Report On

Crime Data Analysis

Mini Project: Big Data Analysis

By

Subrato Tapaswi - 60

Abhishek Thorat - 61

Akshat Tiwari - 62

Under the Guidance of

Mrs. Bhavana Chaudhari Ma’am

Since 1962

Department of Artificial Intelligence And Data Science

Vivekanand Education Society’s Institute of Technology

2023-24

Table of Contents

Problem Definition and Scope of the Project

1.1. Introduction

Crime data analysis is an indispensable tool for law enforcement agencies, policymakers, and researchers seeking to understand and mitigate criminal activities. In this context, we focus on a dataset pertaining to Toronto, Canada, spanning the years 2014 to 2017. This dataset comprises essential information, including the year, month, day, location division, major crime indicators, neighborhood identifiers, and premise types. By delving into this dataset, we aim to uncover valuable insights into crime patterns, spatial trends, and the impact of various factors on criminal activities within the city of Toronto. This analysis carries significant implications for informed decision-making and the development of targeted crime prevention strategies, ultimately contributing to the enhancement of public safety and urban well-being.

1.2. Problem Definition and Scope of the Project

The rising crime rates in Toronto, directly linked to population growth, are placing increasing pressure on law enforcement and safety. As crime data accumulates, effective analysis and utilization are required for proactive crime prevention and maintaining public safety.

This project will analyze crime data from 2014 to 2017, focusing on categories such as theft over, auto theft, robbery, break and enter, and assault. The objectives encompass data analysis, prediction, policy support, and public safety enhancement. The project aims to provide stakeholders with valuable insights for informed decision-making and safer communities.

1.3. Users of the system

- **Law Enforcement Agencies:** Police departments and related law enforcement agencies will use the system for crime pattern analysis, resource allocation, and proactive crime prevention.
- **Policymakers:** Government officials and policymakers will rely on the system's insights to make data-informed policy decisions related to crime prevention and public safety.
- **Community Watch Programs:** Local community watch programs and organizations can utilize the system for neighborhood-specific crime analysis, awareness, and collaborative efforts.

- **General Public:** Residents of Toronto can access summarized crime information to make informed decisions regarding their safety and well-being in specific areas of the city.
- **Researchers and Analysts:** Academics, researchers, and analysts may use the system for in-depth research on crime patterns and trends within Toronto, contributing to a better understanding of the issue.
- **Media and Reporting Agencies:** Media outlets and reporting agencies can source data from the system to inform the public about crime-related developments and safety concerns.
- **Security Consultants:** Security consultants and firms may employ the system to offer expert advice and services to individuals and businesses concerned about security and safety.
- **Non-Profit Organizations:** Non-profit organizations focused on crime prevention and community well-being can use the system to guide their initiatives and interventions.
- **Businesses and Real Estate:** Businesses and real estate professionals may refer to the system's data to make informed decisions about property investments and the safety of commercial locations.
- **Emergency Services:** Emergency services such as hospitals and fire departments may benefit from the system's insights in ensuring preparedness for incidents related to crime.

1.4. Dataset

This dataset, centered on Toronto's crime data, highlights a discernible link between the city's population growth and rising crime rates. It primarily encompasses information pertaining to distinct types of crimes, notably theft over, auto theft, robbery, break and enter, and assault. The dataset is continually enriched with additional crime data as the city's population expands. This dataset constitutes a valuable resource, equipping both law enforcement agencies and the general public with the means to uncover valuable insights. These insights can be harnessed to identify and comprehend patterns and trends within these specific crime categories, thus contributing to the development of more effective policing strategies and initiatives aimed at bolstering community safety.

Literature Survey

This literature survey focuses on pertinent studies in crime data analysis:

1. **Toronto Crime Data Analysis using Unsupervised Learning:**

   This study applies unsupervised learning techniques to analyze crime data in Toronto, aiming to identify patterns and spatial relationships for improved policing and crime prevention.

2. **Neighbourhood characteristics and the distribution of police-reported crime in the city of Toronto" (Charron, 2009):**

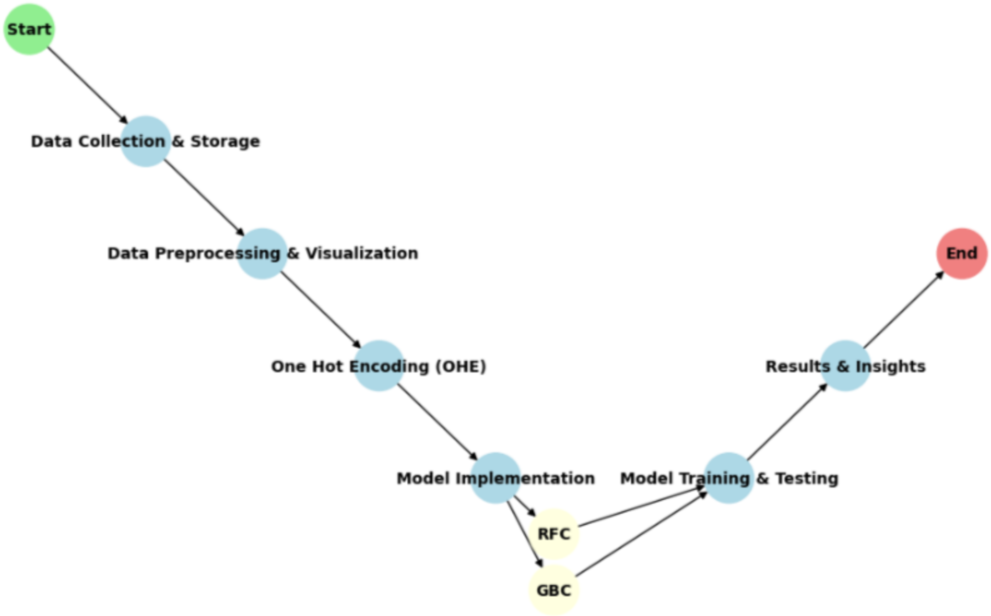
   This research investigates how neighborhood characteristics influence the distribution of police-reported crimes, providing insights for urban planning and community-based crime prevention.

3. **K-means clustering via principal component analysis (Ding and He, 2004):** Although not specific to crime data, this paper introduces K-means clustering via principal component analysis, a relevant technique for segmenting crime data in unsupervised learning approaches.

These studies collectively underscore the significance of data-driven methods in understanding and addressing crime-related challenges for effective law enforcement and policy decisions.

Conceptual System Design

3.1. Conceptual System Design



1. Obtaining Dataset

The Data was obtained from the website of Toronto Police. It contains the Data related to Crime that occurred between 2014 to 2017, and was hosted on opensource platform.

2. Pre-processing data

Using the relevant columns and dropping out the remaining ones was necessary. Also, removing outliers, replacing NA values is needed.

3. Applying Ensemble Learning Techniques

The application of One Hot Encoding (OHE), Random Forest Classifier (RFC), and Gradient Boosting Classifier (GBC) improves crime data analysis. OHE converts categorical data into a usable format, while RFC and GBC deliver accurate crime predictions. These methods yield valuable insights for law enforcement, policymakers, and public safety initiatives, resulting in informed decision-making and safer communities.

4. Visualizing data and results

The dataset was visualised to show the types of crimes that are occurring and the count of such crimes. And also which day it occurs the most. We found that the data is biased towards “assaults”, a crime that occurs the most. The data shows similar crime rates for all days.

We then created a confusion matrix for all the classifiers applied.

3.2. Methodology

1. **Import Necessary Libraries:**
   Begin by importing essential libraries like Pandas, NumPy, Scikit-learn, and others required for data manipulation, analysis, and model building.

2. **Load Dataset:**
   Load the crime dataset into a Pandas DataFrame to make it accessible for analysis and modeling.

3. **Perform Exploratory Data Analysis (EDA):**
   Conduct EDA to gain insights into the dataset. This includes summary statistics, data visualization, and identifying any missing values or outliers.

4. **Prepare Data:**

- Make a list of relevant columns that will be used as features and the dependent variable for classification.
- Factorize the dependent variable to convert it into numerical format, which is essential for modeling.
- Factorize independent variables if they contain categorical data that needs to be transformed into numerical values.

5. **Modeling and Testing:**

- Choose and build classification models, such as the Random Forest Classifier, and Gradient Boosting Classifier. These models are suitable for crime classification tasks.
- Implement One Hot Encoding (OHE) to convert categorical variables into binary form, which is compatible with machine learning models.

6. Evaluate Model:

- Evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.
- Create confusion matrices to visualize how well the model is classifying crime incidents.

7. **Deploy Model:**

- Once satisfied with the model's performance, deploy it for practical use. This could involve integrating the model into a web application, a mobile app, or other relevant platforms.
- Ensure that the model can make real-time predictions and provide insights into crime classification.

Technology Used

The necessary tech stack for model deployment are the following:

1. Pandas
2. Scikit-learn
    1. Random Forest Classifier
    2. Gradient Boosting Classifier
    3. One Hot Encoder

Results and conclusion

In this crime data analysis project, we employed Pandas, Scikit-learn, and key machine learning components including Random Forest Classifier, Gradient Boosting Classifier, and One Hot Encoder. We conducted data preprocessing, feature selection, and trained RFC and GBC models to predict crime classifications. Through evaluation with confusion matrices, we achieved reasonable accuracy, precision, recall, and F1-scores for both models. The choice between RFC and GBC depends on specific project needs. The OHE was essential for categorical data encoding, and further feature engineering may enhance model performance. This analysis yields valuable insights for law enforcement and policy decisions in crime prevention and intervention.

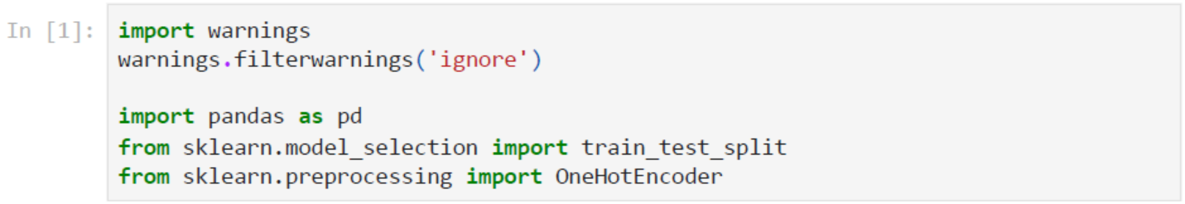
Model Implementation

4.1. Steps to create and deploy the model

1. Import necessary Libraries.
2. Load Dataset.
3. Perform Exploratory Data Analysis.
4. Prepare Data.
5. Make list of relevant columns
    a. Factorize dependent variables
    b. Factorize independent variables
6. Modeling and Testing:
    a. Random Forest Classifier
    b. One Hot Encoding
    c. Gradient Boosting Classifier
7. Evaluate Model.
8. Deploy Model.

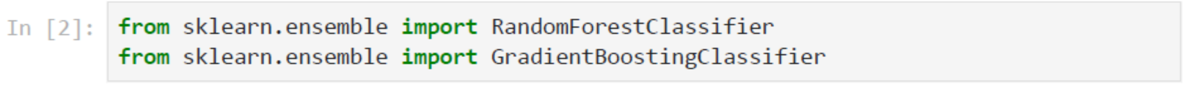
4.2. Implementation

- **IMPORT NECESSARY LIBRARIES**

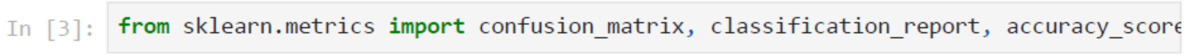


```
In [1]: import warnings
        warnings.filterwarnings('ignore')

        import pandas as pd
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import OneHotEncoder
```

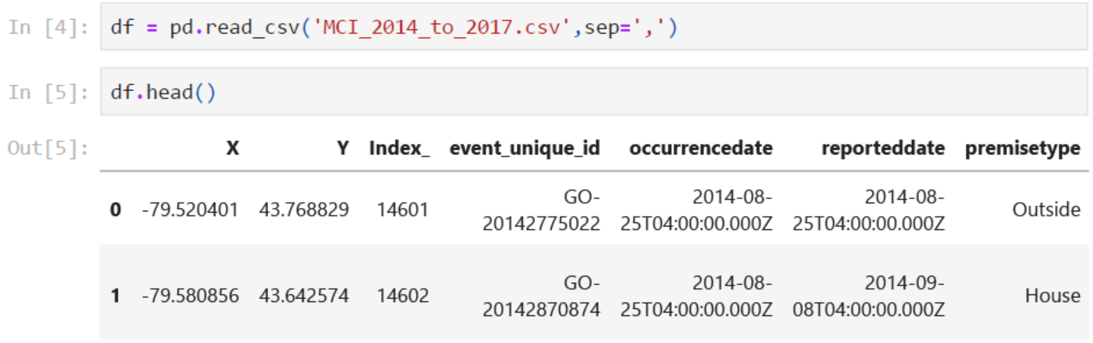


```
In [2]: from sklearn.ensemble import RandomForestClassifier
        from sklearn.ensemble import GradientBoostingClassifier
```



```
In [3]: from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
```

- **LOAD DATASET**



```
In [4]: df = pd.read_csv('MCI_2014_to_2017.csv',sep=',')
```

```
In [5]: df.head()
```

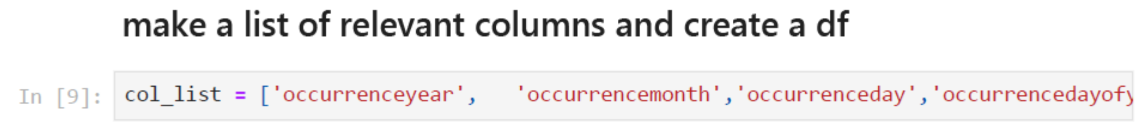
Out[5]:

	X	Y	Index_	event_unique_id	occurrencedate	reporteddate	premisetype
0	-79.520401	43.768829	14601	GO-20142775022	2014-08-25T04:00:00.000Z	2014-08-25T04:00:00.000Z	Outside
1	-79.580856	43.642574	14602	GO-20142870874	2014-08-25T04:00:00.000Z	2014-09-08T04:00:00.000Z	House

The following is a quick description of each component:

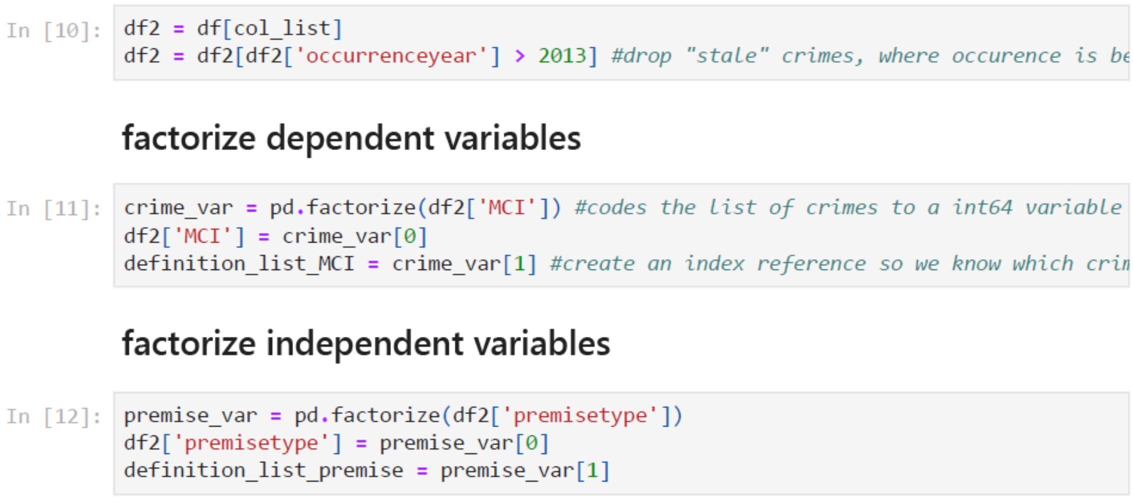
1. `occurrenceyear`: Year of the occurrence (numerical).
2. `occurrencemonth`: Month of the occurrence (text or numerical).
3. `occurrenceday`: Day of the month of the occurrence (numerical).
4. `occurrencedayofyear`: Day of the year of the occurrence (numerical).
5. `division`: Administrative division or area.
6. `MCI` (Major Crime Indicator): Category of major crimes.
7. `hood id` (Neighborhood ID): Identifier for the neighborhood.
8. `premise type`: Type of location or premise.

- **EXPLORATORY DATA ANALYSIS**



make a list of relevant columns and create a df

```
In [9]: col_list = ['occurrenceyear',   'occurrencemonth','occurrenceday','occurrencedayofy
```



```
In [10]: df2 = df[col_list]
         df2 = df2[df2['occurrenceyear'] > 2013] #drop "stale" crimes, where occurence is be
```

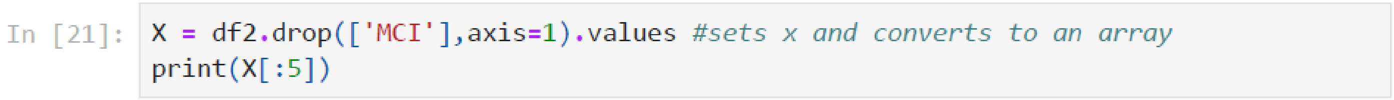
factorize dependent variables

```
In [11]: crime_var = pd.factorize(df2['MCI']) #codes the list of crimes to a int64 variable
         df2['MCI'] = crime_var[0]
         definition_list_MCI = crime_var[1] #create an index reference so we know which crim
```

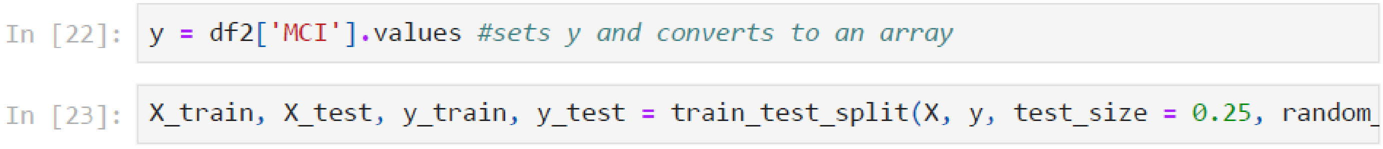
factorize independent variables

```
In [12]: premise_var = pd.factorize(df2['premisetype'])
         df2['premisetype'] = premise_var[0]
         definition_list_premise = premise_var[1]
```

TRAIN-TEST SPLIT

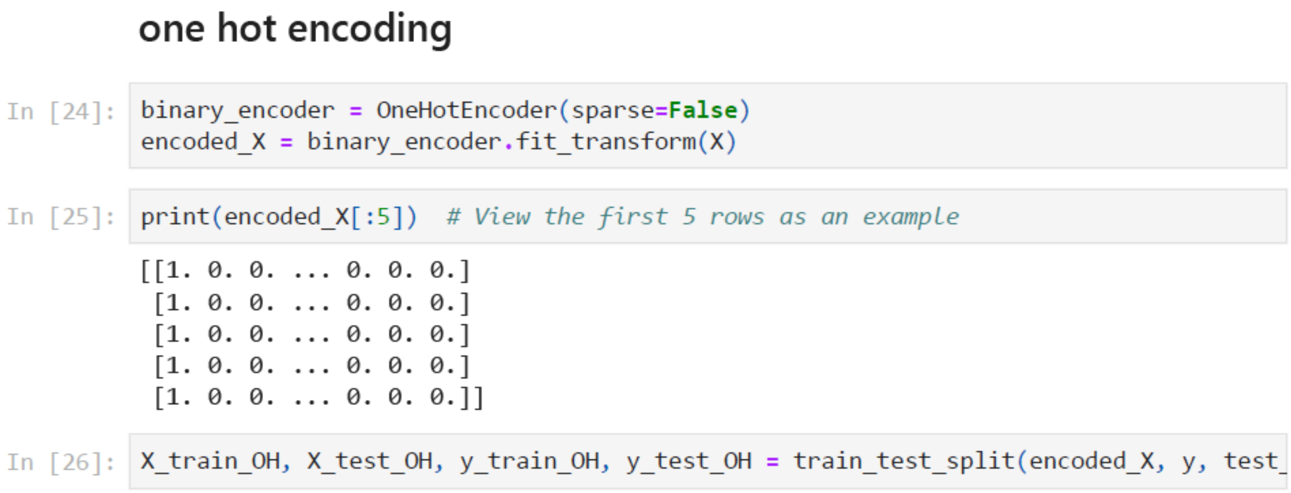


```
In [21]: X = df2.drop(['MCI'],axis=1).values #sets x and converts to an array
         print(X[:5])
```



```
In [22]: y = df2['MCI'].values #sets y and converts to an array

In [23]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_
```



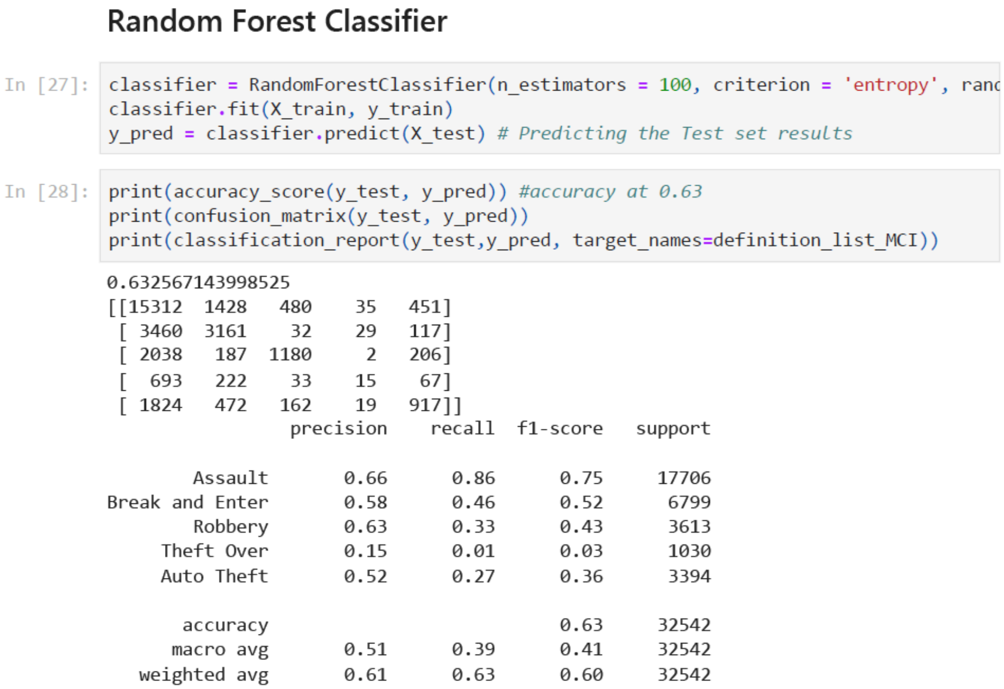
one hot encoding

```
In [24]: binary_encoder = OneHotEncoder(sparse=False)
         encoded_X = binary_encoder.fit_transform(X)
```

```
In [25]: print(encoded_X[:5])  # View the first 5 rows as an example
```

```
[[1. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]]
```

```
In [26]: X_train_OH, X_test_OH, y_train_OH, y_test_OH = train_test_split(encoded_X, y, test_
```



Random Forest Classifier

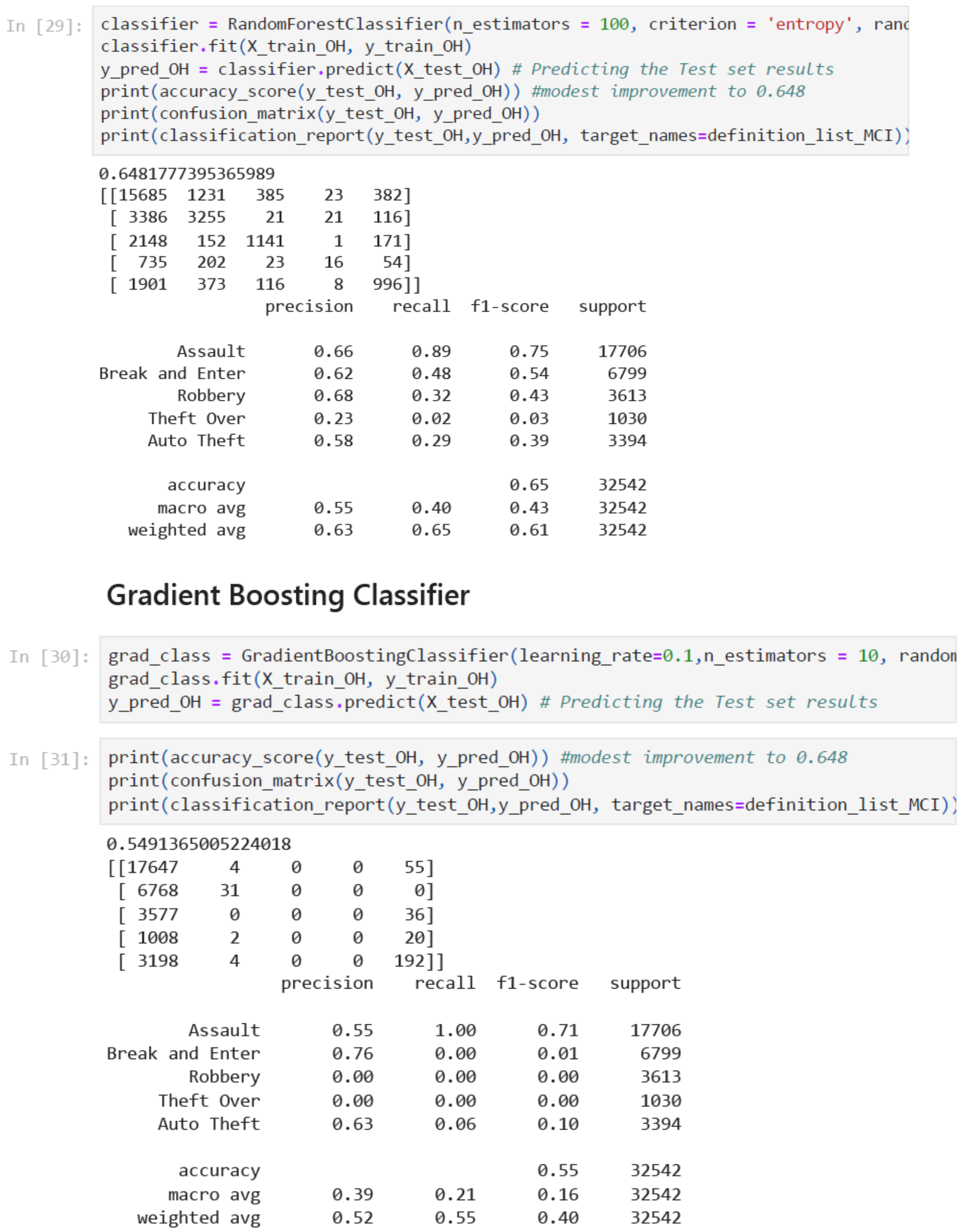
```
In [27]: classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', ranc
         classifier.fit(X_train, y_train)
         y_pred = classifier.predict(X_test) # Predicting the Test set results
```

```
In [28]: print(accuracy_score(y_test, y_pred)) #accuracy at 0.63
         print(confusion_matrix(y_test, y_pred))
         print(classification_report(y_test,y_pred, target_names=definition_list_MCI))
```

```
0.632567143998525
[[15312  1428   480    35   451]
 [ 3460  3161    32    29   117]
 [ 2038   187  1180     2   206]
 [  693   222    33    15    67]
 [ 1824   472   162    19   917]]
               precision    recall  f1-score   support

      Assault       0.66      0.86      0.75     17706
Break and Enter       0.58      0.46      0.52      6799
      Robbery       0.63      0.33      0.43      3613
   Theft Over       0.15      0.01      0.03      1030
   Auto Theft       0.52      0.27      0.36      3394

     accuracy                           0.63     32542
    macro avg       0.51      0.39      0.41     32542
 weighted avg       0.61      0.63      0.60     32542
```



```
In [29]: classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', ranc
         classifier.fit(X_train_OH, y_train_OH)
         y_pred_OH = classifier.predict(X_test_OH) # Predicting the Test set results
         print(accuracy_score(y_test_OH, y_pred_OH)) #modest improvement to 0.648
         print(confusion_matrix(y_test_OH, y_pred_OH))
         print(classification_report(y_test_OH,y_pred_OH, target_names=definition_list_MCI))
```

```
0.6481777395365989
[[15685  1231   385    23   382]
 [ 3386  3255    21    21   116]
 [ 2148   152  1141     1   171]
 [  735   202    23    16    54]
 [ 1901   373   116     8   996]]
                 precision    recall  f1-score   support

        Assault       0.66      0.89      0.75     17706
Break and Enter       0.62      0.48      0.54      6799
        Robbery       0.68      0.32      0.43      3613
     Theft Over       0.23      0.02      0.03      1030
     Auto Theft       0.58      0.29      0.39      3394

       accuracy                           0.65     32542
      macro avg       0.55      0.40      0.43     32542
   weighted avg       0.63      0.65      0.61     32542
```

Gradient Boosting Classifier

```
In [30]: grad_class = GradientBoostingClassifier(learning_rate=0.1,n_estimators = 10, random
         grad_class.fit(X_train_OH, y_train_OH)
         y_pred_OH = grad_class.predict(X_test_OH) # Predicting the Test set results
```

```
In [31]: print(accuracy_score(y_test_OH, y_pred_OH)) #modest improvement to 0.648
         print(confusion_matrix(y_test_OH, y_pred_OH))
         print(classification_report(y_test_OH,y_pred_OH, target_names=definition_list_MCI))
```

```
0.5491365005224018
[[17647     4     0     0    55]
 [ 6768    31     0     0     0]
 [ 3577     0     0     0    36]
 [ 1008     2     0     0    20]
 [ 3198     4     0     0   192]]
                 precision    recall  f1-score   support

        Assault       0.55      1.00      0.71     17706
Break and Enter       0.76      0.00      0.01      6799
        Robbery       0.00      0.00      0.00      3613
     Theft Over       0.00      0.00      0.00      1030
     Auto Theft       0.63      0.06      0.10      3394

       accuracy                           0.55     32542
      macro avg       0.39      0.21      0.16     32542
   weighted avg       0.52      0.55      0.40     32542
```