FLIP ROBO

FAKE NEWS CLASSIFIER PROJECT – PREDICTING WHETHER A NEWS IS FAKE OR NOT

Submitted by:

NATASHA PODDAR

ACKNOWLEDGMENT

The data used in the project was provided in true.csv & Fake.csv file

INTRODUCTION

- Business Problem Framing

Fake news has become one of the biggest problems of our age. It has serious impact on our online as well as offline discourse. One can even go as far as saying that, to date, fake news poses a clear and present danger to western democracy and stability of the society.

- Conceptual Background of the Domain Problem

A basic understanding on news background is needed

- Review of Literature

Fake news's simple meaning is to incorporate information that leads people to the wrong path. Nowadays fake news spreading like water and people share this information without verifying it. This is often done to further or impose certain ideas and is often achieved with political agendas.
For media outlets, the ability to attract viewers to their websites is necessary to generate online advertising revenue. So it is necessary to detect fake news

- Motivation for the Problem Undertaken

This issue is very realistic and common in today’s world and one should know to deal with such situations in the future

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  Firstly missing values were checked

  Correlation with all independent variables and wrt target were checked

  Feature extraction was done through count vectorizer & Tfidvectorizer

  Models were applied to train and test the model

  .

- Data Sources and their formats

  The complete data was provided In true.csv & fake.csv file

- Data Preprocessing Done:
  1. Duplicate values check
  2. Unique & Count of all columns were checked
  3. Missing values were checked
  4. Catagorical data was Encoded
  5. Correlation check
  6. Graphical Univariate Analysis
  7. Feature extraction was done - count vectorizer & Tfidvectorizer

- Hardware and Software Requirements and Tools Used
  1. Pandas – For Data Reading and understanding
  2. Label Encoder –(SK LEARN) – For Encoding the categorical data into numerical ones
  3. Duplicate- To check for duplicate Values
  4. Numpy- For mathematical operations
  5. LOGITSIC REGRESSION (SKLEARN) – Training & Testing the model
  6. DTC (SKLEARN) – Training & Testing the model
  7. GAUSSIAN NB (SKLEARN) – Training & Testing the model

8. RANDOM FOREST CLASSIFIER (SKLEARN) – Training & Testing the model
9. CROSS VAL SCORE – Regularizing the model
10. GRID SEARCH CV- Hyper Tuning the Model for higher accuracy
11. SEABORN- VISUALIZATION LIBRARY -COUNTPLOTS
12. MATPLOTLIB.PY PLOT -Visualization tool

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
    1. Firstly missing values were checked .
    2. Correlation with all independent variables and wrt target were checked
    3. Feature extraction was performed
    4. Models were applied to train and test the model

- Testing of Identified Approaches (Algorithms)
    1. LOGISTIC REGRESSION
    2. DECISION TREE CLASSIFIER
    3. GAUSSIAN NB CLASSIFIER
    4. RANDOM FOREST CLASSIFIER

- Key Metrics for success in solving problem under consideration
    1. ACCURACY SCORE
    2. CONFUSION MATRIX
    3. CLASSIFICATION REPORT
    4. AUC-ROC CURVE
- Visualizations

  Seaborn Library was used along with matplotlib Library for visualizations

  Count plots were made and analysed

- Interpretation of the Results

  RANDOM FOREST REGRESSOR had the highest model accuracy and the difference between CV MEAN SCORE & MODEL ACCURACY

SCORE was the least hence we had hyper tuned the said model and saved the same

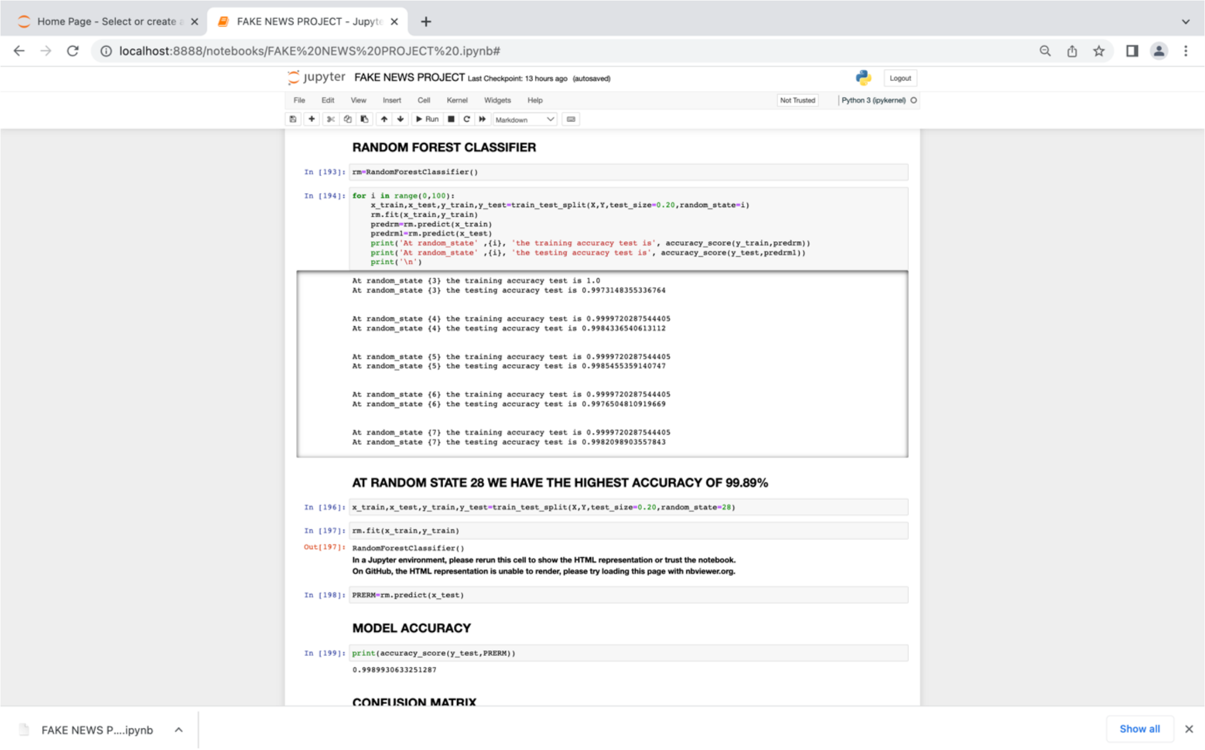
CONCLUSION

- Key Findings and Conclusions of the Study

  Random Forest Regressor after hyper tuning had a higher accuracy and the same model was selected and saved

- Learning Outcomes of the Study in respect of Data Science

  Strong insights were derived from the various visualization tools which helped in understanding the various relationships between the target and other variables



Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
FAKE NEWS P....ipynb
Show all

RANDOM FOREST CLASSIFIER

```
In [193]: rm=RandomForestClassifier()
```

```
In [194]: for i in range(0,100):
              x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=i)
              rm.fit(x_train,y_train)
              predrm=rm.predict(x_train)
              predrml=rm.predict(x_test)
              print('At random_state' ,{i}, 'the training accuracy test is', accuracy_score(y_train,predrm))
              print('At random_state' ,{i}, 'the testing accuracy test is', accuracy_score(y_test,predrml))
              print('\n')
```

```
At random_state {3} the training accuracy test is 1.0
At random_state {3} the testing accuracy test is 0.9973148355336764


At random_state {4} the training accuracy test is 0.9999720287544405
At random_state {4} the testing accuracy test is 0.9984336540613112


At random_state {5} the training accuracy test is 0.9999720287544405
At random_state {5} the testing accuracy test is 0.9985455359140747


At random_state {6} the training accuracy test is 0.9999720287544405
At random_state {6} the testing accuracy test is 0.9976504810919669


At random_state {7} the training accuracy test is 0.9999720287544405
At random_state {7} the testing accuracy test is 0.9982098903557843
```

AT RANDOM STATE 28 WE HAVE THE HIGHEST ACCURACY OF 99.89%

```
In [196]: x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=28)
```

```
In [197]: rm.fit(x_train,y_train)
```

```
Out[197]: RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

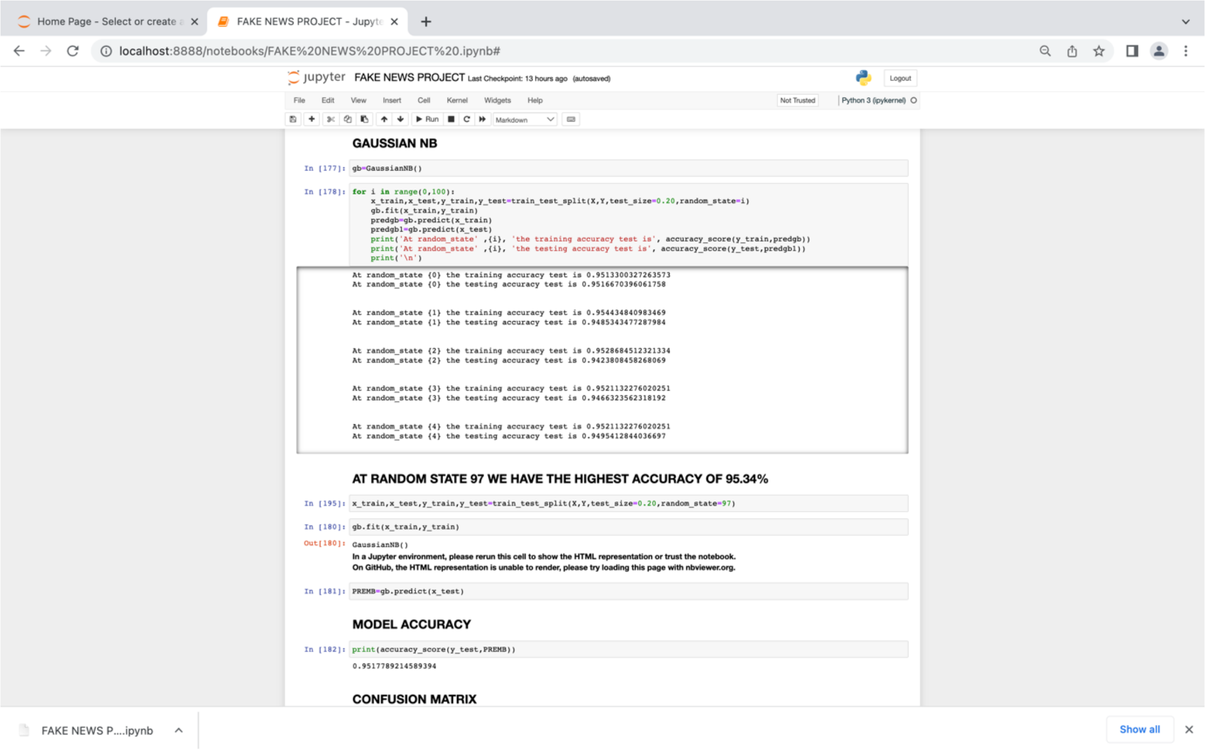
```
In [198]: PRERM=rm.predict(x_test)
```

MODEL ACCURACY

```
In [199]: print(accuracy_score(y_test,PRERM))

0.9989930633251287
```

CONFUSION MATRIX



Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
FAKE NEWS P....ipynb
Show all

GAUSSIAN NB

```
In [177]: gb=GaussianNB()
```

```
In [178]: for i in range(0,100):
              x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=i)
              gb.fit(x_train,y_train)
              predgb=gb.predict(x_train)
              predgbl=gb.predict(x_test)
              print('At random_state' ,{i}, 'the training accuracy test is', accuracy_score(y_train,predgb))
              print('At random_state' ,{i}, 'the testing accuracy test is', accuracy_score(y_test,predgbl))
              print('\n')
```

```
At random_state {0} the training accuracy test is 0.9513300327263573
At random_state {0} the testing accuracy test is 0.9516670396061758


At random_state {1} the training accuracy test is 0.954434840983469
At random_state {1} the testing accuracy test is 0.9485343477287984


At random_state {2} the training accuracy test is 0.9528684512321334
At random_state {2} the testing accuracy test is 0.9423808458268069


At random_state {3} the training accuracy test is 0.9521132276020251
At random_state {3} the testing accuracy test is 0.9466323562318192


At random_state {4} the training accuracy test is 0.9521132276020251
At random_state {4} the testing accuracy test is 0.9495412844036697
```

AT RANDOM STATE 97 WE HAVE THE HIGHEST ACCURACY OF 95.34%

```
In [195]: x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=97)
```

```
In [180]: gb.fit(x_train,y_train)
```

```
Out[180]: GaussianNB()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

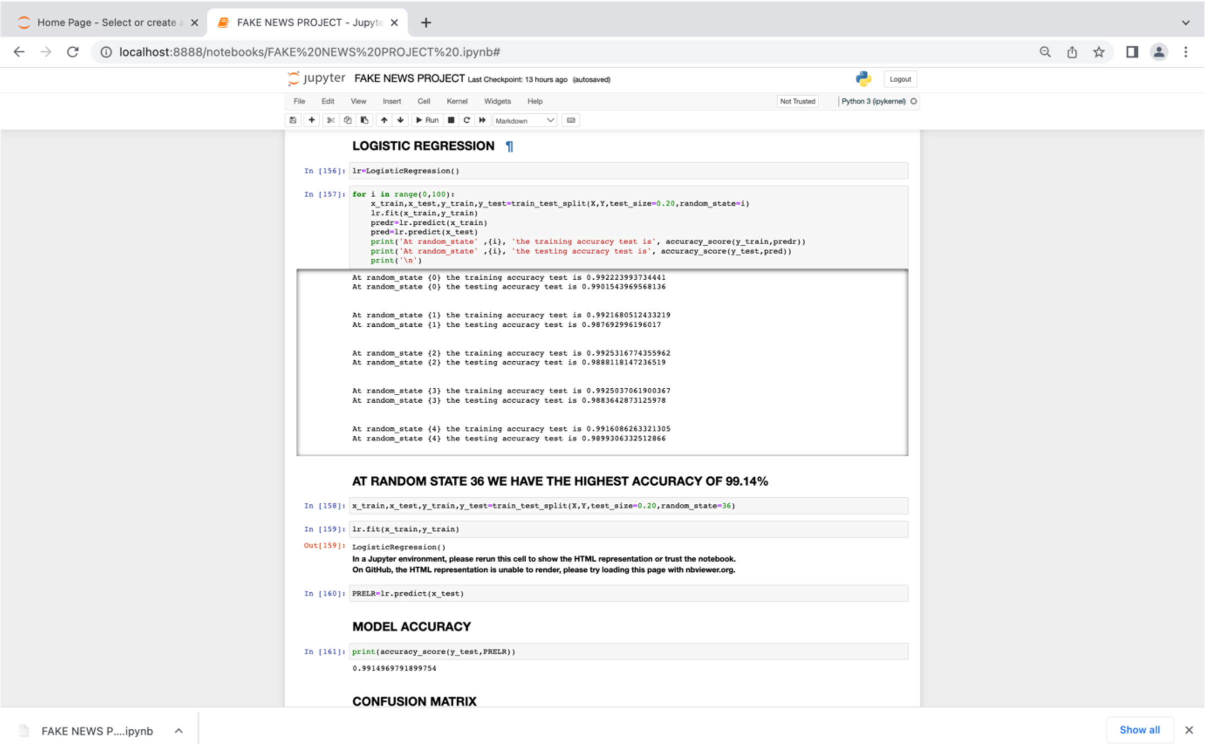
```
In [181]: PREMB=gb.predict(x_test)
```

MODEL ACCURACY

```
In [182]: print(accuracy_score(y_test,PREMB))

0.9517789214589394
```

CONFUSION MATRIX


Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
FAKE NEWS P....ipynb
Show all

LOGISTIC REGRESSION ¶

```
In [156]: lr=LogisticRegression()
```

```
In [157]: for i in range(0,100):
              x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=i)
              lr.fit(x_train,y_train)
              predr=lr.predict(x_train)
              pred=lr.predict(x_test)
              print('At random_state' ,{i}, 'the training accuracy test is', accuracy_score(y_train,predr))
              print('At random_state' ,{i}, 'the testing accuracy test is', accuracy_score(y_test,pred))
              print('\n')
```

```
At random_state {0} the training accuracy test is 0.992223993734441
At random_state {0} the testing accuracy test is 0.9901543969568136


At random_state {1} the training accuracy test is 0.9921680512433219
At random_state {1} the testing accuracy test is 0.987692996196017


At random_state {2} the training accuracy test is 0.9925316774355962
At random_state {2} the testing accuracy test is 0.9888118147236519


At random_state {3} the training accuracy test is 0.9925037061900367
At random_state {3} the testing accuracy test is 0.9883642873125978


At random_state {4} the training accuracy test is 0.9916086263321305
At random_state {4} the testing accuracy test is 0.9899306332512866
```

AT RANDOM STATE 36 WE HAVE THE HIGHEST ACCURACY OF 99.14%

```
In [158]: x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=36)
```

```
In [159]: lr.fit(x_train,y_train)
```

```
Out[159]: LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

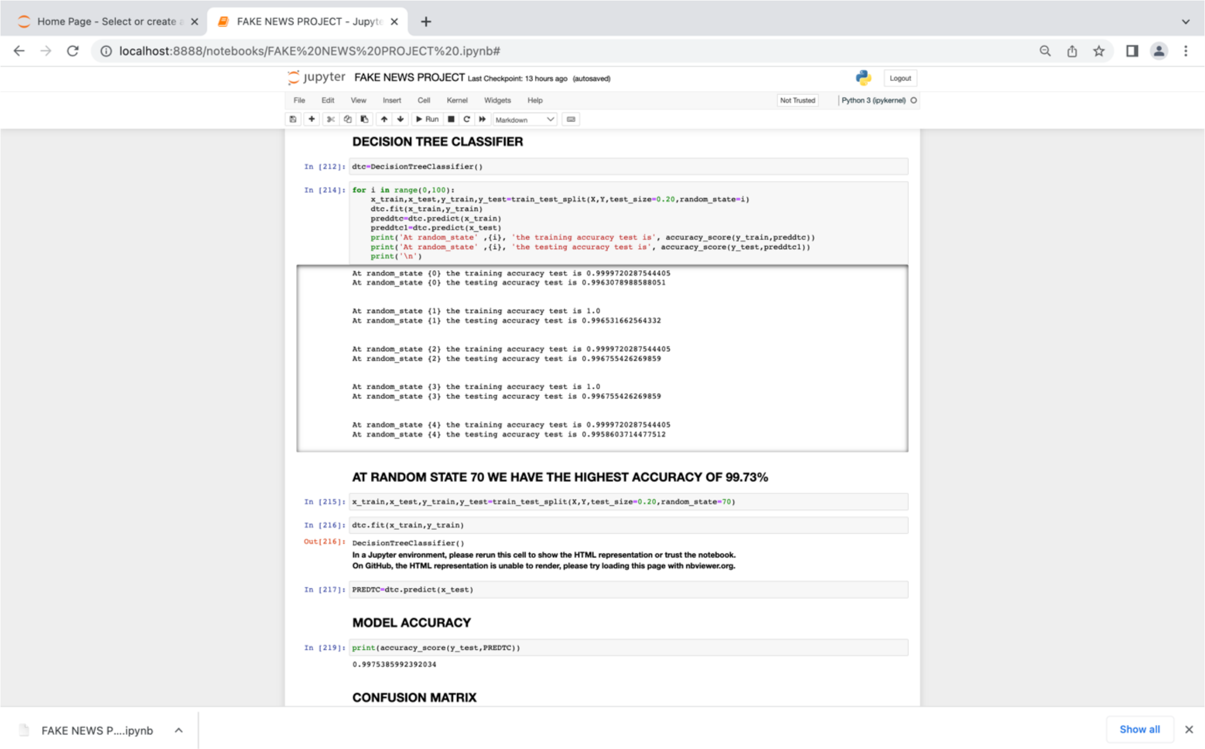
```
In [160]: PRELR=lr.predict(x_test)
```

MODEL ACCURACY

```
In [161]: print(accuracy_score(y_test,PRELR))

0.9914969791899754
```

CONFUSION MATRIX


Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
FAKE NEWS P....ipynb
Show all

DECISION TREE CLASSIFIER

```
In [212]: dtc=DecisionTreeClassifier()
```

```
In [214]: for i in range(0,100):
              x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=i)
              dtc.fit(x_train,y_train)
              preddtc=dtc.predict(x_train)
              preddtc1=dtc.predict(x_test)
              print('At random_state' ,{i}, 'the training accuracy test is', accuracy_score(y_train,preddtc))
              print('At random_state' ,{i}, 'the testing accuracy test is', accuracy_score(y_test,preddtc1))
              print('\n')
```

```
At random_state {0} the training accuracy test is 0.9999720287544405
At random_state {0} the testing accuracy test is 0.9963078988588051


At random_state {1} the training accuracy test is 1.0
At random_state {1} the testing accuracy test is 0.996531662564332


At random_state {2} the training accuracy test is 0.9999720287544405
At random_state {2} the testing accuracy test is 0.996755426269859


At random_state {3} the training accuracy test is 1.0
At random_state {3} the testing accuracy test is 0.996755426269859


At random_state {4} the training accuracy test is 0.9999720287544405
At random_state {4} the testing accuracy test is 0.9958603714477512
```

AT RANDOM STATE 70 WE HAVE THE HIGHEST ACCURACY OF 99.73%

```
In [215]: x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.20,random_state=70)
```

```
In [216]: dtc.fit(x_train,y_train)
```

```
Out[216]: DecisionTreeClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

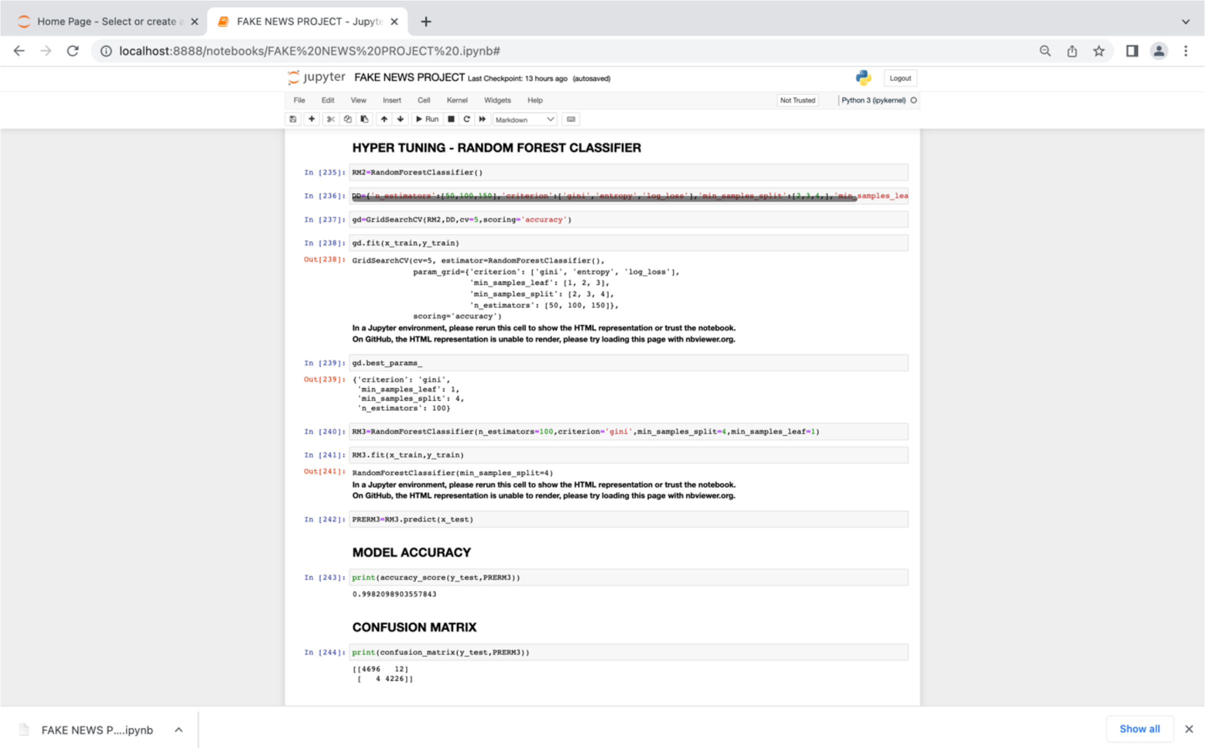
```
In [217]: PREDTC=dtc.predict(x_test)
```

MODEL ACCURACY

```
In [219]: print(accuracy_score(y_test,PREDTC))

0.9975385992392034
```

CONFUSION MATRIX


Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
FAKE NEWS P....ipynb
Show all

HYPER TUNING - RANDOM FOREST CLASSIFIER

```
In [235]: RM2=RandomForestClassifier()

In [236]: DD={'n_estimators':[50,100,150],'criterion':['gini','entropy','log_loss'],'min_samples_split':[2,3,4,],'min_samples_lea

In [237]: gd=GridSearchCV(RM2,DD,cv=5,scoring='accuracy')

In [238]: gd.fit(x_train,y_train)

Out[238]: GridSearchCV(cv=5, estimator=RandomForestClassifier(),
                       param_grid={'criterion': ['gini', 'entropy', 'log_loss'],
                                   'min_samples_leaf': [1, 2, 3],
                                   'min_samples_split': [2, 3, 4],
                                   'n_estimators': [50, 100, 150]},
                       scoring='accuracy')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [239]: gd.best_params_

Out[239]: {'criterion': 'gini',
           'min_samples_leaf': 1,
           'min_samples_split': 4,
           'n_estimators': 100}

In [240]: RM3=RandomForestClassifier(n_estimators=100,criterion='gini',min_samples_split=4,min_samples_leaf=1)

In [241]: RM3.fit(x_train,y_train)

Out[241]: RandomForestClassifier(min_samples_split=4)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [242]: PRERM3=RM3.predict(x_test)
```

MODEL ACCURACY

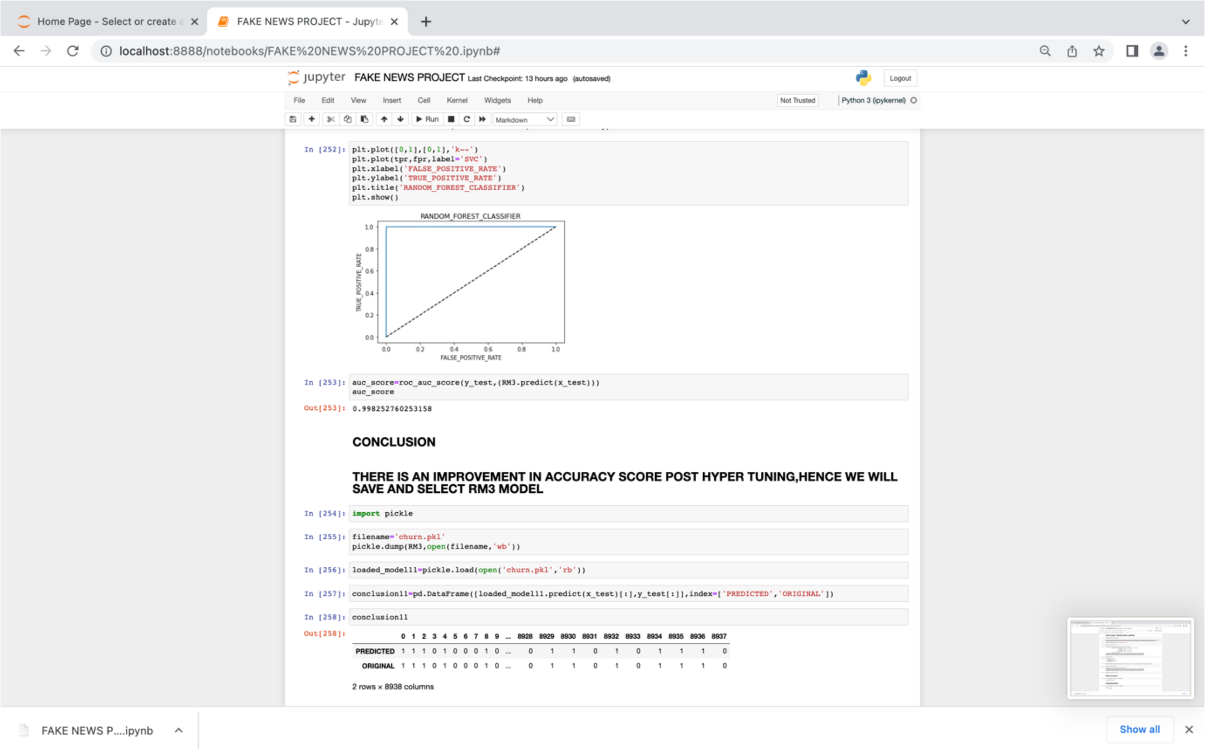
```
In [243]: print(accuracy_score(y_test,PRERM3))

          0.9982098903557843
```

CONFUSION MATRIX

```
In [244]: print(confusion_matrix(y_test,PRERM3))

          [[4696   12]
           [   4 4226]]
```


Home Page - Select or create a
FAKE NEWS PROJECT - Jupyter
localhost:8888/notebooks/FAKE%20NEWS%20PROJECT%20.ipynb#
jupyter FAKE NEWS PROJECT Last Checkpoint: 13 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
RANDOM_FOREST_CLASSIFIER
TRUE_POSITIVE_RATE
FALSE_POSITIVE_RATE
0.0
0.2
0.4
0.6
0.8
1.0
FAKE NEWS P....ipynb
Show all

```
In [252]: plt.plot([0,1],[0,1],'k--')
          plt.plot(tpr,fpr,label='SVC')
          plt.xlabel('FALSE_POSITIVE_RATE')
          plt.ylabel('TRUE_POSITIVE_RATE')
          plt.title('RANDOM_FOREST_CLASSIFIER')
          plt.show()

In [253]: auc_score=roc_auc_score(y_test,(RM3.predict(x_test)))
          auc_score

Out[253]: 0.998252760253158
```

CONCLUSION

THERE IS AN IMPROVEMENT IN ACCURACY SCORE POST HYPER TUNING,HENCE WE WILL SAVE AND SELECT RM3 MODEL

```
In [254]: import pickle

In [255]: filename='churn.pkl'
          pickle.dump(RM3,open(filename,'wb'))

In [256]: loaded_modell1=pickle.load(open('churn.pkl','rb'))

In [257]: conclusionl1=pd.DataFrame([loaded_modell1.predict(x_test)[:],y_test[:]],index=['PREDICTED','ORIGINAL'])

In [258]: conclusionl1

Out[258]:
```

	0	1	2	3	4	5	6	7	8	9	...	8928	8929	8930	8931	8932	8933	8934	8935	8936	8937
PREDICTED	1	1	1	0	1	0	0	0	1	0	...	0	1	1	0	1	0	1	1	1	0
ORIGINAL	1	1	1	0	1	0	0	0	1	0	...	0	1	1	0	1	0	1	1	1	0

2 rows × 8938 columns