NATURAL LANGUAGE PROCESSING PROJECT

Fake News Classifier

By: Swagata Chakraborty

July 2021

1. Introduction:

The spread of fake news on the Internet is a cause of great concern for all members of the society including the government, business, organizations and citizens. We often come across many hoaxes in the social media which negatively influences people. Such news spreads wrong message in the society and creates various ambiguity, misleading and chaos. Fake news is used to plant a seed of mistrust among citizens and has a negative impact on individuals. Thus, the aims of this project is to classify news into two distinct categories: real and fake. We have used three different models and check the accuracy of each.

2. Data:

2.1 Data and its source:

We got the train and test dataset from Kaggle. The train dataframe consists of four columns: the news headline(title), author of the news(author), the news body(text) and the label where 1 implies real and 0 implies fake.

2.2 Data Modification:

We separated the independent features from the dependent features and stored them in separate dataframes. Here independent features are: title, author and text. The dependent feature is the label.

3. Text cleaning and pre-processing:

- We dropped the columns having null values and reset the index of the dataframe.
- We made our analysis on the news headline only.
- We will used **stemming** which is a process of reducing inferred words to their word stem. For example, history and historical has the stem word histori, finally, final and finalize has the stem word fina, going, go and gone has the stem word go. Stemming sometimes gives us stem words having no meaning but it works faster than lemmatization which always

gives a meaningful stem word. Stemming is highly used with sentiment classifiers and spam classifiers.

- We traversed through every headline and removed non-alphabetic characters with space.
- We converted every headline into lower case and split every sentence into a list of words.
- Next, we remove all the unnecessary **stopwords** like I, am, the, is, that etc. These stopwords do not play a significant role in classifying a message to be spam or not. These set of stopwords are present in nltk.corpus library.
- Lastly, we converted remaining words into their respective stem words using PorterStemmer library present in nltk.stem.

4. Count Vectorizer:

This model is also known as bag of words. It is used to convert alphabetic sentences into numeric vectors so that we can feed that to the machine learning model who doesn't understand nominal categorical variables. In a binary bag of words model, presence of words is represented by 0 or 1. However, in bag of words one word is represented by its frequency.
For example, if we have 3 sentences:

S1 => He is a good boy.
S2 => She is a good girl.
S3 => Boy and girl are good.

After text cleaning and pre-processing we will have,

S1 => good boy
S2 => good girl
S3 => boy girl good

Now, if we apply binary bag of words,

SENTENCE	GOOD	BOY	GIRL
S1	1	1	0
S2	1	0	1
S3	1	1	1

But the disadvantage of this model is that it represents different words with the same number which implies same importance. This disadvantage can be overcome by the TFIDF Vectorizer if we want to give different importance to different words.

5. TFIDF Vectorizer:

Term Frequency (TF) – Inverse Document Frequency (IDF) model is used to convert words into numeric vectors and it represents every word by different value of numeric vectors.

TF = Frequency of a word in sentence / Total words in the sentence
IDF = log (No. of sentence / No. of sentences containing that word)

The previous example can be written as,

WORDS	FREQUENCY
good	3
boy	2
girl	2

Now we calculate TF as,

WORDS	S1	S2	S3
good	½	½	1/3
boy	½	0	1/3
girl	0	½	1/3

Now we calculate IDF as,

WORDS	IDF
good	log(3/3) =0
boy	log(3/2)
girl	log(3/2)

The vectors are formed by TF*IDF

SENTENCE	GOOD	BOY	GIRL
S1	0	½ log(3/2)	0

S2	0	0	½ log(3/2)
S3	0	1/3 log(3/2)	1/3 log(3/2)

In this project, we implement both Count Vectorizer and TFIDF Vectorizer and make two different models to compare the which model gives a better accuracy.

6. Multinomial Naïve Bayes Algorithm:

Naïve Bayes Classifier Algorithm is a supervised machine learning algorithm which works on the basic concepts of conditional probability and Bayes' theorem.

Suppose we have a dataset of patient details and we have to predict whether the person is diabetic or not. Let A = event that a person is diabetic, B = event that a person is not diabetic. Let X1, X2, X3… Xn be the health details of the patient X like age, gender, blood pressure, height, weight etc. Probability that patient X is diabetic or not given his health details is calculated as follows,

$$\mathbf{P\ (A|X1*X2*X3\ldots Xn) = [P(X1|A)*P(X2|A)*P(X3|A)\ldots*P(Xn|A)*P(A)]\ /\ P(X1)*\ P(X2)*..\ P(Xn)}$$

$$\mathbf{P\ (B|X1*X2*X3\ldots Xn) = [P(X1|B)*P(X2|B)*P(X3|B)\ldots*P(Xn|B)*P(B)]\ /\ P(X1)*\ P(X2)*..\ P(Xn)}$$

The maximum result becomes the output for the person. Multinomial Naïve Bayes uses term frequency i.e., the number of times a given term appears in a document.

This algorithm is specially used in solving text classification problems where suppose we want to classify the text as having positive sentiment or negative sentiment. For example, if we have some sentences:

S1 => The food is delicious.
S2 => Food is bad.
S3 => The food is good.
S4 => …
…

After text classification and pre-processing we have,

S1 => food delicious
S2 => food bad
S3 => food good
S4 => …

…

Using any model like BOW or TFIDF we convert the sentences (train data) into vectors,

SENTENCE	FOOD	DELICIOUS	BAD	GOOD	SENTIMENT
S1	1	1	0	0	1
S2	1	0	1	0	0
S3	1	0	0	1	1
S4	1	1	0	0	1
S5	0	0	1	0	0

Here sentiment is the dependent feature. Suppose our test sentence is "food is delicious" To find the sentiment of this sentence given the words,

P(Sentiment=1|Sentence)

- ⇨ P(1|x1,x2,x3….xn) where x = different words in the sentence
- ⇨ [P(x1|1)*P(x2|1)*P(x3|1)…..*P(xn|1)*P(1)] / [P(x1)*P(x2)*P(x3)…..*P(xn)]
- ⇨ P(x1|1)*P(x2|1)*P(x3|1)…..*P(xn|1)*P(1) (directly proportional)
- ⇨ (3/4)*(2/2)*(3/5)
- ⇨ 0.45

P(Sentiment=0|Sentence)

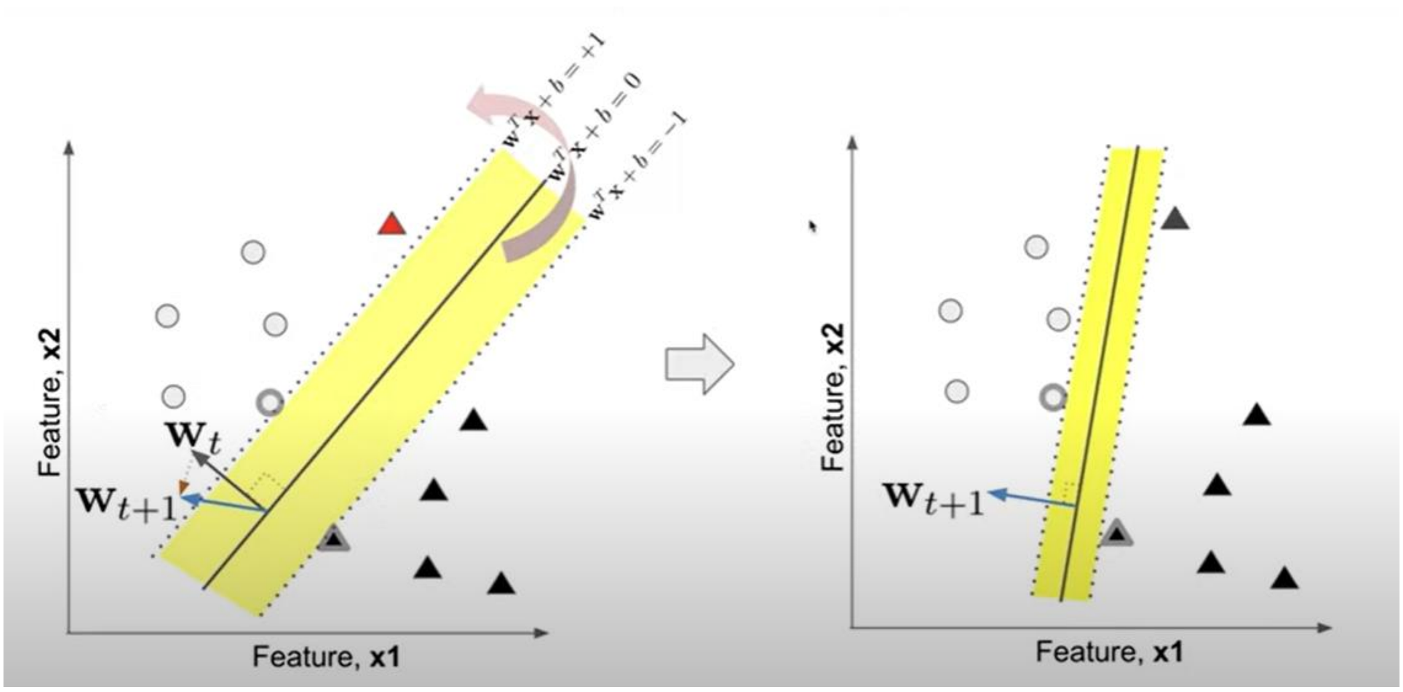
- ⇨ P(0|x1,x2,x3….xn) where x = different words in the sentence
- ⇨ [P(x1|0)*P(x2|0)*P(x3|0)…..*P(xn|0)*P(0)] / [P(x1)*P(x2)*P(x3)…..*P(xn)]
- ⇨ P(x1|0)*P(x2|0)*P(x3|0)…..*P(xn|0)*P(0) (directly proportional)
- ⇨ (1/4)*(0/2)*(2/5)
- ⇨ 0

Since P(Sentiment=1|Sentence) > P(Sentiment=0|Sentence), we can conclude that the sentence is having positive sentiment. This is how Naïve Bayes' classifies text data. However, this algorithm fails when it encounters a new word. If the sentence is "The food is delicious and tasty" it will be classified as negative sentiment although it has a positive sentiment.

7. Passive Aggressive Classifier Algorithm:

This algorithm works extremely well when we are working with huge live datasets from social media websites for large scale learning. It is called so because, if the prediction is correct, the model makes no changes i.e., passive but aggressive because if the prediction is incorrect, it makes changes to the model.

It is a margin-based prediction algorithm like Support Vector Machines. We have a hyperplane and margins on both sides which linearly separates the features. The margin passes through at least one support vector on each side. We choose the hyperplane which has the maximum margin distance. Now when we have a new test point in the features, and if it is correctly classified then the previous hyperplane remains. But if the point is misclassified, then the hyperplane and margins change its position in such a way that the point falls in the correct plane.



8. Model Development:

In the first model:
Techniques used: Stemming, Count Vectorizer, Multinomial Naïve Bayes
Accuracy: 0.902
Techniques used: Stemming, Count Vectorizer, Passive Aggressive Classifier
Accuracy: 0.918

In the second model:
Techniques used: Stemming, TFIDF, Multinomial Naïve Bayes
Accuracy: 0.881
Techniques used: Stemming, TFIDF, Passive Aggressive Classifier
Accuracy: 0.917

9. Conclusion:

Thus, we can conclude that, using passive aggressive classifier we get a better accuracy. Accuracy can be further increased if ‘text’ column of the dataset is used in stemming.