I. Introduction

1. Domain-specific area

Today, in the age of technology, people spend a lot of their times on social media including Facebook, WhatsApp, Instagram, and Telegram. Accessing information online has become more convenient and faster. Most of the social media users may believe what they have read online, and they can easily share fake news without realising it.

There are also people who create fake news for many purposes and intention, it can be the intention to make fake medical news goes viral, or for political motives such as influence public views on some political decisions. It can also be the intention to make money, or to cause damage to the reputation of a person or a company.

These can cause a serious impact on the society if fake news is not properly identified and detected. For example, if people believe in a post with false information about the effects of Covid-19 vaccines, then it can result in unnecessary public panic. If many fake news related to health spread rapidly on social media, people will have difficulty in searching for accurate, reliable, and trustworthy information online.

According to the “Channel News Asia (CAN)” news, it reported that the misinformation about Covid-19 vaccines being shared on social media has threatens Singapore’s Covid-19 vaccination programme.
https://www.channelnewsasia.com/commentary/covid-19-coronavirus-conspiracy-misinformation-fake-news-400276

A local study revealed that about 6 in 10 people has received and encountered fake news about Covid-19 on social media in Singapore.
https://www.channelnewsasia.com/singapore/fake-covid-19-news-study-ncid-messaging-platforms-whatsapp-673131

“The Straits Times” news reported that around 90 percent of the people in a survey of 750 participants has identified at least one out of five fake headlines wrongly which means they identified one of the fake headlines as real.
https://www.straitstimes.com/singapore/4-in-5-singaporeans-confident-in-spotting-fake-news-but-90-per-cent-wrong-when-put-to-the

To help address the issues mentioned above, the natural language processing (NLP) and machine learning techniques will be used to identify fake news and reduce the number of fake news on social media.

2. Objectives

The objective of this project is to create a text classifier that can differentiate fake news from real news, and the goal is to quickly help readers to identify misleading information and disinformation in any news article.

I planned to use machine learning and natural language processing methods. I also aiming to find the best classifier for fake news detection by explore different machine learning classifiers including the Naïve Bayes and logistic regression, and then compare their accuracy results. My goal is to find the model that achieve the highest accuracy.

Early and accurate detection of fake news can avoid it going viral so it can prevent misinformation to be received by majority

3. Dataset

The REAL and FAKE news dataset that I decided to use for my project is from the Kaggle official website. Here is the link to the dataset
https://www.kaggle.com/datasets/nopdev/real-and-fake-news-dataset

This is a publicly available dataset of ‘REAL and FAKE news ‘. Kaggle provide many free datasets, it is easily accessible and available for public download.

There are two csv files in the dataset, namely the ‘Fake.csv’ file and the ‘True.csv’ file. The ‘Fake.csv’ file contains a list of news articles considered as fake news and the ‘True.csv’ file contains a list of news articles considered as real news.

In [1]:
```
# load data
import pandas as pd
import numpy as np

df = pd.read_csv('news.csv')
df.head()
```

Out[1]:

	Unnamed: 0	title	text	label
0	8476	You Can Smell Hillary’s Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE

4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

In [2]:
```
df.shape
```

Out[2]:
```
(6335, 4)
```

There are a total of 6335 news articles and four columns. The first column is just a list of numbers that representing each row. The second column and the third column are the title and text. The last column has labels denoting whether the news is REAL or FAKE.

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6335 entries, 0 to 6334
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  6335 non-null   int64 
 1   title       6335 non-null   object
 2   text        6335 non-null   object
 3   label       6335 non-null   object
dtypes: int64(1), object(3)
memory usage: 198.1+ KB
```

The datatype of the first column is Int64. The datatype of the 'title' column, 'text' column and 'label' column are object.

In [4]:
```
df.isna().sum()
```

Out[4]:
```
Unnamed: 0    0
title         0
text          0
label         0
dtype: int64
```

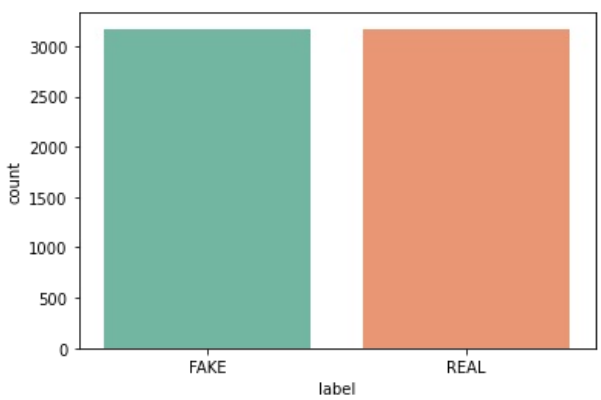
There are no missing values in the dataset.

In [5]:
```
df['label'].value_counts()
```

Out[5]:
```
REAL    3171
FAKE    3164
Name: label, dtype: int64
```

The total number of real news articles is: 3171
The total number of fake news articles is: 3164

In [6]:
```
import matplotlib.pyplot as plt
import seaborn as sns

sns.countplot(x ='label', data = df, palette = "Set2")
plt.show()
```



The dataset is considered balanced and there is no inconsistency in the data.

4. Evaluation methodology

I decided to use confusion matrix to visualize, describe and summarizes the performance of my proposed machine learning classification algorithm or 'fake and real' news classifier. Evaluation metrics including accuracy, precision and recall will also be used to evaluate the performance of my proposed 'fake and real' news classifier.

True Positive (TP): The truth is positive and is predicted positive.
False Negative (FN): The truth is positive but is predicted negative.
True Negative (TN): The truth is negative and is predicted negative.
False Positive (FP): The truth is negative but is predicted positive.

I'm going to use the accuracy metric to measure how often my proposed classifier predicts correctly on the test data. For example, I can use the accuracy metric to calculate the number of correctly predicted news as 'fake' divided by the total number of predictions. Accuracy= (TP+TN)/(TP+TN+FP+FN)

Next example, I can use the precision metric to calculate out of all the news that were predicted as 'fake', how many the classifier got it right. Let's say the classifier predicted 10 news as 'fake'. Then, out of the 10 news that were predicted as 'fake', it only got 5 right which means it only predicted 5 of them correctly, so the precision is equal to 5/ (5+5) = 0.5. Precision = TruePositives / (TruePositives + FalsePositives)

Third example, I can use the recall metric to calculate out of all the actual news labelled as 'fake', how many the classifier got it right on predicting it. Let's say the total number of news labelled as 'fake' actually is 20 (The total 'fake' news truth samples), the classifier only predicted 5 correctly, so 5/(5+15) = 0.25. Recall = TruePositives / (TruePositives + FalseNegatives)

II. Implementation

5. Preprocessing

In [7]:
```
#filetype - csv, load csv file

import pandas as pd
import numpy as np

df = pd.read_csv('news.csv')
print(df.shape) # get the shape of the dataset
df.isna().sum() # calculate the sum of missing values
```

Out[7]:
```
(6335, 4)
Unnamed: 0    0
title         0
text          0
label         0
dtype: int64
```

In [8]:
```
df.head()
```

Out[8]:

	Unnamed: 0	title	text	label
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

In [9]:
```
# getting all the fake news
# select all rows where the 'label'(column) value is equal to 'Fake'

fake = df[df['label'] == 'FAKE']
fake.head()
```

Out[9]:

	Unnamed: 0	title	text	label

0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
5	6903	Tehran, USA	\nI'm not an immigrant, but my grandparents ...	FAKE
6	7341	Girl Horrified At What She Watches Boyfriend D...	Share This Baylee Luciani (left), Screenshot o...	FAKE

In [10]:
```
# getting all the real news
# select all rows where the 'label'(column) value is equal to 'REAL'

real = df[df['label'] == 'REAL']
real.head()
```

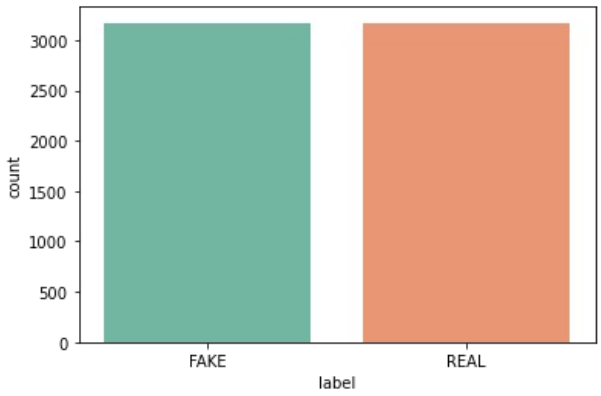
Out[10]:

	Unnamed: 0	title	text	label
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL
7	95	'Britain's Schindler' Dies at 106	A Czech stockbroker who saved more than 650 Je...	REAL
8	4869	Fact check: Trump and Clinton at the 'commande...	Hillary Clinton and Donald Trump made some ina...	REAL
9	2909	Iran reportedly makes new push for uranium con...	Iranian negotiators reportedly have made a las...	REAL

In [11]:
```
print('The total number of real news articles is : ', real.shape[0])
print('The total number of fake news articles is : ', fake.shape[0])
df['label'].value_counts()
```

Out[11]:
```
The total number of real news articles is :  3171
The total number of fake news articles is :  3164
REAL    3171
FAKE    3164
Name: label, dtype: int64
```

In [12]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# count the number of fake news articles and real news articles
sns.countplot(x ='label', data = df, palette = "Set2")
# show the plot, visualize data using matplotlib
plt.show()
```



In [13]:
```
# the first column has no meaningful information and is not useful, so drop the unecessary column


df.drop(['Unnamed: 0'], axis=1, inplace=True)
```

In [14]:
```
df.head()
```

Out[14]:

	title	text	label
0	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE

2	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

In [15]:

```
# combining the title and the text column

df['text'] = df['title'] + " " + df['text']
df.drop('title', axis=1, inplace=True)
df.head()
```

Out[15]:

	text	label
0	You Can Smell Hillary's Fear Daniel Greenfield...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	FAKE
2	Kerry to go to Paris in gesture of sympathy U....	REAL
3	Bernie supporters on Twitter erupt in anger ag...	FAKE
4	The Battle of New York: Why This Primary Matte...	REAL

In [16]:

```
# replace the 'label' column containing the values 'FAKE' and 'REAL' with 0 and  1:

new_df = df.replace({'label': {'FAKE': 0, 'REAL': 1}})
new_df.head()
```

Out[16]:

	text	label
0	You Can Smell Hillary's Fear Daniel Greenfield...	0
1	Watch The Exact Moment Paul Ryan Committed Pol...	0
2	Kerry to go to Paris in gesture of sympathy U....	1
3	Bernie supporters on Twitter erupt in anger ag...	0
4	The Battle of New York: Why This Primary Matte...	1

In [17]:

```
new_df.info() #  get more info about the data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6335 entries, 0 to 6334
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    6335 non-null   object
 1   label   6335 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 99.1+ KB
```

In [18]:

```
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

ps = PorterStemmer()

def process_text(text):
    # use regular expression to remove special characters and numbers
    clean_text = re.sub('[^0-9a-zA-Z]',' ', text)

    # convert all text to lowercase
    clean_text = clean_text.lower()

    # remove extra spaces
    clean_text = re.sub(r'\s+', ' ',clean_text )

    clean_text = clean_text.split()

    # remove the stopwords first, then choose the non-stopwords for stemming
    # stemming: removing a part of a word, reducing a word to its root
    # return the clean text
    clean_text = [ps.stem(word) for word in clean_text if not word in stopwords.words('english')]
    clean_text =' '.join(clean_text)
    return clean_text
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\unnie\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

I removed all the special characters, numbers and stopwords from the test because they do not add any semantic meaning to the test.
I choose to use stemming because lemmatization takes longer time to implement, stemming is a lot faster.

In [19]:
```
new_df['clean_text'] = new_df['text'].apply(process_text)
new_df.head()
```

Out[19]:

	text	label	clean_text
0	You Can Smell Hillary's Fear Daniel Greenfield...	0	smell hillari fear daniel greenfield shillman ...
1	Watch The Exact Moment Paul Ryan Committed Pol...	0	watch exact moment paul ryan commit polit suic...
2	Kerry to go to Paris in gesture of sympathy U....	1	kerri go pari gestur sympathi u secretari stat...
3	Bernie supporters on Twitter erupt in anger ag...	0	berni support twitter erupt anger dnc tri warn...
4	The Battle of New York: Why This Primary Matte...	1	battl new york primari matter primari day new ...

clean text show that all the special characters and stop words are removed, text is in lowercase, extra spaces also removed and the text is being reduced to its root through stemming method

In [20]:
```
# now i'm going to drop the text column because i no longer need it
new_df.drop('text', axis=1, inplace=True)
new_df.head()
```

Out[20]:

	label	clean_text
0	0	smell hillari fear daniel greenfield shillman ...
1	0	watch exact moment paul ryan commit polit suic...
2	1	kerri go pari gestur sympathi u secretari stat...
3	0	berni support twitter erupt anger dnc tri warn...
4	1	battl new york primari matter primari day new ...

In [21]:
```
from sklearn.feature_extraction.text import CountVectorizer

# using CountVectorizer builds the Bag Of Word model
# convert text to a matrix of tokens
cv = CountVectorizer(max_features=4000, ngram_range=(1,3)) # choose max features is 4000
text_bow = cv.fit_transform(new_df['clean_text'])
```

In [22]:
```
# gettting the shape of the text_bow
text_bow.shape
```

Out[22]:
```
(6335, 4000)
```

Machine learning don't understand text, so I must convert the text into a number. CountVectorizer from sklearn will help to bag of words model. It converts the text to a matrix of token. Basically, it helps to create tokens in a list by tokenizing the text, then it counts the number of times the tokens occur in each sentence and lastly it stores the count.

6. Baseline performance

I decided to create a baseline model using decision tree classifier from sklearn to obtain a baseline result, it is easy and fast to build. It is a good model for benchmarking my proposed machine learning classifiers.

I will then compare the baseline result with the performance of my proposed machine learning classifiers which are the Naïve Bayes classifier and Logistic regression classifier.

By looking at the accuracy score of my baseline model, if my proposed classifiers achieve a better accuracy score, it means they are good for 'fake and real' news classification.

In [23]:
```
X = text_bow
```

```
X = text_bow
y = new_df['label'].copy()
```

In [24]:

```
# use train_test_split to split the dataset for training and testing
# 80% for training, 20% for testing
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix, plot_confusion_matrix

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

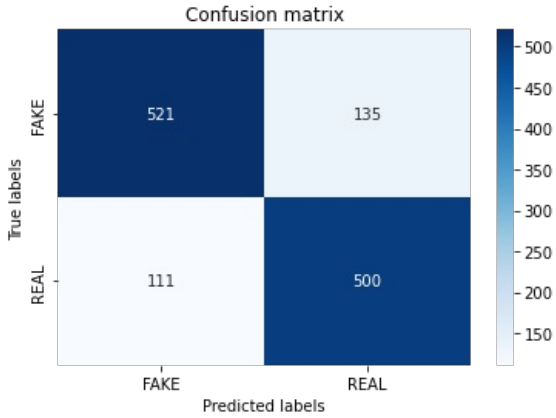
In [25]:

```
# evaluation

from sklearn import tree

knn_classifier = tree.DecisionTreeClassifier()
knn_classifier.fit(X_train, y_train)

pred = knn_classifier.predict(X_test)
# get accuracy score of DecisionTreeClassifier on predicting the test data
test_data_accuracy = accuracy_score(y_test, pred)

print('test accuracy score: ',test_data_accuracy )
print(classification_report(y_test, pred))


knn_ct=pd.crosstab(y_test,pred)
print(knn_ct)

knn_cm = confusion_matrix(y_test, pred)
ax = plt.subplot()
sns.heatmap(knn_cm, annot = True, ax = ax, cmap = 'Blues', fmt = '')

ax.set_xlabel('Predicted labels')
ax.set_ylabel('True labels')
ax.set_title('Confusion matrix')
ax.xaxis.set_ticklabels(['FAKE', 'REAL'])
ax.yaxis.set_ticklabels(['FAKE', 'REAL'])
```

```
test accuracy score:  0.8058405682715075
              precision    recall  f1-score   support

           0       0.82      0.79      0.81       656
           1       0.79      0.82      0.80       611

    accuracy                           0.81      1267
   macro avg       0.81      0.81      0.81      1267
weighted avg       0.81      0.81      0.81      1267

col_0    0    1
label
0      521  135
1      111  500
```

Out[25]:

```
[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



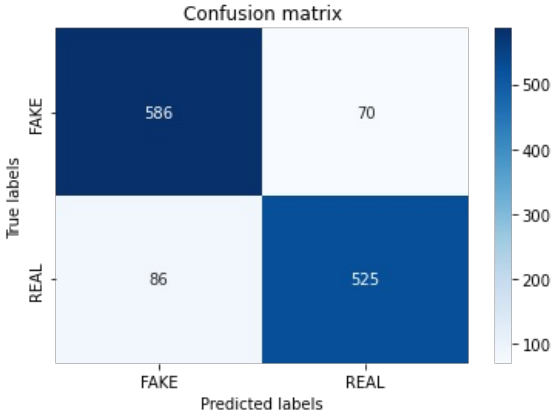
7. Classification approach

My chosen classification approaches are Naïve Bayes and Logistic regression because they are two of the best machine learning algorithms for fake news detection problem. Then, I'll choose the best approach between these two based on their performance results.

In [26]:
```
from sklearn.naive_bayes import MultinomialNB

nb_classifier = MultinomialNB() # Naïve Bayes classifier
nb_classifier.fit(X_train, y_train)


pred = nb_classifier.predict(X_test)
# get accuracy score of Naïve Bayes classifier on predicting the test data
test_data_accuracy = accuracy_score(y_test, pred)

print('test accuracy score: ',test_data_accuracy )
print(classification_report(y_test, pred))

nb_ct=pd.crosstab(y_test,pred)
print(nb_ct)

classes = ['FAKE','REAL']

# plot confusion matrix
nb_cm = confusion_matrix(y_test, pred)
print(nb_cm)
ax = plt.subplot()
sns.heatmap(nb_cm, annot = True, ax = ax, cmap = 'Blues', fmt = '')

ax.set_xlabel('Predicted labels')
ax.set_ylabel('True labels')
ax.set_title('Confusion matrix')
ax.xaxis.set_ticklabels(['FAKE', 'REAL'])
ax.yaxis.set_ticklabels(['FAKE', 'REAL'])
```

```
test accuracy score:  0.8768745067087609
              precision    recall  f1-score   support

           0       0.87      0.89      0.88       656
           1       0.88      0.86      0.87       611

    accuracy                           0.88      1267
   macro avg       0.88      0.88      0.88      1267
weighted avg       0.88      0.88      0.88      1267

col_0    0    1
label
0      586   70
1       86  525
[[586  70]
 [ 86 525]]
```

Out[26]:
```
[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



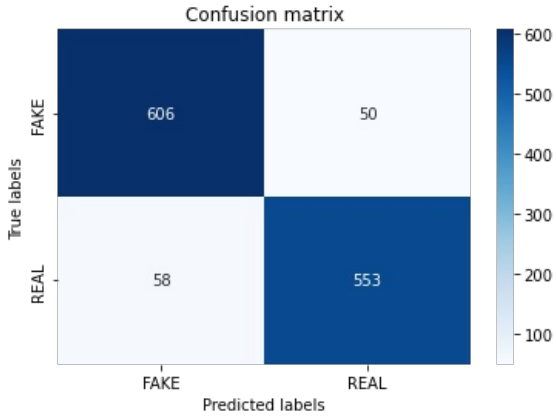
In [27]:
```
from sklearn.linear_model import LogisticRegression

lg_classifier = LogisticRegression()
lg_classifier.fit(X_train, y_train)

pred = lg_classifier.predict(X_test)
# get accuracy score of LogisticRegression classifier on predicting the test data
test_data_accuracy = accuracy_score(y_test, pred)

print('test accuracy score: ',test_data_accuracy )
print(classification_report(y_test, pred))

lg_ct=pd.crosstab(y_test,pred)
print(lg_ct)

classes = ['FAKE','REAL']
```

```
# plot confusion matrix
lg_cm = confusion_matrix(y_test, pred)
print(lg_cm)
ax = plt.subplot()
sns.heatmap(lg_cm, annot = True, ax = ax, cmap = 'Blues', fmt = '')

ax.set_xlabel('Predicted labels')
ax.set_ylabel('True labels')
ax.set_title('Confusion matrix')
ax.xaxis.set_ticklabels(['FAKE', 'REAL'])
ax.yaxis.set_ticklabels(['FAKE', 'REAL'])
```

```
test accuracy score:  0.914759273875296
              precision    recall  f1-score   support

           0       0.91      0.92      0.92       656
           1       0.92      0.91      0.91       611

    accuracy                           0.91      1267
   macro avg       0.91      0.91      0.91      1267
weighted avg       0.91      0.91      0.91      1267

col_0    0    1
label
0      606   50
1       58  553
[[606  50]
 [ 58 553]]
```

```
C:\Users\unnie\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[27]:
```
[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



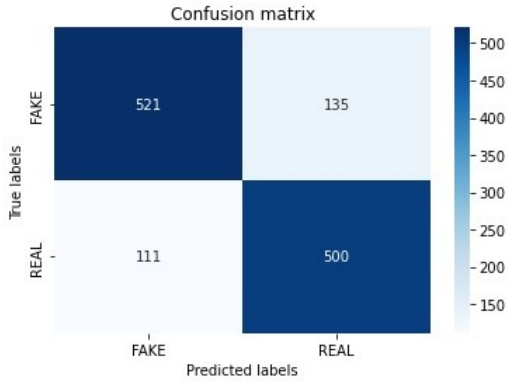
III Conclusions

9. Evaluation

Evaluate the baseline model:

```
test accuracy score:  0.8058405682715075
              precision    recall  f1-score   support

           0       0.82      0.79      0.81       656
           1       0.79      0.82      0.80       611

    accuracy                           0.81      1267
   macro avg       0.81      0.81      0.81      1267
weighted avg       0.81      0.81      0.81      1267

col_0    0    1
label
0      521  135
1      111  500
```

```
[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



Label 0 – represents fake news
Label 1 – represents real news

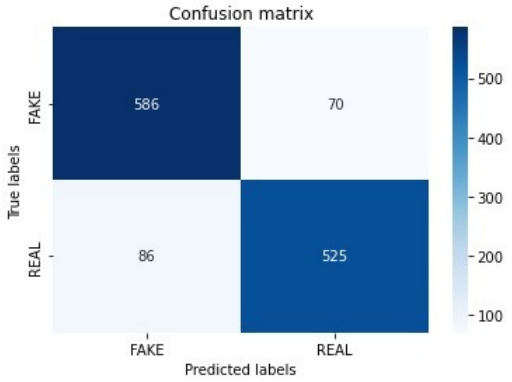
The test accuracy score of the Baseline model is around 81%. The precision scores for label 0 and label 1 are 0.82 and 0.79 respectively. The recall scores for label 0 and label 1 are 0.79 and 0.82 respectively.

The baseline model predicted 521 fake news articles as 'Fake' correctly and it also predicted 500 real news articles as 'Real' correctly.

However, the model predicted wrongly on some fake news and real news. It predicted 135 fake news articles as 'Real' and 111 real news articles as 'Fake'.

Evaluate the Naive Bayes Classifier

```
test accuracy score:  0.8768745067087609
              precision    recall  f1-score   support

           0       0.87      0.89      0.88       656
           1       0.88      0.86      0.87       611

    accuracy                           0.88      1267
   macro avg       0.88      0.88      0.88      1267
weighted avg       0.88      0.88      0.88      1267

col_0    0    1
label
0      586   70
1       86  525
[[586  70]
 [ 86 525]]

[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



The test accuracy score of the Naïve Bayes classifier is around 88%. The precision scores for label 0 and label 1 are 0.87 and 0.88 respectively. The recall scores for label 0 and label 1 are 0.89 and 0.86 respectively.

The Naïve Bayes classifier predicted 586 fake news articles as 'Fake' correctly and it also predicted 525 real news articles as 'Real' correctly.

However, the classifier predicted wrongly on some of the fake news and real news. It predicted 70 fake news articles as 'Real' and 86 real news articles as 'Fake'.

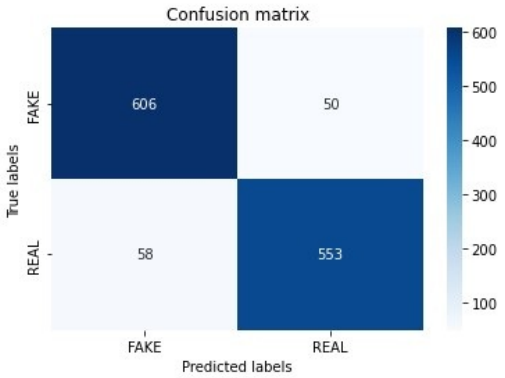
Evaluate the Logistic Regression Classifier

```
test accuracy score:  0.914759273875296
              precision    recall  f1-score   support

           0       0.91      0.92      0.92       656
           1       0.92      0.91      0.91       611
```

```
              1       0.92      0.91      0.91       611

    accuracy                           0.91      1267
   macro avg       0.91      0.91      0.91      1267
weighted avg       0.91      0.91      0.91      1267

col_0    0    1
label
0      606   50
1       58  553
[[606  50]
 [ 58 553]]
```

```
C:\Users\unnie\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs failed to conver
ge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```
[Text(0, 0.5, 'FAKE'), Text(0, 1.5, 'REAL')]
```



The test accuracy score of the Logistic regression classifier is around 91%. The precision scores for both label 0 and label 1 are 0.91 and 0.92. The recall scores for label 0 and label 1 are 0.92 and 0.91 respectively.

The Logistic regression classifier predicted 606 fake news articles as 'Fake' correctly and it also predicted 553 real news articles as 'Real' correctly.

However, the classifier predicted wrongly on some of the fake news and real news. It predicted 50 fake news articles as 'Real' and 58 real news articles as 'Fake'.

10. Summary and conclusions

The results above showed that logistic regression classifier works best on fake news detection, and it outperforms the baseline model and the naïve bayes classifier.

It has achieved the highest accuracy score of 91% on predicting the test data, it’s precision score and recall score also the best. It has the least number of false positive and false negative.

This means the logistic regression classifier works well with binary classification problem, it can not only be used for fake news detection, i can also used it for spam messages or spam emails detection.