**Fake News Detection**

**Abstract**

In our modern era where internet is ubiquitous, everyone relies on various online resources for news. Along with the increase in use of social me-dia platforms like Facebook, Twitter etc. news spread rapidly among millions of users within a very short span of time. The spread of fake news has far reaching consequences like creation of bi-asked opinions to swaying election outcomes for the benefit of certain candidates. Moreover, spammers use appealing news headlines to generate revenue using advertisements via click-baits. In this project, we aim to perform a binary classification of various news articles available online with the help of concepts pertaining to Artificial Intelligence, Natural Language Processing and Machine Learning.

**Introduction**

With the growing popularity of mobile technology and social media, information is accessible at one’s fingertips. Mobile applications and social media platforms have over-thrown traditional print media in the dissemination of news and information. It is only natural that with the convenience and speed that digital media offers, people express prefer-once towards using it for their daily information needs. Not only has it empowered consumers with faster access to di-verse data, it has also provided profit seeking parties with a strong platform to capture a wider audience.

With the outburst of information, it is seemingly tedious for a layman to distinguish whether the news he consumes is real or fake. Fake news is typically published with an intent to mislead or create bias to acquire political or financial gains. Hence it may tend to have luring headlines or interesting content to increase viewership.

In the recent elections of United States, there has been much debate regarding the authenticity of various news reports favoring certain candidates and the political motives behind them. Amidst such growing concerns, the detection of fake news gains utmost importance to prevent its negative im-pacts on individuals and society.

**Data Collection**

The dataset for this project was built with a mix of both real and fake news. Most of the data was manually crawled and extracted, whereas some were used off the shelf. The entire dataset amounted to 44,898 news articles out of which

23,481 were fake news and 21,417were real news.

The data comes from Kaggle, you can download it here:

[**https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset**](https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset)

**Real News**

The News Aggregator Dataset from the UCI Machine Learning Repository was used to extract real news. This dataset consists of links to the originally published news articles in their websites. We extracted these URLS and crawled them to download the news content using Beauti-fulSoup.

We extracted the body content of the articles by removing unnecessary information such as headers, footers, images, advertisements, tables etc. Further, we extracted the text from div tags having content and performed preprocessing steps on them before saving them into CSV files.

**Fake News**

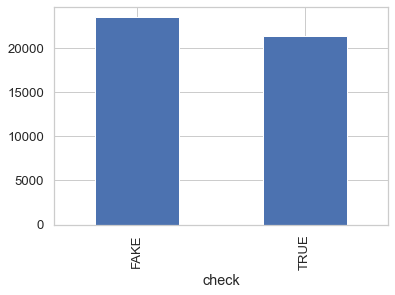
For fake news we used Kaggle’s ‘Getting Real about Fake News’ dataset. The CSV file with data was available off the shelf for use, and we had to perform minimal text processing on this data.

**In the class distribution we want to know:**

How many news are fake and how many are true?

We have 23481 fake news and 21417 true news.

We have a balanced mix of true and fake articles.



**Methodology**

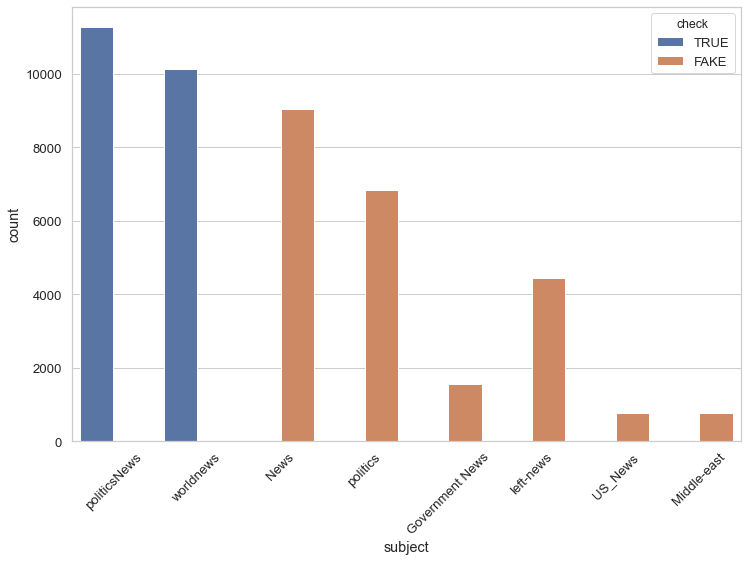
The approach proposed for this project is:

* Data Preprocessing
* Generating News Feature Vector
* Classification

We have politics News, world news as True and the rest of the news which include News, politics, Government News, left-News, US\_News,

Middle-east are Fake.

A picture containing text

Description automatically generated 

**Detecting Fake News With Natural Language**

**Processing ( NLP ):**

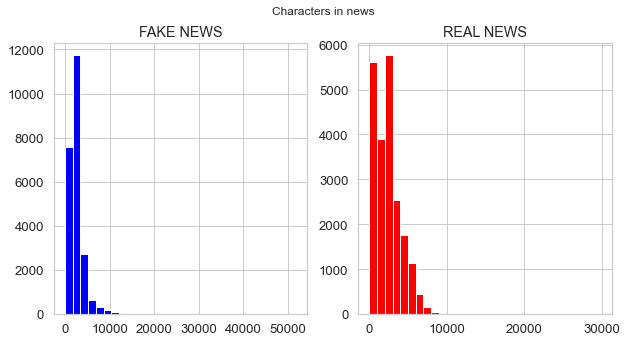
As human being, when we read a sentence or A paragraph, we can interpret the words with the whole documents and understand the context.

It is possible to teach to a computer how to read and understand the difference between real news and the fake news using Natural Language Processing (NLP).

**Exploratory Data Analysis:**

* The purpose of EDA is to enhance our understanding of trends in the dataset without involving complicated machine learning models.
* The last stage of my exploratory data analysis of the text is Word cloud analysis. Word cloud is a great way to represent text data. The size and color of each word that appears in the Word cloud indicate its frequency or importance.
* In the results, we can see how often some words used in the news text in fake or real news.

**Number of characters in news:**

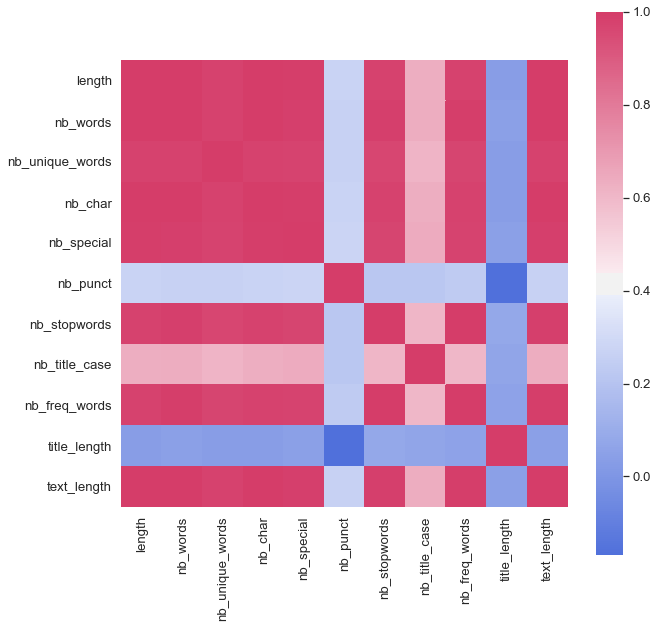
****

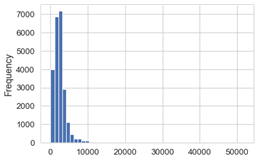
# Number of words in news:

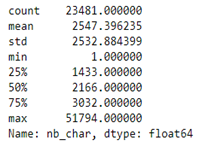
# 

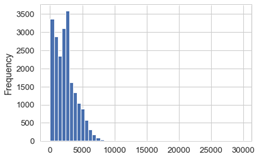
# feature engineering:

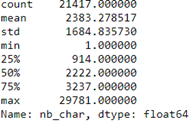
# We can see in heatmap some features have correlated together. For example, nb\_words, nb\_unique\_words, nb\_char have positively correlated with text length and nb\_punct, title\_length have negatively correlated with text length.



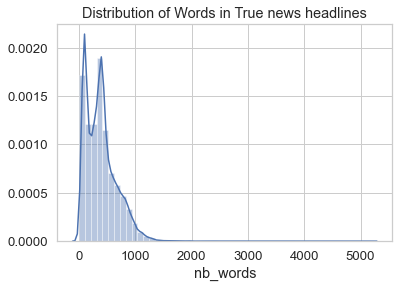
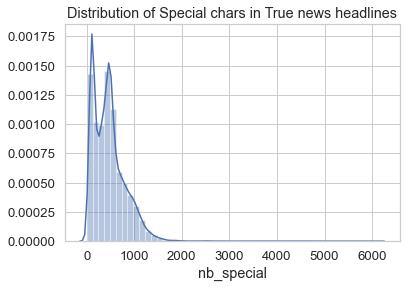
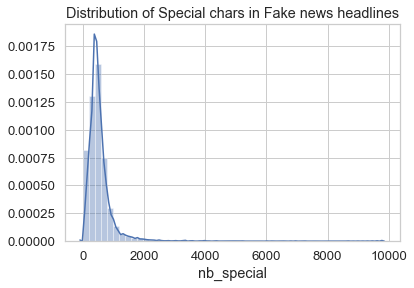
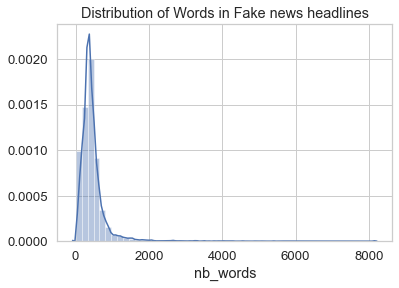








As we can see, from above analysis average number of characters in a sentence in real news is around 2383 while in case of fake news it is around 2547 which is but obvious because fake news generally uses superfluous language with more characters to grab the attention

**  
 **

# Observation:

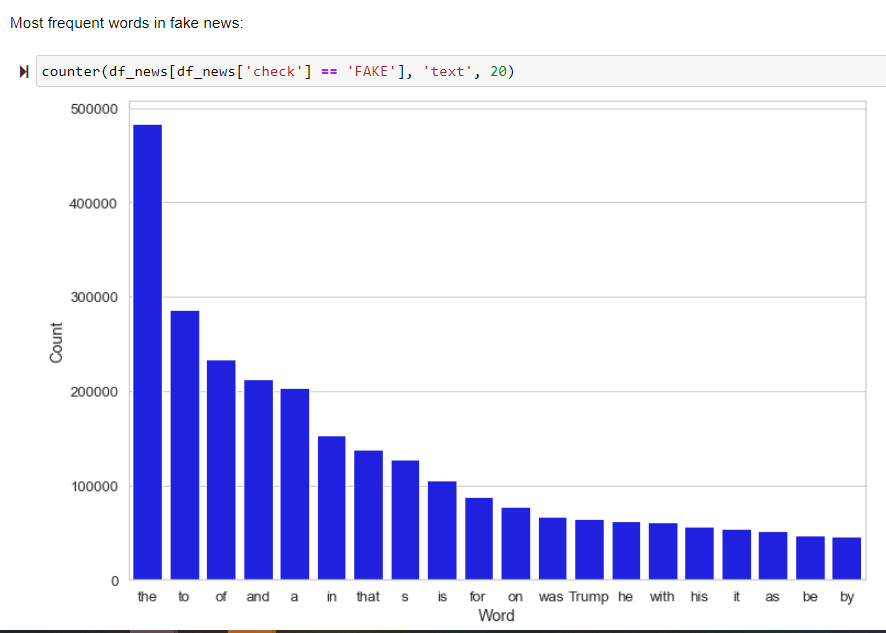
As we can see there are more special characters in fake news than real news because real news is generally to the point no superfluous words or less use of special characters

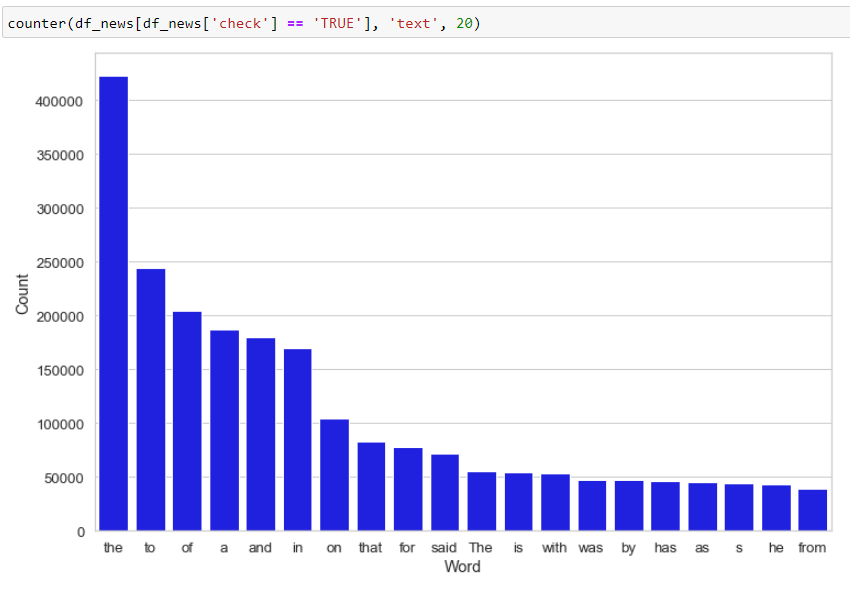
# Visualizing the data with Word cloud:

# The last stage of my exploratory data analysis of the text is Word cloud analysis. Word cloud is a great way to represent text data. The size and color of each word that appears in the Word cloud indicate its frequency or importance.

# 

# 





**Text Preprocessing**

Since most of the data was crawled and extracted manually, we had to first go through the data to understand organization and formatting of text. The data was made uniform and comparable by converting it into a uniform UTF-8 encoding. There were some cases where we encountered weird symbols and letters incompatible with the character set which had to be removed. We noticed that the data from news articles were often organized into paragraphs. So, we performed trimming to get rid of extra spaces and empty lines in text.

**Term Frequency - Inverse Document Frequency**

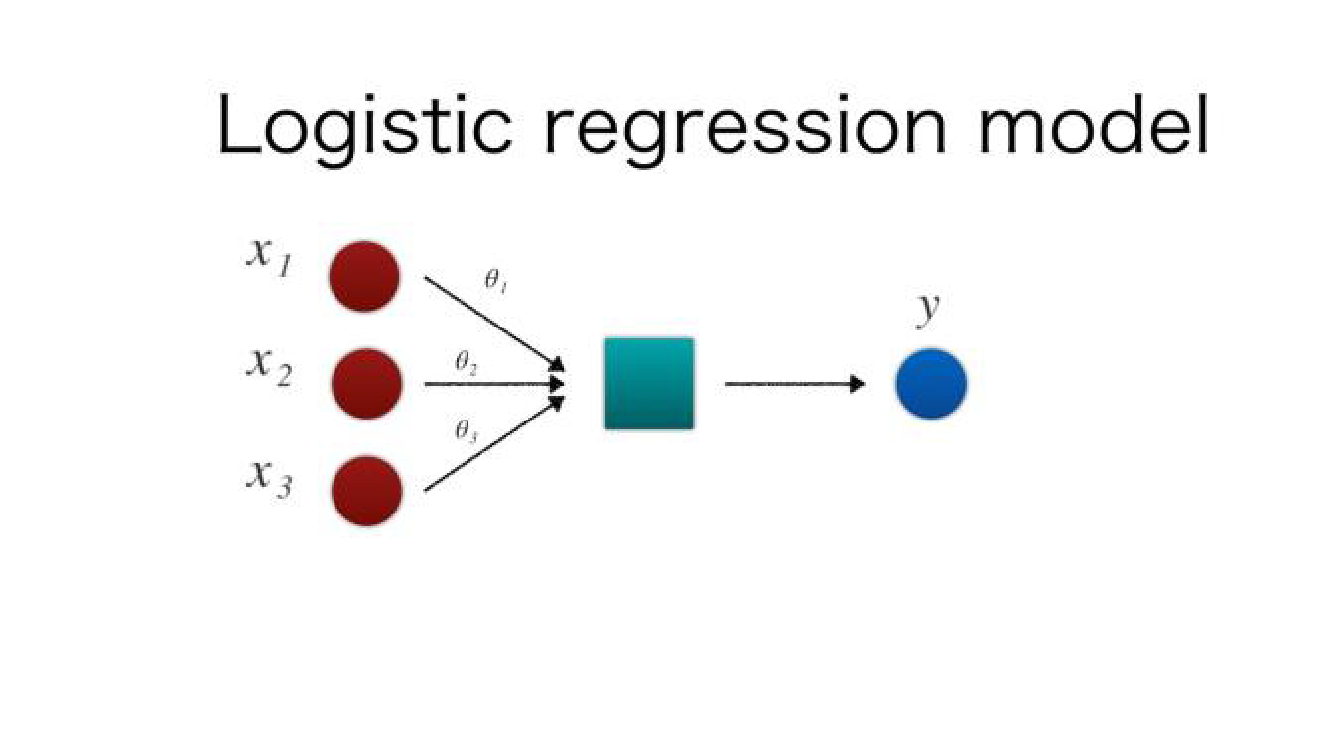
The tf-idf is a statistical measure that reflects the importance of a particular word with respect to a document in a corpus. It is often used in information retrieval and text mining as one of the components for scoring documents and performing searches. It is a weighted measure of how often a word occurs in a document relative to how often it occurs across all documents in the corpus. Term frequency is the number of times a term occurs in a document. Inverse document frequency is the inverse function of the number of documents in which it occurs.

**Prediction Algorithms**

We implemented five different algorithms from scratch for the prediction model which were: Logistic Regression model and the Naïve Bayes classifier model, PassiveAgressiveClassifier model, Support Vector Machine model, Random Forest.

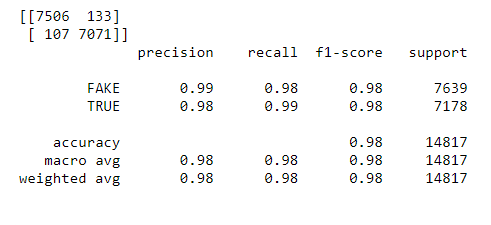
**Logistic Regression**

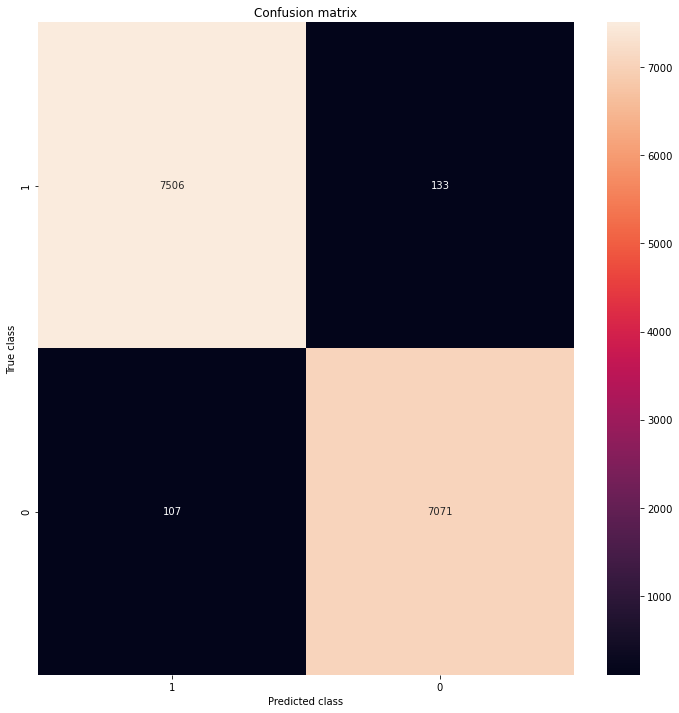
Logistic Regression is a Machine Learning technique used to estimate relationships among variables using statistical methods. This algorithm is great for binary classification problems as it deals with predicting probabilities of classes, and hence our decision to choose this algorithm as our base-line run. It relies on fitting the probability of true scenarios to the proportion of actual true scenarios observed. Also, this algorithm does not require large sample sizes to start giving fairly good results.

****

The Logistic Regression algorithm works by assigning observations to a discrete set of classes and then transforms it using a sigmoid function to give the probability value which can be mapped to a discrete class.

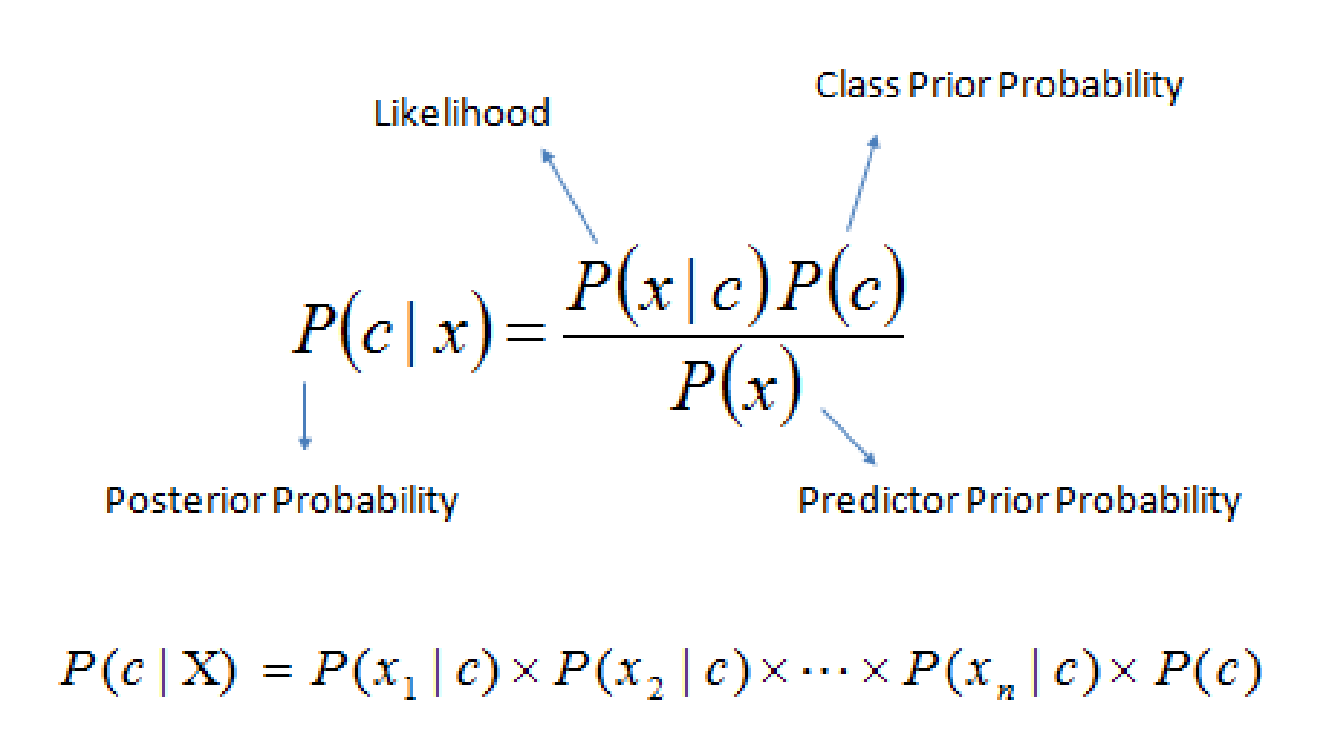
**Logistic Regression Classification Report:**





**Naïve Bayes Classifier**

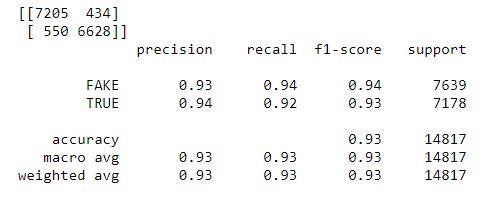
This is a simple yet powerful classification model that works remarkably well. It uses probabilities of the elements belonging to each class to form a prediction. The underlying assumption in the Naïve Bayes model is that the probability-ties of an attribute belonging to a class is independent of the other attributes of that class. Hence the name ‘Naive’.

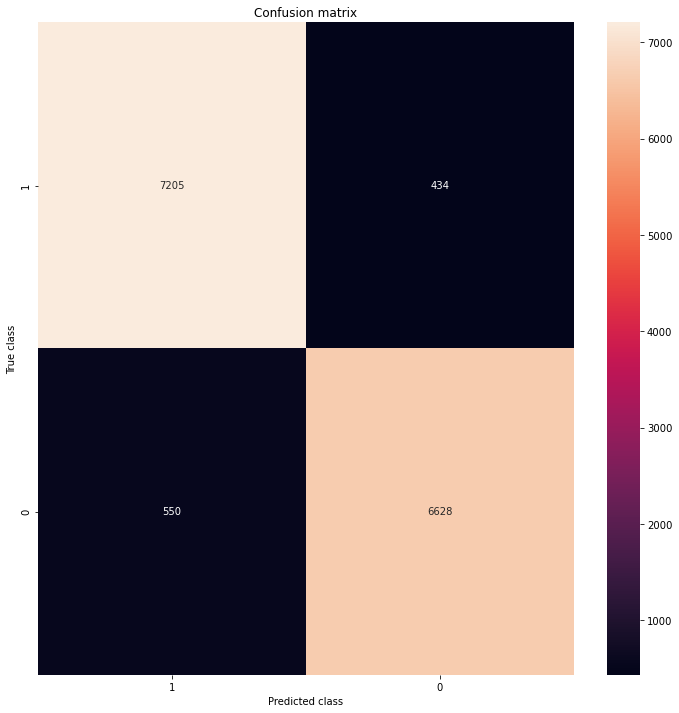
****

In this model we multiply the conditional probabilities of each attribute given the class value, to get the probability of the test data belonging to that class. We arrive at the final prediction by selecting the class that has the highest of the probabilities for the instance belonging to that class.

The advantages of using Naïve Bayes is that it is simple to compute, and it works well in categorizing data as we are using ratios for computation.

**Naïve Bayes Classifier classification report:**

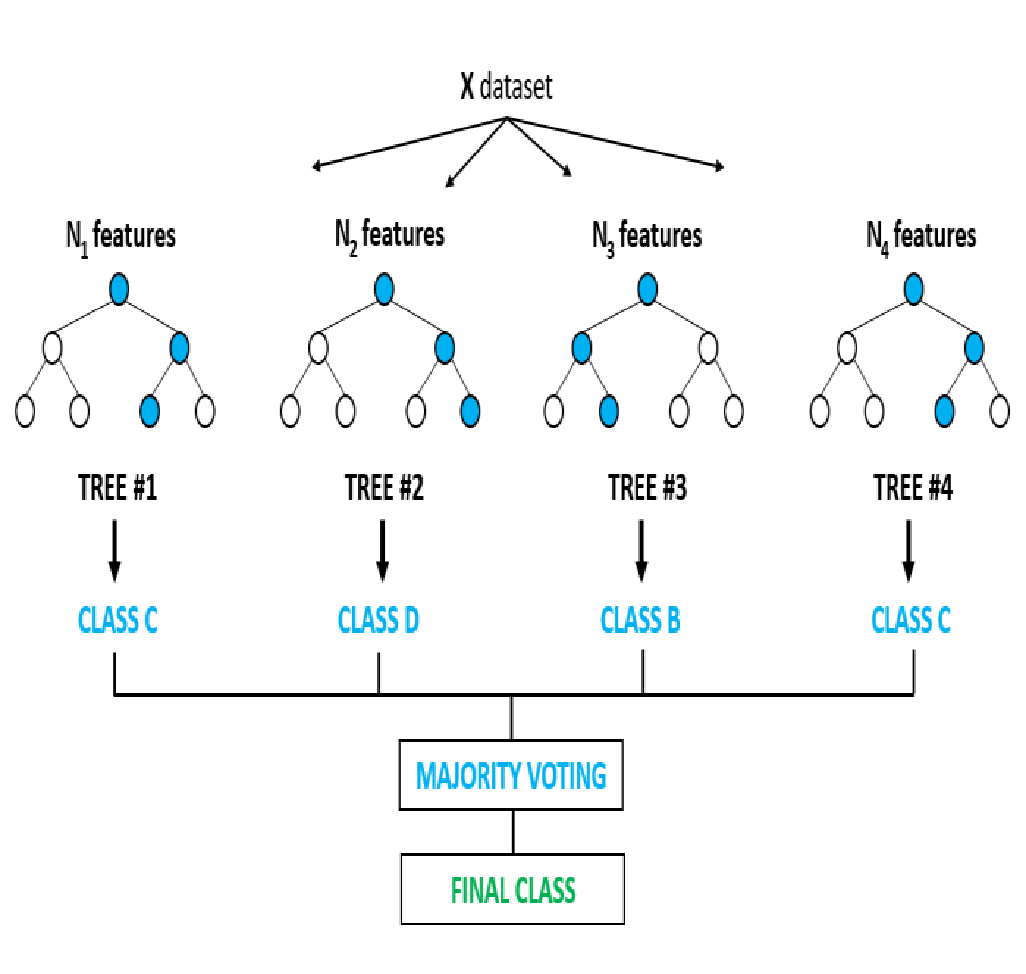




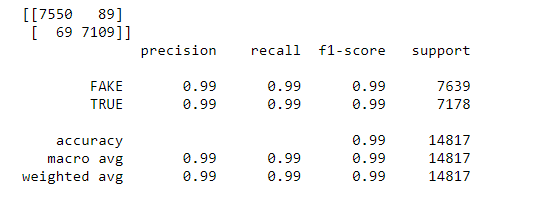
**Random Forest Classifier**

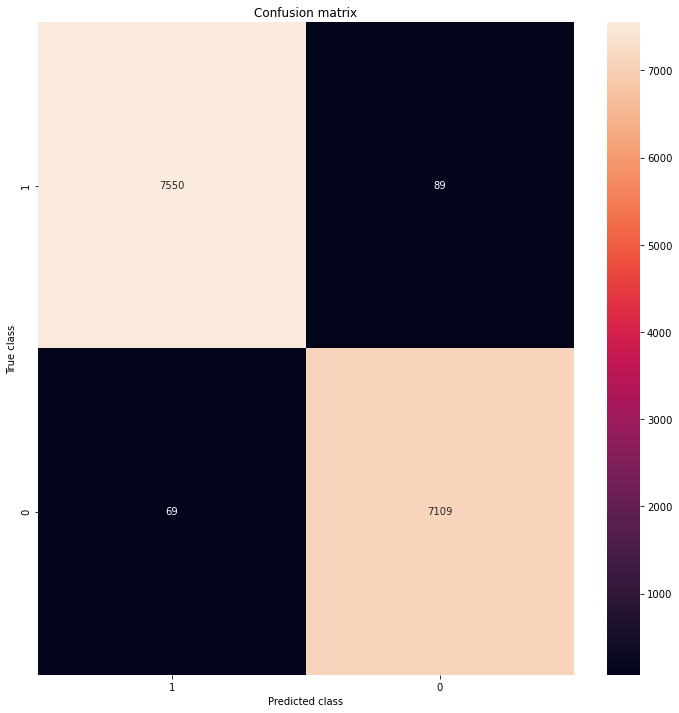
Random Forests are a machine learning method of classification that work by building several decision trees while training the model. It is a kind of additive model that makes predictions from a combination of decisions from base models. Decision trees have huge depth and tend to overfit results. Random forest utilizes multiple decision trees to average out the results.

The Random forest classifier creates a set of decision trees from a subset of the training data. It aggregates the results from different decision trees and then decides the final classification of the test data. The subsets of data used in the decision trees may overlap.



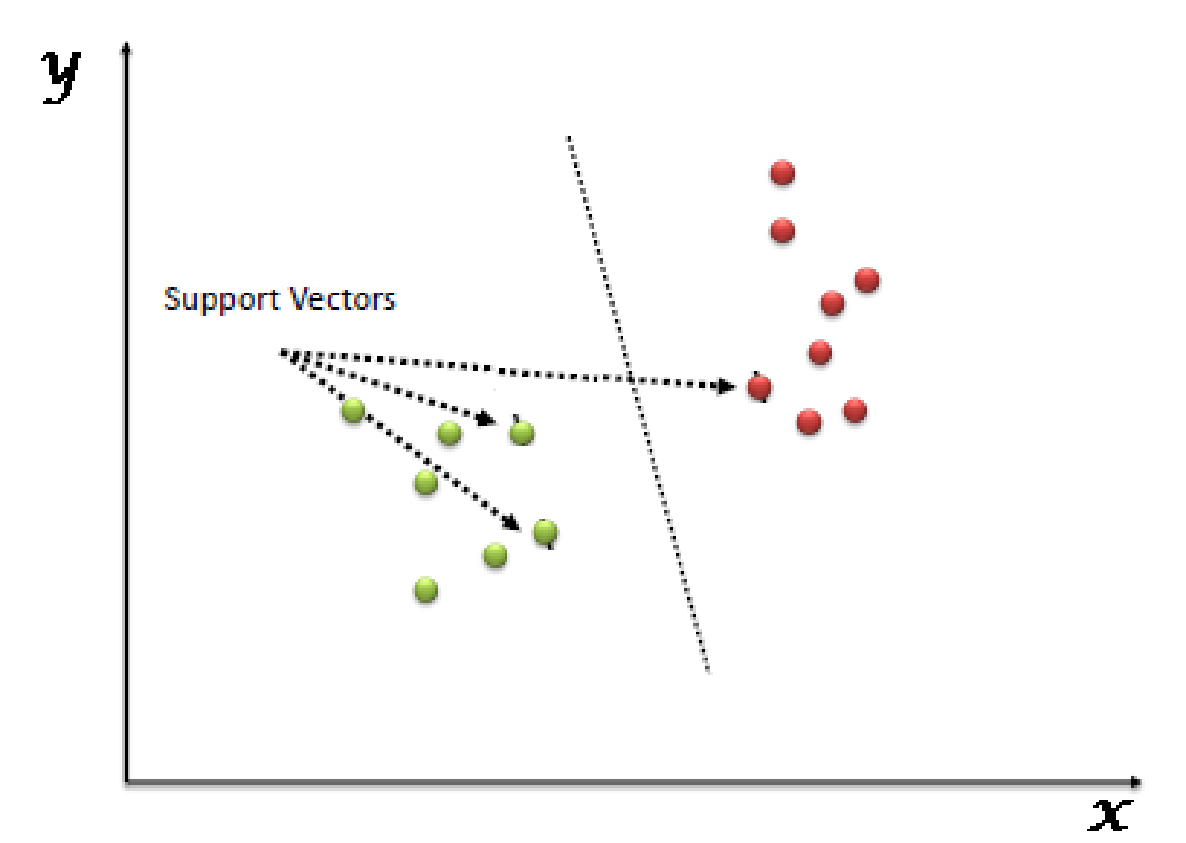
**Random Forest Classification Report:**



****

**Support Vector Machine**

Support Vector Machines are machine learning models that perform supervised learning on data for classification and regression. When given a labeled training dataset, it computes the optimal hyperplane that categorizes the test data.

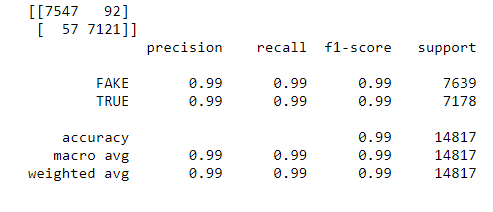
****

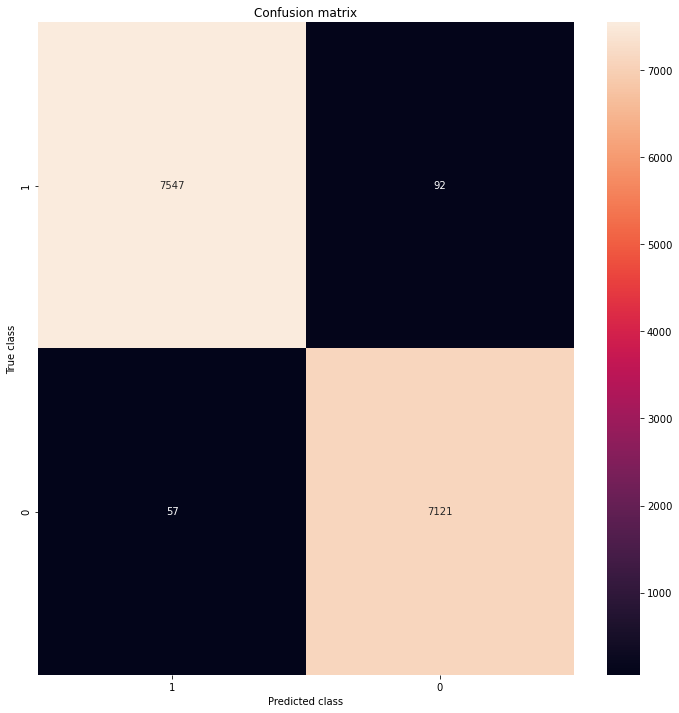
Data points are plotted in a multidimensional space, where the dimension is determined by the number of features at our disposal. The value of each feature is mapped to a point in the coordinate system. The algorithm then performs classification by finding the hyperplane that differentiates the two classes well. The hyperplane having the maximum mar-gin between the two classes in chosen.

The advantages of the SVM model are that it performs very well for high dimensional spaces and also creates a clear margin of separation between data points. The disadvantages of using SVM were that it takes greater time to train the model compared to other models, especially when the dataset is large.

**Support Vector Machine**

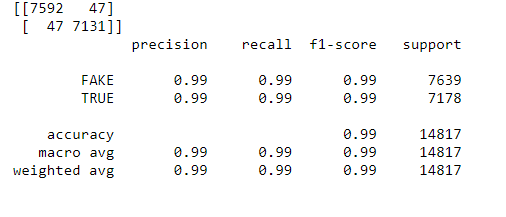
**Classification Report:**

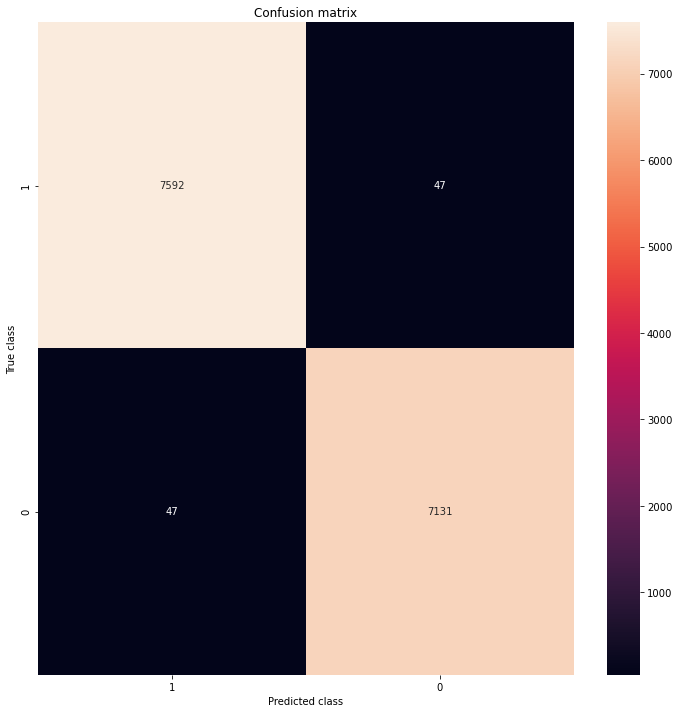




**PassiveAgressiveClassifier**

The passive-aggressive algorithms are a family of algorithms for large-scale learning. Intuitively, passive signifies that if the classification is correct, we should keep the model, and, aggressive signifies that if the classification is incorrect, update the model to adjust to more misclassified examples. Unlike most others, it does not converge, rather it makes updates to correct the loss.



****

**Evaluation Metrics**

We used the following three metrics for the evaluation of our results. The use of more than one matrix helped us evaluate the performance of the models from different perspectives.

**Confusion Matrix**

This is a great visual way to depict the predictions as four categories:

1. False Positive: Predicted as fake news but are actually true news.

2. False Negative: Predicted as true news but are actually fake news.

3. True Positive: Predicted as fake news and are actually fake news.

4. True Negative: Predicted as true news and are actually true news.

**Precision and Recall**

Precision which is also known as the positive predictive value is the ratio of relevant instances to the retrieved in-stances.

Precision = No. of True Positives / (No. of True Positives + No. of False Positives)

Recall which is also known as sensitivity is the proportion of relevant instances retrieved among the total number of relevant instances.

Recall = No. of True Positives / (No. of True Positives + No. of False Negatives)

**Feature Selection and Extraction**

Feature selection was the major part of our text-based classification problem, we used tf-idf vectorization of the news data we collected. But there was a challenge as the dimensions of the vector was quite high which caused models like SVM and Logistic Regression to run for a very long time on large datasets. To resolve the issue, we used some text-based transformation techniques such as stopping. We passed a list of stop words generated using NLTK library, as a parameter to the sklearn TFIDF vectorization class. We also defined the max feature parameter to be assigned to 50000, i.e. the TFIDF class generates a vocabulary and considers only the top max features ordered by term frequency across the corpus. This was only possible after stop word removal as this could have caused an issue as stop words are the most frequent words in the document.

**Classification Accuracy**

This depicts the number of accurate predictions made out of the total number of predictions made.

Classification accuracy is calculated by dividing the total number of correct results by the total number of test data records and multiplying by 100 to get the percentage

**Observed Results:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **Accuracy** |
| **Logistic Regression** | **0.99** | **0.98** | **0.98** |
| **PassiveAgressive** | **0.99** | **0.99** | **0.99** |
| **Naïve-Bayes** | **0.93** | **0.94** | **0.93** |
| **Support Vector Machine** | **0.99** | **0.99** | **0.99** |
| **Random Forest** | **0.99** | **0.99** | **0.99** |
|  |  |  |  |

**Conclusion**

We successfully implemented a machine learning and

Natural Language Processing model to detect whether

An article was fake or real.

We got 7592 articles correctly identified as Fake and 7131

Correctly identified as real. When doing such a classification,

it is important to check that we limit the number of false positives

as they can cause real to be marked as fake.

I would like to choose PassiveAggressiveClassifier method, because this

method has less false positive and false negative.

So overall PassiveAggressiveClassifier Method performed much better in

determining in fake news cases which is around 99%.

**Future Work:**

* Our research focuses on daily news articles which have on average around

1000 words. It is difficult to detect linguistic cues in single (or few) statement

news. Some other methods can be researched upon for these cases.

* For designing a fake news detector for social media like Facebook or twitter,

we can take into account the user information, user authenticity and origin

of the news.