APPENDIX 1

NEWS PAPER REVIEW SYSTEM

END TERM REPORT

by

SNO	NAME	REG NO	ROLL NO
1	A.SATYA SAI RAM DASWANTH	11802428	07
2	S.Y.S.SANKEERTHAN	11802488	14

SECTION: K18HV

Department of Intelligent Systems

School of Computer Science Engineering

Lovely Professional University, Jalandhar

APRIL-2020

APPENDIX 2

Student Declaration

This is to declare that this report has been written by us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be copied, we are shall take full responsibility for it.

A.SATYA SAI RAM DASWANTH

ROLL :07

S. Y.S.SANKEERTHAN

ROLL:14

Place: Jalandhar

Date: 10/04/20

APPENDIX 3

TABLE OF CONTENTS

TITLE **PAGE NO.**

APPENDIX 4

BONAFIDE CERTIFICATE

Certified that this project report "NEWSPAPER REVIEW SYSTEM" is the bonafide work of "A.SATYA SAI RAM DASWANTH , S.Y.S.SANKEERTHAN" who carried out the project work under my supervision.

INTRODUCTION

To build a simpleNEWSPAPER REVIEW SYSTEM in python, we need to know what is the review. A reviewer work is to revie th news circulated around us every day whether it is true or false.It isw very useful in nowadys because there is lot of false news so by using this we can control.

OBJECTIVES OF THE PROJECT ASSIGNED :

The main objectives of the NEWS PAPER REVIEW SYSTEM project are

1. To create a some news combined of false and true news.
2. To acknowledge the public what and which news is correct or which is false.

REVIEWE SYSTEM:

A reviewr system is a system that makes suggestions based on the type of news For example, when you continuously browse thesome news about coronovirus .

Because this issue is ruling the world for the better understanding and now a person want to know which state is effected heavily somebody posting with half knowledge and false news understand that there is a review system working under the hood.

Content based reviewer system:

It works on the generated data of a user. There are only one way in which news is generated, either explicitly or implicitly. A user profile is created using the news generated. It contains the news including true news and false news.

Collaborative Recommender System

This system makes review based on users commented the same news in a similar way. Using item similarity, it can also perform collaborative filtering (like ‘Users who review this news’).

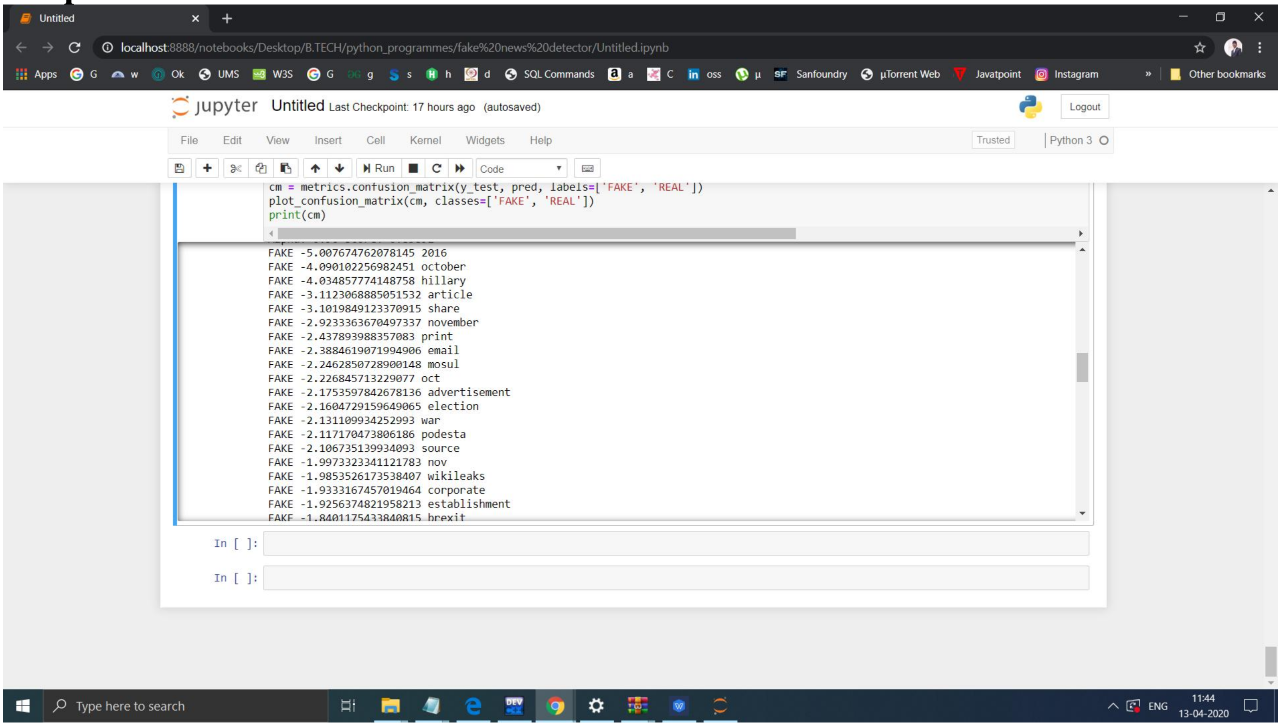
Code:

```
 import pandas as pd
 import matplotlib.pyplot as plt
 from matplotlib import style
 style.use('ggplot')

 web_stats = {"Day":[1,2,3,4,5,6],
        "Visitors":[43,53,34,45,64,34],
        "Bounce_Rate":[65,72,62,64,54,66]}
 df = pd.DataFrame(web_stats)
 print(df)
```

Code:

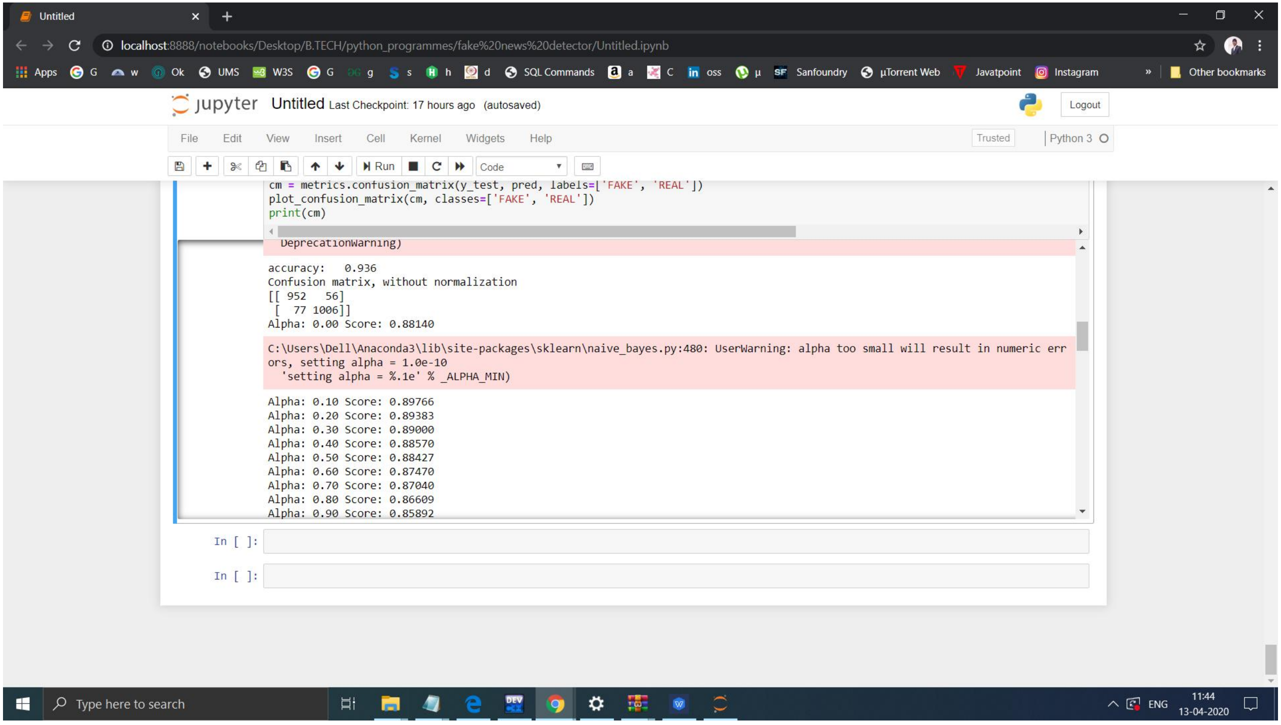
```
import pandas as pd
from sklearn.model_selection import train_test_split
import sklearn
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
from pandas_ml import ConfusionMatrix
from matplotlib import pyplot as plt
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.feature_extraction.text import HashingVectorizer
import itertools
import numpy as np
df =
pd.read_csv("fake_or_real_news.csv",engine='python',encoding="utf-8-
sig")
df.shape
df.head()
df = df.set_index("Unnamed: 0")
df.head()
y = df.label
df.drop("label", axis=1)
X_train, X_test, y_train, y_test = train_test_split(df['text'], y,
test_size=0.33, random_state=53)
count_train = count_vectorizer.fit_transform(X_train)
count_test = count_vectorizer.transform(X_test)
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_df=0.7)
tfidf_train = tfidf_vectorizer.fit_transform(X_train)
tfidf_test = tfidf_vectorizer.transform(X_test)
count_df = pd.DataFrame(count_train.A,
columns=count_vectorizer.get_feature_names())
tfidf_df = pd.DataFrame(tfidf_train.A,
columns=tfidf_vectorizer.get_feature_names())
difference = set(count_df.columns) - set(tfidf_df.columns)
print(difference)
print(count_df.equals(tfidf_df))
print(count_df.head())
print(tfidf_df.head())
def plot_confusion_matrix(cm, classes,normalize=False,title='Confusion
matrix',cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)
 if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
 else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],horizontalalignment="center",color="white" if
cm[i, j] > thresh else "black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
clf = MultinomialNB()
clf.fit(tfidf_train, y_train)
pred = clf.predict(tfidf_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
clf = MultinomialNB()
clf.fit(count_train, y_train)
pred = clf.predict(count_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
linear_clf = PassiveAggressiveClassifier(n_iter=50)
linear_clf.fit(tfidf_train, y_train)
pred = linear_clf.predict(tfidf_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
clf = MultinomialNB(alpha=0.1)
last_score = 0
for alpha in np.arange(0,1,.1):
    nb_classifier = MultinomialNB(alpha=alpha)
    nb_classifier.fit(tfidf_train, y_train)
    pred = nb_classifier.predict(tfidf_test)
    score = metrics.accuracy_score(y_test, pred)
    if score > last_score:
        clf = nb_classifier
    print("Alpha: {:.2f} Score: {:.5f}".format(alpha, score))
def most_informative_feature_for_binary_classification(vectorizer,
classifier, n=100):
    class_labels = classifier.classes_
    feature_names = vectorizer.get_feature_names()
topn_class1 = sorted(zip(classifier.coef_[0], feature_names))[:n]
    topn_class2 = sorted(zip(classifier.coef_[0], feature_names))[-n:]
    for coef, feat in topn_class1:
        print(class_labels[0], coef, feat)
    print()
    for coef, feat in reversed(topn_class2):
        print(class_labels[1], coef, feat)
most_informative_feature_for_binary_classification(tfidf_vectorizer,
linear_clf, n=30)
feature_names = tfidf_vectorizer.get_feature_names()
tokens_with_weights = sorted(list(zip(feature_names, clf.coef_[0])))
hash_vectorizer = HashingVectorizer(stop_words='english',
non_negative=True)
hash_train = hash_vectorizer.fit_transform(X_train)
hash_test = hash_vectorizer.transform(X_test)
clf = MultinomialNB(alpha=.01)
clf.fit(hash_train, y_train)
pred = clf.predict(hash_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
clf = PassiveAggressiveClassifier(n_iter=50)
clf.fit(hash_train, y_train)
pred = clf.predict(hash_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

Snapshots:



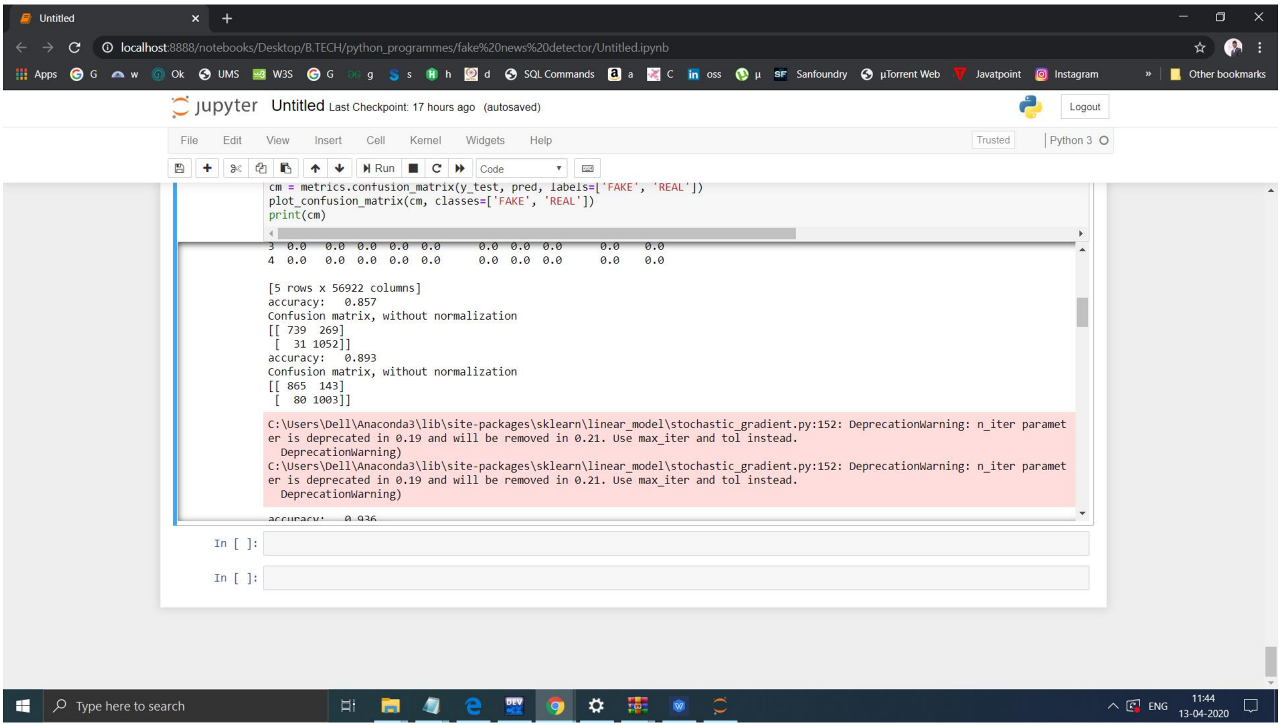
```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
FAKE -5.007674762078145 2016
FAKE -4.090102256982451 october
FAKE -4.034857774148758 hillary
FAKE -3.1123068885051532 article
FAKE -3.1019849123370915 share
FAKE -2.9233363670497337 november
FAKE -2.437893988357083 print
FAKE -2.3884619071994906 email
FAKE -2.2462850728900148 mosul
FAKE -2.226845713229077 oct
FAKE -2.1753597842678136 advertisement
FAKE -2.1604729159649065 election
FAKE -2.131109934252993 war
FAKE -2.117170473806186 podesta
FAKE -2.106735139934093 source
FAKE -1.9973323341121783 nov
FAKE -1.9853526173538407 wikileaks
FAKE -1.9333167457019464 corporate
FAKE -1.9256374821958213 establishment
FAKE -1.8401175433840815 brexit
```



```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

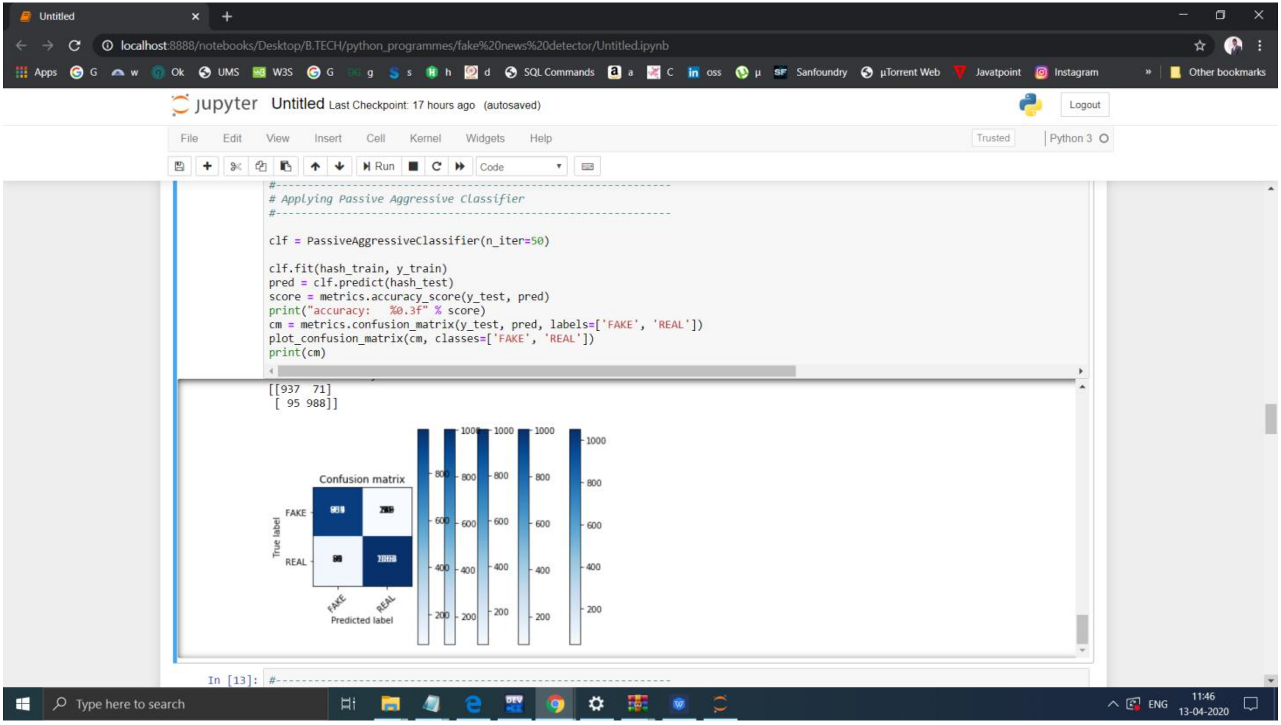
```
  DeprecationWarning)

accuracy:   0.936
Confusion matrix, without normalization
[[ 952   56]
 [  77 1006]]
Alpha: 0.00 Score: 0.88140

C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\naive_bayes.py:480: UserWarning: alpha too small will result in numeric err
ors, setting alpha = 1.0e-10
  'setting alpha = %.1e' % _ALPHA_MIN)

Alpha: 0.10 Score: 0.89766
Alpha: 0.20 Score: 0.89383
Alpha: 0.30 Score: 0.89000
Alpha: 0.40 Score: 0.88570
Alpha: 0.50 Score: 0.88427
Alpha: 0.60 Score: 0.87470
Alpha: 0.70 Score: 0.87040
Alpha: 0.80 Score: 0.86609
Alpha: 0.90 Score: 0.85892
```


Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
11:44
13-04-2020

```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
3  0.0    0.0  0.0  0.0  0.0      0.0  0.0  0.0      0.0    0.0
4  0.0    0.0  0.0  0.0  0.0      0.0  0.0  0.0      0.0    0.0

[5 rows x 56922 columns]
accuracy:   0.857
Confusion matrix, without normalization
[[ 739  269]
 [  31 1052]]
accuracy:   0.893
Confusion matrix, without normalization
[[ 865  143]
 [  80 1003]]
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
  DeprecationWarning)
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
  DeprecationWarning)
accuracy:   0.936
```

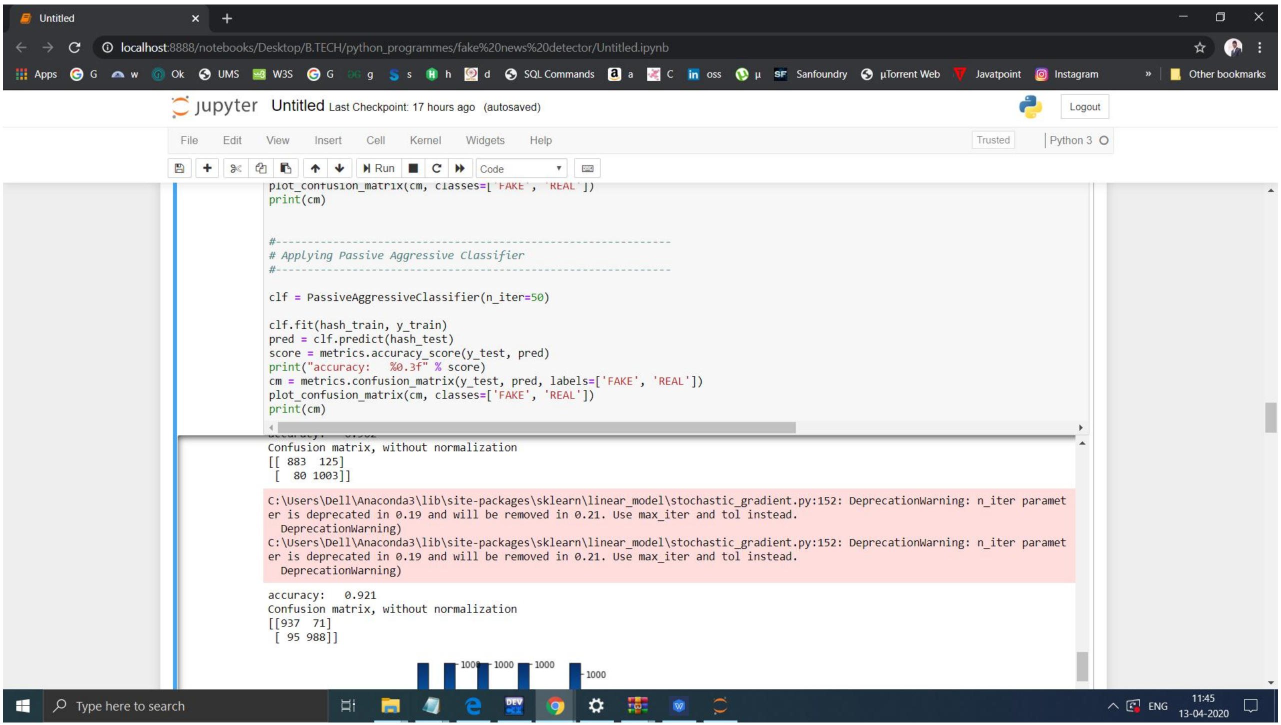
```
In [ ]:

In [ ]:
```


Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Confusion matrix
FAKE
REAL
True label
Predicted label
1000
800
600
400
200
Type here to search
ENG
11:46
13-04-2020

```
#-----------------------------------------------------------------
# Applying Passive Aggressive Classifier
#-----------------------------------------------------------------

clf = PassiveAggressiveClassifier(n_iter=50)

clf.fit(hash_train, y_train)
pred = clf.predict(hash_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
[[937  71]
 [ 95 988]]
```

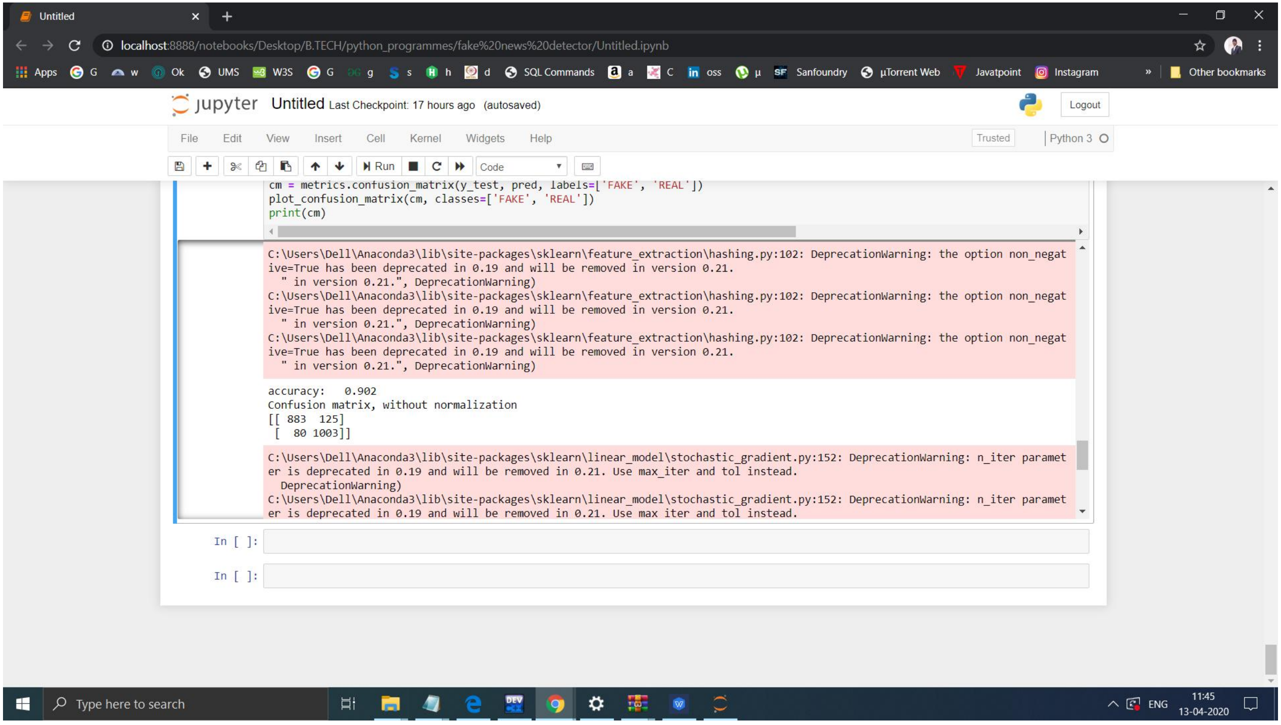
```
In [13]: #-----------------------------------------------------------------
```



Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
1000
1000
1000
1000
Type here to search
ENG
11:45
13-04-2020

```
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)


#-------------------------------------------------------------
# Applying Passive Aggressive Classifier
#-------------------------------------------------------------

clf = PassiveAggressiveClassifier(n_iter=50)

clf.fit(hash_train, y_train)
pred = clf.predict(hash_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:   %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
Confusion matrix, without normalization
[[ 883  125]
 [  80 1003]]

C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
  DeprecationWarning)
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
  DeprecationWarning)

accuracy:   0.921
Confusion matrix, without normalization
[[937  71]
 [ 95 988]]
```

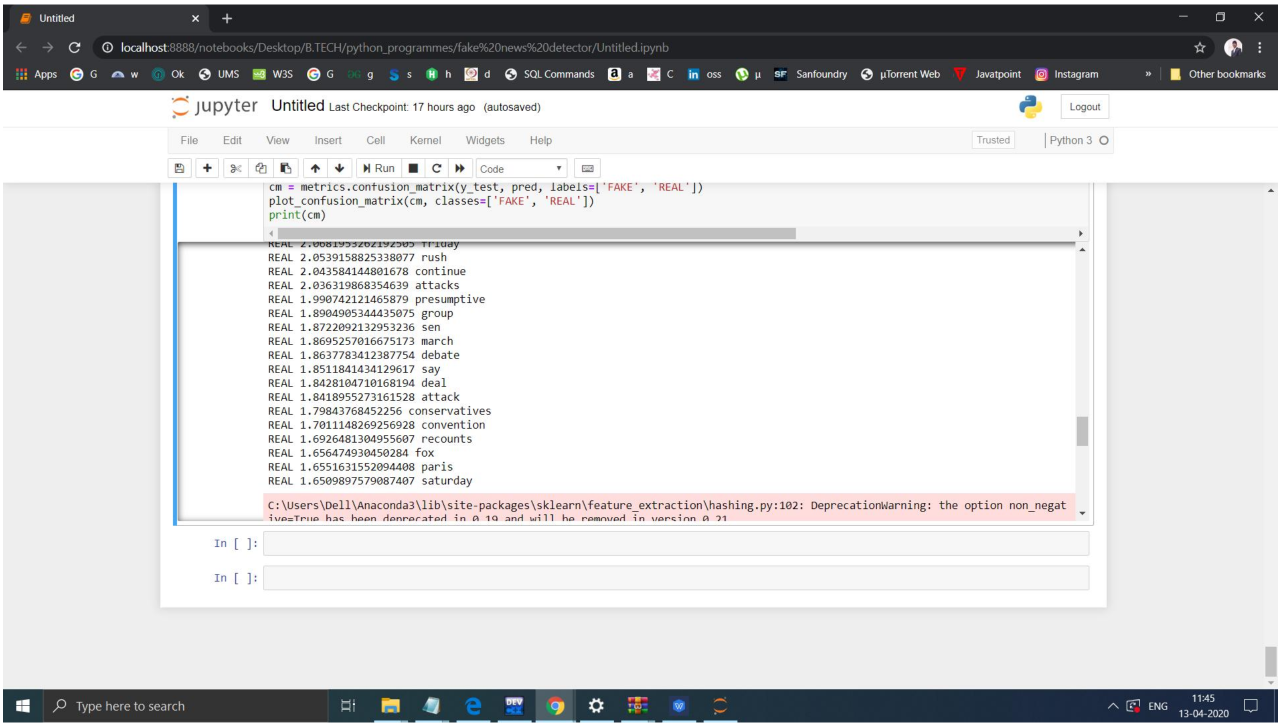


Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
11:45
13-04-2020

```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\feature_extraction\hashing.py:102: DeprecationWarning: the option non_negat
ive=True has been deprecated in 0.19 and will be removed in version 0.21.
  " in version 0.21.", DeprecationWarning)
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\feature_extraction\hashing.py:102: DeprecationWarning: the option non_negat
ive=True has been deprecated in 0.19 and will be removed in version 0.21.
  " in version 0.21.", DeprecationWarning)
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\feature_extraction\hashing.py:102: DeprecationWarning: the option non_negat
ive=True has been deprecated in 0.19 and will be removed in version 0.21.
  " in version 0.21.", DeprecationWarning)

accuracy:   0.902
Confusion matrix, without normalization
[[ 883  125]
 [  80 1003]]

C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
  DeprecationWarning)
C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\linear_model\stochastic_gradient.py:152: DeprecationWarning: n_iter paramet
er is deprecated in 0.19 and will be removed in 0.21. Use max_iter and tol instead.
```

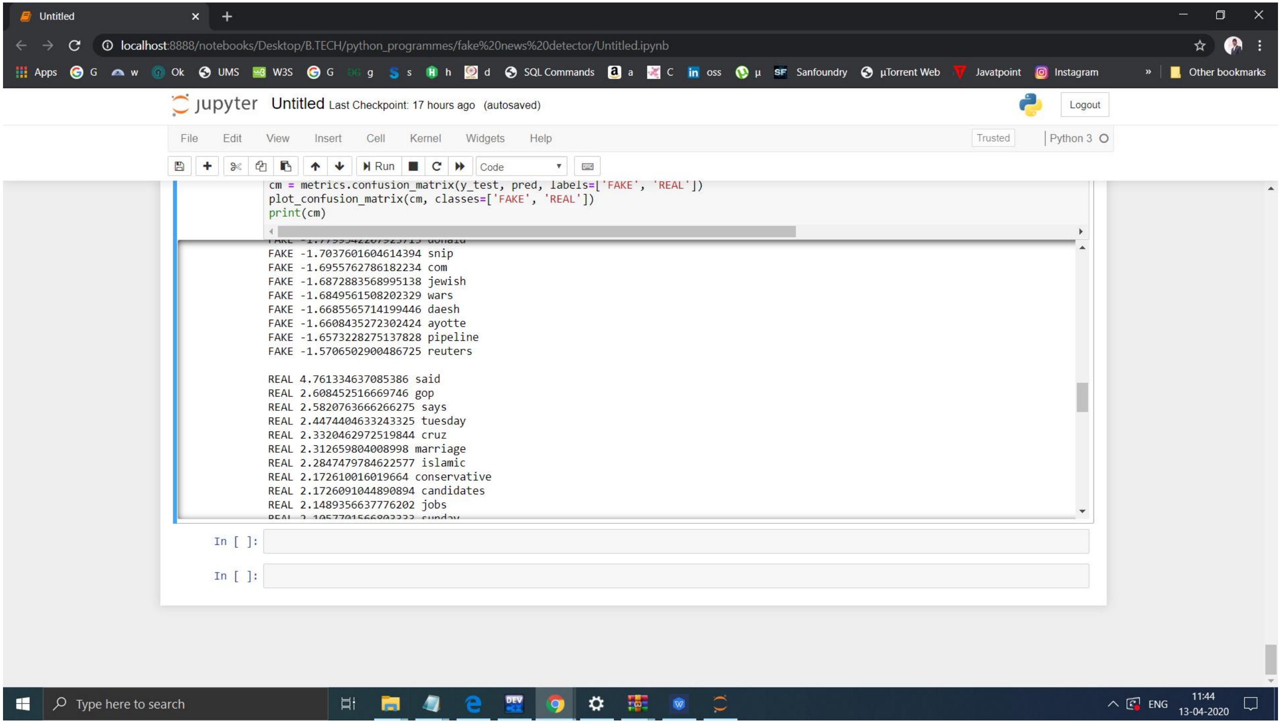
```
In [ ]:

In [ ]:
```


Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
11:45
13-04-2020

```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
REAL 2.0681953262192505 friday
REAL 2.0539158825338077 rush
REAL 2.043584144801678 continue
REAL 2.036319868354639 attacks
REAL 1.990742121465879 presumptive
REAL 1.8904905344435075 group
REAL 1.8722092132953236 sen
REAL 1.8695257016675173 march
REAL 1.8637783412387754 debate
REAL 1.8511841434129617 say
REAL 1.8428104710168194 deal
REAL 1.8418955273161528 attack
REAL 1.79843768452256 conservatives
REAL 1.7011148269256928 convention
REAL 1.6926481304955607 recounts
REAL 1.656474930450284 fox
REAL 1.6551631552094408 paris
REAL 1.6509897579087407 saturday

C:\Users\Dell\Anaconda3\lib\site-packages\sklearn\feature_extraction\hashing.py:102: DeprecationWarning: the option non_negat
ive=True has been deprecated in 0.19 and will be removed in version 0.21
```

```
In [ ]:

In [ ]:
```


Untitled
localhost:8888/notebooks/Desktop/B.TECH/python_programmes/fake%20news%20detector/Untitled.ipynb
jupyter Untitled Last Checkpoint: 17 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
ENG
11:44
13-04-2020

```
cm = metrics.confusion_matrix(y_test, pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['FAKE', 'REAL'])
print(cm)
```

```
FAKE -1.7799542207925715 donald
FAKE -1.7037601604614394 snip
FAKE -1.6955762786182234 com
FAKE -1.6872883568995138 jewish
FAKE -1.6849561508202329 wars
FAKE -1.6685565714199446 daesh
FAKE -1.6608435272302424 ayotte
FAKE -1.6573228275137828 pipeline
FAKE -1.5706502900486725 reuters

REAL 4.761334637085386 said
REAL 2.608452516669746 gop
REAL 2.5820763666266275 says
REAL 2.4474404633243325 tuesday
REAL 2.3320462972519844 cruz
REAL 2.312659804008998 marriage
REAL 2.2847479784622577 islamic
REAL 2.172610016019664 conservative
REAL 2.1726091044890894 candidates
REAL 2.1489356637776202 jobs
REAL 2.1057701566803333 sunday
```

```
In [ ]:

In [ ]:
```