-Heena Rijhwani

FAKE NEWS DETECTOR

Fake news circulation is a hot button issue of the moment with the rising tide of a myriad of fake news reports and fabricated stories. Let us take an example of India. Whatsapp, India's leading messaging platform, is a vehicle for misinformation, particularly during the time of elections. In March 2019, Whatsapp groups were flooded with photographs claiming to show that unprecedented Indian air strikes in Pakistan had been successful. Islamabad claimed that there had been no casualties. According to BBC, the photos were old images that were being shared with false captions. One photo ruled back to a suicide attack in Pakistan in 2014 and another was traced back to an earthquake in Pakistan-administered Kashmir in 2005.
Whatsapp claimed to be deleting 2 million accounts per month as an attempt to cease the generation of fake news. Similar instances have also been encountered by Facebook and other social media applications.
BBC research found that emotional thinking and patriotism surpassed facts in many cases, leading to the generation of fabricated news.This fake news can often lead to violence and mob lynchings in the name of jingoism.

There are certain conditions that are necessary for the spread of fake news. These are:

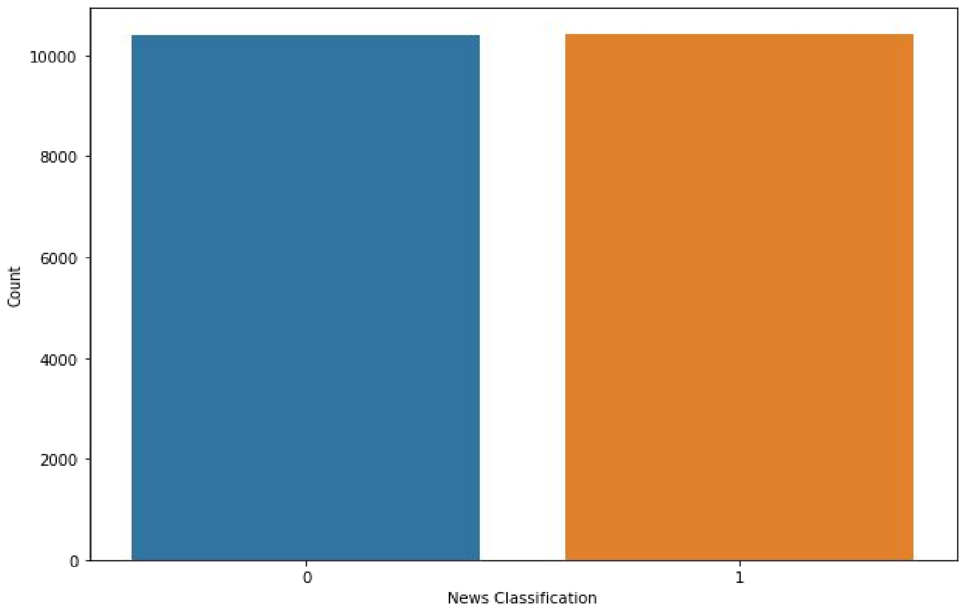
1. The blurring of lines between all types of news
2. Scepticism about the motivations of the news media
3. The flood of digital information and the shift to a high frequency news consumption world
4. The coping mechanisms for dealing with onrushing digital information. These are: selective consumption, preference for images, sender primacy, source agnosticism, nature of the forum, and 'feel' over 'think'
5. The broken link between consumption and sharing
6. The audiences sharing tactics for WhatsApp and Facebook.

Motivations behind sharing are rich and complex and need to be understood to establish why fake news is shared. These are:

1. Sharing to verify within the networks
2. Sharing as a civic duty
3. Sharing for nation building
4. Sharing as an expression of one's socio-political identity.

Here we are creating a model which predicts whether a given news is real or fake.It predicts the trustworthiness of news articles and automates the question of veracity. To start with, we import essential python libraries such as pandas and numpy, followed by importing our dataset (containing id,title,author of the article, text and label). Label 1 represents real news and 0 represents fake news. We explore the data and use seaborn library to plot the count of real and fake news.

```
plt.figure(figsize=(10,7))
sns.countplot(x='label', data=df)
plt.xlabel('News Classification')
plt.ylabel('Count')
```



We can conclude from the plot that the data is balanced.
The next crucial step is Data Cleaning. We check for null values. Few null values were found which were removed using dropna() function.

```
df.isna().any()
df.dropna(inplace=True)
```

Now, we clean the news before classifying it as fake or not. We start by removing special characters from the news-title, then converting it to lower case. Then we tokenize it (split into words) and remove stop words. Stop words are commonly used words in a language. Search engines ignore the stop words while indexing data as well as retrieving results for search queries.We also need to stem the words using PorterStemmer(). This gives the root form of the word. For instance: ‘clean’ for ‘cleaning’, ‘fish’ for ‘fishes’ and so on.

```
for i in range(0,news.shape[0]):
   title = re.sub(pattern='[^a-zA-Z]', repl=' ', string=news.title[i])
   title = title.lower()
   words = title.split()
   words = [word for word in words if word not in set(stopwords.words('englis
h'))]
   words = [ps.stem(word) for word in words]
   title = ' '.join(words)
   corpus.append(title)
```

Once the data is simplified by converting it to lower case,removing special characters, stemming and removing stop words, we build a bag of words model. This will represent text as a bag (multiset) of words and keeps a count of the words. It gives us 5000 features or most frequently used words with a range of 1 to 3.The count vectorizer tokenizes a collection of documents and builds a vocabulary of unique words. It can also encode new documents using this vocabulary.

```
cv = CountVectorizer(max_features=5000, ngram_range=(1,3))
X = cv.fit_transform(corpus).toarray()
```

Now, let us move into the classification models. First we need to split our data into training and testing data since it is a supervised learning model. We train the data using the train set and test it using the test set. 80% of the data has been used for training and the rest 20 % for testing.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
random_state=0)
```

NAIVE BAYES

Naive Bayes are mostly used in natural language processing. Naive Bayes classifier algorithm is a family of algorithms which use Bayes Theorem. It uses the naive assumption that all the features are independent of each other. Bayes theorem calculates the probability P(A|B) where A is the class of possible outcomes and B is the given instance which has to be classified.

$$P(A|B) = P(B|A) * P(A) / P(B)$$

According to our data, the class is 0 or 1, where 0 implies fake news and 1 implies true news. Given a news x, we will compute:

$$P(\text{true news}|x) = P(x|\text{true news}) * P(\text{true news}) / P(x)$$

$$P(\text{fake news}|x) = P(x|\text{false news}) * P(\text{false news}) / P(x)$$

If P(true news|x) > P(false news|x), the algorithm predicts it is a true news. Otherwise, the news will be predicted as fake.

We fit the model to the training set, predict test set results and calculate accuracy, precision and recall. Further, we tune hyperparameters for optimal results and *accuracy.*

```
from sklearn.naive_bayes import MultinomialNBnb_classifier = MultinomialNB()
nb_classifier.fit(X_train, y_train)
nb_y_pred = nb_classifier.predict(X_test)

from sklearn.metrics import accuracy_score, precision_score, recall_score
score1 = accuracy_score(y_test, nb_y_pred)
score2 = precision_score(y_test, nb_y_pred)
score3 = recall_score(y_test, nb_y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))

best_accuracy = 0.0
alpha_val = 0.0
for i in np.arange(0.1,1.1,0.1):
  temp_classifier = MultinomialNB(alpha=i)
  temp_classifier.fit(X_train, y_train)
  temp_y_pred = temp_classifier.predict(X_test)
  score = accuracy_score(y_test, temp_y_pred)
  print("Accuracy score for alpha={} is: {}%".format(round(i,1), round(score*10
0,2)))
  if score>best_accuracy:
    best_accuracy = score
    alpha_val = i
print('--------------------------------------------')
print('The best accuracy is {}% with alpha value as {}'.format(round(best_accur
acy*100, 2), round(alpha_val,1)))
```

We obtain an Accuracy score of 90.16%
Precision score of 0.87 and
Recall score of 0.91
The best accuracy is 90.59% with alpha value as 0.3.

LOGISTIC REGRESSION

Logistic regression is a statistical model in which the response variable takes a discrete value and the explanatory variables can either be continuous or discrete. If the outcome variable takes only two values, then the model is called binary logistic regression model.Here the outcomes can be real news (Y = 1) and false news (Y = 0).Then the probability that a record belongs to a positive class, P(Y = 1), using the binary logistic regression model is given by:

P(Y=1) = e^z/(1 + (e^z))

Logistic Function-The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment.It is an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

y = e^(b0 + b1*x) / (1 + e^(b0 + b1*x))

Here y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x).Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data.

Similarly, we fit the model to the training set, predict test set results and calculate accuracy, precision and recall as well as tune hyperparameters for optimal results and accuracy.

```
from sklearn.linear_model import LogisticRegression
lr_classifier = LogisticRegression(random_state=0)lr_classifier.fit(X_train, y_trai
n)
lr_y_pred = lr_classifier.predict(X_test)
from sklearn.metrics import accuracy_score, precision_score, recall_score
score1 = accuracy_score(y_test, lr_y_pred)
score2 = precision_score(y_test, lr_y_pred)
score3 = recall_score(y_test, lr_y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))

best_accuracy = 0.0
c_val = 0.0
for i in np.arange(0.1,1.1,0.1):
  temp_classifier = LogisticRegression(C=i, random_state=0)
  temp_classifier.fit(X_train, y_train)
  temp_y_pred = temp_classifier.predict(X_test)
  score = accuracy_score(y_test, temp_y_pred)
  print("Accuracy score for C={} is: {}%".format(round(i,1), round(score*100,
2)))
  if score>best_accuracy:
    best_accuracy = score
    c_val = I
print('-------------------------------------------')
print('The best accuracy is {}% with C value as {}'.format(round(best_accuracy
*100, 2), round(c_val,1)))
```

We obtain an Accuracy score of 93.52%

Precision score of 0.89 and
Recall score of 0.97
The best accuracy is 93.63% with C value as 0.8.

We use confusion matrix to evaluate the performance of the model.It compares predicted values and the actual values. We use 4 measures to evaluate the performance:

True positive: The cases in which the predicted values and the actual values are the same, and the value is positive.

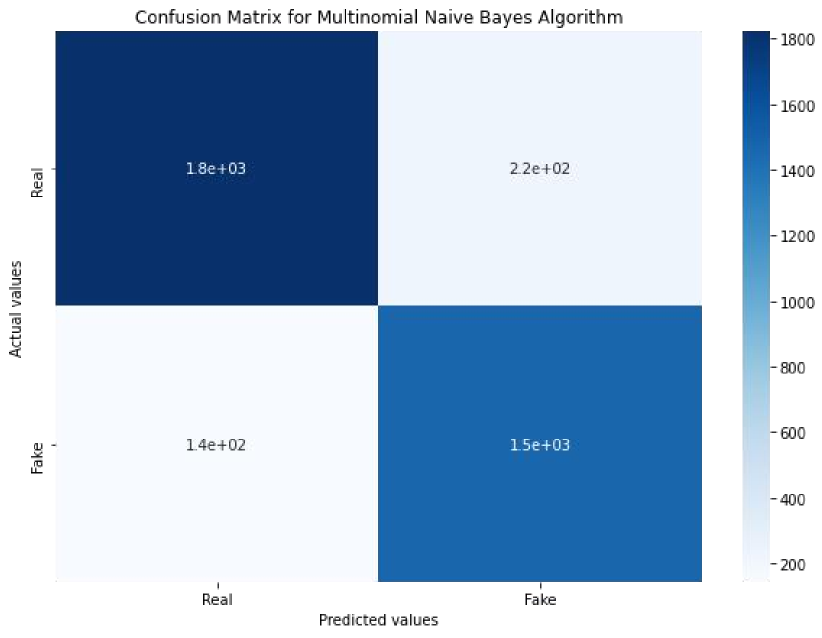
True Negative: The cases in which the predicted values and the actual values are the same, and the value is negative.

False Positive: The cases in which the prediction is ‘YES’ ,but the actual value is ‘NO’.

False Negative: The cases in which the prediction is ‘NO’, but the actual value is ‘YES’.

Confusion Matrix for Multinomial Naive Bayes:

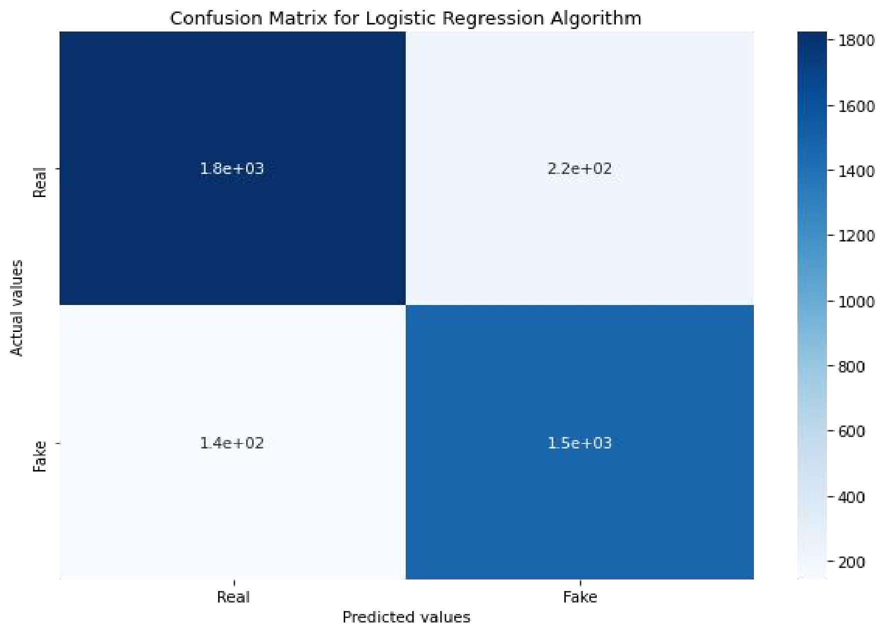
```
from sklearn.metrics import confusion_matrix
nb_cm = confusion_matrix(y_test, nb_y_pred)
plt.figure(figsize=(10,7))
sns.heatmap(data=nb_cm, annot=True, cmap="Blues", xticklabels=['Real', 'Fake'],
 yticklabels=['Real', 'Fake'])
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
plt.title('Confusion Matrix for Multinomial Naive Bayes Algorithm')
plt.show()
```



Confusion Matrix for LoR:

```
from sklearn.metrics import confusion_matrix

lr_cm = confusion_matrix(y_test, lr_y_pred)
plt.figure(figsize=(10,7))
sns.heatmap(data=nb_cm, annot=True, cmap="Blues", xticklabels=['Real', 'Fake
'], yticklabels=['Real', 'Fake'])
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
plt.title('Confusion Matrix for Logistic Regression Algorithm')
plt.show()
```



The last step now is to make predictions.

```
def fake_news(sample_news):
   sample_news = re.sub(pattern='[^a-zA-Z]',repl=' ', string=sample_news)
   sample_news = sample_news.lower()
   sample_news_words = sample_news.split()
   sample_news_words = [word for word in sample_news_words if not word in
 set(stopwords.words('english'))]
   ps = PorterStemmer()
   final_news = [ps.stem(word) for word in sample_news_words]
   final_news = ' '.join(final_news)

   temp = cv.transform([final_news]).toarray()
   return classifier.predict(temp)

# For generating random integer
```

```
from random import randint

# Predicting values
row = randint(0,news_title.shape[0]-1)
sample_news = news_title[row]
print('News: {}'.format(sample_news))
if fake_news(sample_news):
  print('Prediction: This is a FAKE news!')
else:
  print('Prediction: This is a REAL news.')
```

OUTPUTS

```
News: City to Close or Merge 9 Schools That Were in Support Program - Th
e New York Times
Prediction: This is a REAL news.
News: Obama’s Brother: I’m Voting Trump
Prediction: This is a FAKE news!
News: Walter Hautzig, Pianist Whose Talent Helped Him Flee Nazis, Dies at
95 - The New York Times
Prediction: This is a REAL news.
News: Top Congressmen Just Demanded A Federal Investigation Into FBI’s Le
aks To Giuliani
Prediction: This is a FAKE news!
```

CONCLUSION

The paper presents the stages of creating a model for fake news detection. The pre-processing, training and prediction phases have been described and the performance of the model has been assessed using accuracy,precision,recall and confusion matrix. The model fulfills its task of detecting fake news.

However, this is merely a beginning of eradicating fake news. There are more advanced NLP techniques such as BERT,GloVe,ELMo that can be used. Other methods include using stylometry-based provenance, i.e., tracing a text’s writing style back to its producing source and determining whether that source is malicious.Techniques can also be used to evaluate sub-categories of fake news.

REFERENCES

https://www.geeksforgeeks.org/applying-multinomial-naive-bayes-to-nlp-problems/

https://en.wikipedia.org/wiki/Logistic_regression

https://www.forbes.com/sites/bernardmarr/2017/03/01/fake-news-how-big-data-and-ai-can-help/

https://www.datarobot.com/blog/data-science-fails-fake-news-fake-data/

https://www.theguardian.com/technology/2019/feb/06/whatsapp-deleting-two-million-accounts-per-month-to-stop-fake-news https://www.bbc.com/news/world-asia-india-47797151

http://downloads.bbc.co.uk/mediacentre/duty-identity-credibility.pdf