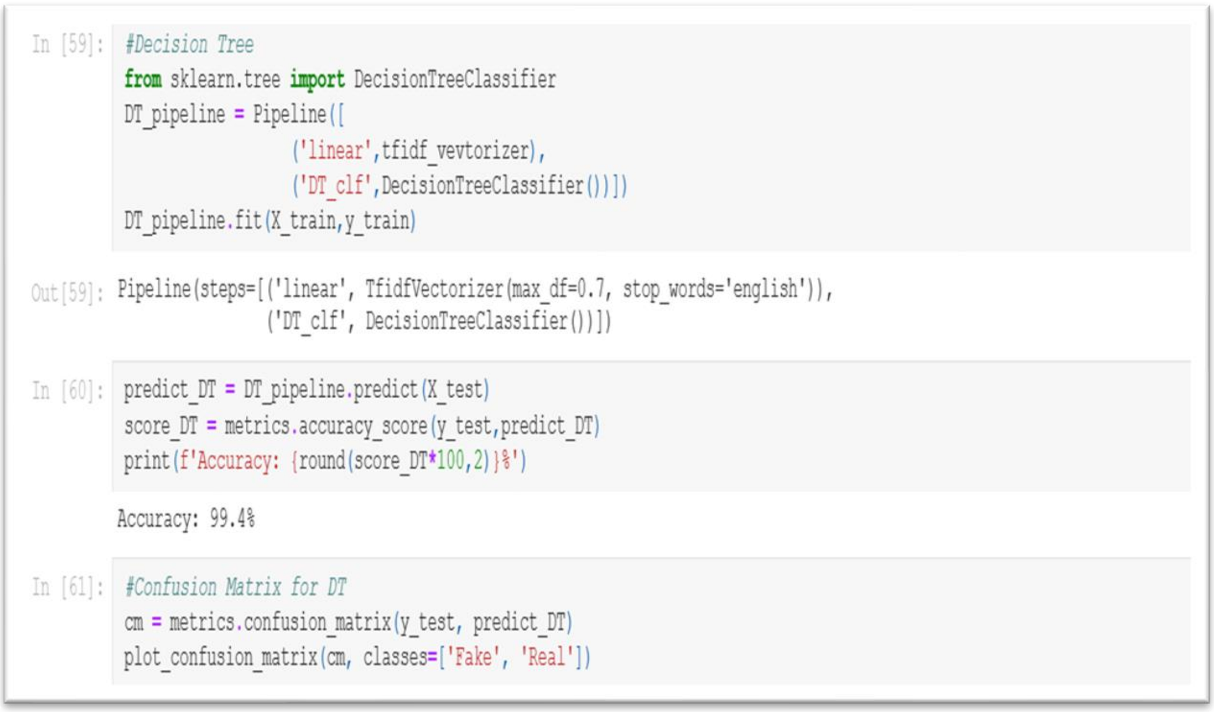
Decision trees require less effort as compared to other algorithms for data preparation during pre-processing. A decision tree does not require normalization of data. A decision tree is easy to explain to technical teams as well as stakeholders.

Cons:

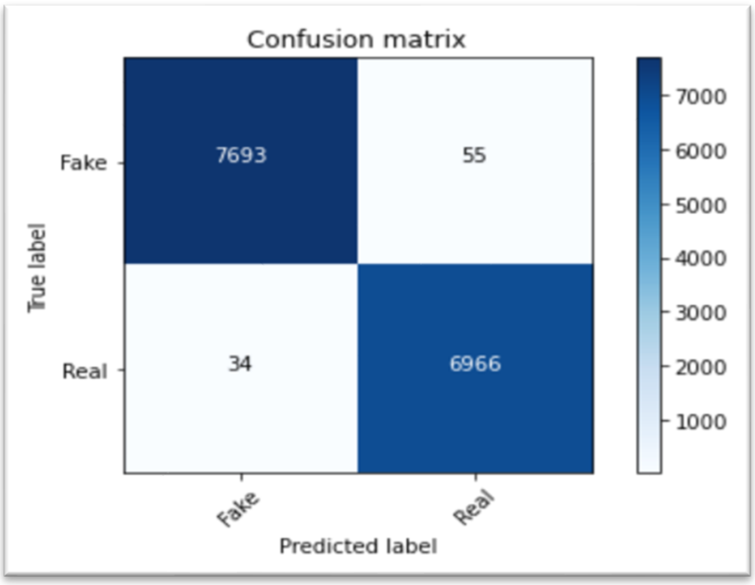
A minor change in the data can cause a large change in the structure of Decision Tree. It takes more time to train the model. It’s training is expensive and complex. It is inadequate for predicting continuous values.

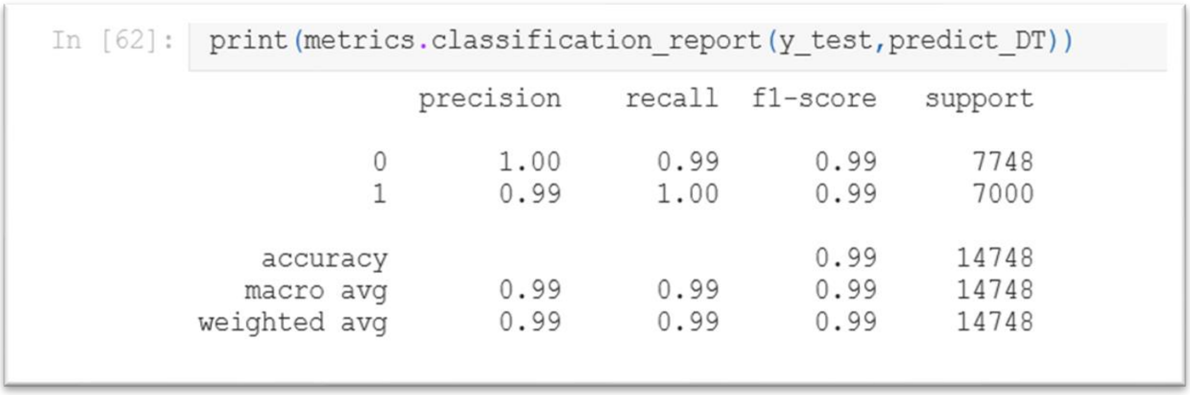
We are using Decision Tree algorithm. We will fit this on X_train and y_train. Then, we’ll predict on the test set from the TfidfVectorizer and calculate the accuracy with accuracy_score () from sklearn.metrics.

We got an accuracy of 94.4% with this model. Finally, let’s print out a confusion matrix to gain insight into the number of false and true negatives and positives.



```
In [59]: #Decision Tree
         from sklearn.tree import DecisionTreeClassifier
         DT_pipeline = Pipeline([
                         ('linear',tfidf_vevtorizer),
                         ('DT_clf',DecisionTreeClassifier())])
         DT_pipeline.fit(X_train,y_train)

Out[59]: Pipeline(steps=[('linear', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('DT_clf', DecisionTreeClassifier())])

In [60]: predict_DT = DT_pipeline.predict(X_test)
         score_DT = metrics.accuracy_score(y_test,predict_DT)
         print(f'Accuracy: {round(score_DT*100,2)}%')

         Accuracy: 99.4%

In [61]: #Confusion Matrix for DT
         cm = metrics.confusion_matrix(y_test, predict_DT)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```





```
In [62]: print(metrics.classification_report(y_test,predict_DT))

              precision    recall  f1-score   support

           0       1.00      0.99      0.99      7748
           1       0.99      1.00      0.99      7000

    accuracy                           0.99     14748
   macro avg       0.99      0.99      0.99     14748
weighted avg       0.99      0.99      0.99     14748
```

Decision Tree Classifier is good classifier for the fake news detection as it shows the accuracy rate of 99% as per above code and we can recognize that the precision of the fake news is 100% with the support of 7748 rows. And the precision of real news is 99% with the support of 7000 rows.

d. <u>Naïve Bayes Classifier:</u>

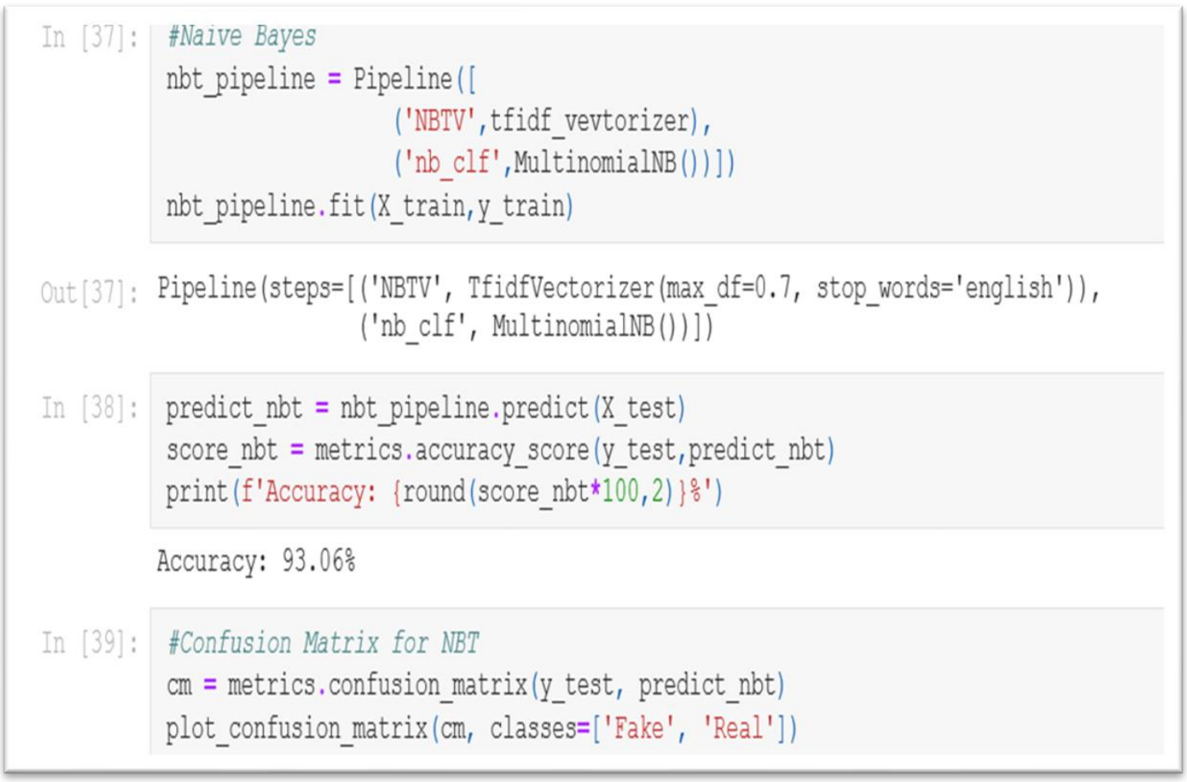
This classification technique is predicated on Bayes theorem, that assumes that the presence of a selected feature during a category is freelance of the presence of the other feature. It provides the simplest way for calculative the posterior chance. the most imitation of Naive Bayes is that the assumption of freelance predictors. Naive Bayes implicitly assumes that each one the attributes area unit reciprocally freelance. In world, it's nearly not possible that we tend to get a group of predictors that area unit utterly freelance.

Pros:

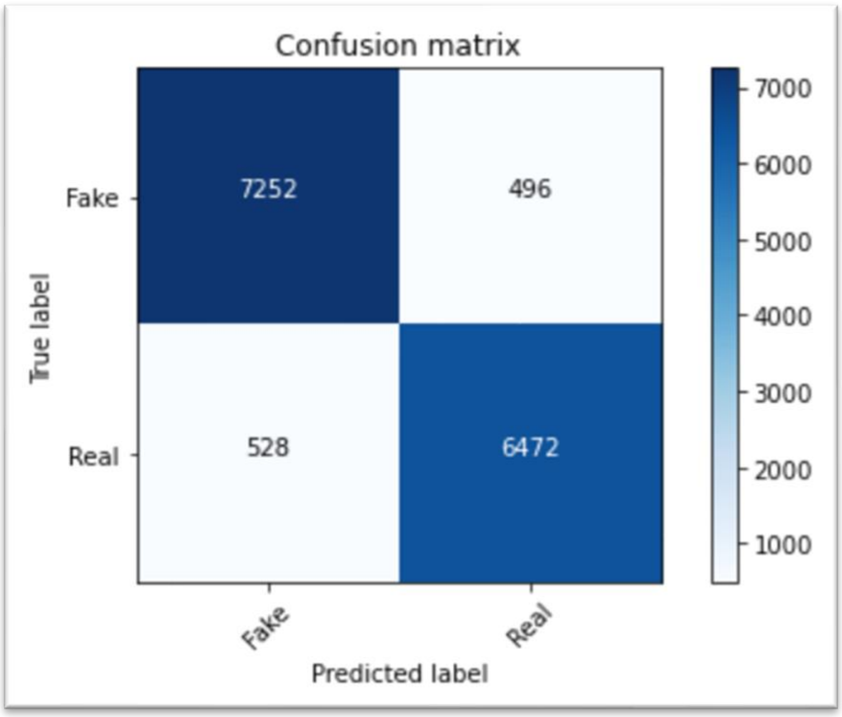
Naïve byes classifier works very fast and can easily predict the class of a test dataset. It performs well with categorical variables in the comparison of numerical variables.

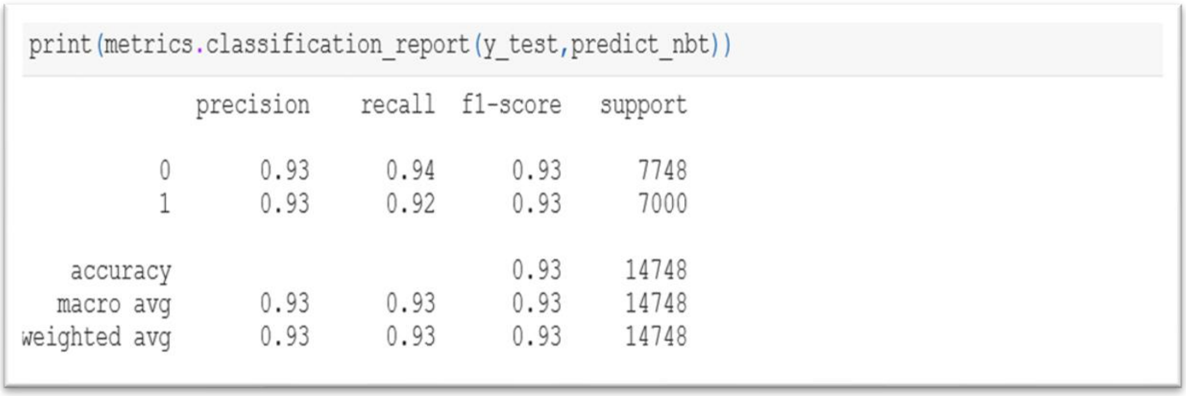
Cons:

If the test dataset has a categorical variable that wasn't present in training dataset, Naïve byes model assign it to zero probability and won't be able to make any prediction in that dataset. It assumes all the features as independent variables which is not possible always in real-time data.



```
In [37]: #Naive Bayes
         nbt_pipeline = Pipeline([
                             ('NBTV',tfidf_vevtorizer),
                             ('nb_clf',MultinomialNB())])
         nbt_pipeline.fit(X_train,y_train)

Out[37]: Pipeline(steps=[('NBTV', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('nb_clf', MultinomialNB())])

In [38]: predict_nbt = nbt_pipeline.predict(X_test)
         score_nbt = metrics.accuracy_score(y_test,predict_nbt)
         print(f'Accuracy: {round(score_nbt*100,2)}%')

         Accuracy: 93.06%

In [39]: #Confusion Matrix for NBT
         cm = metrics.confusion_matrix(y_test, predict_nbt)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```




```
print(metrics.classification_report(y_test,predict_nbt))

              precision    recall  f1-score   support

           0       0.93      0.94      0.93      7748
           1       0.93      0.92      0.93      7000

    accuracy                           0.93     14748
   macro avg       0.93      0.93      0.93     14748
weighted avg       0.93      0.93      0.93     14748
```

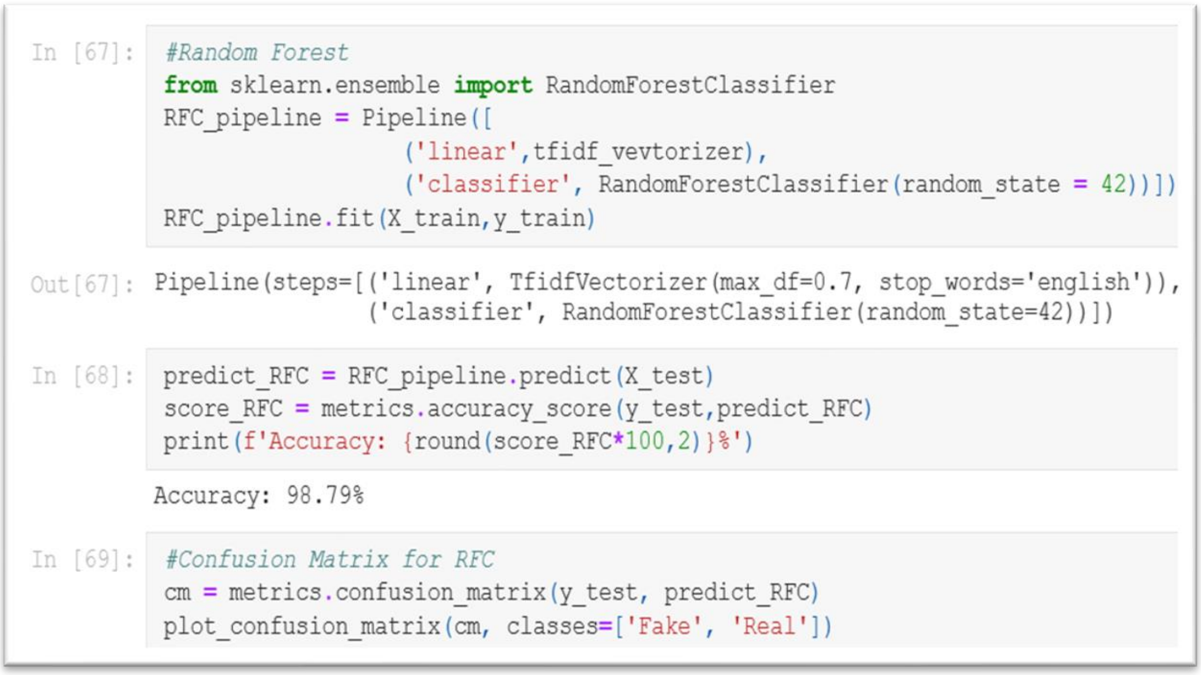
Naïve - Bayes Classifier is good classifier for the fake news detection as it shows the accuracy rate of 93% as per above code and we can recognize that the precision of the fake news is 93% with the support of 7748 rows. And the precision of real news is 93% with the support of 7000 rows.

e. Random Forest:

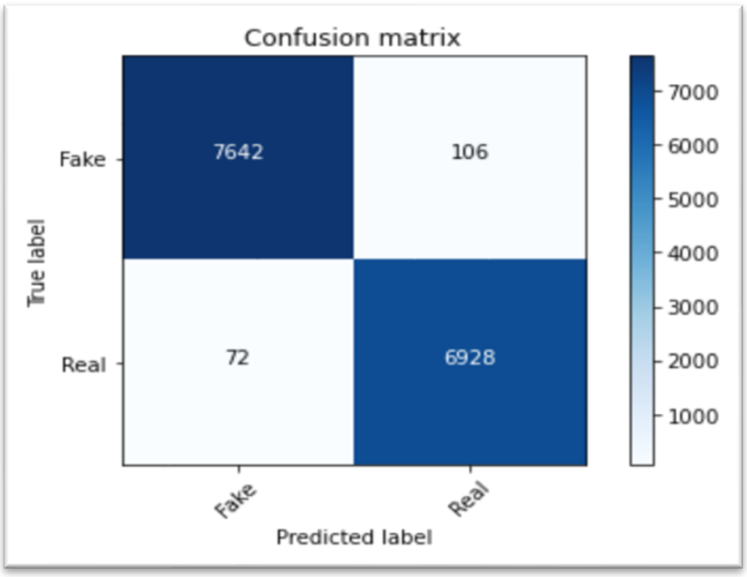
In this classifier, there are different random forests that give a value and a value with more votes is the actual result of this classifier. This algorithm is fast to train but quite slow to create predictions once it is trained.

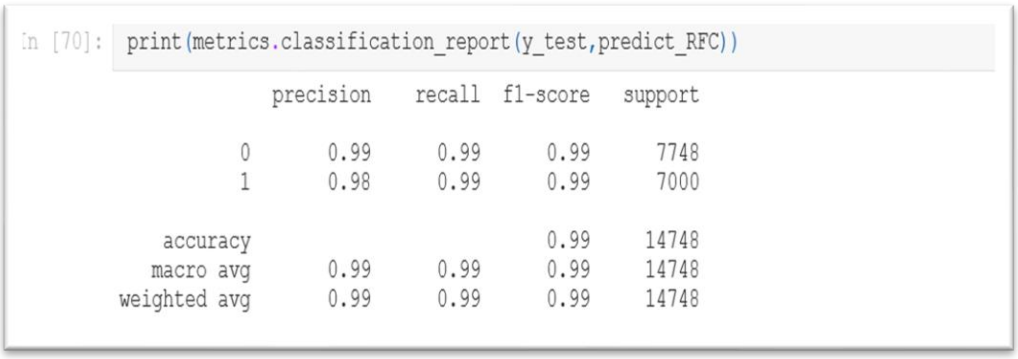
Pros:
Random forest can be used to solve both classification as well as regression problems. It works well with categorical and continuous variables. Random forest handles missing values automatically.

Cons:
Random forest requires more computational power and resources as it creates a lot of trees and combines their output. It requires much more time to train the dataset.



```
In [67]: #Random Forest
         from sklearn.ensemble import RandomForestClassifier
         RFC_pipeline = Pipeline([
                             ('linear',tfidf_vevtorizer),
                             ('classifier', RandomForestClassifier(random_state = 42))])
         RFC_pipeline.fit(X_train,y_train)

Out[67]: Pipeline(steps=[('linear', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('classifier', RandomForestClassifier(random_state=42))])

In [68]: predict_RFC = RFC_pipeline.predict(X_test)
         score_RFC = metrics.accuracy_score(y_test,predict_RFC)
         print(f'Accuracy: {round(score_RFC*100,2)}%')

         Accuracy: 98.79%

In [69]: #Confusion Matrix for RFC
         cm = metrics.confusion_matrix(y_test, predict_RFC)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```





```
In [70]: print(metrics.classification_report(y_test,predict_RFC))

              precision    recall  f1-score   support

           0       0.99      0.99      0.99      7748
           1       0.98      0.99      0.99      7000

    accuracy                           0.99     14748
   macro avg       0.99      0.99      0.99     14748
weighted avg       0.99      0.99      0.99     14748
```

Random Forest is a best classifier for the fake news detection as it shows the accuracy rate of 99% as per above code and we can recognize that the precision of the fake news is 99% with the support of 7748 rows. And the precision of real news is 98% with the support of 7000 rows.

f. <u>Logistic Regression:</u>

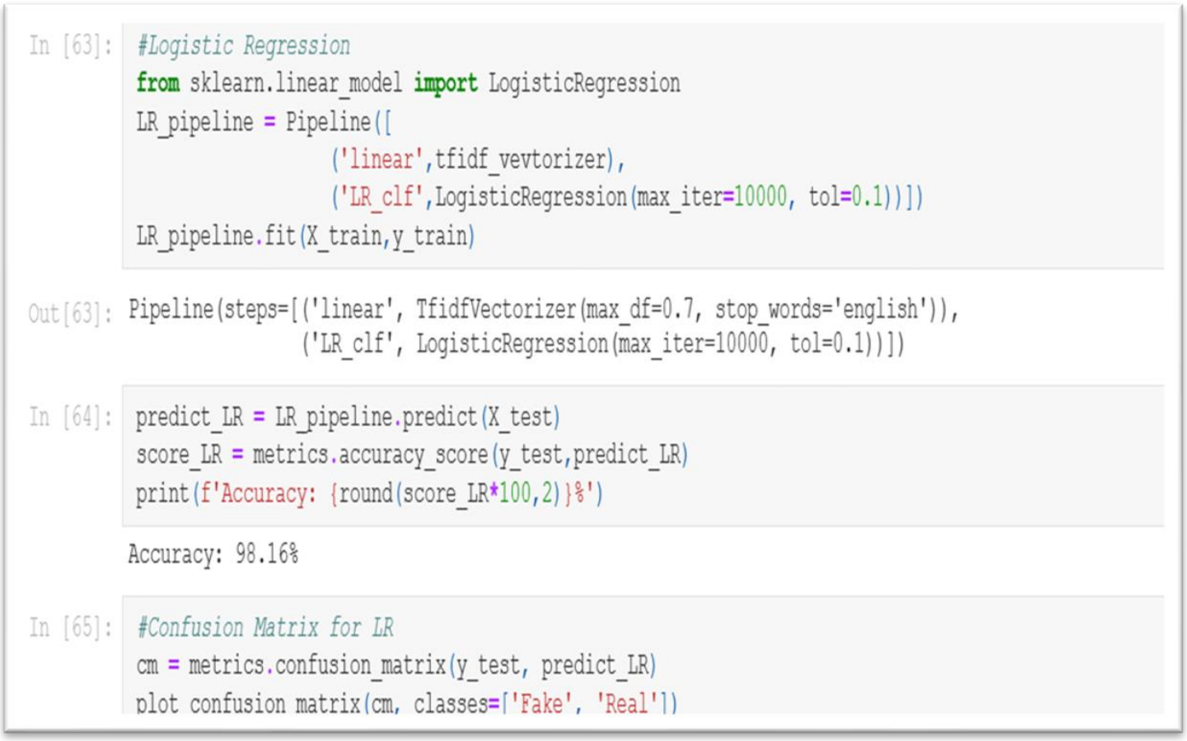
Logistic Regression is a Machine Learning classification algorithmic program that's accustomed predict the likelihood of a categorical variable quantity. it's easier to implement, interpret, and extremely economical to coach. If the quantity of observations is lesser than the quantity of options, supply Regression mustn't be used, otherwise, it's going to result in overfitting.

Pros:

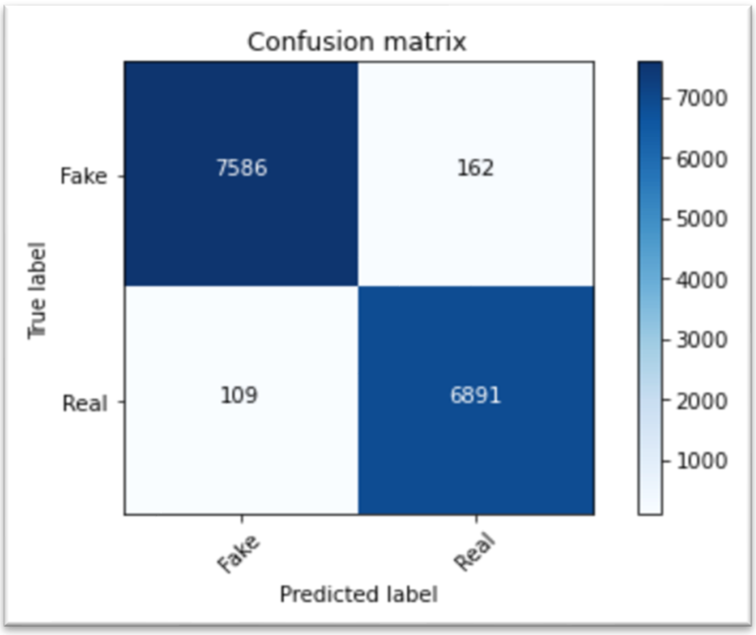
Logistic regression is easier to implement, and it is very efficient to train the dataset. It is quick to identify unknown records. It can easily extend to multiple classes.

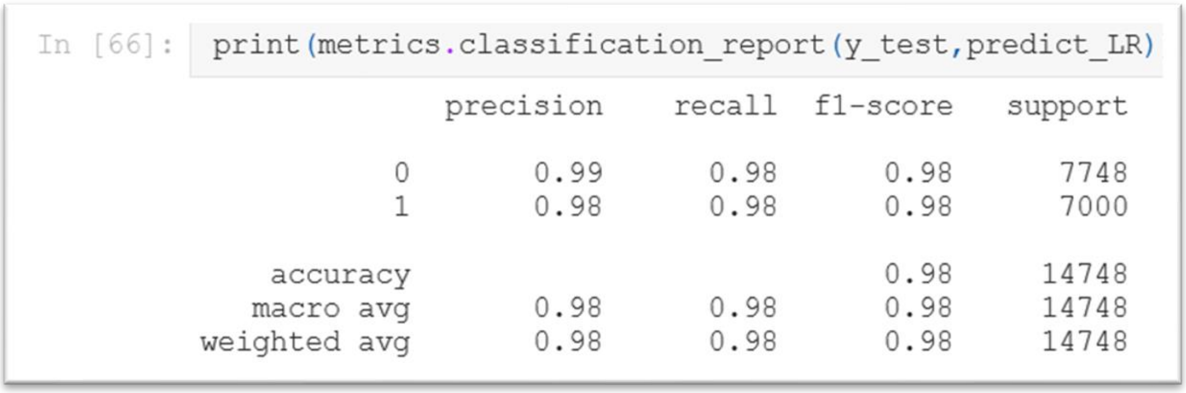
Cons:

Nonlinear problems cannot be solved with logistic regression since they have a linear decision surface. If the number of observations is less than number of features, logistic regression should not be used, otherwise it may lead to overfitting.



```
In [63]: #Logistic Regression
         from sklearn.linear_model import LogisticRegression
         LR_pipeline = Pipeline([
                         ('linear',tfidf_vevtorizer),
                         ('LR_clf',LogisticRegression(max_iter=10000, tol=0.1))])
         LR_pipeline.fit(X_train,y_train)

Out[63]: Pipeline(steps=[('linear', TfidfVectorizer(max_df=0.7, stop_words='english')),
                         ('LR_clf', LogisticRegression(max_iter=10000, tol=0.1))])

In [64]: predict_LR = LR_pipeline.predict(X_test)
         score_LR = metrics.accuracy_score(y_test,predict_LR)
         print(f'Accuracy: {round(score_LR*100,2)}%')

         Accuracy: 98.16%

In [65]: #Confusion Matrix for LR
         cm = metrics.confusion_matrix(y_test, predict_LR)
         plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```





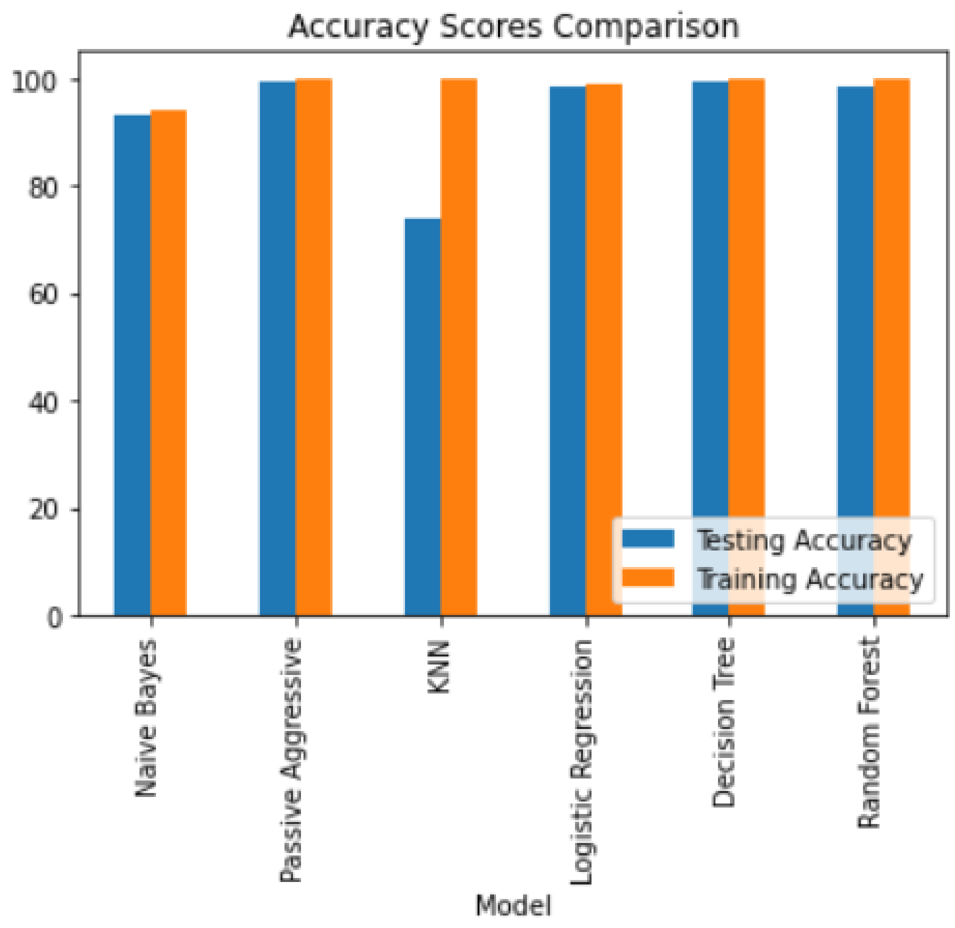
```
In [66]:  print(metrics.classification_report(y_test,predict_LR)
                 precision    recall  f1-score   support

            0       0.99      0.98      0.98      7748
            1       0.98      0.98      0.98      7000

     accuracy                           0.98     14748
    macro avg       0.98      0.98      0.98     14748
 weighted avg       0.98      0.98      0.98     14748
```

Logistic Regression Classifier is a best classifier for the fake news detection as it shows the accuracy rate of 98% as per above code and we can recognize that the precision of the fake news is 99% with the support of 7748 rows. And the precision of real news is 98% with the support of 7000 rows.

12. Candidate Algorithm Selection

Model selection is the process of choosing one among many candidate models for a predictive modelling problem. There may be many competing concerns when performing model selection beyond model performance, such as complexity, maintainability, and available resources. Here we represent the result of several algorithms on modeling fake news detection data. After cleaning and preprocessing data, we trained classification models such as logistic regression, decision tree, random forest, naïve byes, passive aggressive classifier, kNN. The highest performance of classification models are passive aggressive classifier, random forest, and decision tree. We use confusion matrix to calculate the accuracy value. Passive aggressive classifier is more accurate than other algorithms with 99.99 % for training datasets and 99.29 % for testing datasets. kNN accuracy is like passive aggressive classifier for training dataset but it reduced in testing datasets as 74.02 %. So, we cannot pick kNN based on only training datasets. Same with the logistic regression, it has 99.06 % accuracy in training period, but it curbed with approx. 1% as 98.27 %. We choose other two algorithms, decision tree and random forest as per their training and testing accuracy levels 99.99 % respectively. Decision tree and random forest has the similar accuracy in their training period but when it comes to testing data random forest accuracy reduced with 98.54 % and decision tree accuracy reduced with approx. 0.49% as 99.50 % which is better than random forest. So as per the results we found that these three algorithms suit best for our fake news detection datasets.

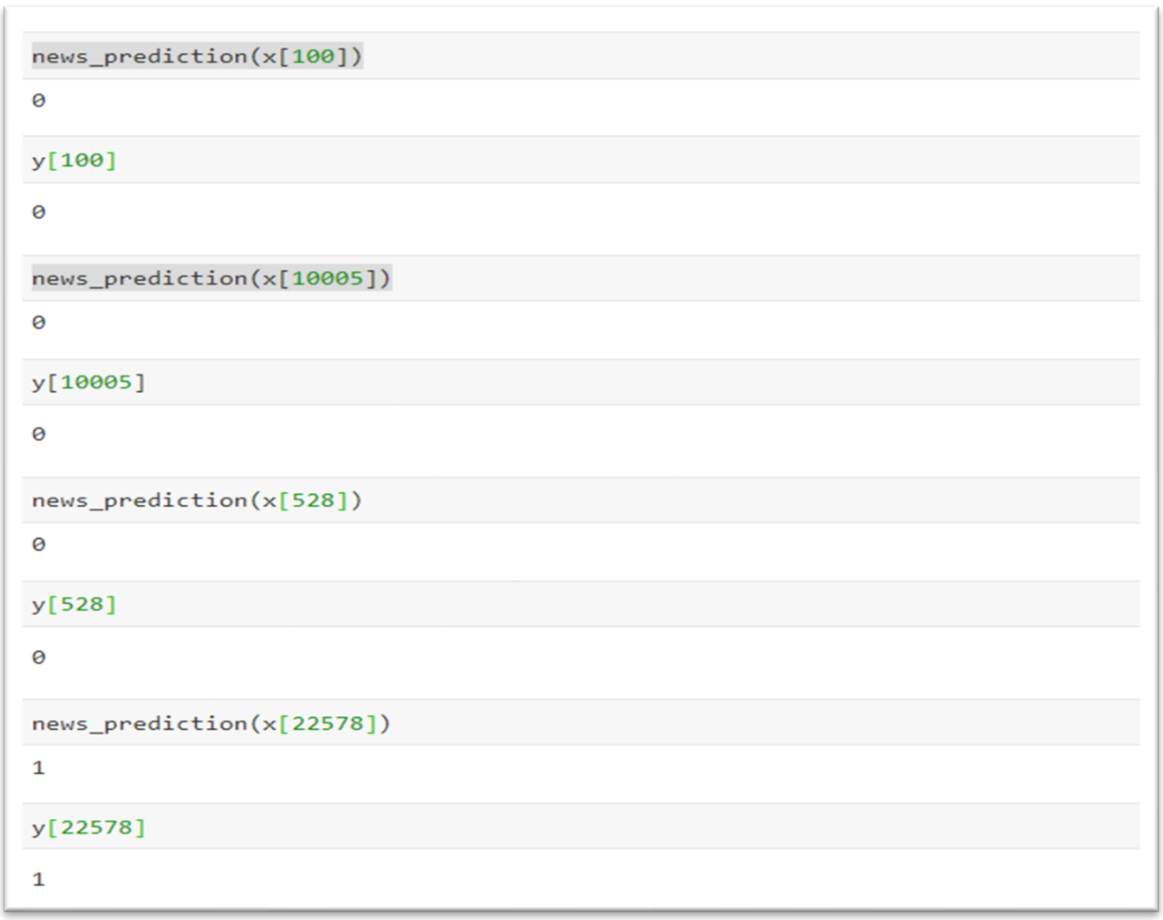
All the models are compared using the test and train accuracy scores. Accuracy is used to compare the models as we have enough observations for the analysis and there is only minimal class imbalance.



KNN and Naïve Bayes models have the least accuracy scores and are excluded from further analysis. All other models have very high testing and training accuracy. Among the rest of the models, Decision Tree model has the highest testing accuracy (99.5%), followed by Passive Aggressive model (99.3%). Random Forest and Logistic Regression models have 98% testing accuracy as well. Decision Tree model is selected as the final model for the analysis.

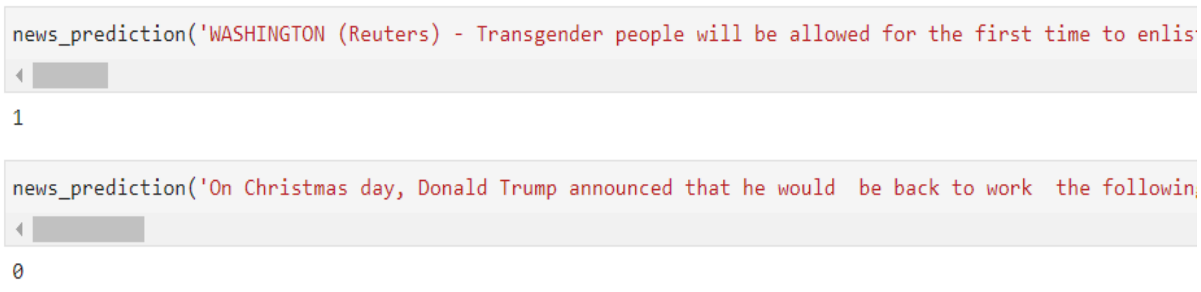
13. Inference

Decision Tree model is selected as the final model for the analysis based on the accuracy scores. Multiple random data points were tested against the predicted value for evaluating the model. X is the input news article and Y is the correct news category. (0 for real and 1 for fake). The Y value is compared with the model prediction and the values are matching.



```
news_prediction(x[100])
0
y[100]
0
news_prediction(x[10005])
0
y[10005]
0
news_prediction(x[528])
0
y[528]
0
news_prediction(x[22578])
1
y[22578]
1
```

Also tested the model using a live news article and the model predicted whether the news is real or fake.



```
news_prediction('WASHINGTON (Reuters) - Transgender people will be allowed for the first time to enlis
1
news_prediction('On Christmas day, Donald Trump announced that he would  be back to work  the followin
0
```

14. Appendix

The project code in python

Fake_News_Detection.
html