Chicago Crime Data Analysis

Indrasena kallam

AIM

- Aim is to explore crime data in Chicago and showcase the implementation of a predictive model for arrests in Chicago. This could help the public institutions in 3 main ways:
- Better create public policy for correctional agencies
- Help focus the countermeasures on negatively impacted crime categories according to the prediction
- Guide the resource allocation by crime categories

Python Libraries Used

Data wrangling using Pandas.

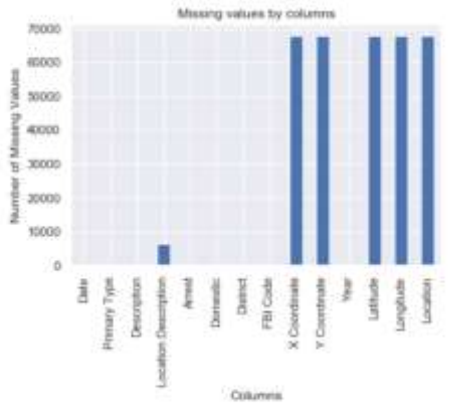
Data visualization using Matplotlib, Seaborn.

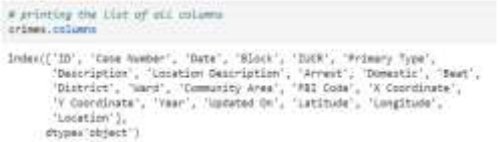
ML libraries Sklearn.

Miscellaneous: NumPy (for math operations, python datetime module)

All About the Dataset

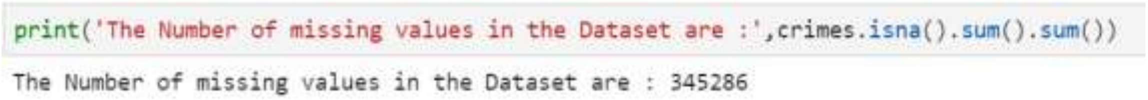
- 6 million Data Records
- Data from 2001-2019
- 22 columns
- 345286 missing values
- 92.5 % retained after dropping Null values.





```
# printing the list of all columns
crimes.columns
```

```
Index(['ID', 'Case Number', 'Date', 'Block', 'IUCR', 'Primary Type',
       'Description', 'Location Description', 'Arrest', 'Domestic', 'Beat',
       'District', 'Ward', 'Community Area', 'FBI Code', 'X Coordinate',
       'Y Coordinate', 'Year', 'Updated On', 'Latitude', 'Longitude',
       'Location'],
      dtype='object')
```



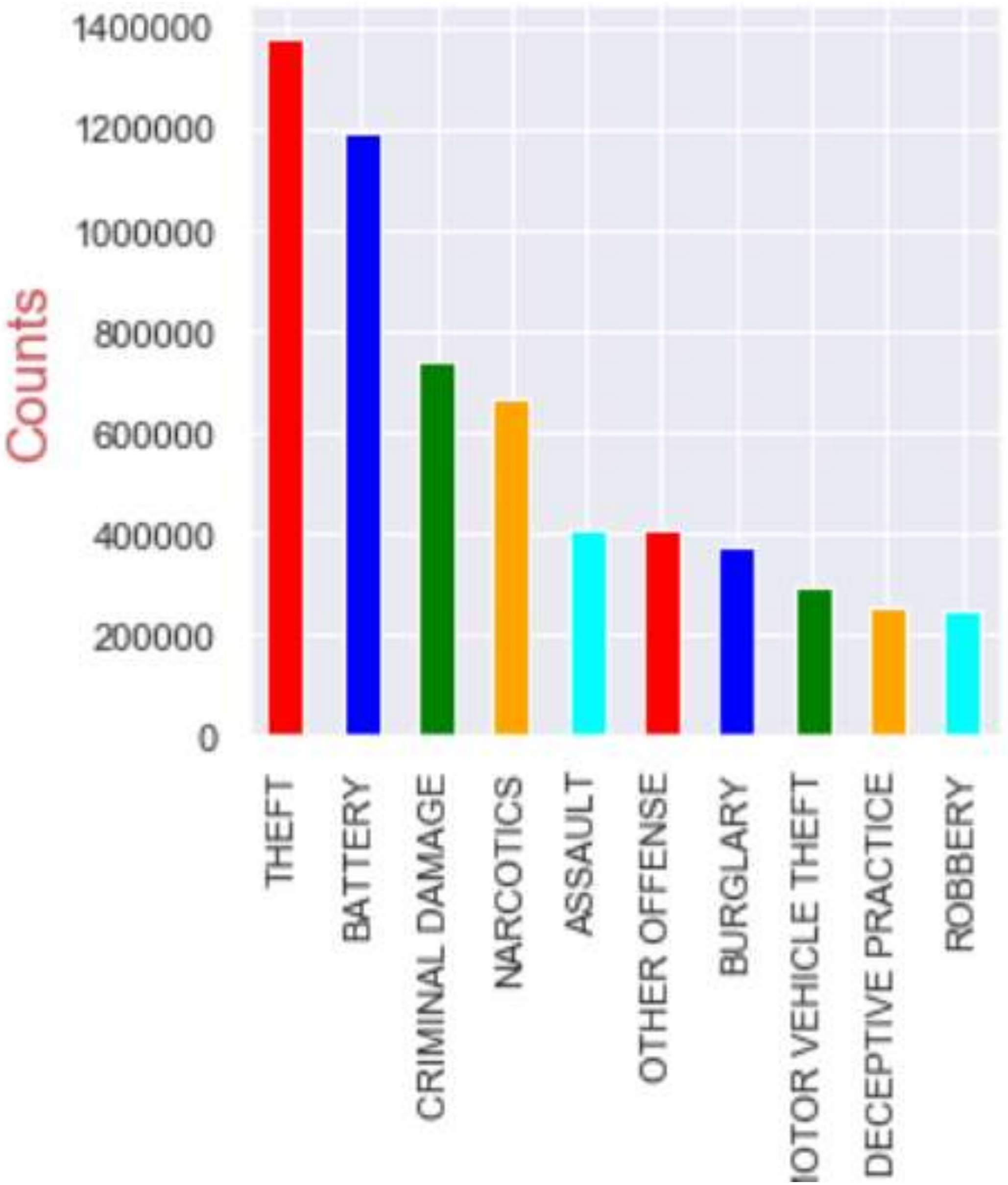
```
print('The Number of missing values in the Dataset are :',crimes.isna().sum().sum())
```

```
The Number of missing values in the Dataset are : 345286
```

Exploratory Data Analysis

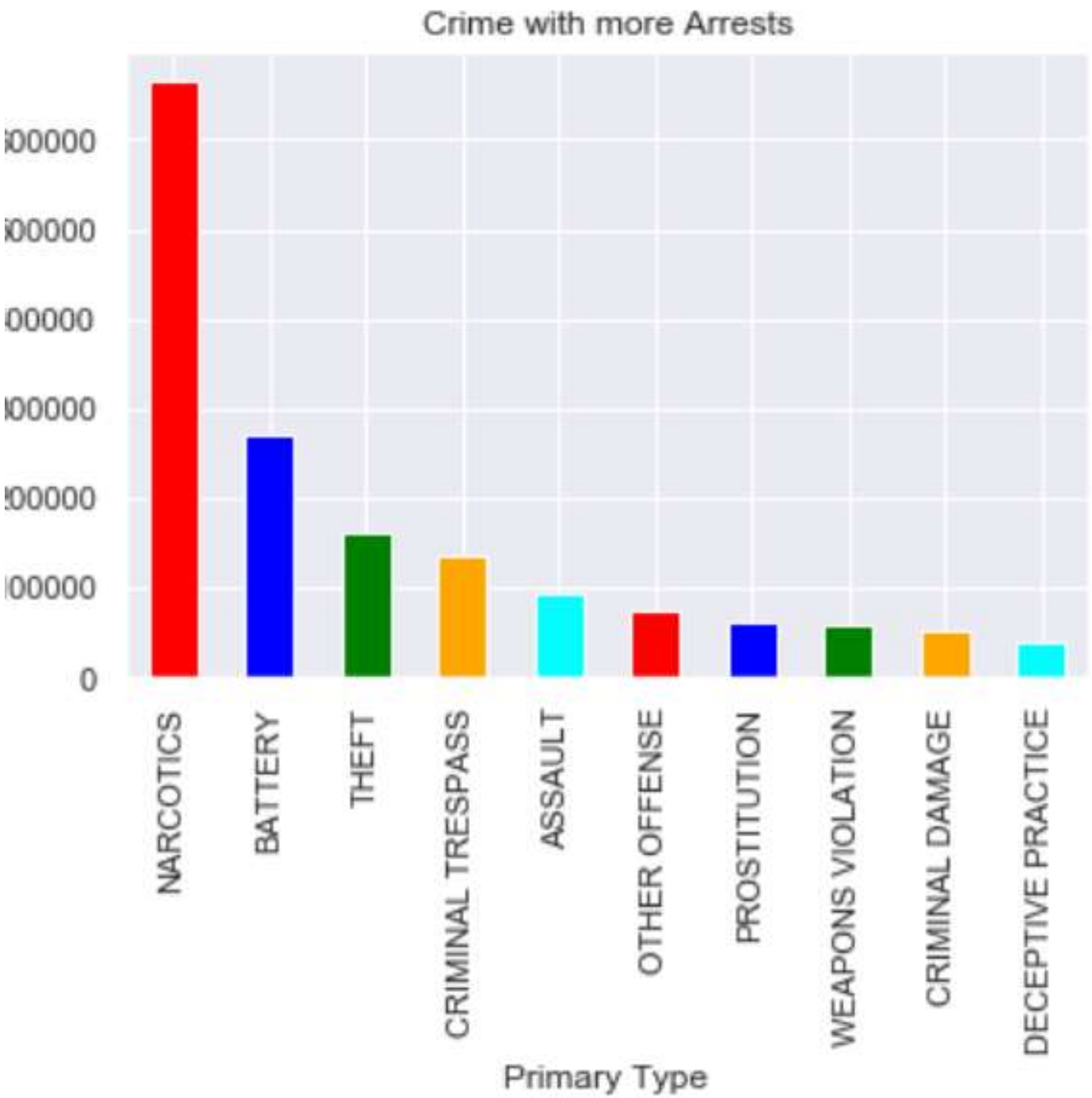
Top crimes

- Theft's were most occurring crimes with an account of 1.34 million
- High counts of Battery and assault indicate the presence of a physically violent community.

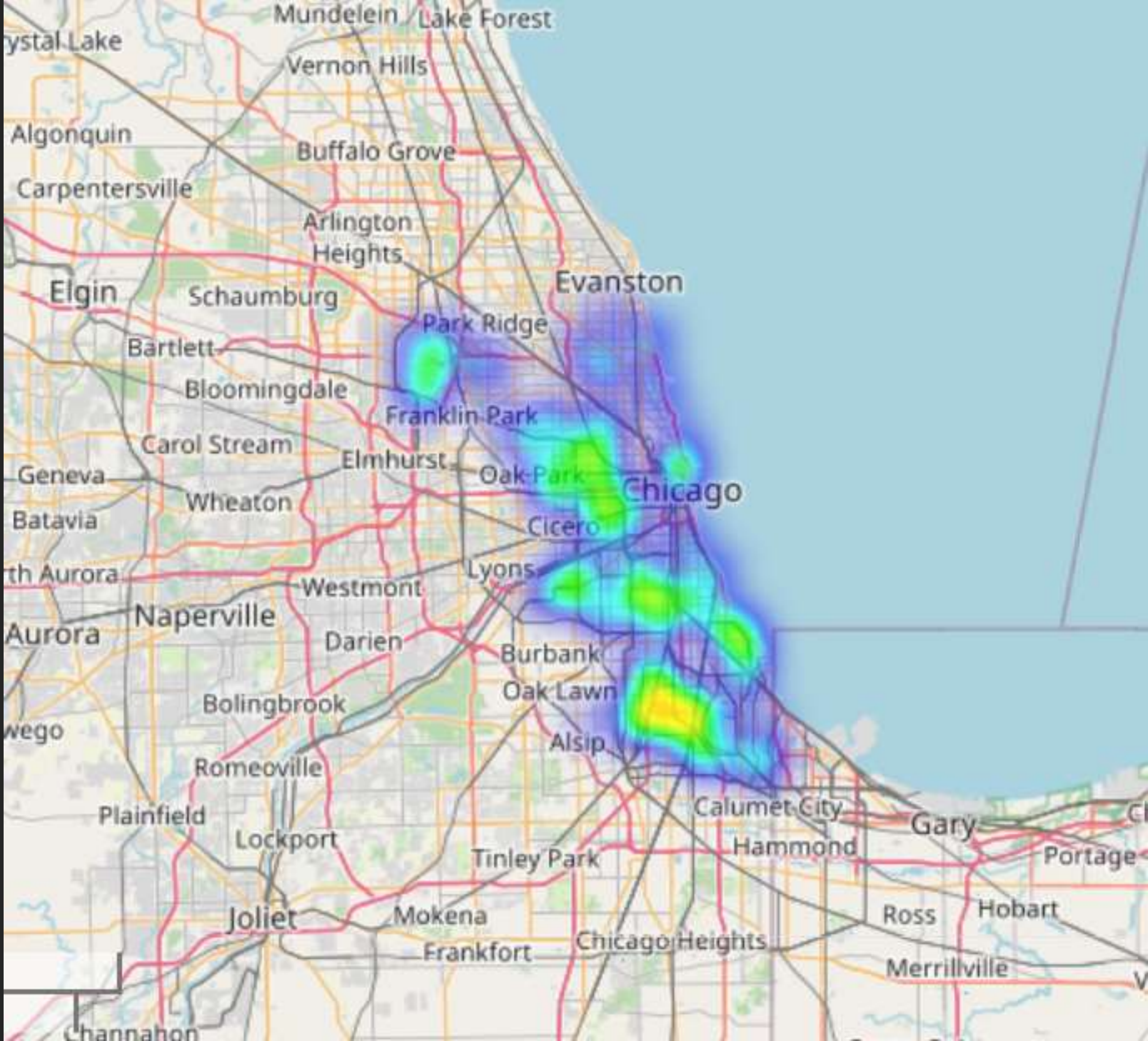


Arrest’s

- 80 percent of the crimes saw no Arrests.
- Most of the crimes seen more No Arrests .
- However it is good to see that ‘Narcotics’ has 99 percent arrest rate, Even the ‘Weapon Violation’ has good Arrest Rate

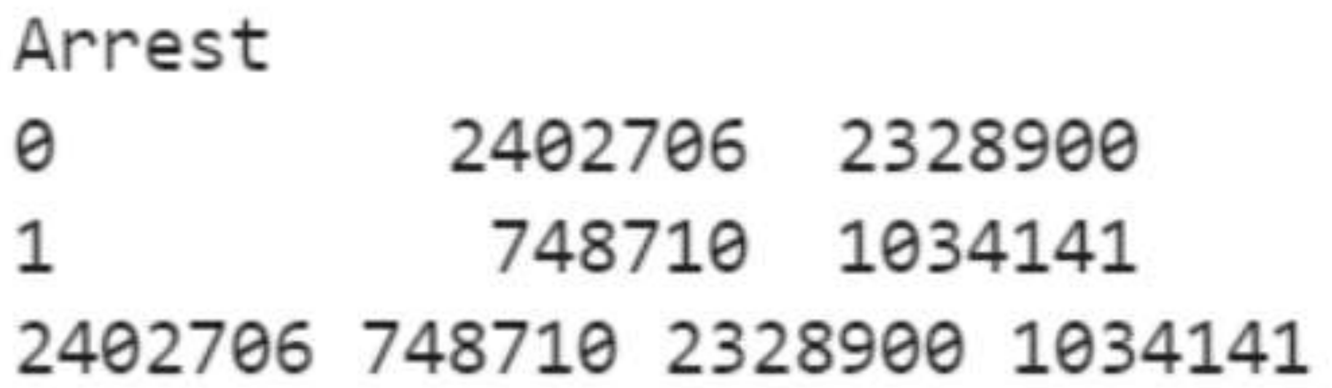


Heat Maps’of Arrest’s in 2019



Hypothesis Testing

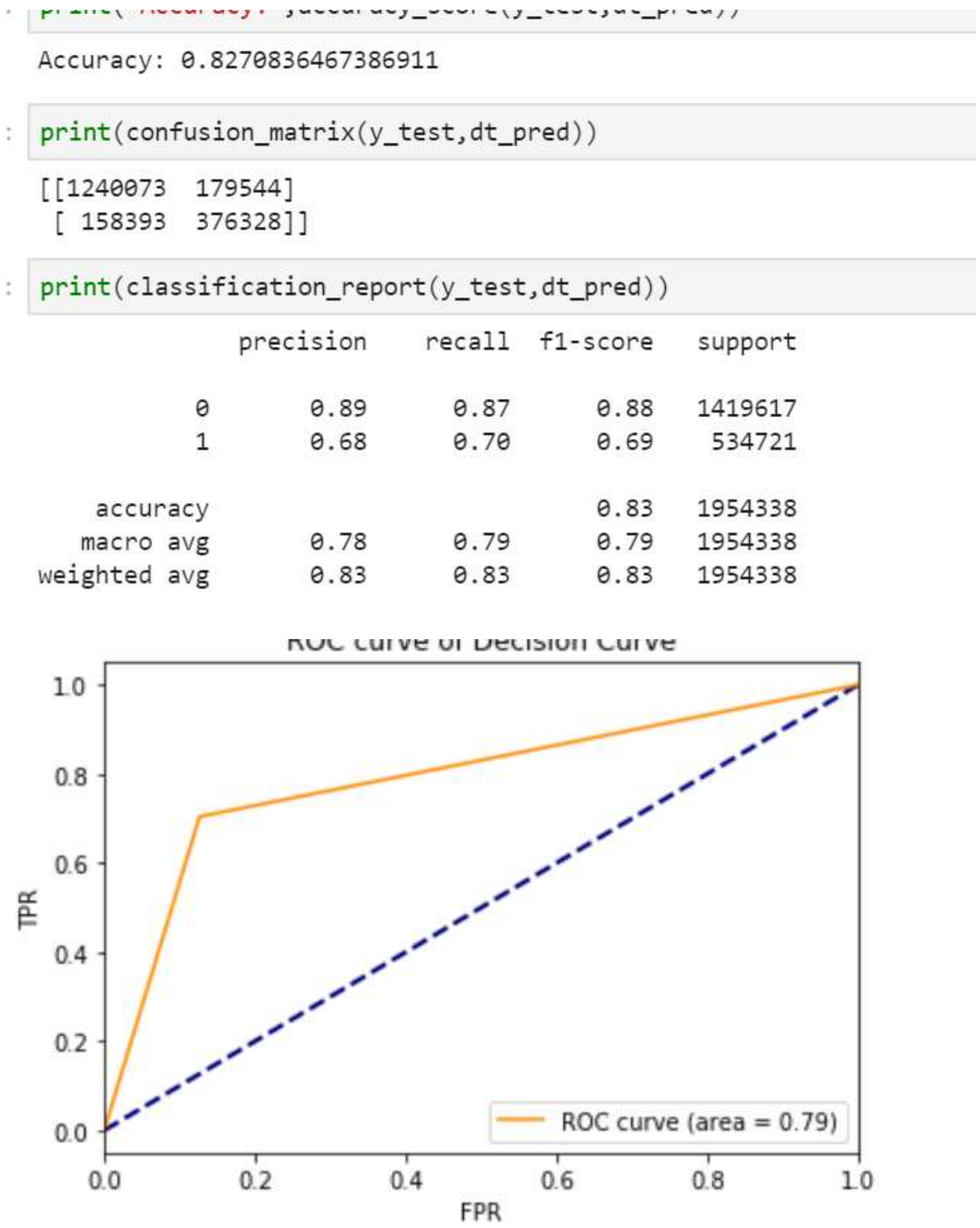
- **Hypothesis Testing :**
- H0 : There is no Association with arrests between the Majority blacks districts and others.
- H1 : There is an association with arrest between the districts.
- The districts 15.0,11.0,10.0,21.0,2.0,7.0,9.0,3.0,6.0,4.0,5.0 majority blacks populated districts, and the rest of the districts.
- Since p value is less 0.5 ,we reject null hypothesis
- There might be racial disparities among the blacks than others.
- Police may be more interested in arresting the blacks.



```
Arrest
0         2402706  2328900
1          748710  1034141
2402706 748710 2328900 1034141
```

Decision Tree

- Accuracy : 82 percent
- Area under the curve is 0.79



```
print('Accuracy:',accuracy_score(y_test,dt_pred))

Accuracy: 0.8270836467386911
```

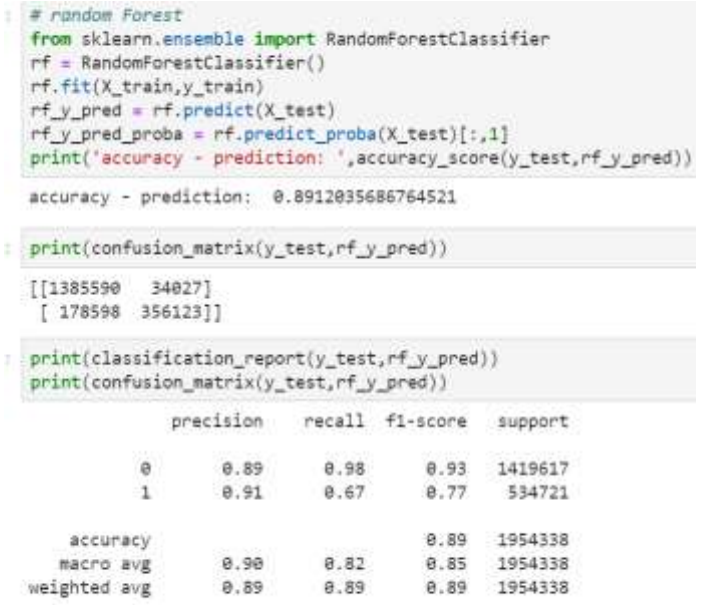
```
print(confusion_matrix(y_test,dt_pred))

[[1240073  179544]
 [ 158393  376328]]
```

```
print(classification_report(y_test,dt_pred))

              precision    recall  f1-score   support

           0       0.89      0.87      0.88   1419617
           1       0.68      0.70      0.69    534721

    accuracy                           0.83   1954338
   macro avg       0.78      0.79      0.79   1954338
weighted avg       0.83      0.83      0.83   1954338
```

ROC curve of Decision Curve

Random Forests

- Accuracy : 89 percent
- Area under the curve is 0.91



```
# random Forest
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(X_train,y_train)
rf_y_pred = rf.predict(X_test)
rf_y_pred_proba = rf.predict_proba(X_test)[:,1]
print('accuracy - prediction: ',accuracy_score(y_test,rf_y_pred))
```

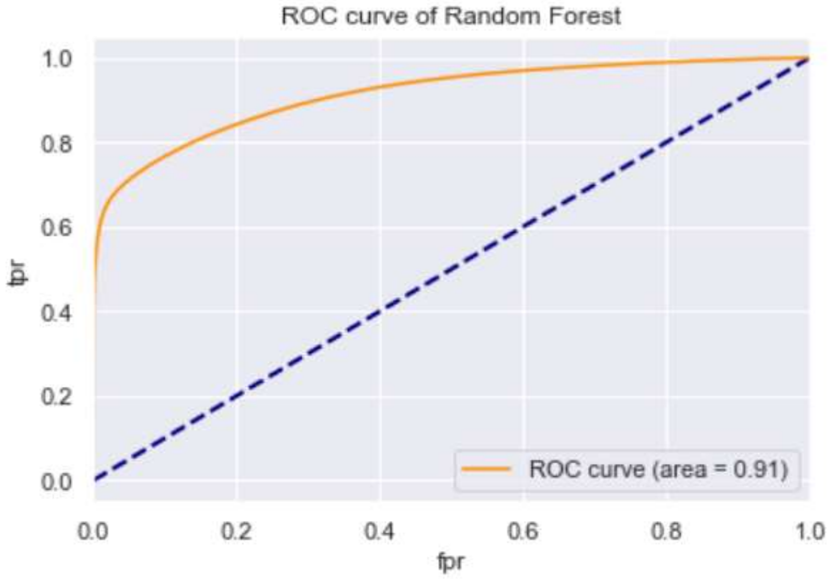
```
accuracy - prediction:  0.8912035686764521
```

```
print(confusion_matrix(y_test,rf_y_pred))
```

```
[[1385590   34027]
 [ 178598  356123]]
```

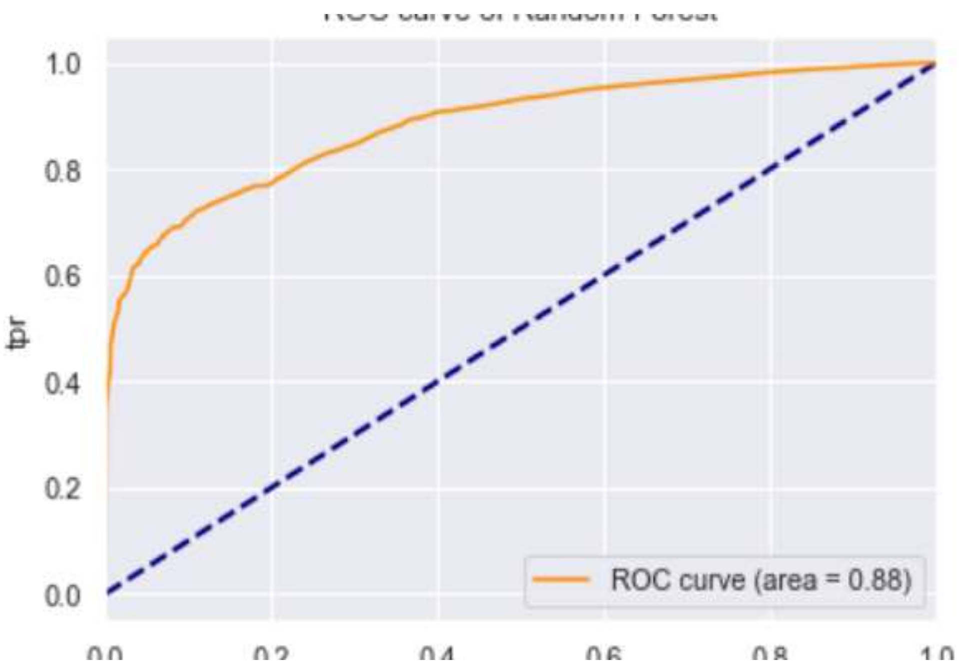
```
print(classification_report(y_test,rf_y_pred))
print(confusion_matrix(y_test,rf_y_pred))
```

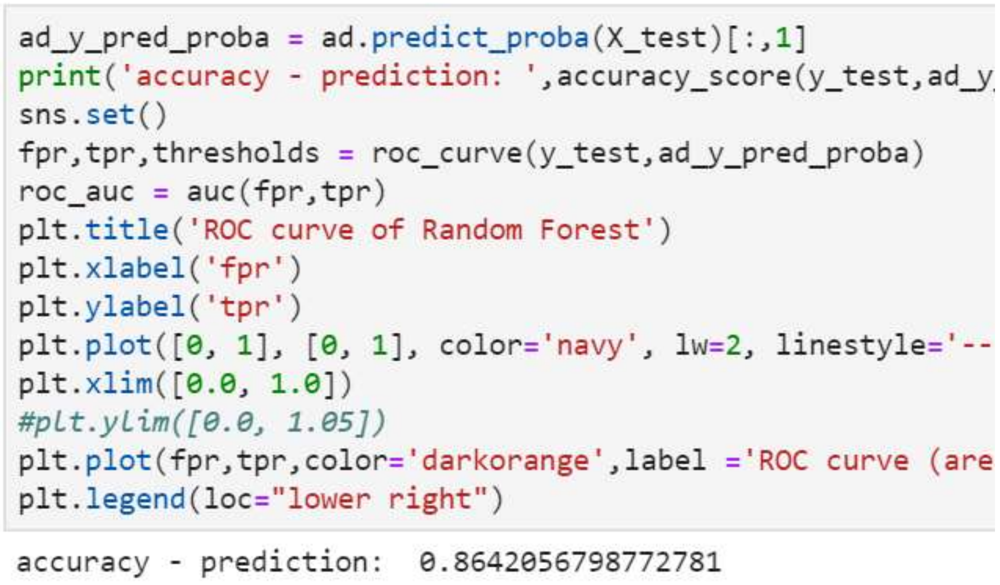
```
              precision    recall  f1-score   support

           0       0.89      0.98      0.93   1419617
           1       0.91      0.67      0.77    534721

    accuracy                           0.89   1954338
   macro avg       0.90      0.82      0.85   1954338
weighted avg       0.89      0.89      0.89   1954338
```



Adaboost

- Accuracy : 86 percent
- Roc curve 0.88



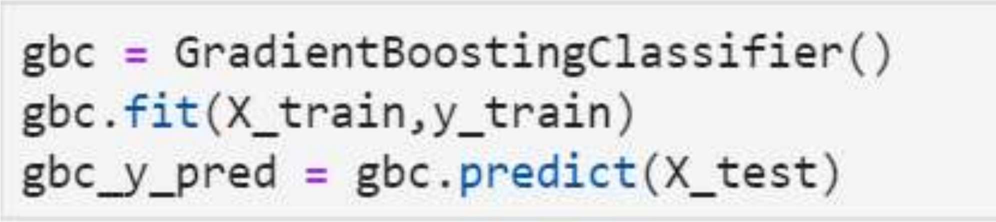


```
ad_y_pred_proba = ad.predict_proba(X_test)[:,1]
print('accuracy - prediction: ',accuracy_score(y_test,ad_y
sns.set()
fpr,tpr,thresholds = roc_curve(y_test,ad_y_pred_proba)
roc_auc = auc(fpr,tpr)
plt.title('ROC curve of Random Forest')
plt.xlabel('fpr')
plt.ylabel('tpr')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--
plt.xlim([0.0, 1.0])
#plt.ylim([0.0, 1.05])
plt.plot(fpr,tpr,color='darkorange',label ='ROC curve (are
plt.legend(loc="lower right")
```

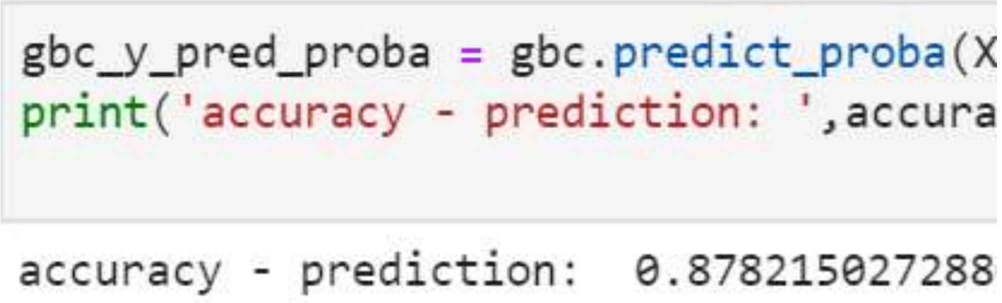
```
accuracy - prediction:  0.8642056798772781
```

Gradient Boosting Classifier

- Accuracy : 87 percent
- Area under the curve : 0.90

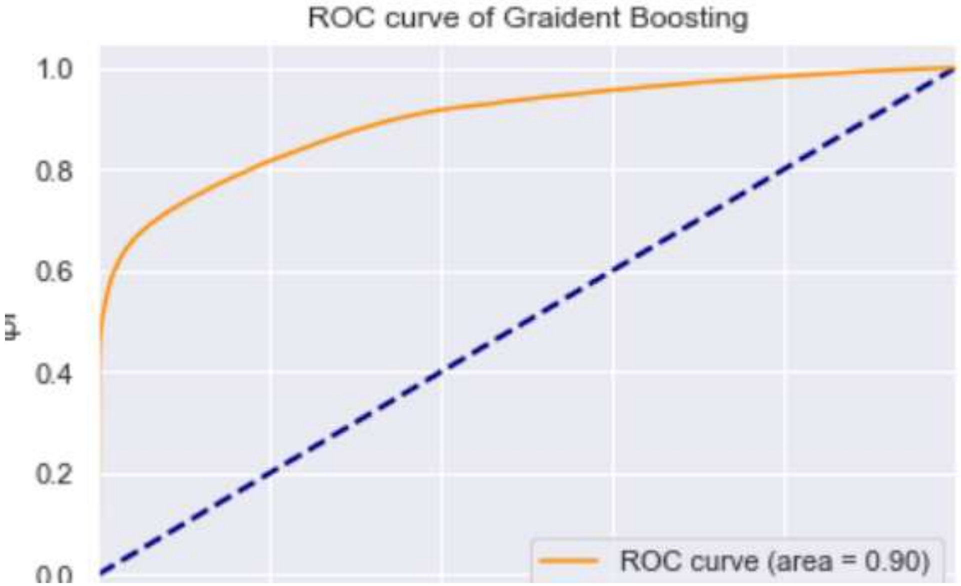


```
gbc = GradientBoostingClassifier()
gbc.fit(X_train,y_train)
gbc_y_pred = gbc.predict(X_test)
```



```
gbc_y_pred_proba = gbc.predict_proba(X
print('accuracy - prediction: ',accura
```

```
accuracy - prediction:  0.878215027288
```



ROC curve of Graident Boosting

Models used for prediction of Arrest

Models	**Accuracy**
Decision Tree	81 percent
Random Forest	89 percent
AdaBoost Classifier	86 percent
Gradient Boosting Classifier	87 percent

Conclusion

- By comparing all these models in terms of accuracy random forests keeps good in predictions, area under the curves are also pretty high which indicates high true positive rate.
- Accuracy is not the main thing that I have to calculate, based on the F1 scores, precision and recall values.
- Random forests model is the best for the prediction of Arrests.

Thank You