CS 5402 INTRODUCTION TO DATA MINING

Semester Project - DATA ANALYSIS OF CRIME DATA IN LOS ANGELES

By: Sumalika Kallu, Pujitha Nimmagadda, Himabindu Pandiri, Venkat Yadav Moolamalla

Concept Description

It would come as no surprise that crime is pervasive in Los Angeles. Owing to a large homeless population, gang activity, and high rates of violent crimes, certain parts of Los Angeles are known to be dangerous. This project aims to improve public safety in Los Angeles. By using the dataset, this project serves as a tool for Los Angeles Police Department to predict future crimes in order to take further actions. This data is transcribed from original crime reports released by the LAPD that are typed on paper and therefore there may be some inaccuracies within the data. Predictive analysis can be used for more efficient placement, scheduling, and routines of police officers, thus optimizing the patrolling in the Los Angeles region. Following are some references in regard to the seriousness of the crime activities in Los Angeles.

https://www.lamag.com/citythinkblog/crime-in-los-angeles/
https://losangeles.cbslocal.com/tag/crime/

Scenario and Research Questions

This project aims to build a predictive model to

Crime forecasting

Predictive analysis can be used for more efficient placement, scheduling, and routines of police officers, thus optimizing the patrolling in the Los Angeles region.

Gathering insights from data visualizations

Visualizations aid human perception and encourage

Research Questions

Some potential research questions that can be addressed include:

a) Which day of the week is most violent in LA?

b) Which year was the highest crime rate recorded in between 2010 and 2020?

c) What are the areas where crime is more likely to occur at a particular time?

d) What are the crimes that are more likely to occur?

Data Source and Collection

The dataset contains crime data in the Los Angeles area dated from 2010 to 2020. Following is the link to the dataset: https://www.kaggle.com/sumaiaparveenshupti/los-angeles-crime-data-20102020

Attributes Description

DR_NO: Division of Records Number: Official file number made up of a 2 digit year, area ID, and 5 digits. API Field Name: MM/DD/YYYY. DATE OCC: MM/DD/YYYY. TIME OCC: In 24 hour military time. AREA: The LAPD has 21 Community Police Stations referred to as Geographic Areas within the department. These Geographic Areas are sequentially numbered from 1-21. AREA NAME: The 21 Geographic Areas or Patrol Divisions are also given a name designation that references a landmark or the surrounding community that it is responsible for. For example 77th Street Division is located at the intersection of South Broadway and 77th Street, serving neighborhoods in South Los Angeles. Rpt Dist No: A four-digit code that represents a sub-area within a Geographic Area. All crime records reference the "RD" that it occurred in for statistical comparisons. Crm Cd: Indicates the crime committed. (Same as Crime Code 1) Crm Cd Desc: Defines the Crime Code provided. Mocodes: Modus Operandi: Activities associated with the suspect in commission of the crime. Vict Age: Two character numeric. Vict Sex: F - Female M - Male X - Unknown. Vict Descent: Descent Code: A - Other Asian B - Black C - Chinese D - Cambodian F - Filipino G - Guamanian H - Hispanic/Latin/Mexican I - American Indian/Alaskan Native J - Japanese K - Korean L - Laotian O - Other P - Pacific Islander S - Samoan U - Hawaiian V - Vietnamese W - White X - Unknown Z - Asian Indian. Premis Cd: The type of structure, vehicle, or location where the crime took place. Premis Desc: Defines the Premise Code provided. Weapon Used Cd: The type of weapon used in the crime. Weapon Desc: Defines the Weapon Used Code provided. Status: Status of the case. (IC is the default). Status DEsc: Defines the Status Code provided. Crm Cd 1: Indicates the crime committed. Crime Code 1 is the primary and most serious one. Crime Code 2, 3, and 4 are respectively less serious offenses. Lower crime class numbers are more serious. Crm Cd 2: May contain a code for an additional crime, less serious than Crime Code 1. Crm Cd 3: May contain a code for an additional crime, less serious than Crime Code 1. Crm Cd 4: May contain a code for an additional crime, less serious than Crime Code 1. LOCATION: Street address of crime incident rounded to the nearest hundred block to maintain anonymity. Cross Street: Cross Street of rounded Address. LAT: Latitude. LON: Longitude.

Attribute	Description
DR_NO	MM/DD/YYYY
API Field Name	MM/DD/YYYY
DATE OCC	MM/DD/YYYY

TIME OCC	In 24 hour military time
AREA	The LAPD has 21 Community Police Stations referred to as Geographic Areas within the department. These Geographic Areas are sequentially numbered from 1-21
AREA NAME	The 21 Geographic Areas or Patrol Divisions are also given a name designation that references a landmark or the surrounding community that it is responsible for. For example 77th Street Division is located at the intersection of South Broadway and 77th Street, serving neighborhoods in South Los Angeles
Rpt Dist No	A four-digit code that represents a sub-area within a Geographic Area. All crime records reference the "RD" that it occurred in for statistical comparisons
Crm Cd	Indicates the crime committed (Same as Crime Code 1)
Crm Cd Desc	Defines the Crime Code provided
Mocodes	Modus Operandi: Activities associated with the suspect in commission of the crime
Vict Age	Two character numeric
Vict Sex	F - Female M - Male X - Unknown
Vict Descent	Descent Code: A - Other Asian B - Black C - Chinese D - Cambodian F - Filipino G - Guamanian H - Hispanic/Latin/Mexican I - American Indian/Alaskan Native J - Japanese K - Korean L - Laotian O - Other P - Pacific Islander S - Samoan U - Hawaiian V - Vietnamese W - White X - Unknown Z - Asian Indian
Premis Cd	The type of structure, vehicle, or location where the crime took place
Weapon Used Cd	The type of weapon used in the crime
Weapon Desc	Defines the Weapon Used Code provided
Status	Status of the case. (IC is the default)
Status DEsc	Defines the Status Code provided
Crm Cd 1	Indicates the crime committed. Crime Code 1 is the primary and most serious one. Crime Code 2, 3, and 4 are respectively less serious offenses. Lower crime class numbers are more serious
Crm Cd 2	May contain a code for an additional crime, less

	serious than Crime Code 1
Crm Cd 3	May contain a code for an additional crime, less serious than Crime Code 1
Crm Cd 4	May contain a code for an additional crime, less serious than Crime Code 1
LOCATION	Street address of crime incident rounded to the nearest hundred block to maintain anonymity
Cross Street	Cross Street of rounded Address
LAT	Latitude
LON	Longitude

Data Import and Wrangling

The data being used in this project consists of two datasets. The first one consists of crime records dated from 2010 to 2019. The second one consists of crime records dated from 2020 to present.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
import plotly.graph_objects as go
from sklearn.preprocessing import scale
from sklearn.decomposition import PCA
import scipy as sp
import warnings
warnings.filterwarnings('ignore')

from sklearn import metrics
from sklearn.model_selection import train_test_split, cross_val_score,
GridSearchCV
from sklearn.metrics import mean_squared_error, r2_score,
accuracy_score, roc_auc_score
from sklearn.preprocessing import StandardScaler, MinMaxScaler,
LabelEncoder, OneHotEncoder
from sklearn.linear_model import LinearRegression, LogisticRegression,
Ridge, Lasso, ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.tree import export_graphviz
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
precision_score, recall_score, roc_auc_score, roc_curve, f1_score

from lightgbm import LGBMRegressor
```

```python
dat=pd.read_csv('Crime_Data_from_2010_to_2019.csv')
dat1=pd.read_csv('Crime_Data_from_2020_to_Present.csv')

pd.set_option("display.max_columns",None)
dat
crimes=[dat,dat1]
crimes=pd.concat(crimes)

crimes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2394173 entries, 0 to 276583
Data columns (total 29 columns):
 #   Column          Dtype  
---  ------          -----  
 0   DR_NO           int64  
 1   Date Rptd       object 
 2   DATE OCC        object 
 3   TIME OCC        int64  
 4   AREA            float64
 5   AREA NAME       object 
 6   Rpt Dist No     int64  
 7   Part 1-2        int64  
 8   Crm Cd          int64  
 9   Crm Cd Desc     object 
 10  Mocodes         object 
 11  Vict Age        int64  
 12  Vict Sex        object 
 13  Vict Descent    object 
 14  Premis Cd       float64
 15  Premis Desc     object 
 16  Weapon Used Cd  float64
 17  Weapon Desc     object 
 18  Status          object 
 19  Status Desc     object 
 20  Crm Cd 1        float64
 21  Crm Cd 2        float64
 22  Crm Cd 3        float64
 23  Crm Cd 4        float64
 24  LOCATION        object 
 25  Cross Street    object 
 26  LAT             float64
 27  LON             float64
 28  AREA            float64
dtypes: float64(10), int64(6), object(13)
memory usage: 548.0+ MB
```

```python
# Checking null values in the dataset
crimes.isnull().sum()
```

```
DR_NO                    0
Date Rptd                0
DATE OCC                 0
TIME OCC                 0
AREA                276584
AREA NAME                0
Rpt Dist No              0
Part 1-2                 0
Crm Cd                   0
Crm Cd Desc              0
Mocodes             266144
Vict Age                 0
Vict Sex            233088
Vict Descent        233139
Premis Cd               57
Premis Desc            284
Weapon Used Cd     1581707
Weapon Desc        1581708
Status                   3
Status Desc              0
Crm Cd 1                12
Crm Cd 2           2231485
Crm Cd 3           2389876
Crm Cd 4           2394044
LOCATION                 0
Cross Street       1988864
LAT                      0
LON                      0
AREA               2117589
dtype: int64
```

Data Cleansing

It might not be the best approach to remove the rows containing missing values if such rows are abundant. They might contain valuable data in other columns and we don't want to skew the data towards an inaccurate state. Areas like machine learning and data mining face severe issues in the accuracy of their model predictions because of poor quality of data caused by missing values. In this process, the missing values in the dataset have been dealt with using various functions offered by Pandas like isnull() and fillna() along with methods like 'bfill' and 'pad'. Missing data can occur when no information is provided for one or more items or for a whole unit.

```
#Dropping all unncessary columns as they will not be used for data
analysis
crimes_new = crimes
crimes_new.drop(columns = ['DR_NO','Date Rptd','AREA','Part 1-
2','Mocodes','Crm Cd 1','Crm Cd 2','Crm Cd 3','Crm Cd 4','Cross
Street','LOCATION'], axis = 1, inplace = True)
```

```
#Checking the null values in the dataset
crimes_new.isnull().sum()
```

```
DATE OCC                 0
TIME OCC                 0
AREA                276584
AREA NAME                0
Rpt Dist No              0
Crm Cd                   0
Crm Cd Desc              0
Vict Age                 0
Vict Sex            233088
Vict Descent        233139
Premis Cd               57
Premis Desc            284
Weapon Used Cd     1581707
Weapon Desc        1581708
Status                   3
Status Desc              0
LAT                      0
LON                      0
dtype: int64
```

We can see there are way too many Null values in the dataset which is a bane for data analysis part. The results that we want to project will not be accurate due to this issue. So dropping the null values in the dataset to get it more refined.

```
crimes_new.dropna(axis = 0, inplace = True)
crimes_new.reset_index(drop = True, inplace = True)
crimes_new.head()
```

```
                 DATE OCC  TIME OCC  AREA   AREA NAME  Rpt Dist No  
Crm Cd  \
0  01/05/2010 12:00:00 AM       150   6.0  Hollywood          646  
900
1  01/02/2010 12:00:00 AM      2100   1.0    Central          176  
122
2  01/08/2010 12:00:00 AM      2100   1.0    Central          157  
230
3  01/09/2010 12:00:00 AM       230   1.0    Central          171  
230
4  01/14/2010 12:00:00 AM      1445   1.0    Central          118  
624

                                      Crm Cd Desc  Vict Age Vict 
Sex  \
0                        VIOLATION OF COURT ORDER        47         F  

1                                 RAPE, ATTEMPTED        47         F  
```

```
2  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT       51        M

3  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT       30        M

4                        BATTERY - SIMPLE ASSAULT       38        F


  Vict Descent  Premis Cd    Premis Desc  Weapon Used Cd  \
0            W      101.0         STREET           102.0
1            H      103.0          ALLEY           400.0
2            B      710.0  OTHER PREMISE           500.0
3            H      108.0    PARKING LOT           400.0
4            B      101.0         STREET           400.0

                                      Weapon Desc Status   Status Desc
\
0                                        HAND GUN     IC   Invest Cont

1  STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)     IC   Invest Cont

2                     UNKNOWN WEAPON/OTHER WEAPON     AA  Adult Arrest

3  STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)     IC   Invest Cont

4  STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)     IC   Invest Cont


       LAT       LON
0  34.1016 -118.3295
1  34.0387 -118.2488
2  34.0435 -118.2427
3  34.0450 -118.2640
4  34.0640 -118.2375
```

Some columns can be removed if they are not required in the further analysis. The 'Date Rptd' column has been removed as its values are similar to the 'DATE OCC' column values. The 'AREA' column is a duplicate column and hence has been removed to avoid redundancy. No information about the 'Part 1-2' attribute has been given in the description of the dataset. The 'Crm Cd 3' and 'Crm Cd 4' columns mostly have null values (NaN values) and thus have been removed.

```
#Checking for null values after cleansing
crimes_new.isnull().sum()
```

```
DATE OCC          0
TIME OCC          0
AREA              0
AREA NAME         0
Rpt Dist No       0
```

```
Crm Cd            0
Crm Cd Desc       0
Vict Age          0
Vict Sex          0
Vict Descent      0
Premis Cd         0
Premis Desc       0
Weapon Used Cd    0
Weapon Desc       0
Status            0
Status Desc       0
LAT               0
LON               0
dtype: int64
```

Extracting the date,day name,month and year from DATE OCC column which will help analyze the data more effectively.

```
crimes_new['DATE OCC'] = pd.to_datetime(crimes_new['DATE OCC'])
crimes_new['Year'] = crimes_new['DATE OCC'].dt.year
crimes_new['Month'] = crimes_new['DATE OCC'].dt.month
crimes_new['Day'] = crimes_new['DATE OCC'].dt.day
crimes_new['TIME OCC'] = crimes_new['TIME 
OCC'].astype(str).str.zfill(4)
crimes_new['HOUR OCC'] = crimes_new['TIME OCC'].apply(lambda t: 
int(t[:2]))
```

```
crimes_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 711135 entries, 0 to 711134
Data columns (total 22 columns):
 #   Column          Non-Null Count   Dtype         
---  ------          --------------   -----         
 0   DATE OCC        711135 non-null  datetime64[ns]
 1   TIME OCC        711135 non-null  object        
 2   AREA            711135 non-null  float64       
 3   AREA NAME       711135 non-null  object        
 4   Rpt Dist No     711135 non-null  int64         
 5   Crm Cd          711135 non-null  int64         
 6   Crm Cd Desc     711135 non-null  object        
 7   Vict Age        711135 non-null  int64         
 8   Vict Sex        711135 non-null  object        
 9   Vict Descent    711135 non-null  object        
 10  Premis Cd       711135 non-null  float64       
 11  Premis Desc     711135 non-null  object        
 12  Weapon Used Cd  711135 non-null  float64       
 13  Weapon Desc     711135 non-null  object        
 14  Status          711135 non-null  object        
 15  Status Desc     711135 non-null  object        
 16  LAT             711135 non-null  float64       
```

```
 17  LON             711135 non-null  float64
 18  Year            711135 non-null  int64
 19  Month           711135 non-null  int64
 20  Day             711135 non-null  int64
 21  HOUR OCC        711135 non-null  int64
dtypes: datetime64[ns](1), float64(5), int64(7), object(9)
memory usage: 119.4+ MB
```

We can see the new columns Year, Month, Day have been created after separation from the DATE OCC column

```
#Arranging columns containing numeric values to show in the beginning
of the dataframe.
crimes_new_desc = crimes_new[['Crm Cd Desc', 'Premis Desc', 'Weapon
Desc', 'Status Desc']]
crimes_new = crimes_new[['DATE OCC','Year', 'Month', 'Day', 'TIME
OCC', 'AREA ', 'AREA NAME', 'Crm Cd',
                         'Vict Age', 'Vict Sex', 'Vict Descent',
'Premis Cd', 'Weapon Used Cd', 'Status']]

crimes_new.head()

    DATE OCC  Year  Month  Day TIME OCC  AREA   AREA NAME  Crm Cd
Vict Age  \
0 2010-01-05  2010      1    5     0150    6.0  Hollywood     900
47
1 2010-01-02  2010      1    2     2100    1.0    Central     122
47
2 2010-01-08  2010      1    8     2100    1.0    Central     230
51
3 2010-01-09  2010      1    9     0230    1.0    Central     230
30
4 2010-01-14  2010      1   14     1445    1.0    Central     624
38

  Vict Sex Vict Descent  Premis Cd  Weapon Used Cd Status
0        F            W      101.0           102.0     IC
1        F            H      103.0           400.0     IC
2        M            B      710.0           500.0     AA
3        M            H      108.0           400.0     IC
4        F            B      101.0           400.0     IC

#Converting column names to upper case for better readability
crimes_new.columns = map(str.upper, crimes_new.columns)

crimes_new.head()

    DATE OCC  YEAR  MONTH  DAY TIME OCC  AREA   AREA NAME  CRM CD
VICT AGE  \
0 2010-01-05  2010      1    5     0150    6.0  Hollywood     900
47
```

```
1 2010-01-02  2010      1    2     2100    1.0    Central     122
47
2 2010-01-08  2010      1    8     2100    1.0    Central     230
51
3 2010-01-09  2010      1    9     0230    1.0    Central     230
30
4 2010-01-14  2010      1   14     1445    1.0    Central     624
38

  VICT SEX VICT DESCENT  PREMIS CD  WEAPON USED CD STATUS
0        F            W      101.0           102.0     IC
1        F            H      103.0           400.0     IC
2        M            B      710.0           500.0     AA
3        M            H      108.0           400.0     IC
4        F            B      101.0           400.0     IC
```

Further transforming the cleansed data by adding Crime Code Descrption in the dataset. The Crime Code description will help us understand what particular crime was committed and what charges did the police place on the accused

```
crimes_new.insert(8,'CRM CD DESC',crimes_new_desc['Crm Cd Desc'])
crimes_new = crimes_new[['DATE OCC','YEAR', 'MONTH', 'DAY', 'TIME
OCC', 'AREA ', 'AREA NAME', 'CRM CD','CRM CD DESC',
                         'VICT AGE', 'VICT SEX', 'VICT DESCENT',
'PREMIS CD', 'WEAPON USED CD', 'STATUS']]

crimes_new.head()

    DATE OCC  YEAR  MONTH  DAY TIME OCC  AREA   AREA NAME  CRM CD  \
0 2010-01-05  2010      1    5     0150    6.0  Hollywood     900
1 2010-01-02  2010      1    2     2100    1.0    Central     122
2 2010-01-08  2010      1    8     2100    1.0    Central     230
3 2010-01-09  2010      1    9     0230    1.0    Central     230
4 2010-01-14  2010      1   14     1445    1.0    Central     624

                                      CRM CD DESC  VICT AGE VICT
SEX  \
0                         VIOLATION OF COURT ORDER        47         F

1                                  RAPE, ATTEMPTED        47         F

2  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT        51         M

3  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT        30         M

4                         BATTERY - SIMPLE ASSAULT        38         F


  VICT DESCENT  PREMIS CD  WEAPON USED CD STATUS
0            W      101.0           102.0     IC
```

```
1              H      103.0            400.0     IC
2              B      710.0            500.0     AA
3              H      108.0            400.0     IC
4              B      101.0            400.0     IC
```

```
#Saving the cleansed dataset into a new csv file to reduce computational power
crimes_new.describe()
crimes_new.to_csv('crimes_new.csv',index = False)
```

This new saved dataset will be used for data analysis and Machine Learning modelling.

Data Exploration

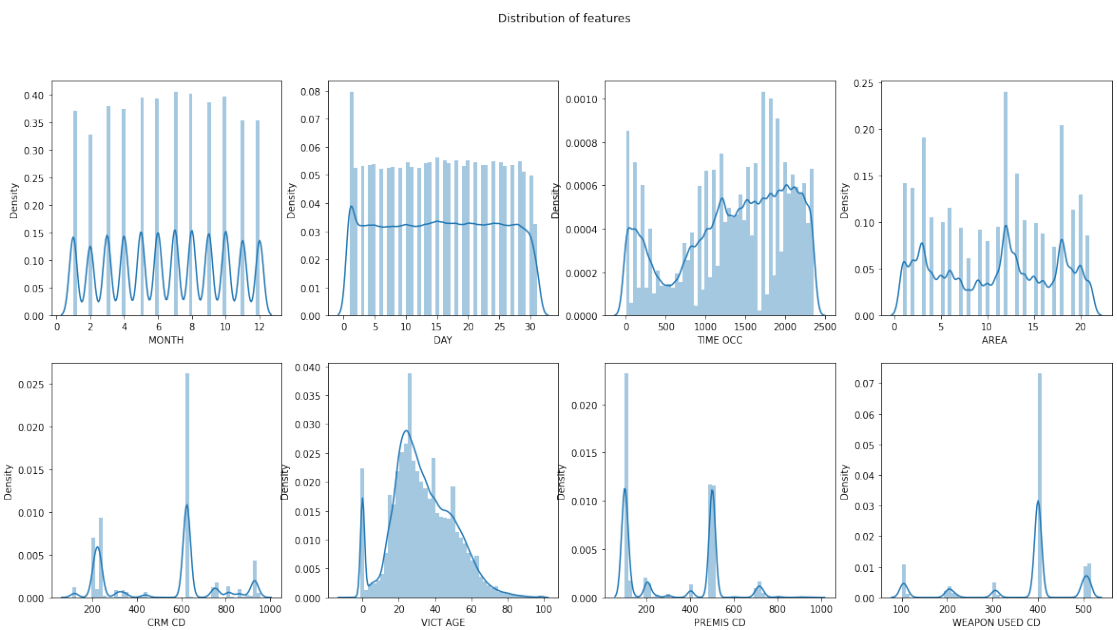
Now we will do check the distribution of key features in the dataset

```
#Checking distribution of each feature
crimes_new_distribution = crimes_new.iloc[:, [2, 3, 4, 5, 7, 9, 12, 13]]
crimes_new_distribution
```

```
        MONTH  DAY TIME OCC  AREA   CRM CD  VICT AGE  PREMIS CD  \
0           1    5     0150   6.0      900        47      101.0
1           1    2     2100   1.0      122        47      103.0
2           1    8     2100   1.0      230        51      710.0
3           1    9     0230   1.0      230        30      108.0
4           1   14     1445   1.0      624        38      101.0
...       ...  ...      ...   ...      ...       ...        ...
711130      1   20     2000  18.0      930        18      108.0
711131      2   23     2220   9.0      210        30      101.0
711132      2   22     0840   5.0      627        14      109.0
711133      3   28     0400   6.0      648         0      706.0
711134      1    6     2100  20.0      930        46      102.0

        WEAPON USED CD
0                102.0
1                400.0
2                500.0
3                400.0
4                400.0
...                ...
711130           511.0
711131           107.0
711132           400.0
711133           506.0
711134           400.0

[711135 rows x 8 columns]
```

```python
fig, axs = plt.subplots(nrows = 2, ncols = 4, figsize=(20,10))

for i, feature in enumerate(crimes_new_distribution.columns):
    row = int(i/4)
    col = i % 4
    sns.distplot(crimes_new_distribution.iloc[:,i], ax = axs[row]
[col])

plt.suptitle("Distribution of features")
plt.tight_layout
```

```
<function matplotlib.pyplot.tight_layout(*, pad=1.08, h_pad=None,
w_pad=None, rect=None)>
```



We can see that the features in the crimes are less biased as per the plots. But the features 'CRM CD', 'PREMIS CD' and 'WEAPON USED CD' need to be transformed.

```python
Crm_Cd_log = np.log1p(crimes_new['CRM CD'])
Premis_Cd_log = np.log1p(crimes_new['PREMIS CD'])
Weapon_used_Cd_log = np.log1p(crimes_new['WEAPON USED CD'])

crimes_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 711135 entries, 0 to 711134
Data columns (total 15 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE OCC        711135 non-null  datetime64[ns]
 1   YEAR            711135 non-null  int64
 2   MONTH           711135 non-null  int64
 3   DAY             711135 non-null  int64
```

```
 4   TIME OCC        711135 non-null  object
 5   AREA            711135 non-null  float64
 6   AREA NAME       711135 non-null  object
 7   CRM CD          711135 non-null  int64
 8   CRM CD DESC     711135 non-null  object
 9   VICT AGE        711135 non-null  int64
 10  VICT SEX        711135 non-null  object
 11  VICT DESCENT    711135 non-null  object
 12  PREMIS CD       711135 non-null  float64
 13  WEAPON USED CD  711135 non-null  float64
 14  STATUS          711135 non-null  object
dtypes: datetime64[ns](1), float64(3), int64(5), object(6)
memory usage: 81.4+ MB
```

```
crimes_new.insert(8, 'CRM CD LOG', Crm_Cd_log)
```

```
crimes_new.insert(12,'PREMIS LOG', Premis_Cd_log)
```

```
crimes_new.insert(14,'WEAPON LOG', Weapon_used_Cd_log)
```

```
crimes_new = crimes_new[['DATE OCC','YEAR','MONTH','DAY','TIME
OCC','AREA ','AREA NAME','CRM CD','CRM CD DESC','CRM CD LOG','VICT
AGE','VICT DESCENT',
                        'VICT SEX','PREMIS CD','PREMIS LOG','WEAPON
USED CD','WEAPON LOG','STATUS']]
```

```
crimes_new.head()
```

```
    DATE OCC  YEAR  MONTH  DAY TIME OCC  AREA   AREA NAME  CRM CD  \
0 2010-01-05  2010      1    5     0150    6.0  Hollywood     900
1 2010-01-02  2010      1    2     2100    1.0    Central     122
2 2010-01-08  2010      1    8     2100    1.0    Central     230
3 2010-01-09  2010      1    9     0230    1.0    Central     230
4 2010-01-14  2010      1   14     1445    1.0    Central     624

                                      CRM CD DESC  CRM CD LOG  VICT
AGE  \
0                         VIOLATION OF COURT ORDER    6.803505
47
1                                  RAPE, ATTEMPTED    4.812184
47
2  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT    5.442418
51
3  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT    5.442418
30
4                         BATTERY - SIMPLE ASSAULT    6.437752
38

  VICT DESCENT VICT SEX  PREMIS CD  PREMIS LOG  WEAPON USED CD  WEAPON
LOG  \
0            W        F      101.0    4.624973           102.0
```

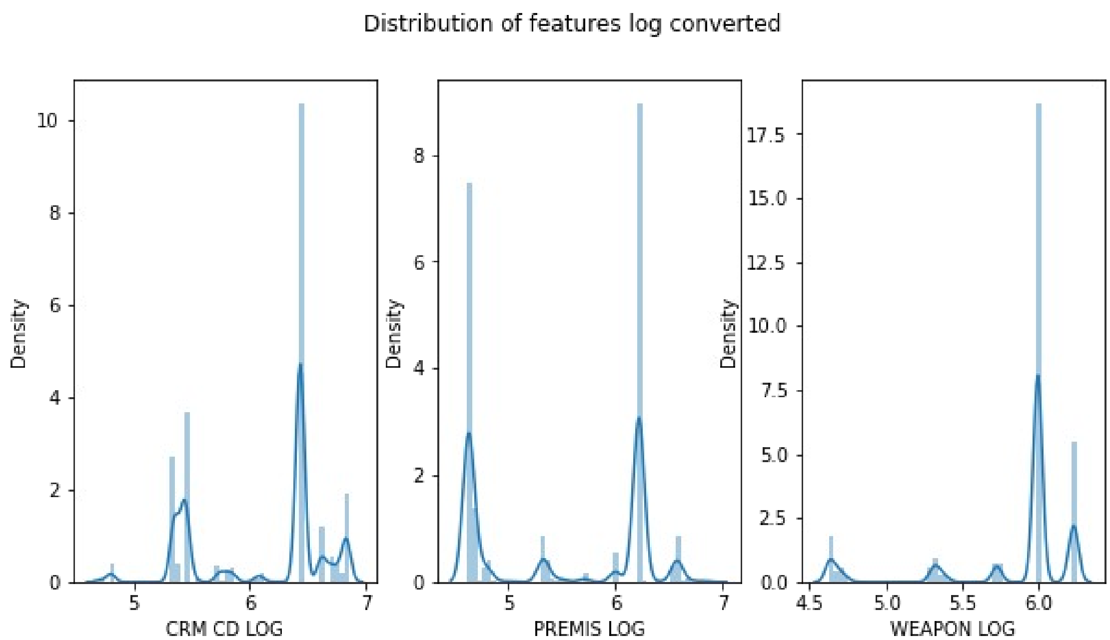
```
4.634729
1            H        F      103.0    4.644391          400.0
5.993961
2            B        M      710.0    6.566672          500.0
6.216606
3            H        M      108.0    4.691348          400.0
5.993961
4            B        F      101.0    4.624973          400.0
5.993961

  STATUS
0     IC
1     IC
2     AA
3     IC
4     IC
```

```
#Checking distribution of columns with log transformation
crimes_new_distribution_log = crimes_new[['CRM CD LOG','PREMIS
LOG','WEAPON LOG']]
crimes_new_distribution_log
```

```
        CRM CD LOG  PREMIS LOG  WEAPON LOG
0         6.803505    4.624973    4.634729
1         4.812184    4.644391    5.993961
2         5.442418    6.566672    6.216606
3         5.442418    4.691348    5.993961
4         6.437752    4.624973    5.993961
...            ...         ...         ...
711130    6.836259    4.691348    6.238325
711131    5.351858    4.624973    4.682131
711132    6.442540    4.700480    5.993961
711133    6.475433    6.561031    6.228511
711134    6.836259    4.634729    5.993961

[711135 rows x 3 columns]
```

```
#Checking distribution of each features
fig, axs = plt.subplots(ncols = 3, figsize=(10,5))

for i, feature in enumerate(crimes_new_distribution_log.columns):
    col = i % 3
    sns.distplot(crimes_new_distribution_log.iloc[:,i], ax = axs[col])

plt.suptitle("Distribution of features log converted")
plt.tight_layout
```

```
<function matplotlib.pyplot.tight_layout(*, pad=1.08, h_pad=None,
w_pad=None, rect=None)>
```



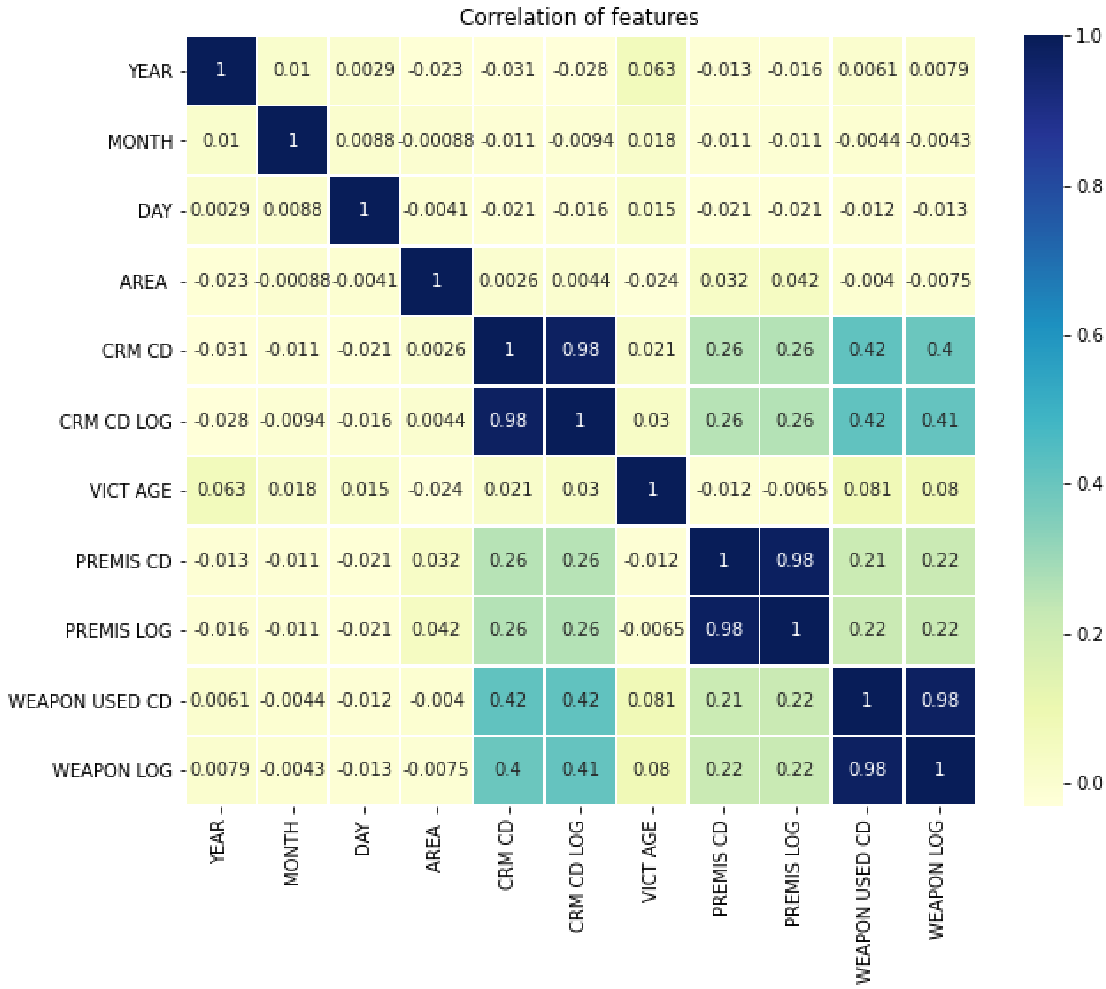

```
crimes_new.corr(method = 'pearson')
```

```
                    YEAR     MONTH       DAY      AREA    CRM CD  CRM
CD LOG  \
YEAR            1.000000  0.009967  0.002929 -0.022538 -0.030839   -
0.027787
MONTH           0.009967  1.000000  0.008815 -0.000880 -0.011464   -
0.009434
DAY             0.002929  0.008815  1.000000 -0.004069 -0.021456   -
0.016092
AREA           -0.022538 -0.000880 -0.004069  1.000000  0.002551
0.004392
CRM CD         -0.030839 -0.011464 -0.021456  0.002551  1.000000
0.979572
CRM CD LOG     -0.027787 -0.009434 -0.016092  0.004392  0.979572
1.000000
VICT AGE        0.063116  0.018425  0.014684 -0.023876  0.020803
0.030283
PREMIS CD      -0.013103 -0.010949 -0.021097  0.032292  0.256931
0.256596
PREMIS LOG     -0.015650 -0.010523 -0.021338  0.042298  0.262251
0.261661
WEAPON USED CD  0.006072 -0.004433 -0.012339 -0.004040  0.419802
0.419368
WEAPON LOG      0.007931 -0.004283 -0.013027 -0.007489  0.396109
0.409126

                VICT AGE  PREMIS CD  PREMIS LOG  WEAPON USED CD
WEAPON LOG
YEAR            0.063116  -0.013103   -0.015650        0.006072
```

```
0.007931
MONTH            0.018425  -0.010949   -0.010523       -0.004433   -
0.004283
DAY              0.014684  -0.021097   -0.021338       -0.012339   -
0.013027
AREA            -0.023876   0.032292    0.042298       -0.004040   -
0.007489
CRM CD           0.020803   0.256931    0.262251        0.419802
0.396109
CRM CD LOG       0.030283   0.256596    0.261661        0.419368
0.409126
VICT AGE         1.000000  -0.011828   -0.006537        0.081034
0.080216
PREMIS CD       -0.011828   1.000000    0.983640        0.211041
0.216050
PREMIS LOG      -0.006537   0.983640    1.000000        0.217225
0.221882
WEAPON USED CD  0.081034    0.211041    0.217225        1.000000
0.978076
WEAPON LOG       0.080216   0.216050    0.221882        0.978076
1.000000
```

```
#Correlation Heatmap
plt.figure(figsize = (10,8))
plt.title("Correlation of features")
sns.heatmap(crimes_new.corr(), annot = True, linewidth = 0.5, cmap =
"YlGnBu")
```

```
<AxesSubplot:title={'center':'Correlation of features'}>
```



Now we can see that the correlation is high for Crimes, Premis and Weapon Used columns.

```
from sklearn.preprocessing import StandardScaler

features = ['YEAR', 'MONTH','DAY','AREA ','CRM CD','CRM CD LOG','VICT
AGE',
            'PREMIS CD','PREMIS LOG','WEAPON USED CD','WEAPON LOG']

x = crimes_new.loc[:,features].values

x = StandardScaler().fit_transform(x)

crimes_new.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 711135 entries, 0 to 711134
Data columns (total 18 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE OCC        711135 non-null  datetime64[ns]
 1   YEAR            711135 non-null  int64
```

```
 2   MONTH           711135 non-null  int64
 3   DAY             711135 non-null  int64
 4   TIME OCC        711135 non-null  object
 5   AREA            711135 non-null  float64
 6   AREA NAME       711135 non-null  object
 7   CRM CD          711135 non-null  int64
 8   CRM CD DESC     711135 non-null  object
 9   CRM CD LOG      711135 non-null  float64
 10  VICT AGE        711135 non-null  int64
 11  VICT DESCENT    711135 non-null  object
 12  VICT SEX        711135 non-null  object
 13  PREMIS CD       711135 non-null  float64
 14  PREMIS LOG      711135 non-null  float64
 15  WEAPON USED CD  711135 non-null  float64
 16  WEAPON LOG      711135 non-null  float64
 17  STATUS          711135 non-null  object
dtypes: datetime64[ns](1), float64(6), int64(5), object(6)
memory usage: 97.7+ MB
```

```
crimes_new = crimes_new[['DATE OCC','TIME OCC','AREA NAME','CRM CD 
DESC','VICT DESCENT','VICT SEX','STATUS','YEAR','MONTH','DAY','AREA
','CRM CD','CRM CD LOG',
                       'VICT AGE','PREMIS CD','PREMIS LOG','WEAPON 
USED CD','WEAPON LOG']]
```

Principal Component Analysis

Normalization

Normalization is a technique often applied as part of data preparation for Data mining. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

```
 data_norm = scale(crimes_new.loc[:,'YEAR':'WEAPON LOG'])
```

```
pd.DataFrame(data_norm).describe().transpose()
```

```
       count          mean       std       min       25%       50%
75%  \
0   711135.0  2.115613e-14  1.000001 -1.608534 -0.921230  0.109725
0.797029
1   711135.0 -6.186843e-17  1.000001 -1.634508 -0.747886  0.138737
0.729819
2   711135.0 -3.676935e-18  1.000001 -1.636321 -0.850620  0.047323
0.833024
3   711135.0  1.092689e-16  1.000001 -1.586059 -0.940925  0.188059
0.833193
4   711135.0 -2.589841e-17  1.000001 -1.716509 -1.218026  0.418661
0.426969
5   711135.0 -3.892356e-15  1.000001 -2.541343 -1.226515  0.559151
```

```
0.564883
6   711135.0 -5.211656e-17  1.000001 -2.566516 -0.659221 -0.122794
0.711648
7   711135.0  7.969357e-17  1.000001 -1.029541 -1.024827 -0.336543
0.860882
8   711135.0 -1.337885e-16  1.000001 -1.119715 -1.107156  0.029155
0.934289
9   711135.0 -4.096585e-18  1.000001 -2.384747  0.251988  0.251988
0.251988
10  711135.0 -1.287807e-15  1.000001 -2.742336  0.336648  0.336648
0.336648

         max
0   1.484333
1   1.616441
2   1.730967
3   1.639610
4   1.797798
5   1.323504
6   3.930209
7   3.071876
8   1.782304
9   1.274936
10  0.908101
```

```
#Correlation of normalized data
pd.DataFrame(data_norm).corr()
```

```
          0         1         2         3         4         5
6   \
0   1.000000  0.009967  0.002929 -0.022538 -0.030839 -0.027787
0.063116
1   0.009967  1.000000  0.008815 -0.000880 -0.011464 -0.009434
0.018425
2   0.002929  0.008815  1.000000 -0.004069 -0.021456 -0.016092
0.014684
3  -0.022538 -0.000880 -0.004069  1.000000  0.002551  0.004392 -
0.023876
4  -0.030839 -0.011464 -0.021456  0.002551  1.000000  0.979572
0.020803
5  -0.027787 -0.009434 -0.016092  0.004392  0.979572  1.000000
0.030283
6   0.063116  0.018425  0.014684 -0.023876  0.020803  0.030283
1.000000
7  -0.013103 -0.010949 -0.021097  0.032292  0.256931  0.256596 -
0.011828
8  -0.015650 -0.010523 -0.021338  0.042298  0.262251  0.261661 -
0.006537
9   0.006072 -0.004433 -0.012339 -0.004040  0.419802  0.419368
0.081034
```

```
10  0.007931 -0.004283 -0.013027 -0.007489  0.396109  0.409126
0.080216

           7         8         9        10
0  -0.013103 -0.015650  0.006072  0.007931
1  -0.010949 -0.010523 -0.004433 -0.004283
2  -0.021097 -0.021338 -0.012339 -0.013027
3   0.032292  0.042298 -0.004040 -0.007489
4   0.256931  0.262251  0.419802  0.396109
5   0.256596  0.261661  0.419368  0.409126
6  -0.011828 -0.006537  0.081034  0.080216
7   1.000000  0.983640  0.211041  0.216050
8   0.983640  1.000000  0.217225  0.221882
9   0.211041  0.217225  1.000000  0.978076
10  0.216050  0.221882  0.978076  1.000000
```

```
#Finding number of principal components
principal = PCA(n_components = 9)

principal.fit(data_norm)
```

```
PCA(n_components=9)
```

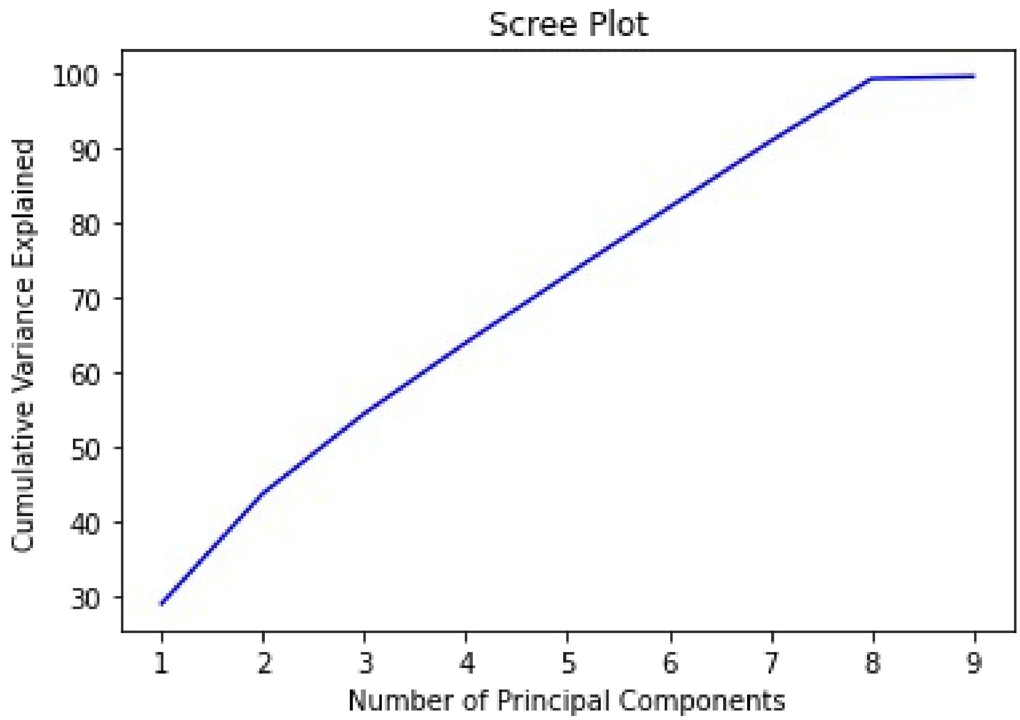
```
var = principal.explained_variance_ratio_

print(var)
```

```
[0.28955881 0.14763633 0.10740264 0.09518346 0.09130871 0.09020902
 0.08898931 0.08441117 0.00252441]
```

```
#Cumulative Variance explains
var1 = np.cumsum(np.round(principal.explained_variance_ratio_,
decimals=3)*100)

print(var1)
```

```
[29.  43.8 54.5 64.  73.1 82.1 91.  99.4 99.7]
```

```
var1 = pd.DataFrame(var1, index=np.arange(1,10))
plt.plot(var1,color='blue')
plt.title('Scree Plot')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Variance Explained')
plt.savefig('scree_plot.png',dpi=100,bbox_inches='tight')
```



Based on the scree plot it is evident that the minimum number of components required for 90% variance is 7. Now, we will implement 7 component solution for better results

7 Component Solution

```
principal7 = PCA(n_components = 7)
principal7.fit(data_norm)
data_pca7 = principal7.transform(data_norm)

# Convert the numpy array to pandas DataFrame
data_pca7 = pd.DataFrame(data_pca7)
data_pca7.columns = ["PC"+str(i) for i in range(1,8)]

# Show the head of the DataFrame
data_pca7.head(10)
```

```
        PC1       PC2       PC3       PC4       PC5       PC6
PC7
0  1.613825 -0.295995  3.897082  0.867887 -1.842939  0.354256
0.117921
1  2.181549 -0.865803 -1.626922 -1.703642 -2.190721  0.514631 -
1.310871
2 -0.952557  1.617956 -2.120890 -1.280256 -1.773104  1.101883 -
1.354292
3  1.506907 -0.990802 -0.595665 -1.691847 -1.689559  1.104430 -
1.427944
4  0.025927 -1.731841  0.850602 -0.636949 -1.365096  1.444046 -
1.044208
```

```
5  1.644552 -1.101862 -0.927982 -1.348408 -1.339813  1.493442 -
1.251953
6 -2.856159  1.109886  0.296132 -0.837388 -1.290983  1.664094 -
1.376834
7  1.624706 -1.143221 -1.074825 -1.125958 -1.274421  1.560402 -
1.136853
8  1.640994 -1.194460 -1.181665 -0.857948 -0.707711  2.209024 -
1.036946
9  0.209000  1.359806 -0.827096 -1.571417 -0.705535  2.355488 -
1.635728
```

```
data_pca7.corr()
```

```
              PC1           PC2           PC3           PC4
PC5  \
PC1  1.000000e+00  1.976918e-15 -1.436053e-15 -2.078815e-16
7.351742e-17
PC2  1.976918e-15  1.000000e+00  8.938567e-16  1.990582e-15 -
1.686421e-17
PC3 -1.436053e-15  8.938567e-16  1.000000e+00 -5.737145e-15
2.240712e-15
PC4 -2.078815e-16  1.990582e-15 -5.737145e-15  1.000000e+00
2.619779e-15
PC5  7.351742e-17 -1.686421e-17  2.240712e-15  2.619779e-15
1.000000e+00
PC6 -6.356473e-17 -1.458020e-15 -6.625990e-16 -1.077285e-15 -
1.864548e-15
PC7  2.273833e-17 -5.489502e-17 -5.173365e-15  5.701365e-15
3.060857e-16

              PC6           PC7
PC1 -6.356473e-17  2.273833e-17
PC2 -1.458020e-15 -5.489502e-17
PC3 -6.625990e-16 -5.173365e-15
PC4 -1.077285e-15  5.701365e-15
PC5 -1.864548e-15  3.060857e-16
PC6  1.000000e+00 -1.180327e-15
PC7 -1.180327e-15  1.000000e+00
```

```
principal7.components_[[0]]
```

```
array([[ 0.01275271,  0.00915518,  0.01910766, -0.01072   , -
0.44165743,
        -0.4433216 , -0.03835499, -0.34440768, -0.34759006, -
0.43008746,
        -0.42637585]])
```

```
#Manually creating the first component
np.dot(data_norm, principal7.components_[[0]].reshape(11,1))[0:10]
```

```
array([[ 1.61382535],
       [ 2.1815494 ],
       [-0.95255705],
       [ 1.50690669],
       [ 0.0259268 ],
       [ 1.64455249],
       [-2.85615878],
       [ 1.6247057 ],
       [ 1.64099407],
       [ 0.20900008]])
```

2 Component Solution

```
# Select the number of components
pca2 = PCA(n_components=2)
pca2.fit(data_norm)
data_pca2 = pca2.fit_transform(data_norm)

# Convert the numpy array to pandas DataFrame
data_pca2 = pd.DataFrame(data_pca2)
# data_pca2.index = df.name
data_pca2.columns = ["PC"+str(i) for i in range(1,3)]

# Show the head of the DataFrame
data_pca2.head()
```

```
        PC1       PC2
0  1.613825 -0.295995
1  2.181549 -0.865803
2 -0.952557  1.617956
3  1.506907 -0.990802
4  0.025927 -1.731841
```

```
pd.DataFrame(pca2.components_.transpose(),
             index=crimes_new.loc[:,'YEAR':'WEAPON LOG'].columns,
             columns=["PC"+str(i) for i in range(1,3)])
```

```
                     PC1       PC2
YEAR            0.012753 -0.033442
MONTH           0.009155 -0.015136
DAY             0.019108 -0.021902
AREA           -0.010720  0.081779
CRM CD         -0.441657 -0.173757
CRM CD LOG     -0.443322 -0.177222
VICT AGE       -0.038355 -0.118238
PREMIS CD      -0.344408  0.606199
PREMIS LOG     -0.347590  0.602964
WEAPON USED CD -0.430087 -0.307606
WEAPON LOG     -0.426376 -0.300402
```

Further transforming the dataset for the efficency in plots

```python
crimes_new['DATE OCC'] = pd.to_datetime(crimes_new['DATE OCC'])
crimes_new['Year'] = crimes_new['DATE OCC'].dt.year
crimes_new['Month'] = crimes_new['DATE OCC'].dt.month_name()
crimes_new['Day'] = crimes_new['DATE OCC'].dt.day_name()
crimes_new['TIME OCC'] = crimes_new['TIME
OCC'].astype(str).str.zfill(4)
crimes_new['HOUR OCC'] = crimes_new['TIME OCC'].apply(lambda t:
int(t[:2]))

crimes_new.head()
```

```
    DATE OCC TIME OCC  AREA NAME  \
0 2010-01-05     0150  Hollywood
1 2010-01-02     2100    Central
2 2010-01-08     2100    Central
3 2010-01-09     0230    Central
4 2010-01-14     1445    Central

                                      CRM CD DESC VICT DESCENT VICT
SEX  \
0                         VIOLATION OF COURT ORDER            W
F
1                                  RAPE, ATTEMPTED            H
F
2  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            B
M
3  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            H
M
4                         BATTERY - SIMPLE ASSAULT            B
F

  STATUS  YEAR  MONTH  DAY  AREA   CRM CD  CRM CD LOG  VICT AGE
PREMIS CD  \
0     IC  2010      1    5    6.0     900    6.803505        47
101.0
1     IC  2010      1    2    1.0     122    4.812184        47
103.0
2     AA  2010      1    8    1.0     230    5.442418        51
710.0
3     IC  2010      1    9    1.0     230    5.442418        30
108.0
4     IC  2010      1   14    1.0     624    6.437752        38
101.0

   PREMIS LOG  WEAPON USED CD  WEAPON LOG  Year    Month       Day
HOUR OCC
0    4.624973           102.0    4.634729  2010  January   Tuesday
1
1    4.644391           400.0    5.993961  2010  January  Saturday
21
```

```
2    6.566672           500.0    6.216606  2010  January    Friday
21
3    4.691348           400.0    5.993961  2010  January  Saturday
2
4    4.624973           400.0    5.993961  2010  January  Thursday
14
```

```
crimes_new_dropped = crimes_new.drop(columns = ['MONTH','DAY'])

#Saving the transformed again dataset into a new csv file
crimes_new_dropped.describe()
crimes_new_dropped.to_csv('crimes_new1.csv',index = False)

crimes_new1 = pd.read_csv('crimes_new1.csv')

crimes_new1.head()
```

```
     DATE OCC  TIME OCC  AREA NAME  \
0  2010-01-05       150  Hollywood
1  2010-01-02      2100    Central
2  2010-01-08      2100    Central
3  2010-01-09       230    Central
4  2010-01-14      1445    Central

                                      CRM CD DESC VICT DESCENT VICT
SEX  \
0                         VIOLATION OF COURT ORDER            W
F
1                                  RAPE, ATTEMPTED            H
F
2  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            B
M
3  ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            H
M
4                         BATTERY - SIMPLE ASSAULT            B
F

  STATUS  YEAR  AREA   CRM CD  CRM CD LOG  VICT AGE  PREMIS CD  PREMIS
LOG  \
0     IC  2010    6.0     900    6.803505        47      101.0
4.624973
1     IC  2010    1.0     122    4.812184        47      103.0
4.644391
2     AA  2010    1.0     230    5.442418        51      710.0
6.566672
3     IC  2010    1.0     230    5.442418        30      108.0
4.691348
4     IC  2010    1.0     624    6.437752        38      101.0
4.624973

   WEAPON USED CD  WEAPON LOG  Year    Month       Day  HOUR OCC
```

```
0           102.0    4.634729  2010  January   Tuesday         1
1           400.0    5.993961  2010  January  Saturday        21
2           500.0    6.216606  2010  January    Friday        21
3           400.0    5.993961  2010  January  Saturday         2
4           400.0    5.993961  2010  January  Thursday        14
```

```
# Process MinMaxScaling in order to make heatmap
crimes_scaled = crimes_new.copy()
except_features = ['MONTH','AREA NAME', 'VICT DESCENT','VICT
SEX','STATUS',
                   'CRM CD DESC','Month','Day'] # features on this
list will not be scaled
features = np.array(crimes_new.drop(except_features, axis=1,
inplace=False).columns).reshape(-1, 1)

for feature in features:
    scaler = MinMaxScaler()
    scaler.fit(crimes_scaled[feature])
    crimes_scaled[feature] = scaler.transform(crimes_scaled[feature])

crimes_scaled
```

```
        DATE OCC  TIME OCC  AREA NAME  \
0       0.001096  0.063189  Hollywood
1       0.000274  0.890161    Central
2       0.001917  0.890161    Central
3       0.002191  0.097116    Central
4       0.003561  0.612383    Central
...          ...       ...        ...
711130  0.905505  0.847752  Southeast
711131  0.914818  0.941052   Van Nuys
711132  0.914544  0.355810     Harbor
711133  0.923856  0.169211  Hollywood
711134  0.901671  0.890161    Olympic

                                          CRM CD DESC VICT DESCENT
VICT SEX  \
0                             VIOLATION OF COURT ORDER            W
F
1                                      RAPE, ATTEMPTED            H
F
2       ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            B
M
3       ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            H
M
4                             BATTERY - SIMPLE ASSAULT            B
F
...                                                ...          ...
...
711130          CRIMINAL THREATS - NO WEAPON DISPLAYED            B
```

```
F
711131                                              ROBBERY            W
F
711132         CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT            W
F
711133                                                ARSON            X
X
711134          CRIMINAL THREATS - NO WEAPON DISPLAYED            B
F

       STATUS  YEAR  MONTH       DAY  AREA     CRM CD  CRM CD LOG
VICT AGE  \
0          IC   0.0      1  0.133333   0.25  0.933806    0.972010
0.522936
1          IC   0.0      1  0.033333   0.00  0.014184    0.047651
0.522936
2          AA   0.0      1  0.233333   0.00  0.141844    0.340202
0.559633
3          IC   0.0      1  0.266667   0.00  0.141844    0.340202
0.366972
4          IC   0.0      1  0.433333   0.00  0.607565    0.802229
0.440367
...       ...   ...    ...       ...    ...       ...         ...
...
711130     IC   1.0      1  0.633333   0.85  0.969267    0.987214
0.256881
711131     IC   1.0      2  0.733333   0.40  0.118203    0.298165
0.366972
711132     AO   1.0      2  0.700000   0.20  0.611111    0.804452
0.220183
711133     IC   1.0      3  0.900000   0.25  0.635934    0.819721
0.091743
711134     IC   1.0      1  0.166667   0.95  0.969267    0.987214
0.513761

        PREMIS CD  PREMIS LOG  WEAPON USED CD  WEAPON LOG  Year
Month  \
0        0.000000    0.000000        0.002410    0.006011   0.0
January
1        0.002299    0.008613        0.720482    0.843456   0.0
January
2        0.700000    0.861300        0.961446    0.980631   0.0
January
3        0.008046    0.029443        0.720482    0.843456   0.0
January
4        0.000000    0.000000        0.720482    0.843456   0.0
January
...           ...         ...             ...         ...   ...
...
711130   0.008046    0.029443        0.987952    0.994012   1.0
```

```
January
711131   0.000000    0.000000        0.014458    0.035216   1.0
February
711132   0.009195    0.033494        0.720482    0.843456   1.0
February
711133   0.695402    0.858797        0.975904    0.987966   1.0
March
711134   0.001149    0.004328        0.720482    0.843456   1.0
January

             Day  HOUR OCC
0        Tuesday  0.043478
1       Saturday  0.913043
2         Friday  0.913043
3       Saturday  0.086957
4       Thursday  0.608696
...          ...       ...
711130    Sunday  0.869565
711131  Saturday  0.956522
711132    Friday  0.347826
711133  Thursday  0.173913
711134    Sunday  0.913043

[711135 rows x 22 columns]
```

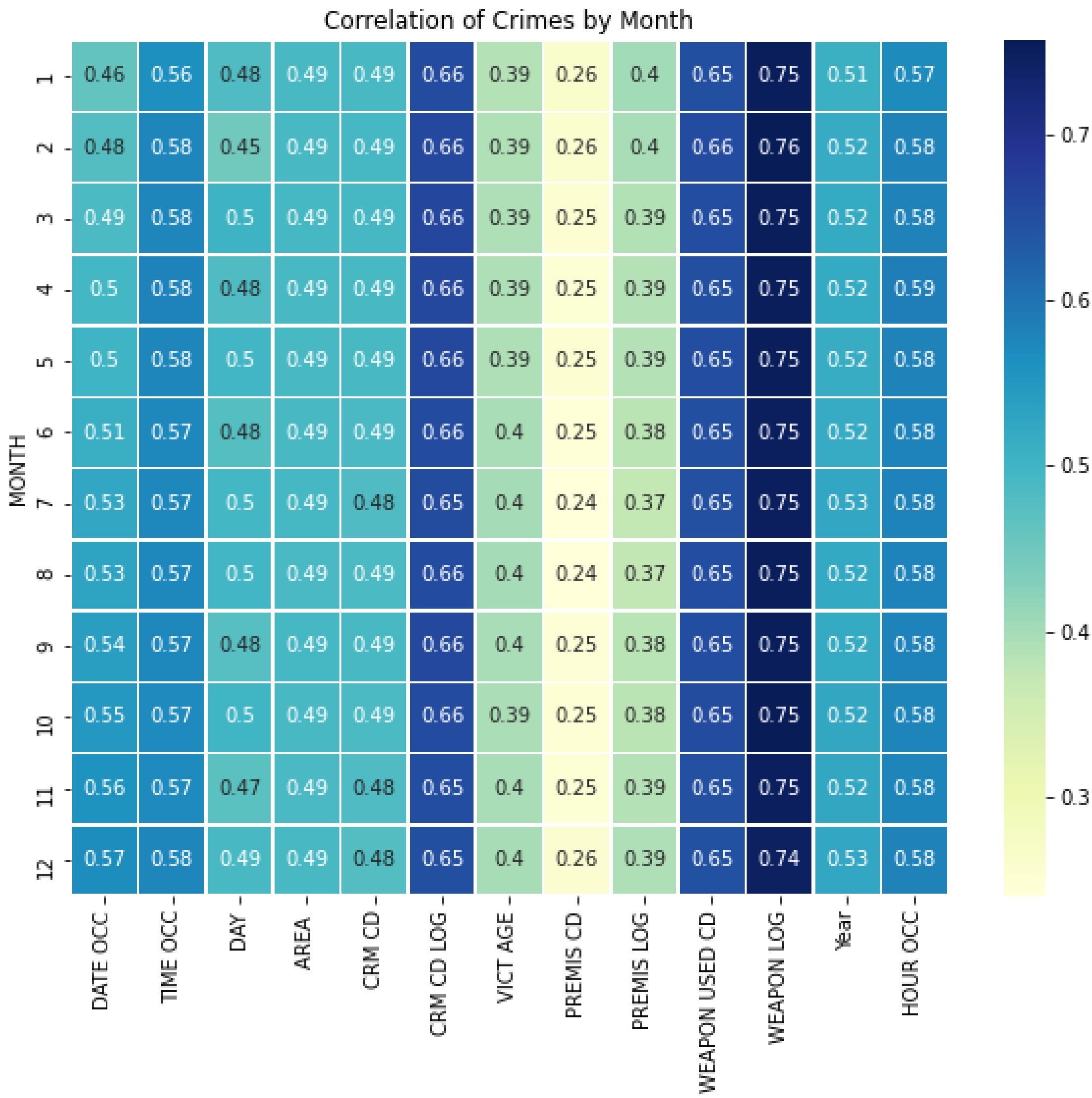
```
# Create DataFrame processed groupby on 'Month'
crimes_month = crimes_scaled.groupby(by='MONTH').mean()
crimes_month.drop(['YEAR'], axis=1, inplace=True)
crimes_month
```

```
       DATE OCC  TIME OCC       DAY     AREA     CRM CD  CRM CD LOG
VICT AGE  \
MONTH

1      0.463244  0.564732  0.482521  0.491595  0.490942    0.657477
0.386817
2      0.479116  0.576359  0.449560  0.490611  0.493031    0.661451
0.391768
3      0.488237  0.576371  0.502715  0.493145  0.493568    0.662240
0.391923
4      0.496007  0.579142  0.484053  0.491734  0.491835    0.660412
0.392374
5      0.501086  0.575646  0.497135  0.491951  0.491436    0.660711
0.392342
6      0.512196  0.574422  0.476680  0.492124  0.488587    0.657445
0.396909
7      0.526512  0.573673  0.495252  0.493935  0.484226    0.654156
0.400939
8      0.531352  0.573808  0.498387  0.489203  0.486705    0.656371
0.400261
```

```
9      0.538955  0.574433  0.475547  0.493246  0.490586    0.660133
0.396644
10     0.550211  0.572828  0.499208  0.491350  0.487382    0.656872
0.394926
11     0.559031  0.572122  0.470995  0.490008  0.483104    0.652839
0.396248
12     0.568284  0.576497  0.491592  0.491143  0.479883    0.650467
0.398355

       PREMIS CD  PREMIS LOG  WEAPON USED CD  WEAPON LOG      Year
HOUR OCC
MONTH

1       0.261408    0.402133        0.651528    0.750993  0.510139
0.570906
2       0.259786    0.397063        0.656312    0.756307  0.518639
0.582516
3       0.254301    0.389387        0.653123    0.752157  0.519707
0.582585
4       0.251872    0.386155        0.651463    0.750710  0.519075
0.585383
5       0.253021    0.387049        0.653096    0.752758  0.515470
0.581821
6       0.246215    0.380005        0.650207    0.749776  0.518565
0.580630
7       0.240832    0.373708        0.649062    0.748697  0.525171
0.579732
8       0.242364    0.374600        0.651715    0.751684  0.521087
0.579958
9       0.248838    0.381964        0.653324    0.753420  0.520309
0.580582
10      0.249446    0.381520        0.654533    0.754766  0.523469
0.578897
11      0.252125    0.387325        0.649142    0.748822  0.524092
0.578239
12      0.255279    0.393666        0.646106    0.744742  0.525055
0.582695
```

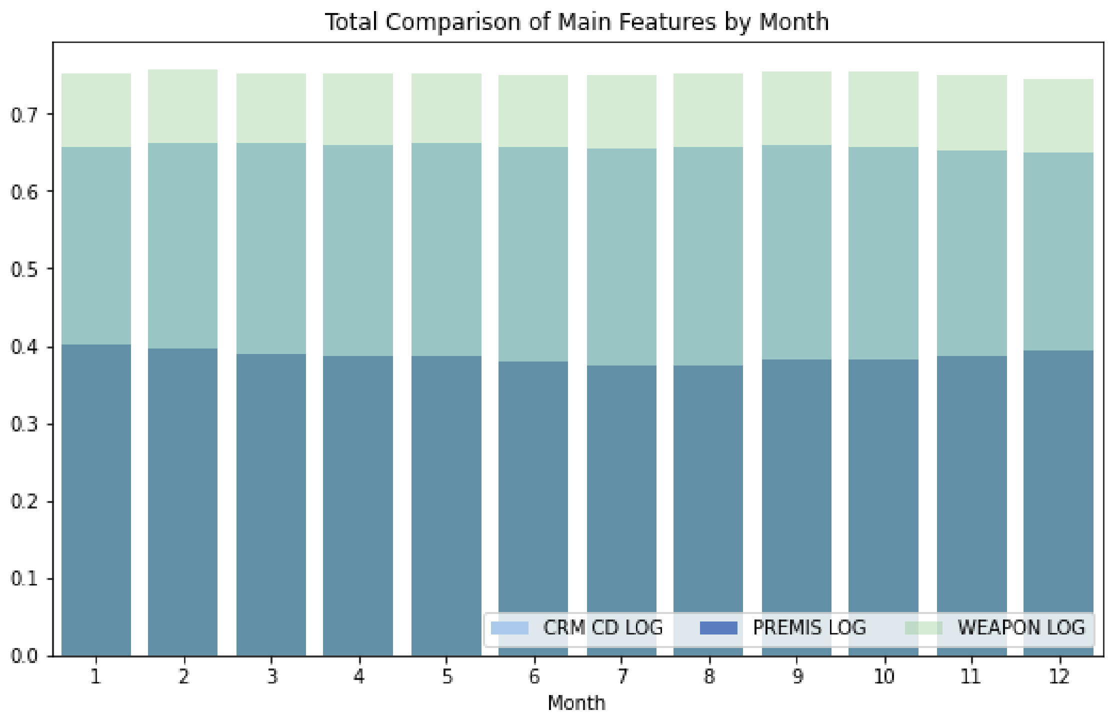
```python
# Plot heatmap
plt.figure(figsize=(10, 8))
plt.title('Correlation of Crimes by Month')
sns.heatmap(crimes_month, annot=True, linewidths=.5, cmap="YlGnBu")
```

```
<AxesSubplot:title={'center':'Correlation of Crimes by Month'},
ylabel='MONTH'>
```




```
# Plot barplot
crimes_month = crimes_scaled.groupby(by='MONTH',
as_index=False).mean()
crimes_month.drop(['YEAR'], axis=1, inplace=True)

sns.set_theme="whitegrid"
f, ax = plt.subplots(figsize=(10, 6))

## Plot 'Crm Cd Log'
sns.set_color_codes("pastel")
sns.barplot(x="MONTH", y="CRM CD LOG", data=crimes_month, label="CRM
CD LOG", color="b")

## Plot 'Premis Cd Log'
sns.set_color_codes("muted")
sns.barplot(x="MONTH", y="PREMIS LOG", data=crimes_month,
```

```
label="PREMIS LOG", color="b")

## Plot 'Weapon Used Cd Log'
sns.set_color_codes("muted")
sns.barplot(x="MONTH", y="WEAPON LOG", data=crimes_month,
label="WEAPON LOG", color="g", alpha=0.3)

ax.legend(ncol=3, loc="lower right", frameon=True)
ax.set(ylabel="", xlabel="Month")
plt.title("Total Comparison of Main Features by Month")

plt.show()
```



We can see there is a high correlation of crimes for the month of October, November and December which means the crime rate is high in the months of November and December.

```
# Process MinMaxScaling in order to make heatmap
crimes_scaled = crimes_new.copy()
except_features = ['DAY','AREA NAME', 'VICT DESCENT', 'VICT
SEX','STATUS',
                   'CRM CD DESC','Month','Day'] # features on this
list will not be scaled
features = np.array(crimes_new.drop(except_features, axis=1,
inplace=False).columns).reshape(-1, 1)

for feature in features:
    scaler = MinMaxScaler()
```

```
    scaler.fit(crimes_scaled[feature])
    crimes_scaled[feature] = scaler.transform(crimes_scaled[feature])

# Create DataFrame processed groupby on 'Day'
crimes_day = crimes_scaled.groupby(by='DAY').mean()
crimes_day.drop(['YEAR'], axis=1, inplace=True)
crimes_day
```

```
     DATE OCC  TIME OCC     MONTH     AREA     CRM CD  CRM CD LOG
VICT AGE  \
DAY

1    0.493399  0.519655  0.466396  0.505721  0.539552    0.689486
0.358272
2    0.514247  0.579402  0.512465  0.490316  0.481734    0.652014
0.400056
3    0.519065  0.581208  0.501875  0.494036  0.484101    0.654103
0.401345
4    0.515693  0.578802  0.504037  0.489427  0.490847    0.660193
0.398327
5    0.522058  0.574406  0.504503  0.485466  0.489499    0.659013
0.399226
6    0.516227  0.577760  0.501537  0.492006  0.487166    0.657128
0.398237
7    0.517622  0.578143  0.506835  0.490201  0.490130    0.659541
0.396602
8    0.521291  0.574352  0.508661  0.492877  0.490475    0.659499
0.397671
9    0.521276  0.573344  0.508715  0.493784  0.485292    0.655644
0.398672
10   0.514577  0.577457  0.506460  0.491428  0.488087    0.657286
0.396665
11   0.520432  0.575664  0.509451  0.492039  0.488443    0.658817
0.395713
12   0.515524  0.581483  0.501883  0.489246  0.489134    0.658754
0.397574
13   0.520959  0.579645  0.506621  0.490813  0.486659    0.655963
0.394795
14   0.519388  0.578867  0.504130  0.490461  0.485275    0.655400
0.397505
15   0.515799  0.574392  0.501467  0.494630  0.493948    0.661320
0.392663
16   0.514894  0.578134  0.502157  0.493277  0.488384    0.658749
0.396439
17   0.513559  0.576433  0.500487  0.490237  0.486837    0.656848
0.395419
18   0.519713  0.575165  0.507914  0.490361  0.487237    0.657389
0.395605
19   0.517296  0.578374  0.501672  0.493268  0.487294    0.658445
0.396087
```

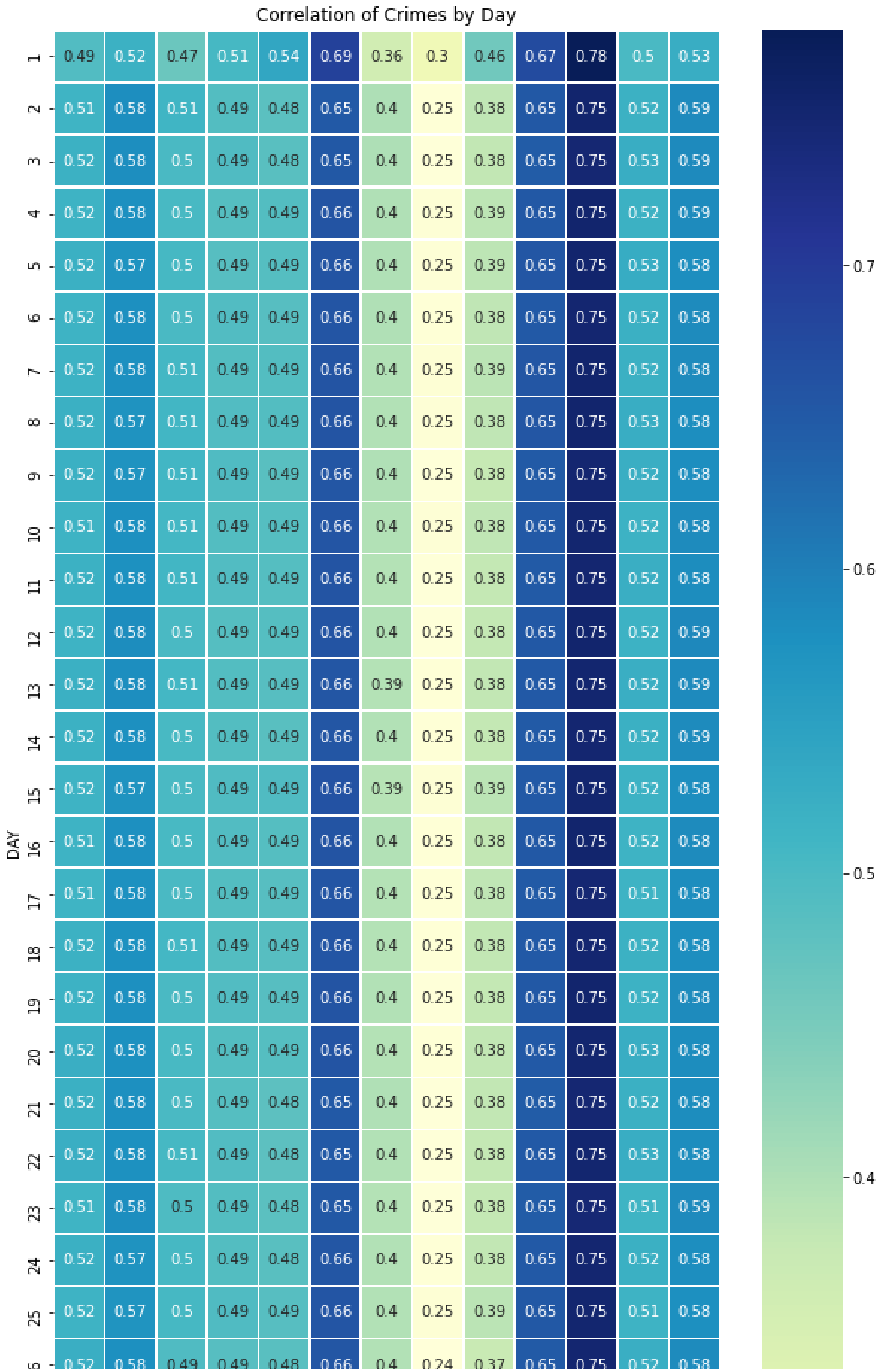
20	0.523859	0.576005	0.503186	0.489916	0.489384	0.658781	0.395041
21	0.521863	0.577210	0.500613	0.488568	0.480517	0.651320	0.399848
22	0.524792	0.576243	0.505719	0.492270	0.480630	0.651642	0.397216
23	0.514060	0.580959	0.497538	0.490934	0.482908	0.653153	0.395480
24	0.522513	0.574759	0.504232	0.489783	0.484268	0.655811	0.397136
25	0.515911	0.571874	0.504349	0.493871	0.486052	0.656415	0.396031
26	0.523997	0.578300	0.494342	0.485857	0.484596	0.655056	0.396487
27	0.523090	0.578010	0.493385	0.490917	0.481617	0.652797	0.394412
28	0.525023	0.573412	0.495365	0.489463	0.483365	0.653664	0.396186
29	0.523661	0.572752	0.518259	0.491895	0.480609	0.650769	0.396785
30	0.520520	0.575266	0.525147	0.491743	0.479833	0.650892	0.396736
31	0.523764	0.580156	0.509855	0.490705	0.477489	0.648282	0.397469

DAY	PREMIS CD	PREMIS LOG	WEAPON USED CD	WEAPON LOG	Year	HOUR OCC
1	0.301553	0.459264	0.674385	0.777235	0.500740	0.526638
2	0.249786	0.384637	0.653013	0.752926	0.518920	0.585532
3	0.247415	0.381457	0.647256	0.747290	0.525048	0.587405
4	0.251473	0.386594	0.653576	0.753255	0.520783	0.585064
5	0.250853	0.385324	0.654516	0.753339	0.527495	0.580516
6	0.249409	0.383680	0.650527	0.749766	0.521014	0.583861
7	0.252943	0.387496	0.654980	0.754270	0.521721	0.584378
8	0.250156	0.384376	0.654840	0.754276	0.525304	0.580341
9	0.246462	0.379021	0.648943	0.748082	0.524979	0.579334
10	0.250419	0.383774	0.652009	0.750767	0.517466	0.583670
11	0.247259	0.380219	0.649894	0.749831	0.523363	

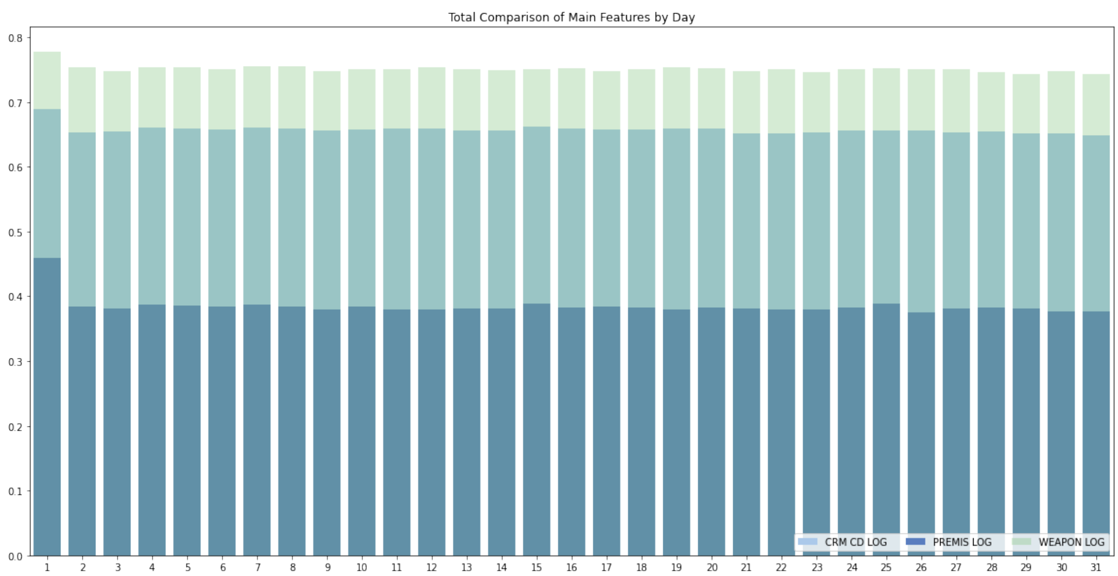
```
0.581653
12    0.246907    0.379190        0.653706    0.753166  0.518375
0.587725
13    0.248566    0.381772        0.652085    0.750335  0.523628
0.585817
14    0.248233    0.381140        0.649228    0.748286  0.521828
0.585110
15    0.252521    0.387976        0.651758    0.750348  0.517811
0.580631
16    0.248840    0.382695        0.652279    0.752286  0.516429
0.584268
17    0.248949    0.383288        0.648395    0.747710  0.514814
0.582468
18    0.249592    0.383230        0.650461    0.750081  0.520589
0.581204
19    0.246392    0.379906        0.653105    0.753230  0.518235
0.584512
20    0.249211    0.382785        0.652028    0.751136  0.525070
0.582213
21    0.247224    0.381376        0.646986    0.746952  0.522807
0.583285
22    0.246333    0.379221        0.650402    0.750500  0.525236
0.582305
23    0.246677    0.380117        0.647036    0.745884  0.513844
0.587196
24    0.248453    0.382139        0.650201    0.749968  0.522245
0.580796
25    0.252298    0.388233        0.652692    0.751794  0.514603
0.577881
26    0.243564    0.374843        0.649949    0.749752  0.524293
0.584356
27    0.247571    0.380346        0.650134    0.750422  0.523085
0.584216
28    0.248957    0.382757        0.646942    0.746311  0.524722
0.579361
29    0.247285    0.380847        0.644752    0.743512  0.520606
0.578695
30    0.244217    0.376755        0.646396    0.746610  0.516114
0.581383
31    0.244290    0.376335        0.643161    0.742651  0.521004
0.586317
```

```python
# Plot heatmap
plt.figure(figsize=(10, 20))
plt.title('Correlation of Crimes by Day')
sns.heatmap(crimes_day, annot=True, linewidths=.5, cmap="YlGnBu")
```

```
<AxesSubplot:title={'center':'Correlation of Crimes by Day'},
ylabel='DAY'>
```


Correlation of Crimes by Day
DAY
1 0.49 0.52 0.47 0.51 0.54 0.69 0.36 0.3 0.46 0.67 0.78 0.5 0.53
2 0.51 0.58 0.51 0.49 0.48 0.65 0.4 0.25 0.38 0.65 0.75 0.52 0.59
3 0.52 0.58 0.5 0.49 0.48 0.65 0.4 0.25 0.38 0.65 0.75 0.53 0.59
4 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.39 0.65 0.75 0.52 0.59
5 0.52 0.57 0.5 0.49 0.49 0.66 0.4 0.25 0.39 0.65 0.75 0.53 0.58
6 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
7 0.52 0.58 0.51 0.49 0.49 0.66 0.4 0.25 0.39 0.65 0.75 0.52 0.58
8 0.52 0.57 0.51 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.53 0.58
9 0.52 0.57 0.51 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
10 0.51 0.58 0.51 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
11 0.52 0.58 0.51 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
12 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.59
13 0.52 0.58 0.51 0.49 0.49 0.66 0.39 0.25 0.38 0.65 0.75 0.52 0.59
14 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.59
15 0.52 0.57 0.5 0.49 0.49 0.66 0.39 0.25 0.39 0.65 0.75 0.52 0.58
16 0.51 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
17 0.51 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.51 0.58
18 0.52 0.58 0.51 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
19 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
20 0.52 0.58 0.5 0.49 0.49 0.66 0.4 0.25 0.38 0.65 0.75 0.53 0.58
21 0.52 0.58 0.5 0.49 0.48 0.65 0.4 0.25 0.38 0.65 0.75 0.52 0.58
22 0.52 0.58 0.51 0.49 0.48 0.65 0.4 0.25 0.38 0.65 0.75 0.53 0.58
23 0.51 0.58 0.5 0.49 0.48 0.65 0.4 0.25 0.38 0.65 0.75 0.51 0.59
24 0.52 0.57 0.5 0.49 0.48 0.66 0.4 0.25 0.38 0.65 0.75 0.52 0.58
25 0.52 0.57 0.5 0.49 0.49 0.66 0.4 0.25 0.39 0.65 0.75 0.51 0.58
0.52 0.58 0.49 0.49 0.48 0.66 0.4 0.24 0.37 0.65 0.75 0.52 0.58
0.7
0.6
0.5
0.4

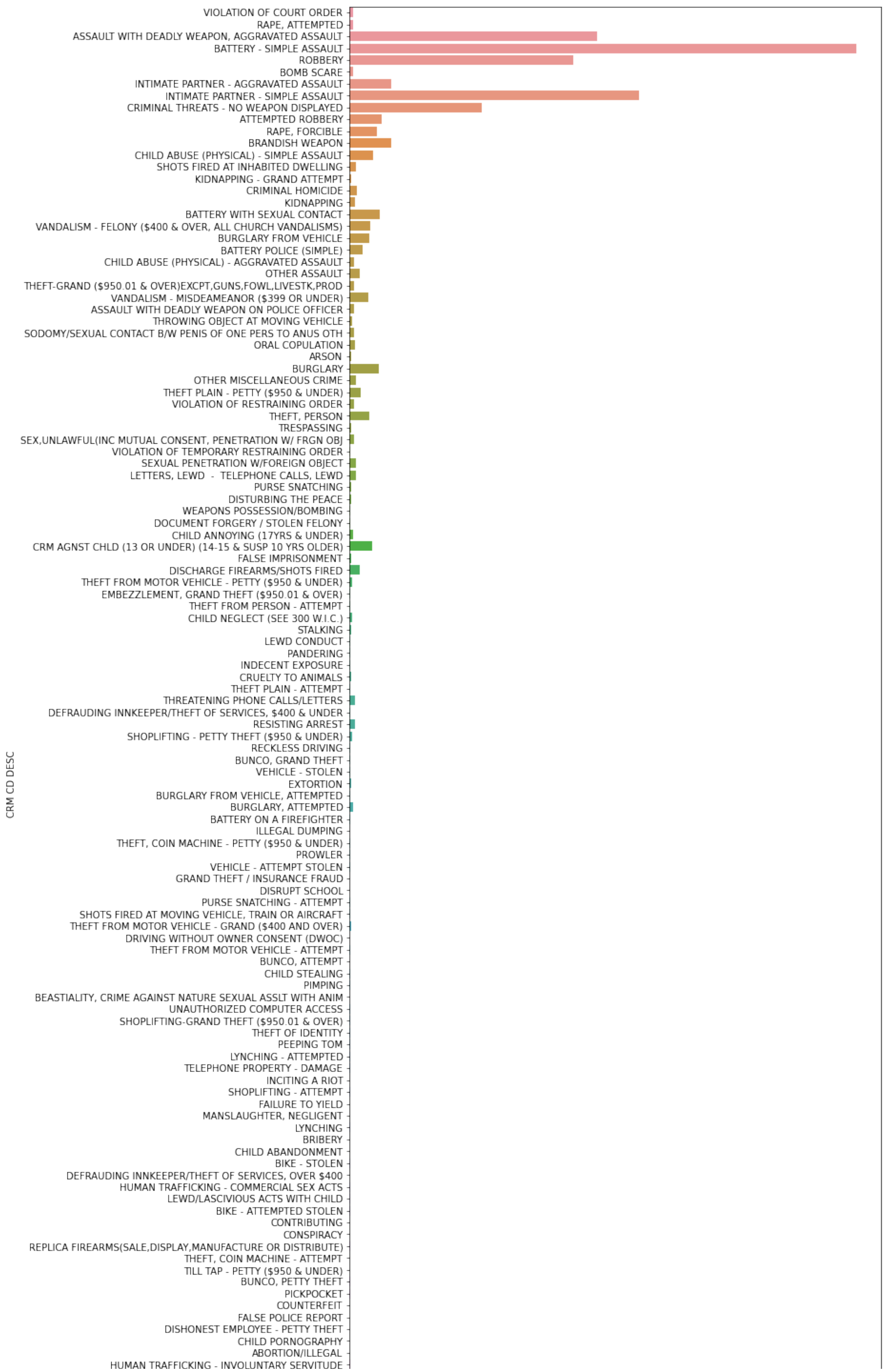
```python
# Plot barplot
crimes_day = crimes_scaled.groupby(by='DAY', as_index=False).mean()
crimes_day.drop(['YEAR'], axis=1, inplace=True)

sns.set_theme="whitegrid"
f, ax = plt.subplots(figsize=(20, 10))

## Plot 'Crm Cd Log'
sns.set_color_codes("pastel")
sns.barplot(x="DAY", y="CRM CD LOG", data=crimes_day, label="CRM CD
LOG", color="b")

## Plot 'Premis Cd Log'
sns.set_color_codes("muted")
sns.barplot(x="DAY", y="PREMIS LOG", data=crimes_day, label="PREMIS
LOG", color="b")

## Plot 'Weapon Used Cd Log'
sns.set_color_codes("muted")
sns.barplot(x="DAY", y="WEAPON LOG", data=crimes_day, label="WEAPON
LOG", color="g", alpha=0.3)

ax.legend(ncol=3, loc="lower right", frameon=True)
ax.set(ylabel="", xlabel="Day")
plt.title("Total Comparison of Main Features by Day")

plt.show()
```



Visualizations: # 1 Visualizing Crimes frequency

```python
plt.figure(figsize=(10,30))
count = sns.countplot(
```

```
        data = crimes_new1,
        y = 'CRM CD DESC',
)
plt.show()
```


CRM CD DESC
VIOLATION OF COURT ORDER
RAPE, ATTEMPTED
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
BATTERY - SIMPLE ASSAULT
ROBBERY
BOMB SCARE
INTIMATE PARTNER - AGGRAVATED ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
ATTEMPTED ROBBERY
RAPE, FORCIBLE
BRANDISH WEAPON
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
SHOTS FIRED AT INHABITED DWELLING
KIDNAPPING - GRAND ATTEMPT
CRIMINAL HOMICIDE
KIDNAPPING
BATTERY WITH SEXUAL CONTACT
VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
BURGLARY FROM VEHICLE
BATTERY POLICE (SIMPLE)
CHILD ABUSE (PHYSICAL) - AGGRAVATED ASSAULT
OTHER ASSAULT
THEFT-GRAND ($950.01 & OVER)EXCPT,GUNS,FOWL,LIVESTK,PROD
VANDALISM - MISDEAMEANOR ($399 OR UNDER)
ASSAULT WITH DEADLY WEAPON ON POLICE OFFICER
THROWING OBJECT AT MOVING VEHICLE
SODOMY/SEXUAL CONTACT B/W PENIS OF ONE PERS TO ANUS OTH
ORAL COPULATION
ARSON
BURGLARY
OTHER MISCELLANEOUS CRIME
THEFT PLAIN - PETTY ($950 & UNDER)
VIOLATION OF RESTRAINING ORDER
THEFT, PERSON
TRESPASSING
SEX,UNLAWFUL(INC MUTUAL CONSENT, PENETRATION W/ FRGN OBJ
VIOLATION OF TEMPORARY RESTRAINING ORDER
SEXUAL PENETRATION W/FOREIGN OBJECT
LETTERS, LEWD - TELEPHONE CALLS, LEWD
PURSE SNATCHING
DISTURBING THE PEACE
WEAPONS POSSESSION/BOMBING
DOCUMENT FORGERY / STOLEN FELONY
CHILD ANNOYING (17YRS & UNDER)
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)
FALSE IMPRISONMENT
DISCHARGE FIREARMS/SHOTS FIRED
THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
EMBEZZLEMENT, GRAND THEFT ($950.01 & OVER)
THEFT FROM PERSON - ATTEMPT
CHILD NEGLECT (SEE 300 W.I.C.)
STALKING
LEWD CONDUCT
PANDERING
INDECENT EXPOSURE
CRUELTY TO ANIMALS
THEFT PLAIN - ATTEMPT
THREATENING PHONE CALLS/LETTERS
DEFRAUDING INNKEEPER/THEFT OF SERVICES, $400 & UNDER
RESISTING ARREST
SHOPLIFTING - PETTY THEFT ($950 & UNDER)
RECKLESS DRIVING
BUNCO, GRAND THEFT
VEHICLE - STOLEN
EXTORTION
BURGLARY FROM VEHICLE, ATTEMPTED
BURGLARY, ATTEMPTED
BATTERY ON A FIREFIGHTER
ILLEGAL DUMPING
THEFT, COIN MACHINE - PETTY ($950 & UNDER)
PROWLER
VEHICLE - ATTEMPT STOLEN
GRAND THEFT / INSURANCE FRAUD
DISRUPT SCHOOL
PURSE SNATCHING - ATTEMPT
SHOTS FIRED AT MOVING VEHICLE, TRAIN OR AIRCRAFT
THEFT FROM MOTOR VEHICLE - GRAND ($400 AND OVER)
DRIVING WITHOUT OWNER CONSENT (DWOC)
THEFT FROM MOTOR VEHICLE - ATTEMPT
BUNCO, ATTEMPT
CHILD STEALING
PIMPING
BEASTIALITY, CRIME AGAINST NATURE SEXUAL ASSLT WITH ANIM
UNAUTHORIZED COMPUTER ACCESS
SHOPLIFTING-GRAND THEFT ($950.01 & OVER)
THEFT OF IDENTITY
PEEPING TOM
LYNCHING - ATTEMPTED
TELEPHONE PROPERTY - DAMAGE
INCITING A RIOT
SHOPLIFTING - ATTEMPT
FAILURE TO YIELD
MANSLAUGHTER, NEGLIGENT
LYNCHING
BRIBERY
CHILD ABANDONMENT
BIKE - STOLEN
DEFRAUDING INNKEEPER/THEFT OF SERVICES, OVER $400
HUMAN TRAFFICKING - COMMERCIAL SEX ACTS
LEWD/LASCIVIOUS ACTS WITH CHILD
BIKE - ATTEMPTED STOLEN
CONTRIBUTING
CONSPIRACY
REPLICA FIREARMS(SALE,DISPLAY,MANUFACTURE OR DISTRIBUTE)
THEFT, COIN MACHINE - ATTEMPT
TILL TAP - PETTY ($950 & UNDER)
BUNCO, PETTY THEFT
PICKPOCKET
COUNTERFEIT
FALSE POLICE REPORT
DISHONEST EMPLOYEE - PETTY THEFT
CHILD PORNOGRAPHY
ABORTION/ILLEGAL
HUMAN TRAFFICKING - INVOLUNTARY SERVITUDE

As per the graph, the Crime Code Description 'BATTERY - SIMPLE ASSAULT' has been most reported in Los Angeles in the last 10 years. Whereas, 'SHOTS FIRED AT MOVING VEHICLE, TRAIN OR AIRCRAFT', 'DRIVING WITHOUT OWNER CONSENT(DWOC)','FALSE POLICE REPORT' are some of the least reported crimes which shows the rate of occurence of these crimes may be very low in the past 10 years.

2 Visualizing which day of the week is most violent in LA

This visualization address the research question of the most violent day in a week in Los Angeles

```
labels = crimes_new1['Day'].unique()
values=[]
for each in labels:
    values.append(len(crimes_new1[crimes_new1['Day']==each]))
fig = go.Figure(data=[go.Pie(labels=labels, values=values, hole=.3)])
fig.show()
```


```
{"data":[{"values":
[94960,109121,102671,95820,113537,99166,95860],"labels":
["Tuesday","Saturday","Friday","Thursday","Sunday","Monday","Wednesday
"],"hole":0.3,"type":"pie"}],"config":{"plotlyServerURL":"https://
plot.ly"},"layout":{"template":{"data":{"contourcarpet":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"contourcarpet"}],"scattermapbox"
:[{"type":"scattermapbox","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"mesh3d":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"mesh3d"}],"heatmap":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmap"}],"pie":
[{"automargin":true,"type":"pie"}],"carpet":[{"aaxis":
{"linecolor":"white","minorgridcolor":"white","endlinecolor":"#2a3f5f"
,"startlinecolor":"#2a3f5f","gridcolor":"white"},"baxis":
{"linecolor":"white","minorgridcolor":"white","endlinecolor":"#2a3f5f"
,"startlinecolor":"#2a3f5f","gridcolor":"white"},"type":"carpet"}],"ba
r":[{"error_x":{"color":"#2a3f5f"},"error_y":
{"color":"#2a3f5f"},"type":"bar","marker":{"line":
{"width":0.5,"color":"#E5ECF6"},"pattern":
{"solidity":0.2,"fillmode":"overlay","size":10}}}],"barpolar":
[{"type":"barpolar","marker":{"line":
{"width":0.5,"color":"#E5ECF6"},"pattern":
{"solidity":0.2,"fillmode":"overlay","size":10}}}],"scatter3d":
[{"line":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatter3d","marker":
{"colorbar":{"outlinewidth":0,"ticks":""}}}],"contour":[{"colorbar":
```

{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"contour"}],"histogram2d":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2d"}],"scatterpolar":
[{"type":"scatterpolar","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"histogram":
[{"type":"histogram","marker":{"pattern":
{"solidity":0.2,"fillmode":"overlay","size":10}}}],"histogram2dcontour
":[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2dcontour"}],"parcoords":[{"line":
{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"parcoords"}],"scatterpolargl":
[{"type":"scatterpolargl","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"heatmapgl":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmapgl"}],"scattercarpet":
[{"type":"scattercarpet","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"choropleth":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"choropleth"}],"scatterternary":
[{"type":"scatterternary","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"scatter":[{"fillpattern":
{"solidity":0.2,"fillmode":"overlay","size":10},"type":"scatter"}],"ta
ble":[{"cells":{"fill":{"color":"#EBF0F8"},"line":
{"color":"white"}},"header":{"fill":{"color":"#C8D4E3"},"line":
{"color":"white"}},"type":"table"}],"scattergeo":
[{"type":"scattergeo","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}],"surface":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"surface"}],"scattergl":

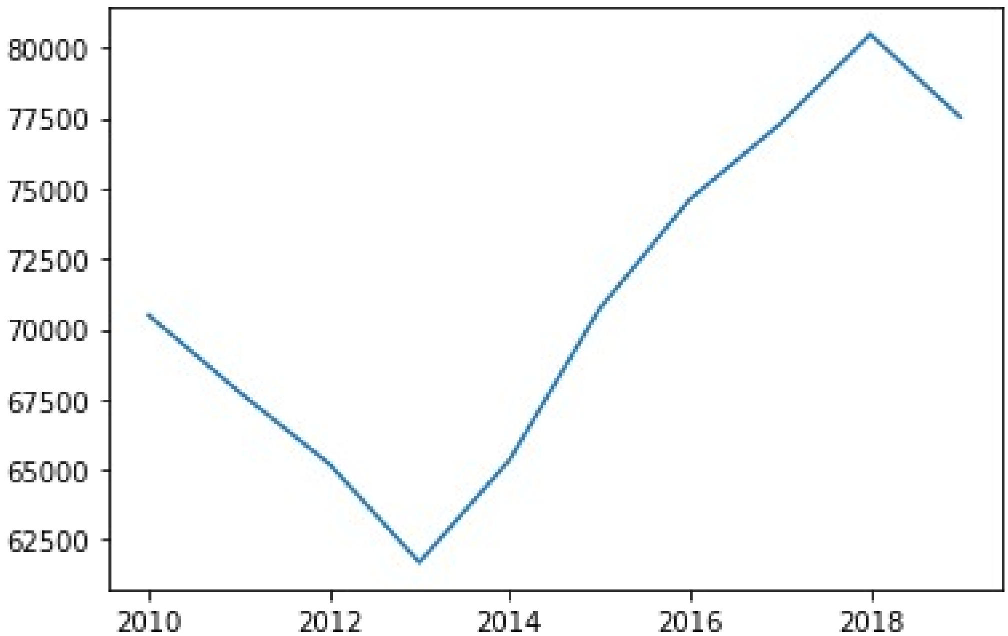
[{"type":"scattergl","marker":{"colorbar":
{"outlinewidth":0,"ticks":""}}}]},"layout":{"ternary":{"aaxis":
{"linecolor":"white","ticks":"","gridcolor":"white"},"baxis":
{"linecolor":"white","ticks":"","gridcolor":"white"},"caxis":
{"linecolor":"white","ticks":"","gridcolor":"white"},"bgcolor":"#E5ECF
6"},"autotypenumbers":"strict","shapedefaults":{"line":
{"color":"#2a3f5f"}},"annotationdefaults":
{"arrowwidth":1,"arrowcolor":"#2a3f5f","arrowhead":0},"coloraxis":
{"colorbar":{"outlinewidth":0,"ticks":""}},"title":{"x":5.0e-
2},"hoverlabel":{"align":"left"},"colorscale":{"diverging":
[[0,"#8e0152"],[0.1,"#c51b7d"],[0.2,"#de77ae"],[0.3,"#f1b6da"],
[0.4,"#fde0ef"],[0.5,"#f7f7f7"],[0.6,"#e6f5d0"],[0.7,"#b8e186"],
[0.8,"#7fbc41"],[0.9,"#4d9221"],[1,"#276419"]],"sequentialminus":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]],"sequential":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]]},"hovermode":"closest","mapbox":
{"style":"light"},"paper_bgcolor":"white","scene":{"zaxis":
{"linecolor":"white","showbackground":true,"zerolinecolor":"white","gr
idwidth":2,"ticks":"","backgroundcolor":"#E5ECF6","gridcolor":"white"}
,"xaxis":
{"linecolor":"white","showbackground":true,"zerolinecolor":"white","gr
idwidth":2,"ticks":"","backgroundcolor":"#E5ECF6","gridcolor":"white"}
,"yaxis":
{"linecolor":"white","showbackground":true,"zerolinecolor":"white","gr
idwidth":2,"ticks":"","backgroundcolor":"#E5ECF6","gridcolor":"white"}
},"font":{"color":"#2a3f5f"},"xaxis":{"linecolor":"white","title":
{"standoff":15},"zerolinewidth":2,"automargin":true,"zerolinecolor":"w
hite","ticks":"","gridcolor":"white"},"polar":{"angularaxis":
{"linecolor":"white","ticks":"","gridcolor":"white"},"radialaxis":
{"linecolor":"white","ticks":"","gridcolor":"white"},"bgcolor":"#E5ECF
6"},"plot_bgcolor":"#E5ECF6","geo":
{"subunitcolor":"white","lakecolor":"white","landcolor":"#E5ECF6","sho
wland":true,"showlakes":true,"bgcolor":"white"},"yaxis":
{"linecolor":"white","title":
{"standoff":15},"zerolinewidth":2,"automargin":true,"zerolinecolor":"w
hite","ticks":"","gridcolor":"white"},"colorway":
["#636efa","#EF553B","#00cc96","#ab63fa","#FFA15A","#19d3f3","#FF6692"
,"#B6E880","#FF97FF","#FECB52"]}}}}

We can see that Sunday is the day where there are more crimes reported with 16% or 113,578 cases.

Pattern indicating rise and fall of crime rate in Los Angeles

```
crimeByYear =
crimes_new1['YEAR'].value_counts(sort=False).sort_index()
crimeByYear.plot(kind = 'line')
```

```
<AxesSubplot:>
```



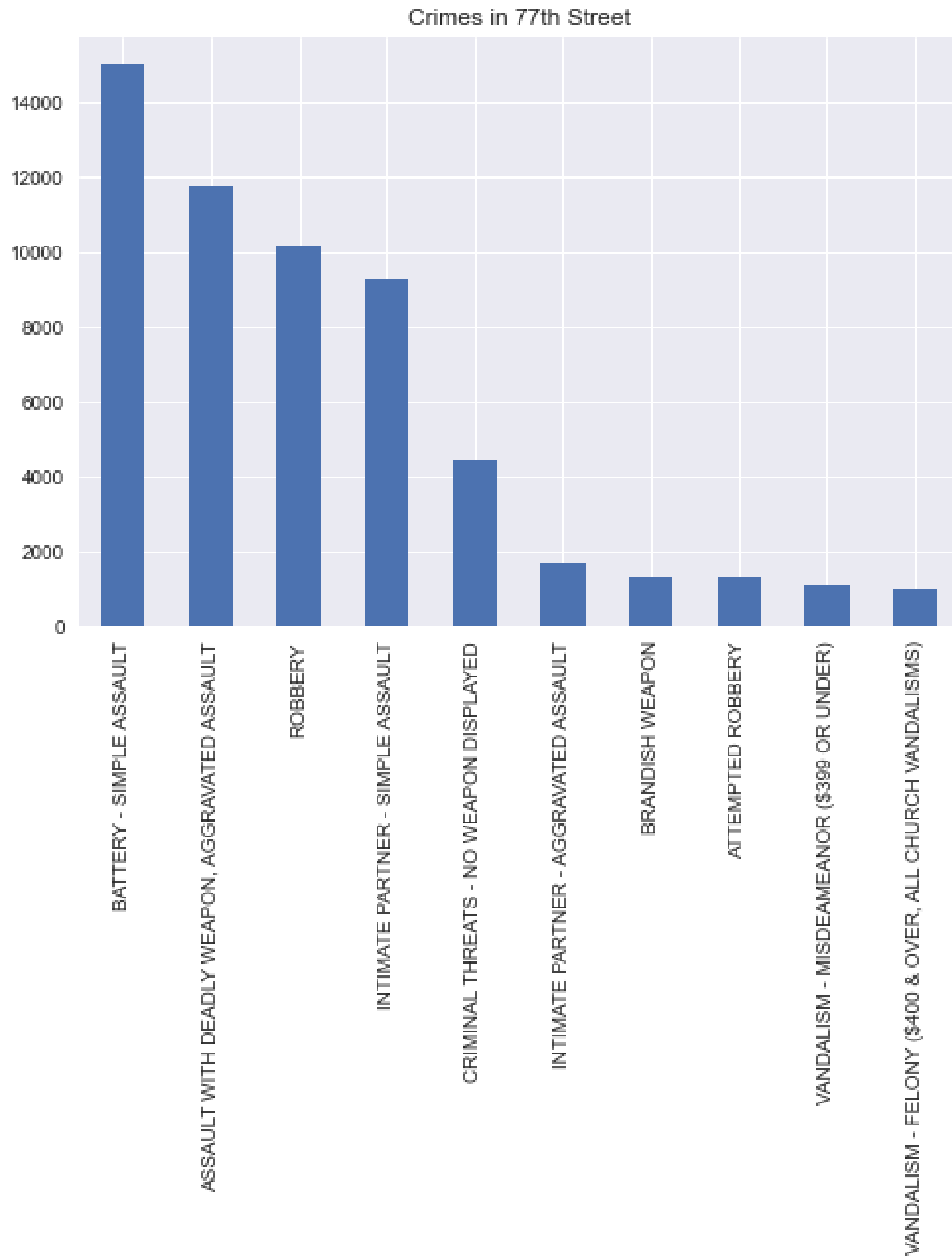
As per the line graph, the crime rate peaked in the two-year period between 2016 and 2018.

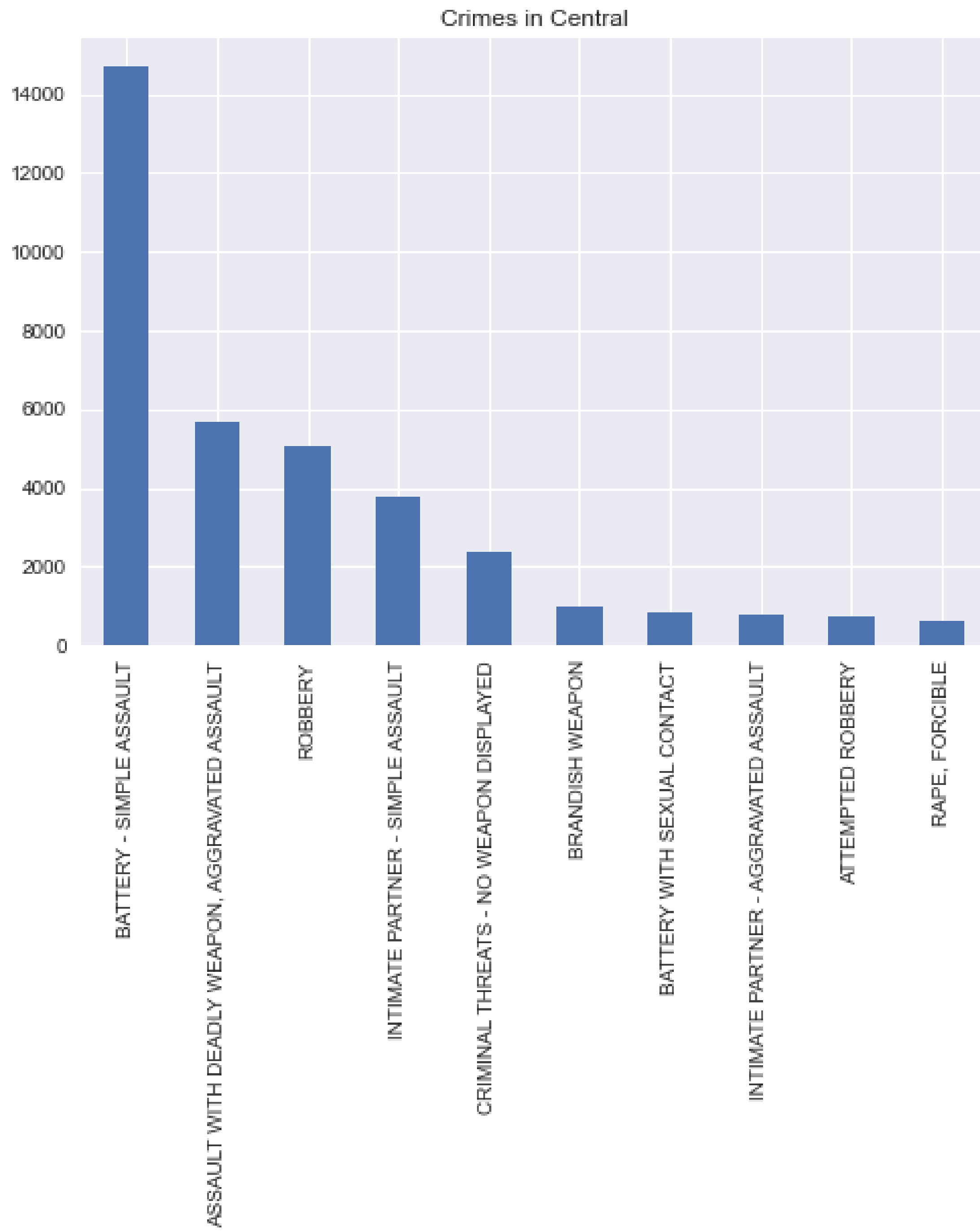
Area Wise Crime Analysis

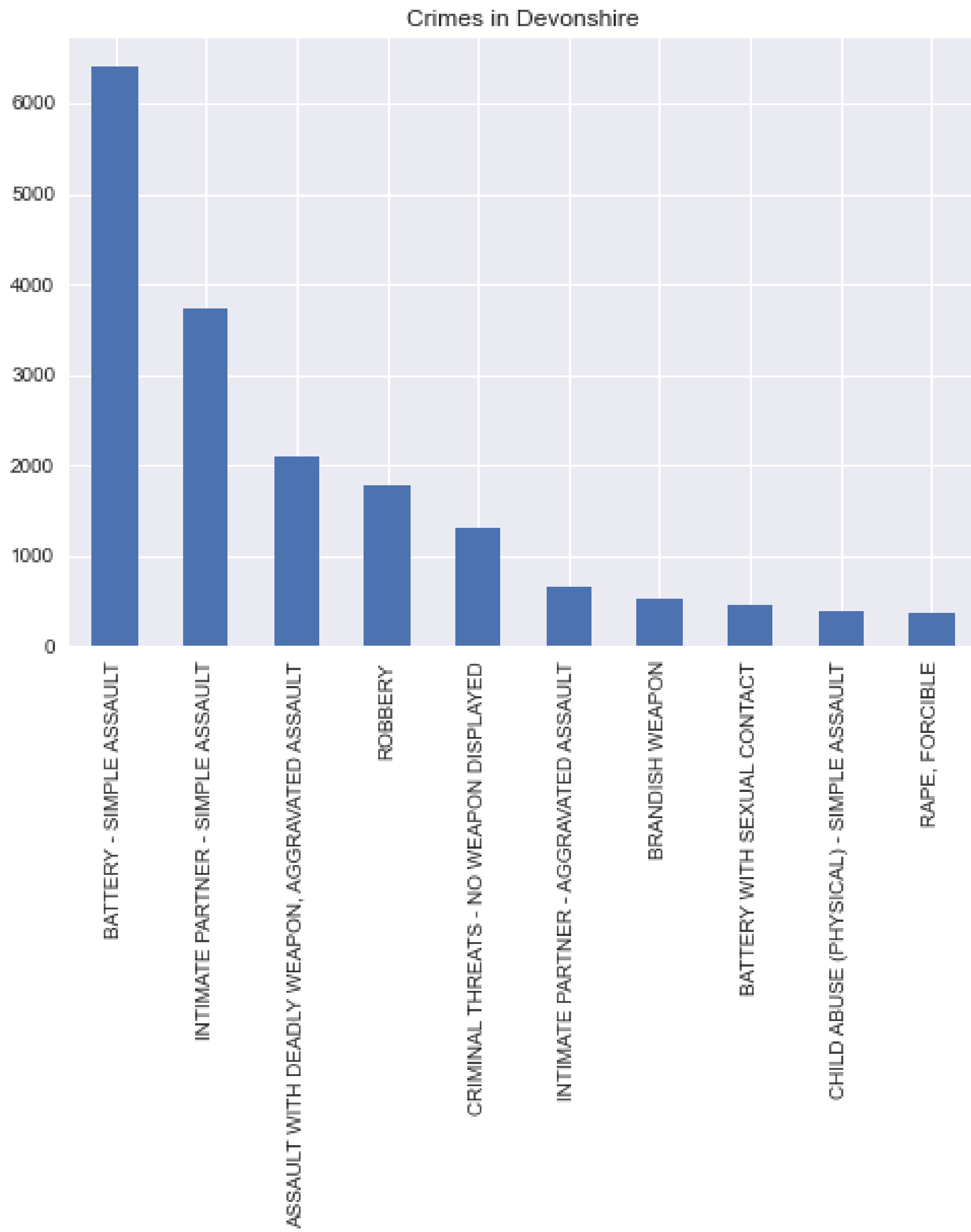
An area-wise Crime Analysis has been visualized via bar plots for the 21 areas in Los Angeles.

```
plt.style.use("seaborn")
color=plt.cm.ocean(np.linspace(0,2,5))
crimeByArea = crimes_new1['AREA NAME'].value_counts().sort_index()
crimeCommonType = {}
for area in crimeByArea.keys():
    crimeArea = crimes_new[crimes_new1['AREA NAME'] == area]['CRM CD
DESC'].value_counts()[:10]
    for crType in crimeArea.keys():
        if not crType in crimeCommonType:
            crimeCommonType[crType] = [area]
        else:
            crimeCommonType[crType].append(area)
    crimeArea = crimeArea.plot(kind = 'bar',title = "Crimes in " +
```

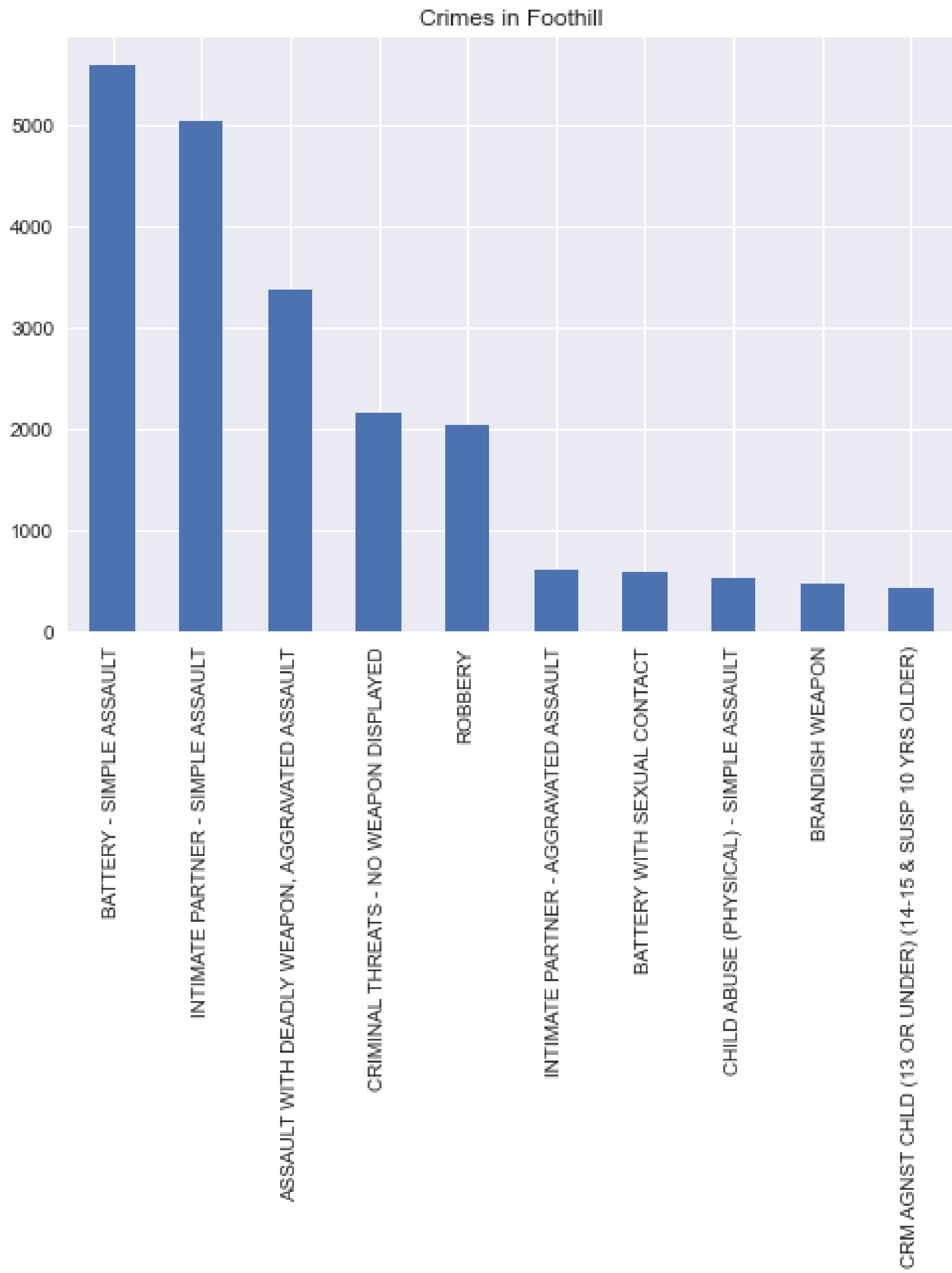
```
area)
    plt.show()
```



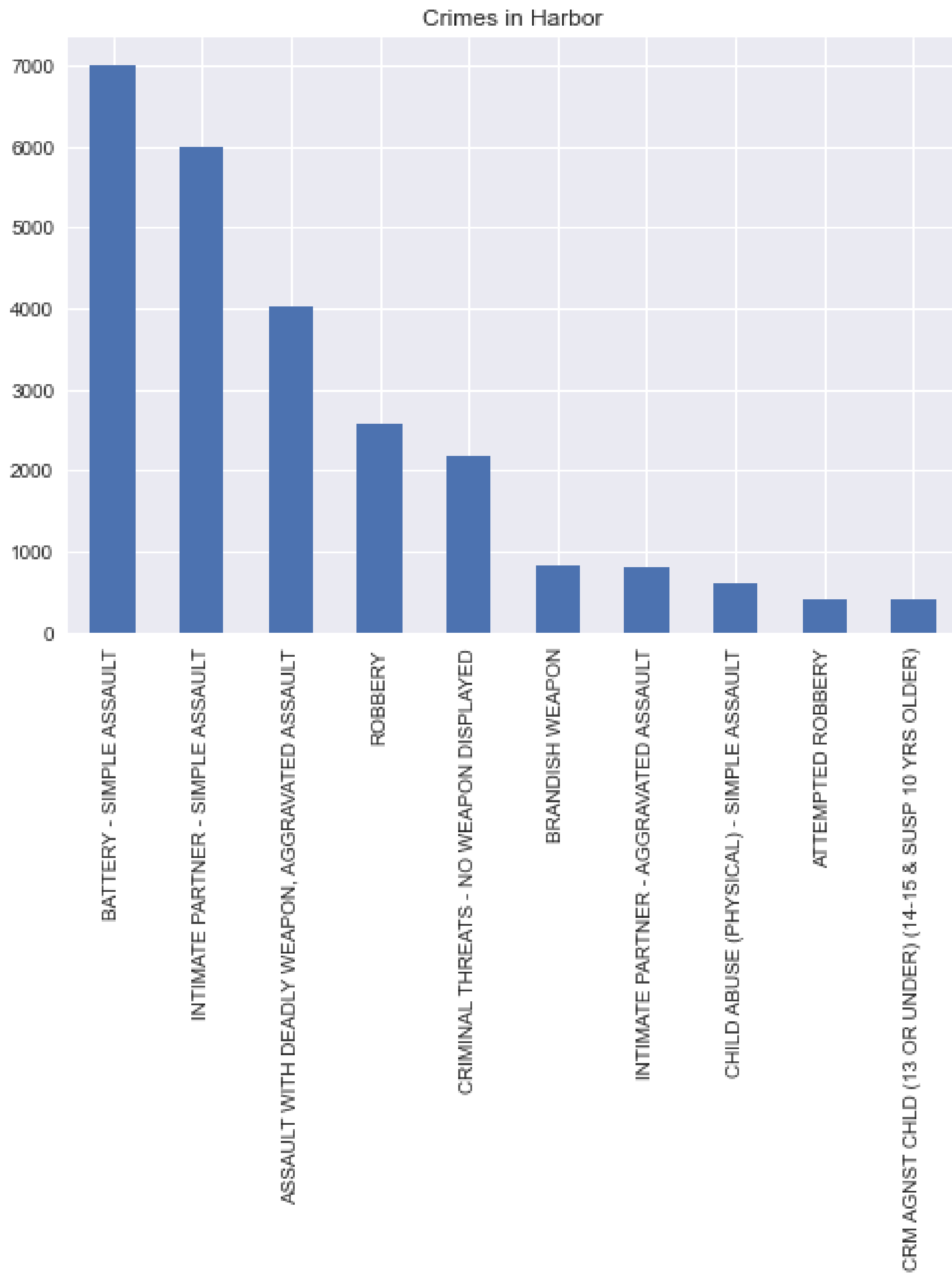

Crimes in Central
0
2000
4000
6000
8000
10000
12000
14000
BATTERY - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
INTIMATE PARTNER - SIMPLE ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
INTIMATE PARTNER - AGGRAVATED ASSAULT
ATTEMPTED ROBBERY
RAPE, FORCIBLE



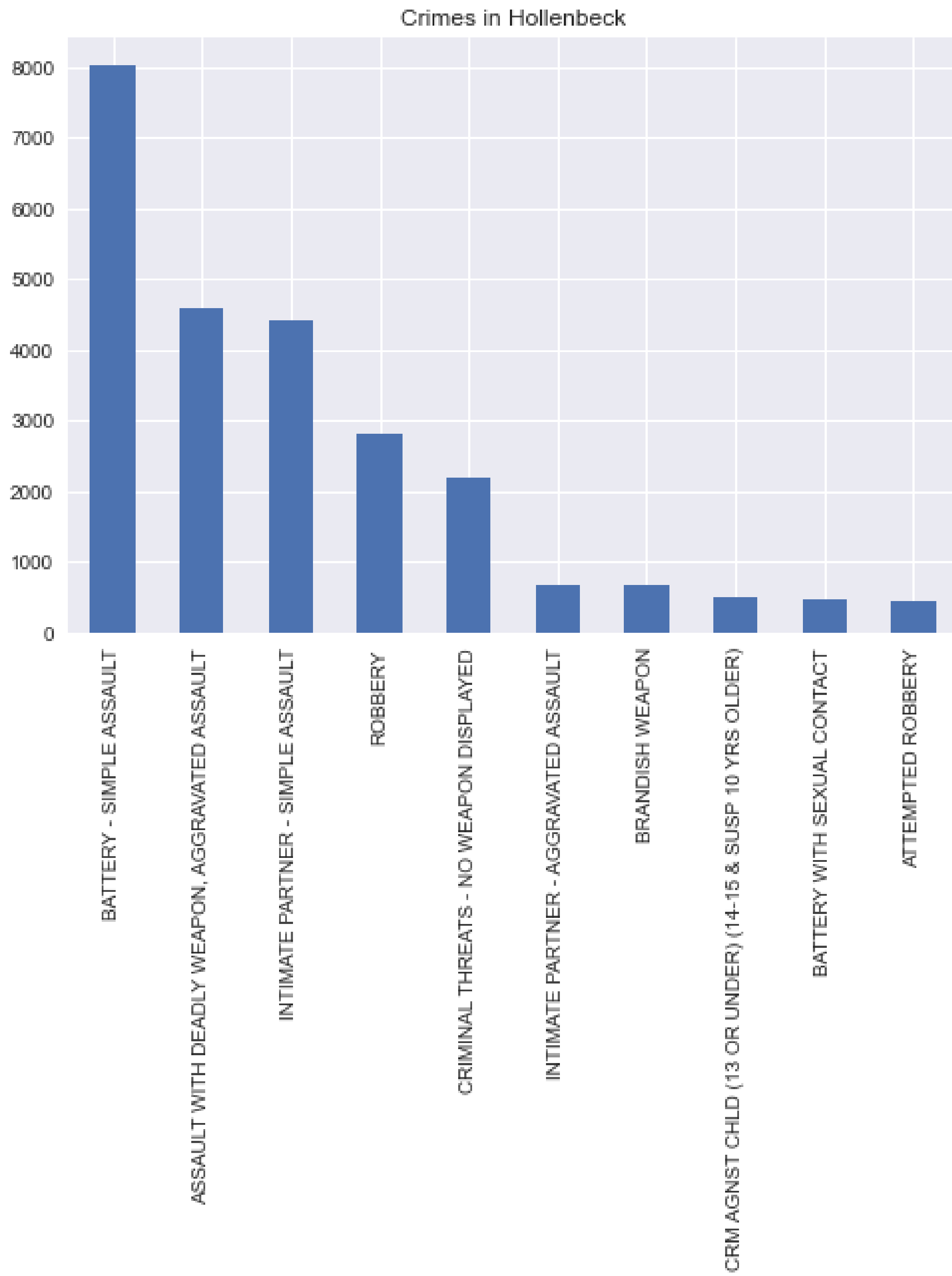
Crimes in Devonshire
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
INTIMATE PARTNER - AGGRAVATED ASSAULT
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
RAPE, FORCIBLE

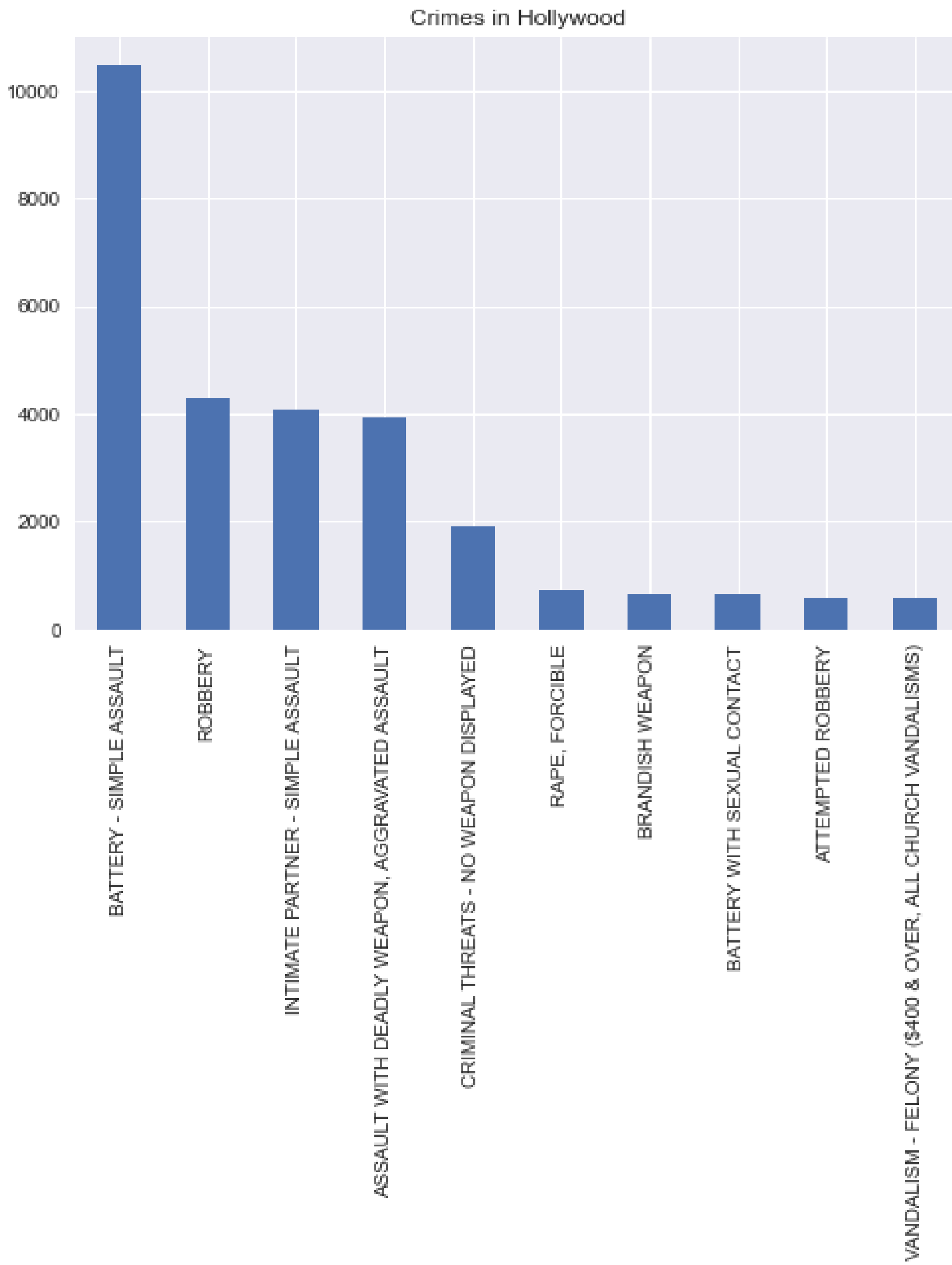


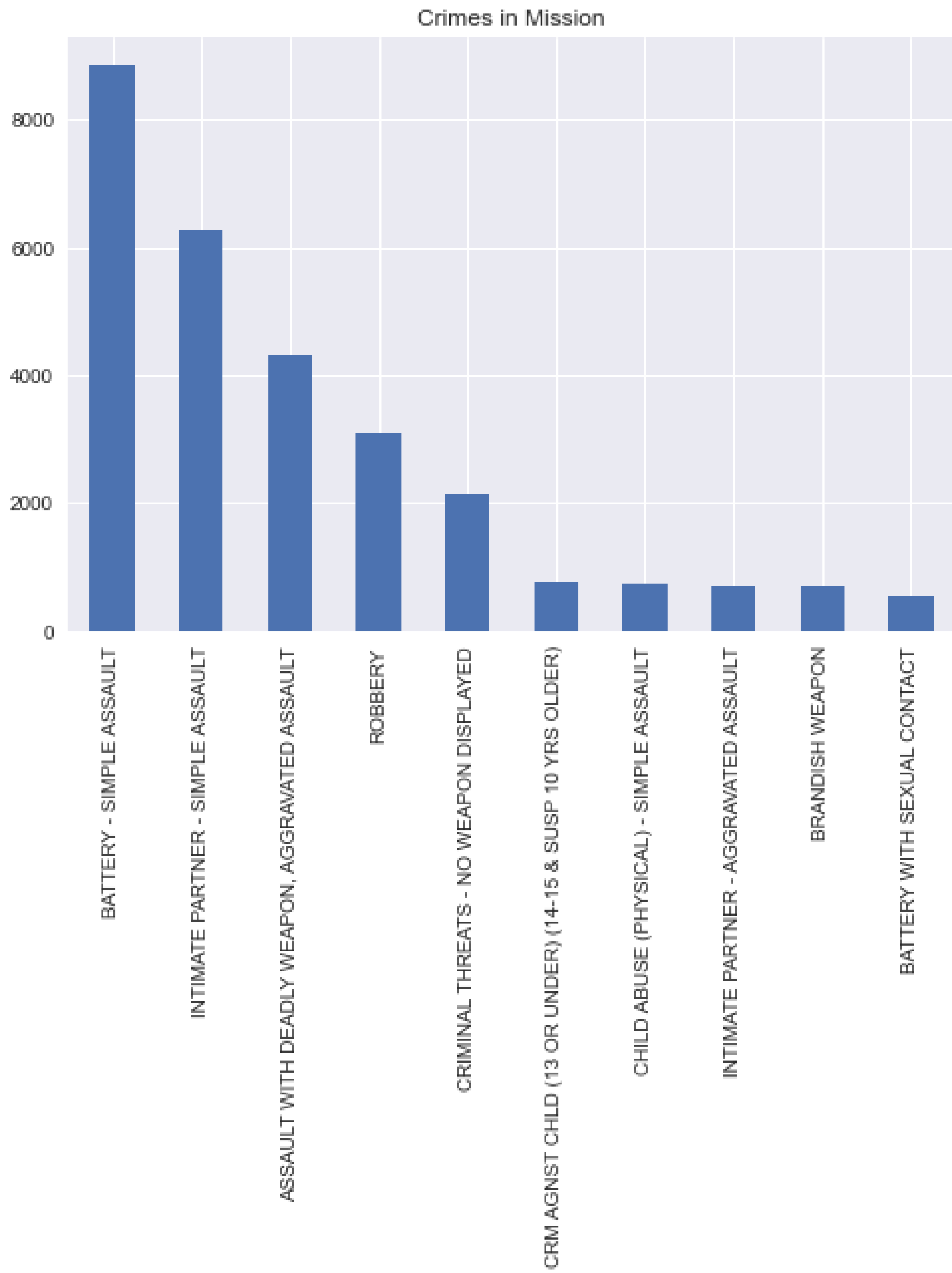
Crimes in Foothill
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
ROBBERY
INTIMATE PARTNER - AGGRAVATED ASSAULT
BATTERY WITH SEXUAL CONTACT
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
BRANDISH WEAPON
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)



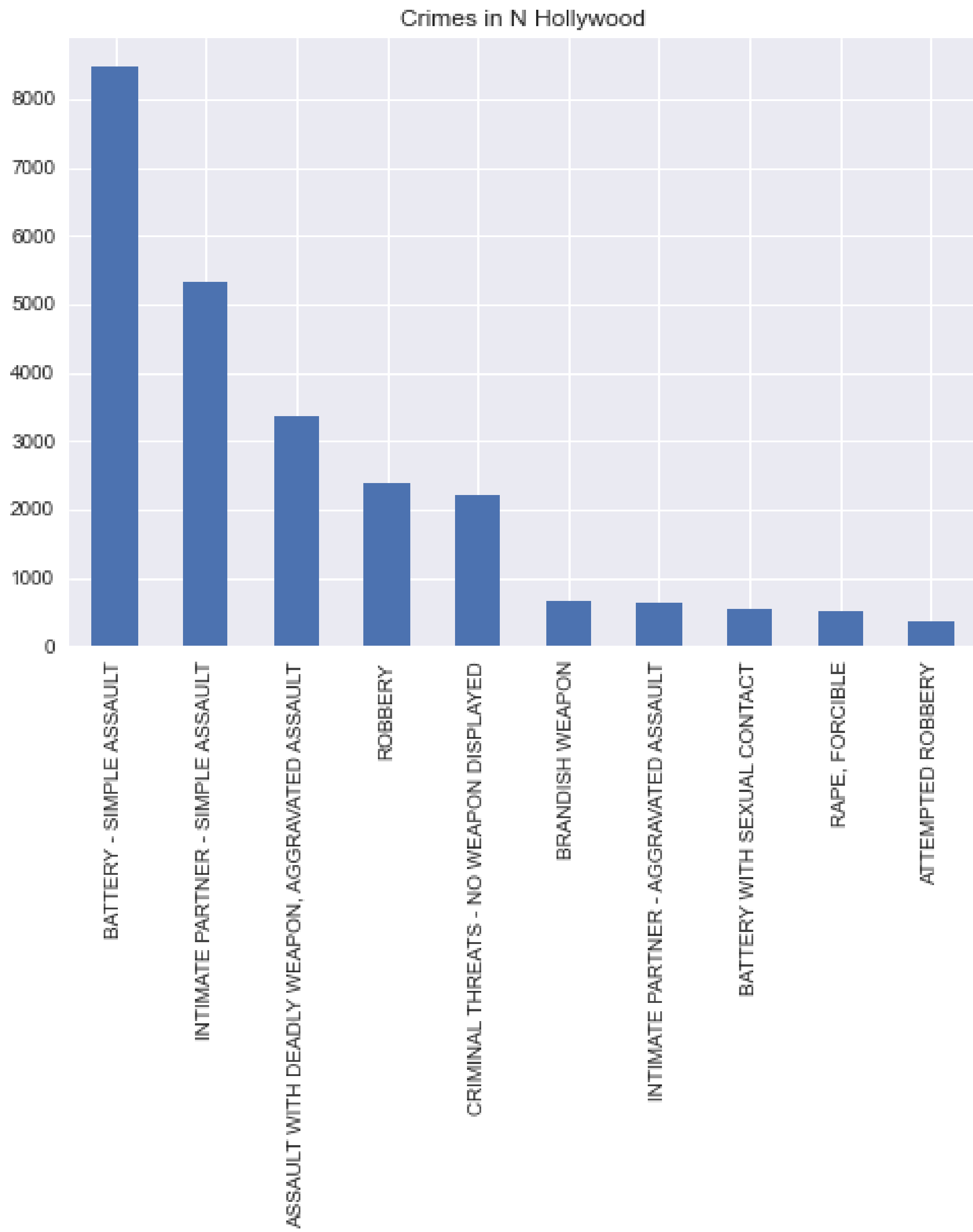
Crimes in Harbor
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
ATTEMPTED ROBBERY
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)


Crimes in Hollenbeck
8000
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
INTIMATE PARTNER - AGGRAVATED ASSAULT
BRANDISH WEAPON
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)
BATTERY WITH SEXUAL CONTACT
ATTEMPTED ROBBERY

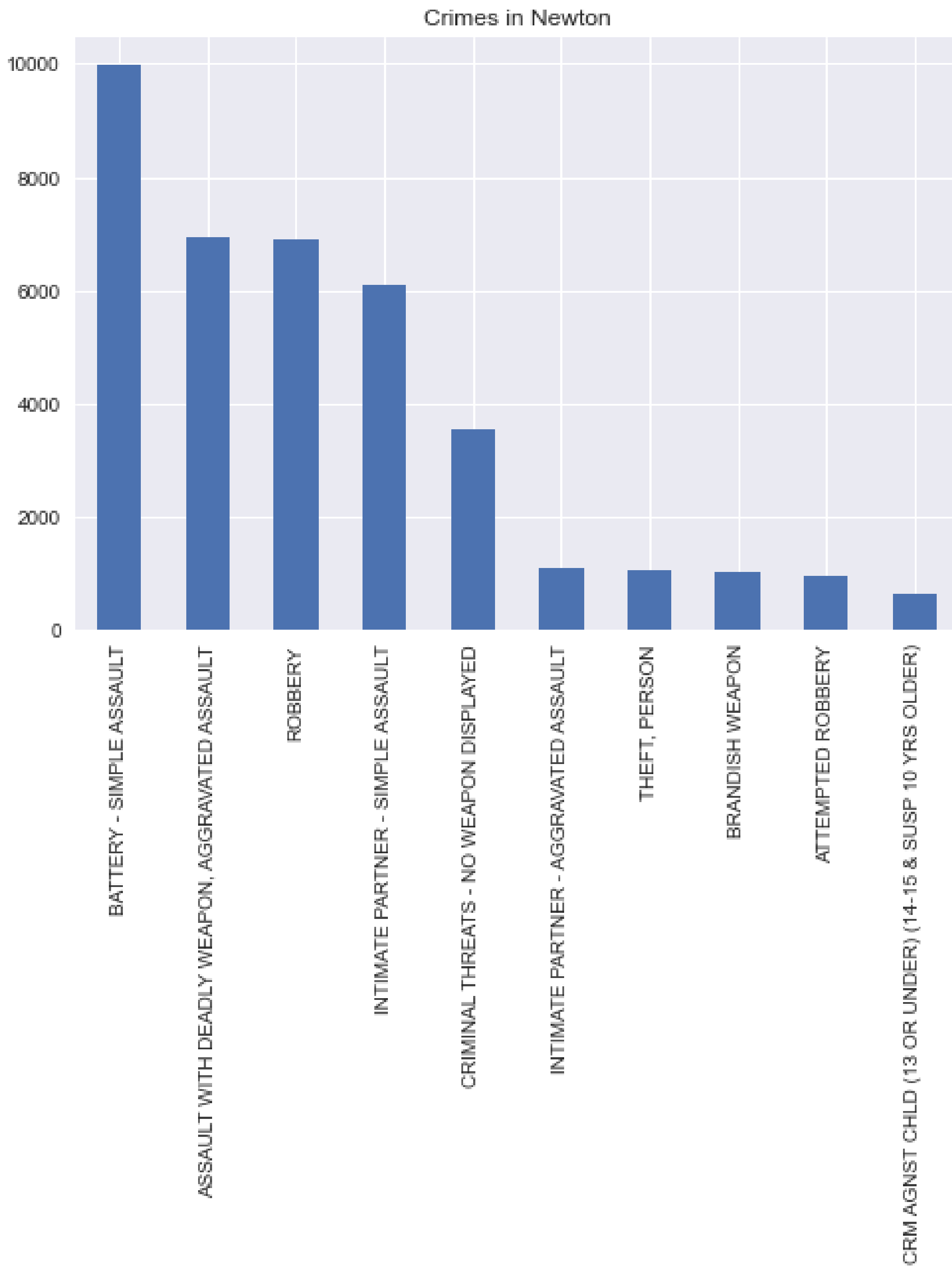

Crimes in Hollywood
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
ROBBERY
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
RAPE, FORCIBLE
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
ATTEMPTED ROBBERY
VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)

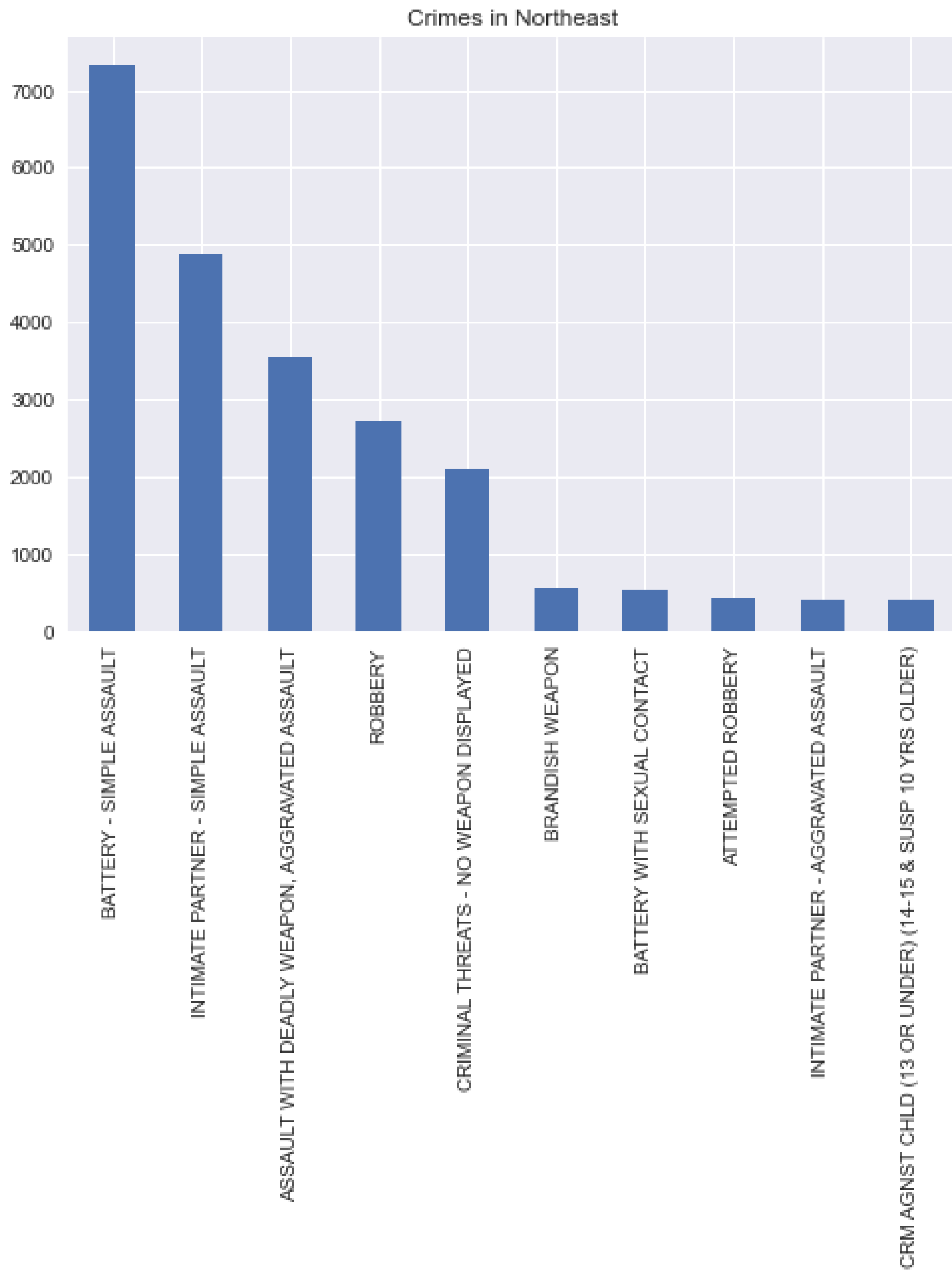


Crimes in Mission
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
INTIMATE PARTNER - AGGRAVATED ASSAULT
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT

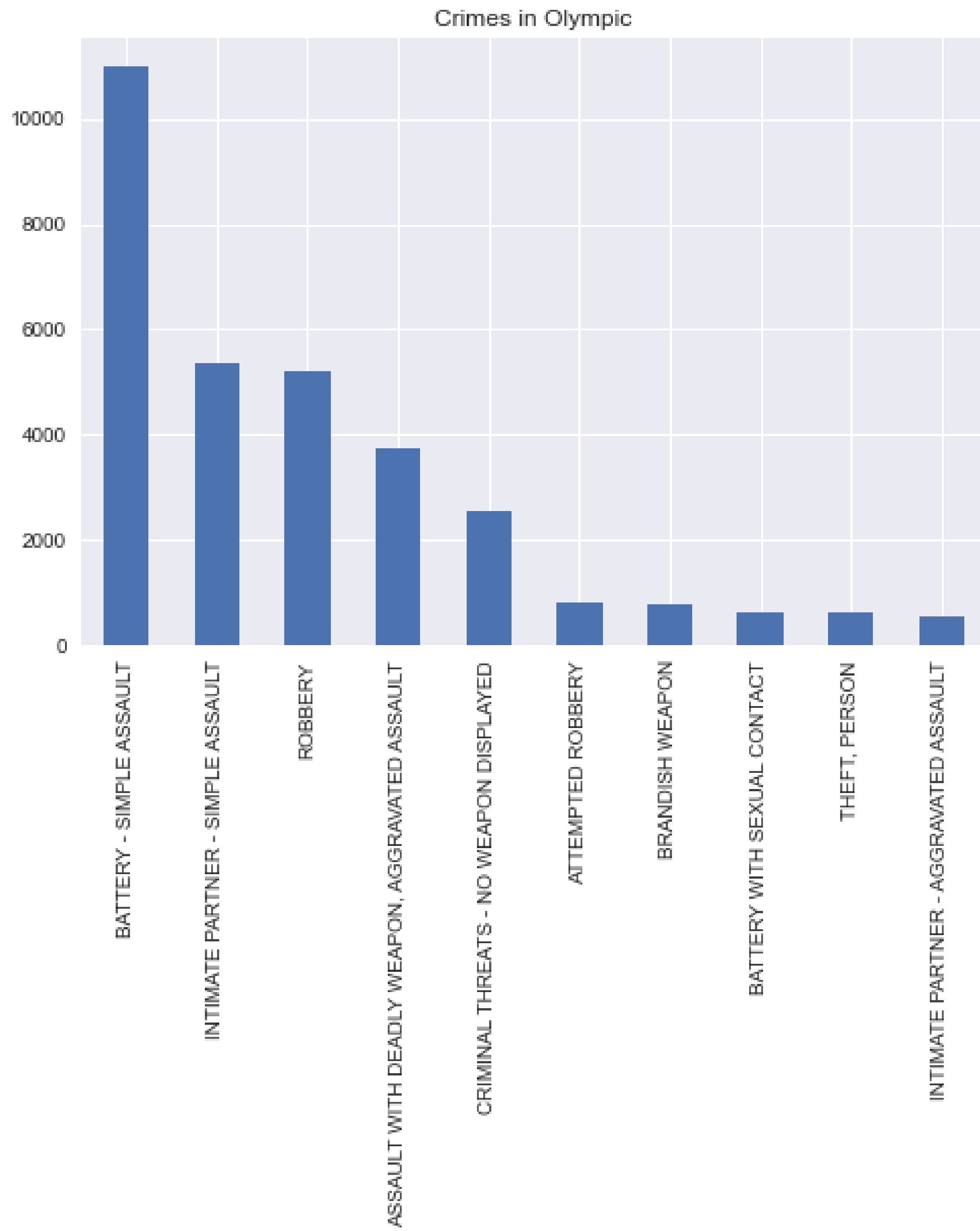


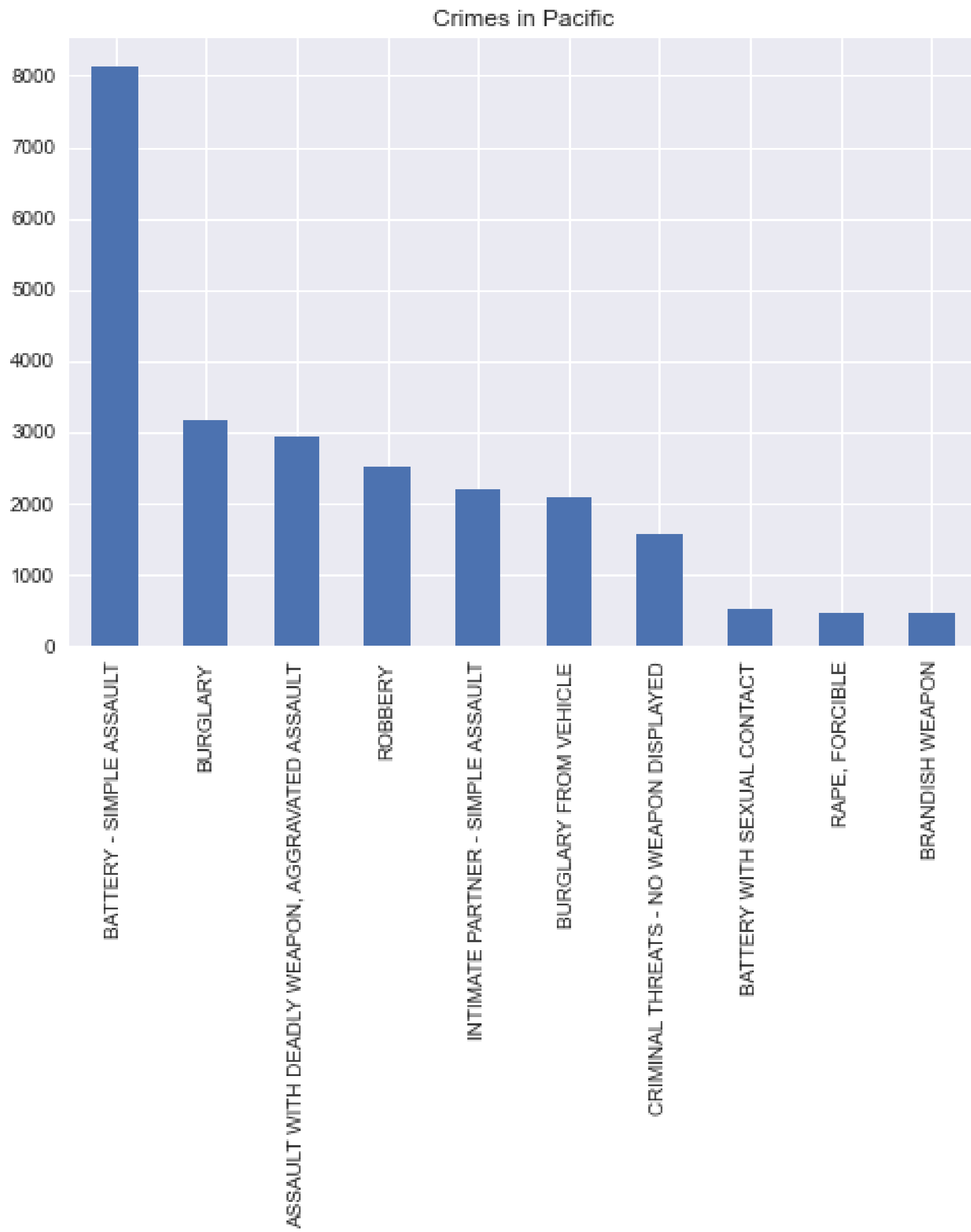
Crimes in N Hollywood
8000
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
BATTERY WITH SEXUAL CONTACT
RAPE, FORCIBLE
ATTEMPTED ROBBERY


Crimes in Newton
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
INTIMATE PARTNER - SIMPLE ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
INTIMATE PARTNER - AGGRAVATED ASSAULT
THEFT, PERSON
BRANDISH WEAPON
ATTEMPTED ROBBERY
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)

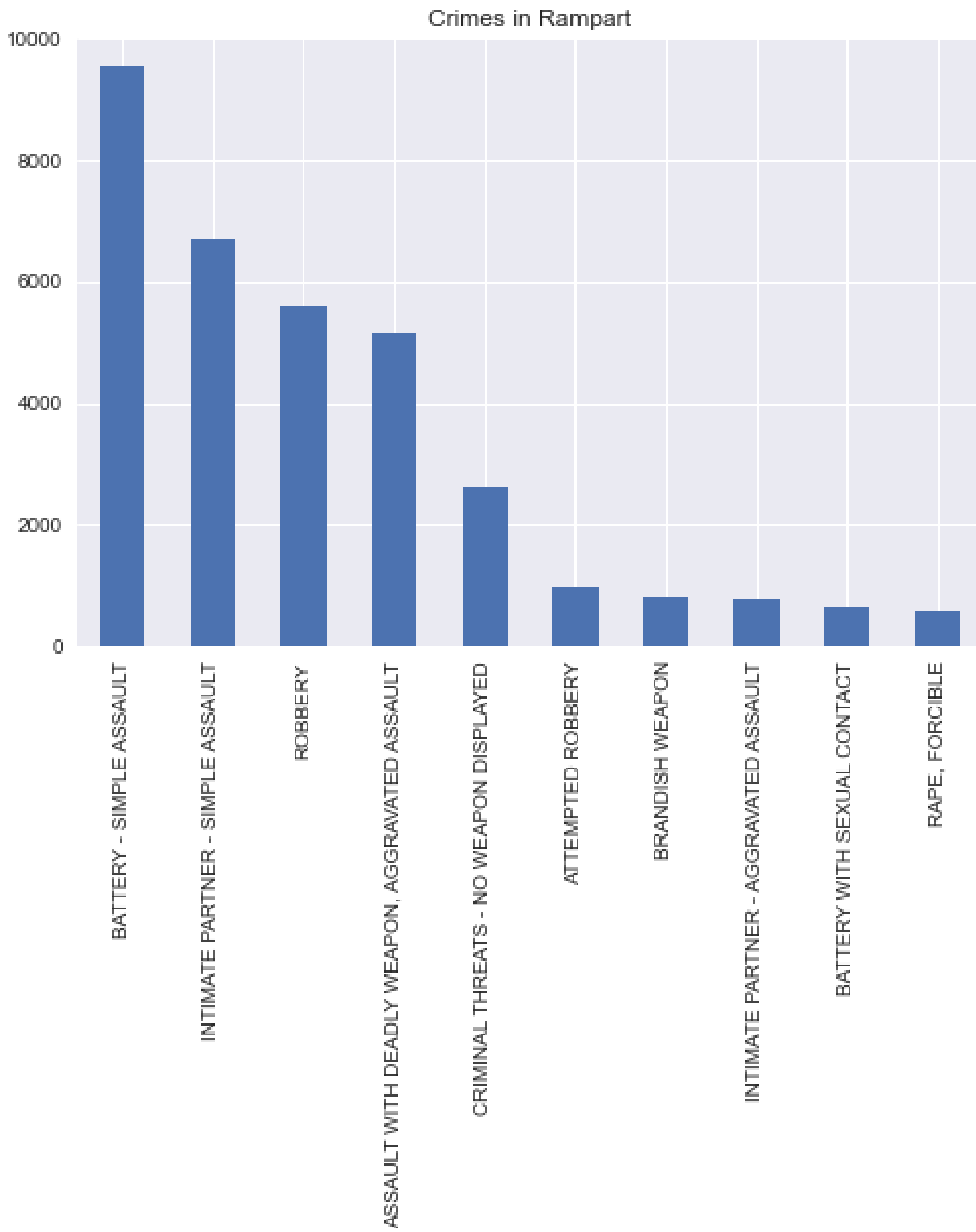


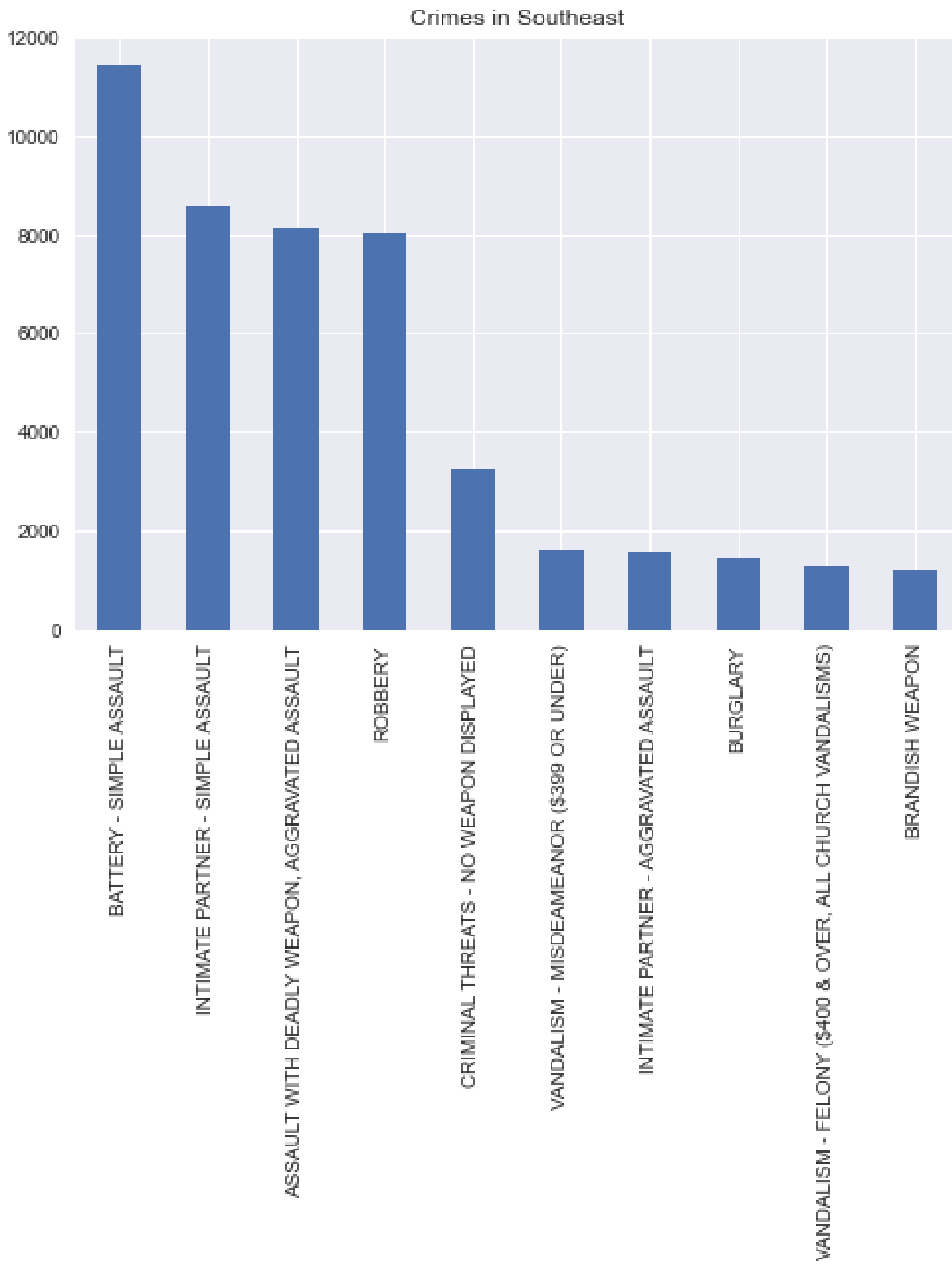
Crimes in Northeast
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
ATTEMPTED ROBBERY
INTIMATE PARTNER - AGGRAVATED ASSAULT
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)

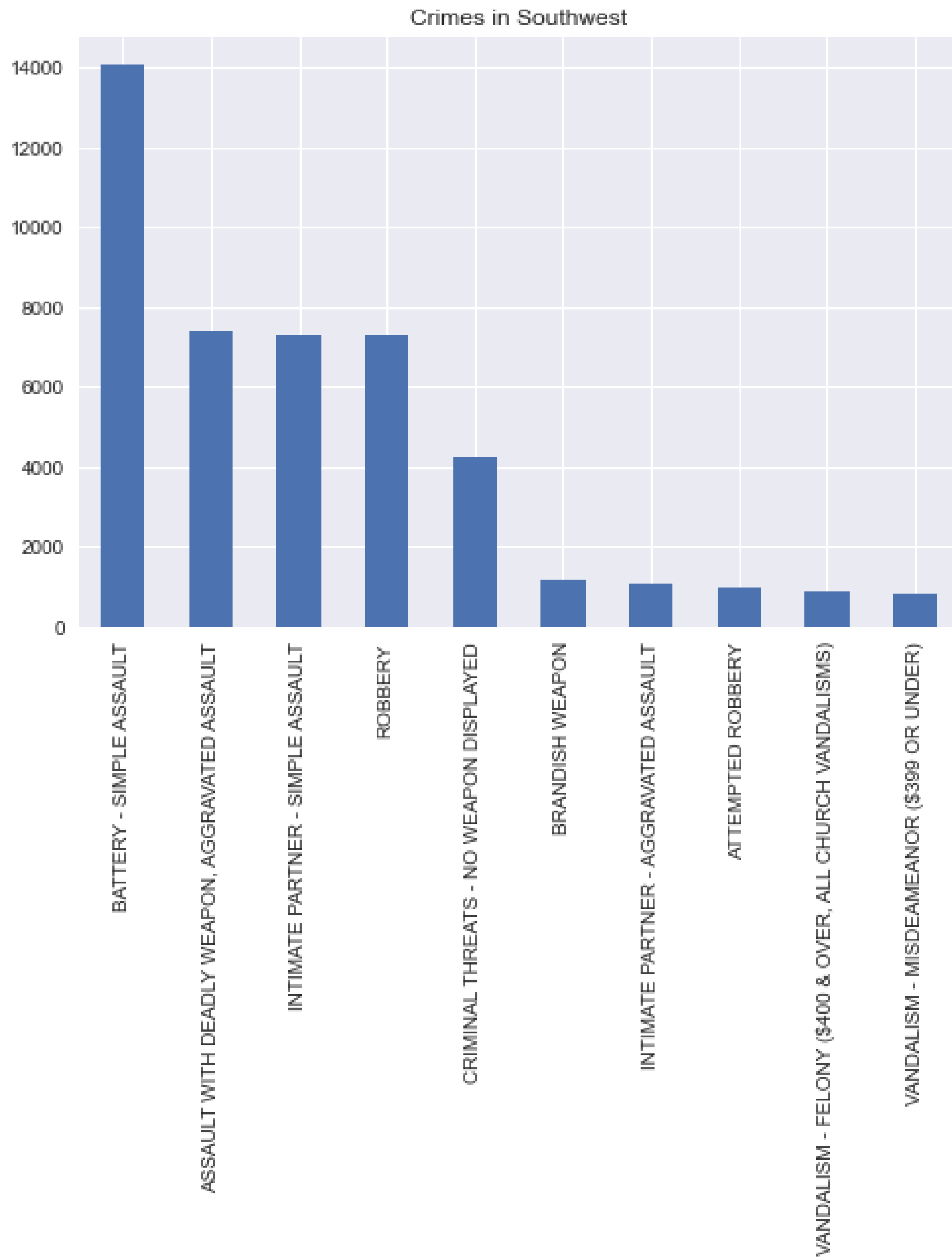

Crimes in Olympic
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ROBBERY
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
ATTEMPTED ROBBERY
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
THEFT, PERSON
INTIMATE PARTNER - AGGRAVATED ASSAULT

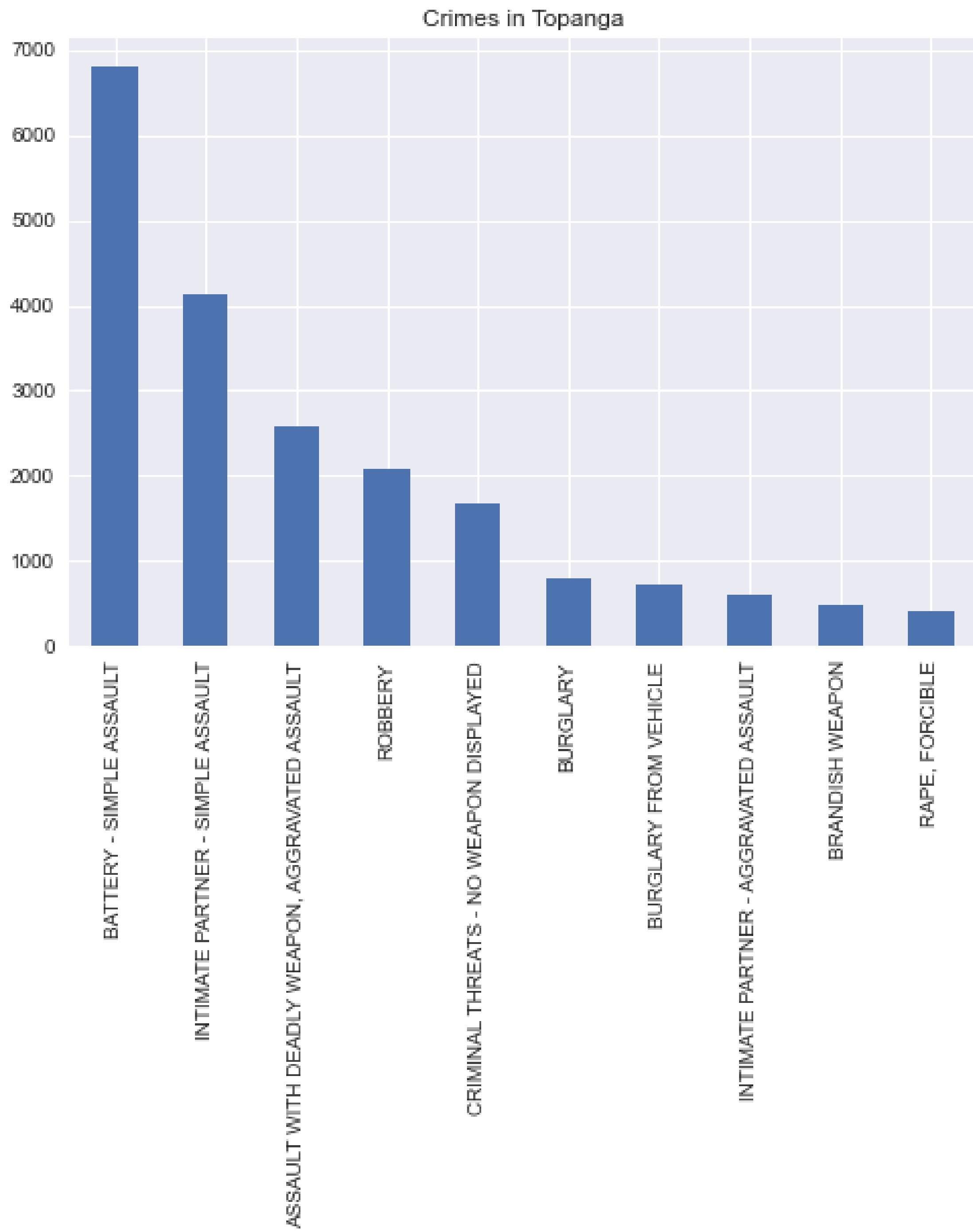


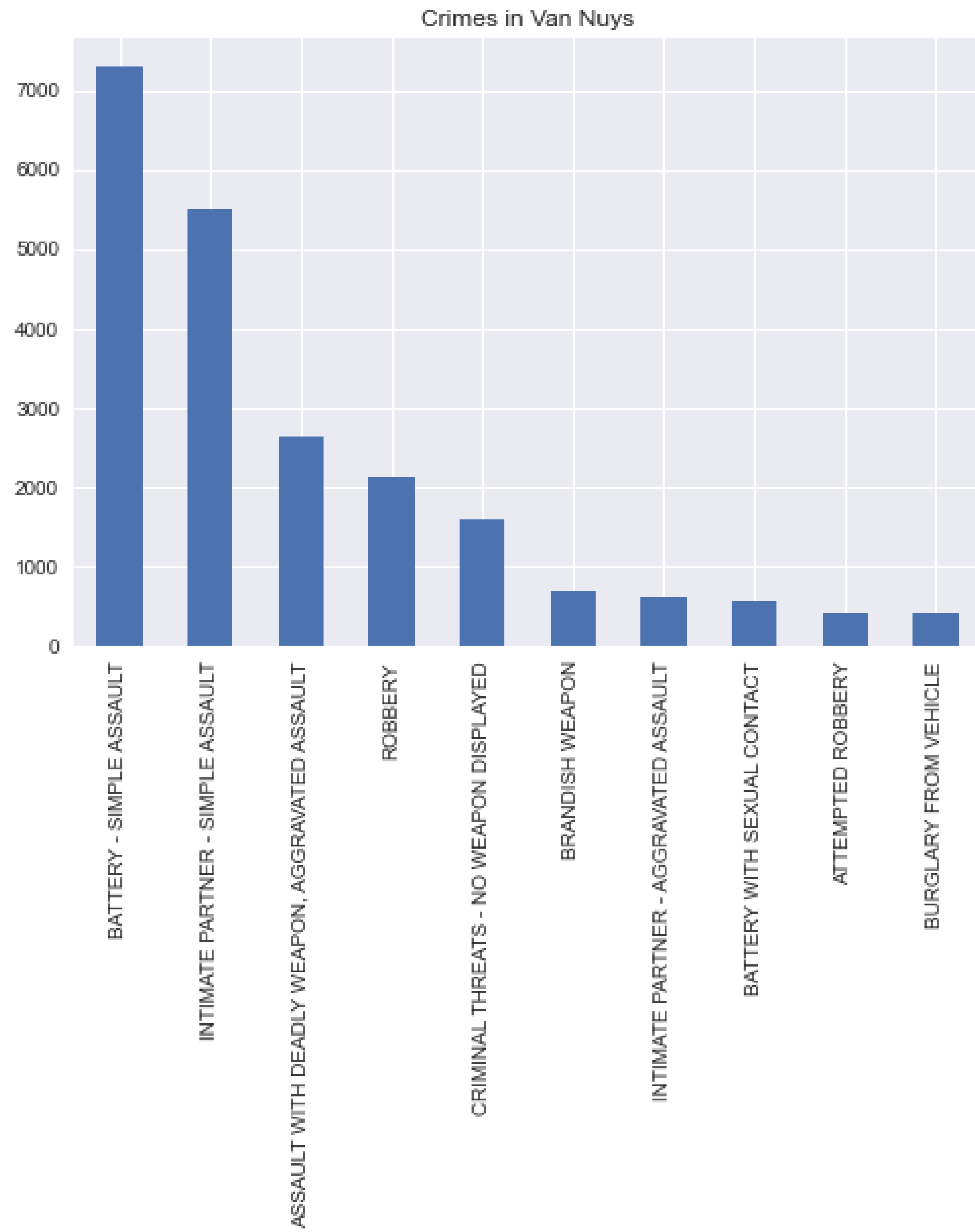
Crimes in Pacific
0
1000
2000
3000
4000
5000
6000
7000
8000
BATTERY - SIMPLE ASSAULT
BURGLARY
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
INTIMATE PARTNER - SIMPLE ASSAULT
BURGLARY FROM VEHICLE
CRIMINAL THREATS - NO WEAPON DISPLAYED
BATTERY WITH SEXUAL CONTACT
RAPE, FORCIBLE
BRANDISH WEAPON

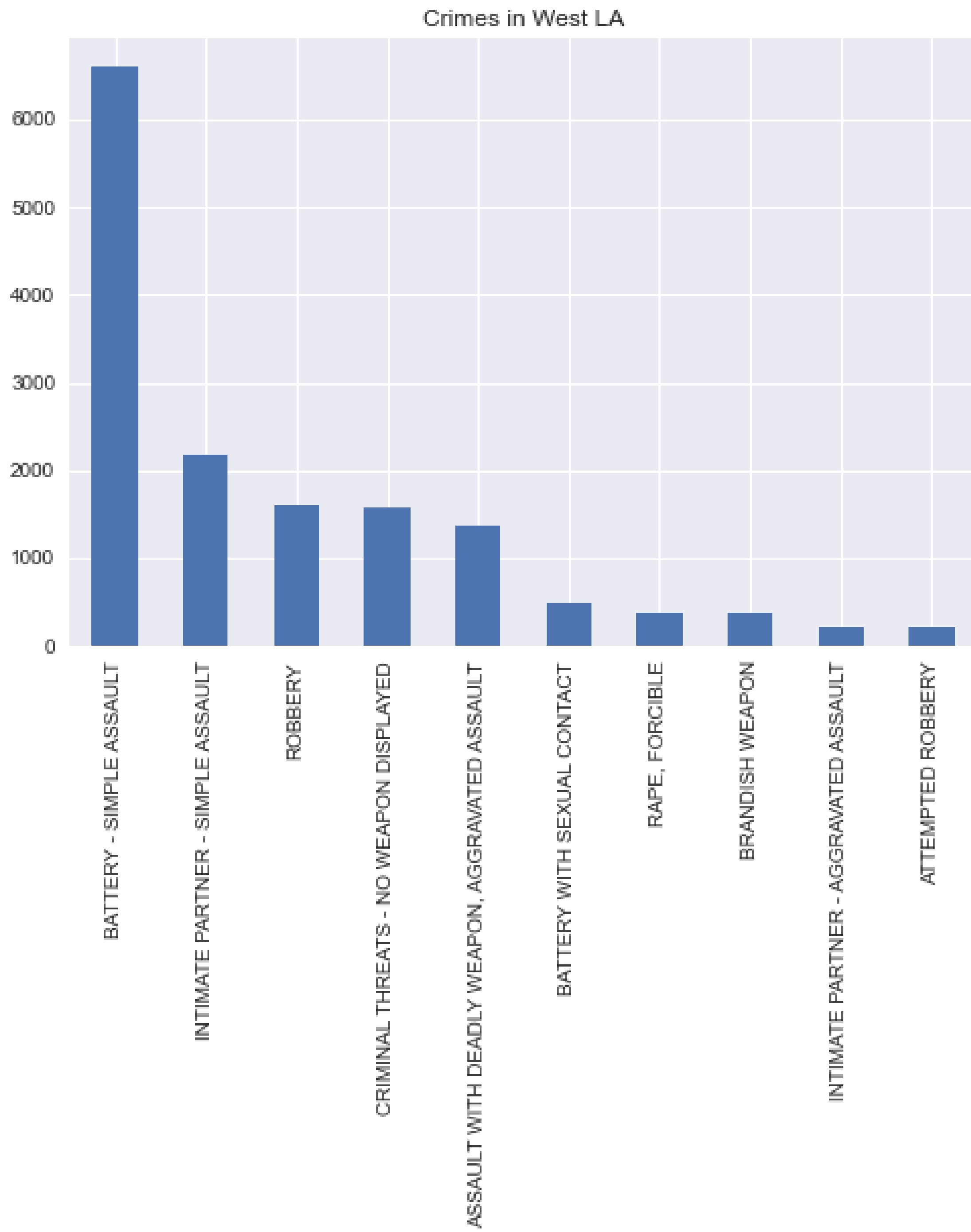

Crimes in Rampart
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ROBBERY
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
CRIMINAL THREATS - NO WEAPON DISPLAYED
ATTEMPTED ROBBERY
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
BATTERY WITH SEXUAL CONTACT
RAPE, FORCIBLE


Crimes in Southeast
12000
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
VANDALISM - MISDEAMEANOR ($399 OR UNDER)
INTIMATE PARTNER - AGGRAVATED ASSAULT
BURGLARY
VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
BRANDISH WEAPON

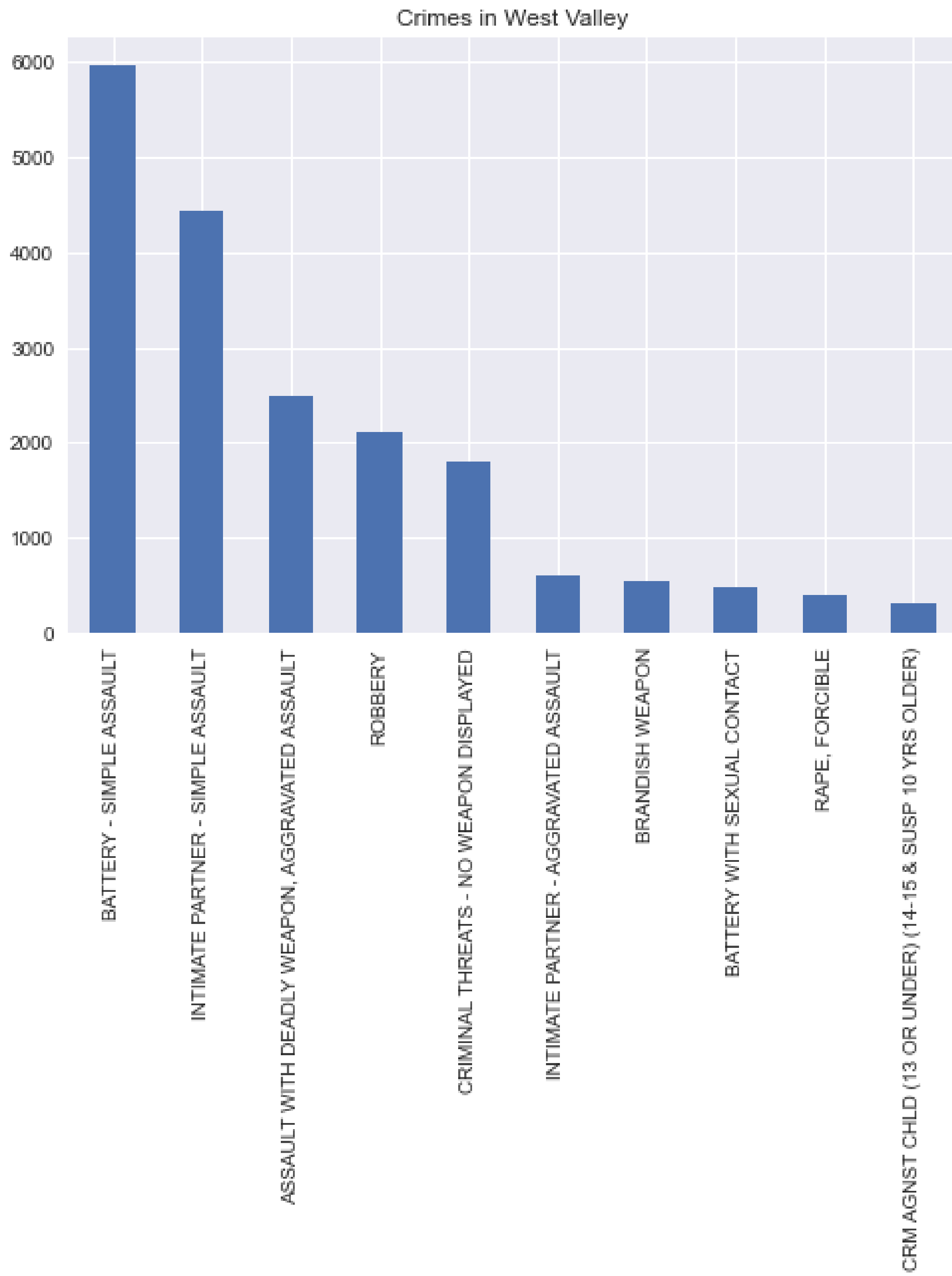

Crimes in Southwest
14000
12000
10000
8000
6000
4000
2000
0
BATTERY - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
ATTEMPTED ROBBERY
VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
VANDALISM - MISDEAMEANOR ($399 OR UNDER)

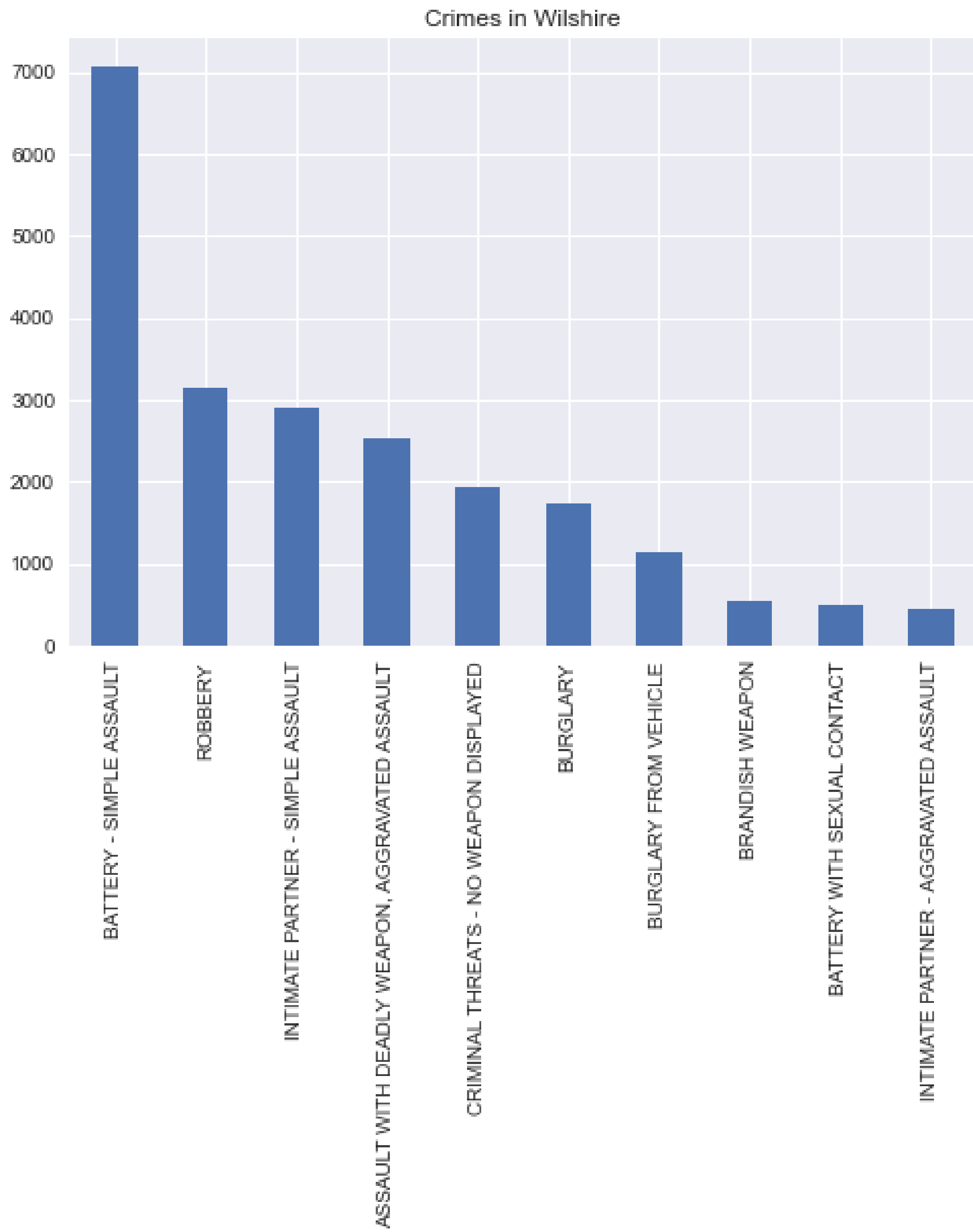

Crimes in Topanga
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BURGLARY
BURGLARY FROM VEHICLE
INTIMATE PARTNER - AGGRAVATED ASSAULT
BRANDISH WEAPON
RAPE, FORCIBLE


Crimes in Van Nuys
7000
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
BATTERY WITH SEXUAL CONTACT
ATTEMPTED ROBBERY
BURGLARY FROM VEHICLE



Crimes in West LA
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
BATTERY WITH SEXUAL CONTACT
RAPE, FORCIBLE
BRANDISH WEAPON
INTIMATE PARTNER - AGGRAVATED ASSAULT
ATTEMPTED ROBBERY


Crimes in West Valley
6000
5000
4000
3000
2000
1000
0
BATTERY - SIMPLE ASSAULT
INTIMATE PARTNER - SIMPLE ASSAULT
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
ROBBERY
CRIMINAL THREATS - NO WEAPON DISPLAYED
INTIMATE PARTNER - AGGRAVATED ASSAULT
BRANDISH WEAPON
BATTERY WITH SEXUAL CONTACT
RAPE, FORCIBLE
CRM AGNST CHLD (13 OR UNDER) (14-15 & SUSP 10 YRS OLDER)



Detecting outliers with boxplot rule

```
#Calculation of first quantile
q1 = crimes_new1['VICT AGE'].quantile(0.25)

#Calculation of third quantile
q3 = crimes_new1['VICT AGE'].quantile(0.75)
```

```
# Calculate the interquantile range (IQR)
IQR = q3 - q1

#Evaluating the outliers
crimes_new1[crimes_new1['VICT AGE']<q1 - 1.5*IQR]
crimes_new1[crimes_new1['VICT AGE']>q3 + 1.5*IQR]
```

```
          DATE OCC  TIME OCC    AREA NAME  \
120     2010-06-04       800      Central
310     2010-01-05       100      Central
866     2010-03-28      2215      Central
976     2010-04-15      1130      Central
1095    2010-05-02       100      Central
...            ...       ...          ...
709793  2019-05-07      1800  77th Street
709911  2019-09-12      2340    Northeast
710025  2019-04-30       230      Pacific
710189  2019-06-12      1500     Foothill
711021  2019-10-27       155      West LA

                                           CRM CD DESC VICT DESCENT
VICT SEX  \
120                           BATTERY - SIMPLE ASSAULT            H
F
310            VANDALISM - MISDEAMEANOR ($399 OR UNDER)            B
M
866                           BATTERY - SIMPLE ASSAULT            H
M
976     ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT            H
F
1095                                    RAPE, FORCIBLE            W
F
...                                                ...          ...
...
709793                        BATTERY - SIMPLE ASSAULT            B
M
709911                        BATTERY - SIMPLE ASSAULT            W
M
710025                                       BURGLARY            W
F
710189               INTIMATE PARTNER - SIMPLE ASSAULT            W
F
711021                        BATTERY - SIMPLE ASSAULT            O
M

       STATUS  YEAR  AREA   CRM CD  CRM CD LOG  VICT AGE  PREMIS CD  \
120        IC  2010   1.0     624    6.437752        83      501.0
310        AA  2010   1.0     745    6.614726        99      203.0
866        IC  2010   1.0     624    6.437752        99      210.0
```

```
976         AA  2010    1.0     230    5.442418        83      502.0
1095        AO  2010    1.0     121    4.804021        86      502.0
...        ...   ...    ...     ...         ...       ...        ...
709793      AO  2019   12.0     624    6.437752        83      501.0
709911      AO  2019   11.0     624    6.437752        89      210.0
710025      AA  2019   14.0     310    5.739793        88      501.0
710189      AO  2019   16.0     626    6.440947        96      719.0
711021      IC  2019    8.0     624    6.437752        86      733.0

        PREMIS LOG  WEAPON USED CD  WEAPON LOG  Year      Month
Day  \
120       6.218600           400.0    5.993961  2010       June
Friday
310       5.318120           400.0    5.993961  2010    January
Tuesday
866       5.351858           400.0    5.993961  2010      March
Sunday
976       6.220590           400.0    5.993961  2010      April
Thursday
1095      6.220590           400.0    5.993961  2010        May
Sunday
...            ...             ...         ...   ...        ...
...
709793    6.218600           400.0    5.993961  2019        May
Tuesday
709911    5.351858           400.0    5.993961  2019  September
Thursday
710025    6.218600           400.0    5.993961  2019      April
Tuesday
710189    6.579251           400.0    5.993961  2019       June
Wednesday
711021    6.598509           400.0    5.993961  2019    October
Sunday

        HOUR OCC
120            8
310            1
866           22
976           11
1095           1
...          ...
709793        18
709911        23
710025         2
710189        15
711021         1

[4077 rows x 20 columns]
```

Although there are 4077 outliers, removing them would not make much of a difference due to a large dataset but we are keeping them for now as they will be required for further analysis of the dataset

```
crimes_train = crimes_new.drop(columns = ['CRM CD DESC','AREA NAME',
                                          'VICT DESCENT',
'STATUS','Year','Month','Day'] ,axis=1, inplace=True)

#VICT SEX is a categorical variable so creating dummies for the
variable to transform the data and use the data for training
crimes_train = pd.get_dummies(crimes_new,columns = ['VICT SEX'])
crimes_train
```

```
         DATE OCC TIME OCC  YEAR  MONTH  DAY  AREA   CRM CD  CRM CD
LOG  \
0      2010-01-05     0150  2010      1    5   6.0     900
6.803505
1      2010-01-02     2100  2010      1    2   1.0     122
4.812184
2      2010-01-08     2100  2010      1    8   1.0     230
5.442418
3      2010-01-09     0230  2010      1    9   1.0     230
5.442418
4      2010-01-14     1445  2010      1   14   1.0     624
6.437752
...           ...      ...   ...    ...  ...   ...     ...          ..
.
711130 2019-01-20     2000  2019      1   20  18.0     930
6.836259
711131 2019-02-23     2220  2019      2   23   9.0     210
5.351858
711132 2019-02-22     0840  2019      2   22   5.0     627
6.442540
711133 2019-03-28     0400  2019      3   28   6.0     648
6.475433
711134 2019-01-06     2100  2019      1    6  20.0     930
6.836259

        VICT AGE  PREMIS CD  PREMIS LOG  WEAPON USED CD  WEAPON LOG
HOUR OCC  \
0             47      101.0    4.624973           102.0    4.634729
1
1             47      103.0    4.644391           400.0    5.993961
21
2             51      710.0    6.566672           500.0    6.216606
21
3             30      108.0    4.691348           400.0    5.993961
2
4             38      101.0    4.624973           400.0    5.993961
14
```

```
...          ...        ...         ...              ...         ...
...
711130        18      108.0    4.691348            511.0    6.238325
20
711131        30      101.0    4.624973            107.0    4.682131
22
711132        14      109.0    4.700480            400.0    5.993961
8
711133         0      706.0    6.561031            506.0    6.228511
4
711134        46      102.0    4.634729            400.0    5.993961
21

        VICT SEX_F  VICT SEX_H  VICT SEX_M  VICT SEX_N  VICT SEX_X
0                1           0           0           0           0
1                1           0           0           0           0
2                0           0           1           0           0
3                0           0           1           0           0
4                1           0           0           0           0
...            ...         ...         ...         ...         ...
711130           1           0           0           0           0
711131           1           0           0           0           0
711132           1           0           0           0           0
711133           0           0           0           0           1
711134           1           0           0           0           0

[711135 rows x 19 columns]
```

```python
# MinMax Scaling of DataFrame
features = np.array(crimes_train.columns).reshape(-1, 1)

for feature in features:
    scaler = MinMaxScaler()
    scaler.fit(crimes_train[feature])
    crimes_train[feature] = scaler.transform(crimes_train[feature])

crimes_train
```

```
        DATE OCC  TIME OCC  YEAR     MONTH       DAY  AREA     CRM CD
\
0       0.001096  0.063189   0.0  0.000000  0.133333   0.25  0.933806

1       0.000274  0.890161   0.0  0.000000  0.033333   0.00  0.014184

2       0.001917  0.890161   0.0  0.000000  0.233333   0.00  0.141844

3       0.002191  0.097116   0.0  0.000000  0.266667   0.00  0.141844

4       0.003561  0.612383   0.0  0.000000  0.433333   0.00  0.607565
```

```
...          ...       ...   ...       ...       ...    ...       ...

711130  0.905505  0.847752   1.0  0.000000  0.633333   0.85  0.969267

711131  0.914818  0.941052   1.0  0.090909  0.733333   0.40  0.118203

711132  0.914544  0.355810   1.0  0.090909  0.700000   0.20  0.611111

711133  0.923856  0.169211   1.0  0.181818  0.900000   0.25  0.635934

711134  0.901671  0.890161   1.0  0.000000  0.166667   0.95  0.969267


        CRM CD LOG  VICT AGE  PREMIS CD  PREMIS LOG  WEAPON USED CD  \
0         0.972010  0.522936   0.000000    0.000000        0.002410
1         0.047651  0.522936   0.002299    0.008613        0.720482
2         0.340202  0.559633   0.700000    0.861300        0.961446
3         0.340202  0.366972   0.008046    0.029443        0.720482
4         0.802229  0.440367   0.000000    0.000000        0.720482
...            ...       ...        ...         ...             ...
711130    0.987214  0.256881   0.008046    0.029443        0.987952
711131    0.298165  0.366972   0.000000    0.000000        0.014458
711132    0.804452  0.220183   0.009195    0.033494        0.720482
711133    0.819721  0.091743   0.695402    0.858797        0.975904
711134    0.987214  0.513761   0.001149    0.004328        0.720482

        WEAPON LOG  HOUR OCC  VICT SEX_F  VICT SEX_H  VICT SEX_M  VICT
SEX_N  \
0         0.006011  0.043478         1.0         0.0         0.0
0.0
1         0.843456  0.913043         1.0         0.0         0.0
0.0
2         0.980631  0.913043         0.0         0.0         1.0
0.0
3         0.843456  0.086957         0.0         0.0         1.0
0.0
4         0.843456  0.608696         1.0         0.0         0.0
0.0
...            ...       ...         ...         ...         ...
...
711130    0.994012  0.869565         1.0         0.0         0.0
0.0
711131    0.035216  0.956522         1.0         0.0         0.0
0.0
711132    0.843456  0.347826         1.0         0.0         0.0
0.0
711133    0.987966  0.173913         0.0         0.0         0.0
0.0
711134    0.843456  0.913043         1.0         0.0         0.0
```

```
0.0

        VICT SEX_X
0              0.0
1              0.0
2              0.0
3              0.0
4              0.0
...            ...
711130         0.0
711131         0.0
711132         0.0
711133         1.0
711134         0.0

[711135 rows x 19 columns]
```

```
# Define features and label for training
train_features = crimes_new[['YEAR', 'MONTH', 'DAY', 'TIME OCC', 'AREA
', 'VICT AGE', 'PREMIS LOG', 'WEAPON LOG']]
train_label = crimes_new['CRM CD LOG'].astype(int)

# Split datasets with 80-20% split
X_train, X_test, y_train, y_test = train_test_split(train_features,
train_label, test_size=0.2, random_state=11)

print('Shape of X_train: ', X_train.shape)
print('Shape of X_test: ', X_test.shape)
print('Shape of y_train: ', y_train.shape)
print('Shape of y_test: ', y_test.shape)
```

```
Shape of X_train:  (568908, 8)
Shape of X_test:  (142227, 8)
Shape of y_train:  (568908,)
Shape of y_test:  (142227,)
```

```
y_train.unique()
```

```
array([5, 6, 4])
```

We will scale the training and testing data for obtaining optimal results

```
scaler = StandardScaler().fit(X_train)

train_X_scale = scaler.transform(X_train)
train_X_scale = pd.DataFrame(train_X_scale)
train_X_scale.columns = X_train.columns

train_X_scale.describe().transpose()
```

```
               count          mean       std       min       25%
50%  \
```

```
YEAR         568908.0 -2.848868e-14  1.000001 -1.608438 -0.921054 
0.110023
MONTH        568908.0  6.569524e-17  1.000001 -1.634168 -0.747523 
0.139122
DAY          568908.0  2.539134e-17  1.000001 -1.636776 -0.850542 
0.048010
TIME OCC     568908.0 -1.278934e-17  1.000001 -1.988302 -0.668676 
0.138659
AREA         568908.0 -4.683596e-18  1.000001 -1.586993 -0.941594 
0.187853
VICT AGE     568908.0  2.172189e-16  1.000001 -2.567541 -0.660515 -
0.124164
PREMIS LOG   568908.0 -1.206944e-15  1.000001 -1.121010 -1.108450 
0.033048
WEAPON LOG   568908.0  1.801748e-15  1.000001 -2.743473  0.336023 
0.336023

                  75%       max  
YEAR         0.797407  1.484791  
MONTH        0.730219  1.616865  
DAY          0.834244  1.732796  
TIME OCC     0.843242  1.472963  
AREA         0.833252  1.640000  
VICT AGE     0.710160  3.928267  
PREMIS LOG   0.933044  1.781079  
WEAPON LOG   0.336023  0.907571  
```

```
test_X_scale = scaler.transform(X_test)
test_X_scale = pd.DataFrame(train_X_scale)
test_X_scale.columns = X_train.columns

test_X_scale.describe().transpose()
```

```
                count          mean       std       min       25%  
50%  \
YEAR         568908.0 -2.848868e-14  1.000001 -1.608438 -0.921054 
0.110023
MONTH        568908.0  6.569524e-17  1.000001 -1.634168 -0.747523 
0.139122
DAY          568908.0  2.539134e-17  1.000001 -1.636776 -0.850542 
0.048010
TIME OCC     568908.0 -1.278934e-17  1.000001 -1.988302 -0.668676 
0.138659
AREA         568908.0 -4.683596e-18  1.000001 -1.586993 -0.941594 
0.187853
VICT AGE     568908.0  2.172189e-16  1.000001 -2.567541 -0.660515 -
0.124164
PREMIS LOG   568908.0 -1.206944e-15  1.000001 -1.121010 -1.108450 
0.033048
WEAPON LOG   568908.0  1.801748e-15  1.000001 -2.743473  0.336023
```

```
0.336023

                  75%       max
YEAR        0.797407  1.484791
MONTH       0.730219  1.616865
DAY         0.834244  1.732796
TIME OCC    0.843242  1.472963
AREA        0.833252  1.640000
VICT AGE    0.710160  3.928267
PREMIS LOG  0.933044  1.781079
WEAPON LOG  0.336023  0.907571
```

We will be using this training data for ML models and test data to predict the performance of the models

Predictive Modelling using Data Mining Methods

We will be doing our predictions of type of crime committed based on the performance of various data mining models

Logistic Regression

Logistic Regression is a classification technique used in data mining and machine learning. It uses a logistic function to model the dependent variable.

```
model = LogisticRegression(solver = 'liblinear')

model.fit(X_train,y_train)

pred_y_lr = model.predict(X_test)

pred_prob_lr = model.predict_proba(X_test)

#Predicting the performance metrics of the model.
from sklearn.metrics import classification_report, confusion_matrix, 
accuracy_score
result = classification_report(y_test,pred_y_lr)
print(result)
result1 = accuracy_score(y_test,pred_y_lr)
print("Accuracy score:",result1)
```

```
              precision    recall  f1-score   support

           4       0.00      0.00      0.00      2847
           5       0.79      0.38      0.51     48292
           6       0.73      0.95      0.83     91088

    accuracy                           0.74    142227
```

```
   macro avg       0.50      0.44      0.45    142227
weighted avg       0.73      0.74      0.70    142227
```

```
Accuracy score: 0.7382775422388154
```

```
#Calculating AUC for this model
print('AUC SCORE:
{0:0.3f}'.format(roc_auc_score(y_test,pred_prob_lr,multi_class =
'ovr')))
```

```
AUC SCORE: 0.715
```

The AUC score for this model is **71.9** which is very low. The accuracy is **74**. Checking other models for better performance

Random Forest

Random forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.

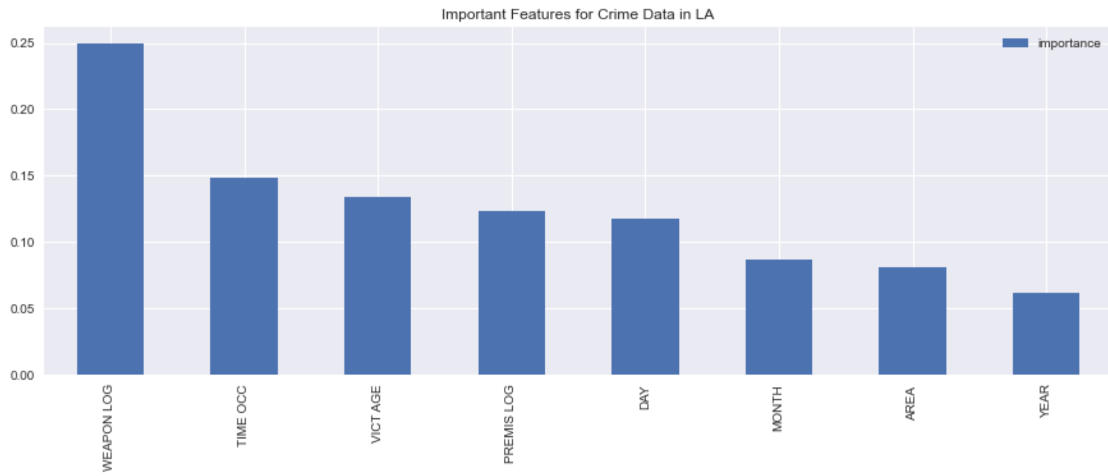
```
rf_reg = RandomForestClassifier(n_estimators = 40)
```

```
rf_reg.fit(X_train,y_train)
```

```
RandomForestClassifier(n_estimators=40)
```

```
#We will use feature importance to define the best predictors
rf_reg.feature_importances_

feature_importances = pd.DataFrame(rf_reg.feature_importances_,
                                  index = X_train.columns,
                                  columns = ['importance'])
feature_importances =
feature_importances.sort_values('importance',ascending = False)

feature_importances.plot(kind = 'bar',figsize = (15,5),
                        title ='Important Features for Crime Data in
LA')
```

```
<AxesSubplot:title={'center':'Important Features for Crime Data in
LA'}>
```




```
pred_y_rf = rf_reg.predict(X_test)

pred_prob_rf = rf_reg.predict_proba(X_test)#Predicting probability as regular prediction for classifier will not work properly

#Performance metrics
result = classification_report(y_test,pred_y_lr)
print(result)
result1 = accuracy_score(y_test,pred_y_lr)
print("Accuracy score:",result1)
```

```
              precision    recall  f1-score   support

           4       0.00      0.00      0.00      2847
           5       0.79      0.38      0.51     48292
           6       0.73      0.95      0.83     91088

    accuracy                           0.74    142227
   macro avg       0.50      0.44      0.45    142227
weighted avg       0.73      0.74      0.70    142227

Accuracy score: 0.7382775422388154
```

```
result3 = roc_auc_score(y_test,pred_prob_rf,multi_class = 'ovr')
print("AUC Score:",result3)
```

```
AUC Score: 0.7929022063931249
```

The accuracy for Random Forest is **74** and auc score **79.6**. Now let us count the number of values obtained for crime code

```
#Based on model predictors, we count the number of values obtained for each crime code
p = pred_y_rf.tolist()

count = dict()
for i in p:
```

```
    count[i] = count.get(i,0)+1
count
```

```
{6: 99827, 5: 42302, 4: 98}
```

```
data = count
pd.DataFrame.from_dict(data,orient = 'index', dtype = None,columns = 
['CRM CD LOG'])
```

```
   CRM CD LOG
6       99827
5       42302
4          98
```

Using this model, we observed that Rape, Assaults and violation of court order are the crimes that ere reported the highest in the past 20 years.

K-NEAREST NEIGHBORS

K-nearest neighbors (KNN) is a type of supervised learning algorithm used for both regression and classification. KNN tries to predict the correct class for the test data by calculating the distance between the test data and all the training points.

```
classifier_knn= KNeighborsClassifier(n_neighbors=5)
classifier_knn.fit(X_train,y_train)
```

```
KNeighborsClassifier()
```

```
pred_y_knn= classifier_knn.predict(X_test)
```

```
pred_prob_knn = classifier_knn.predict_proba(X_test)
```

```
#Performance metrics
result = classification_report(y_test,pred_y_knn)
print(result)
result1=accuracy_score(y_test,pred_y_knn)
print("accuracy score")
print(result1)
```

```
              precision    recall  f1-score   support

           4       0.09      0.02      0.03      2847
           5       0.42      0.31      0.35     48292
           6       0.67      0.78      0.72     91088

    accuracy                           0.60    142227
   macro avg       0.39      0.37      0.37    142227
weighted avg       0.57      0.60      0.58    142227

accuracy score
0.6034226975187552
```

```
result3 = metrics.roc_auc_score(y_test, pred_prob_knn, multi_class =
'ovr')
print("AUC Score:",result3)
```

```
AUC Score: 0.5651080799259621
```

The accuracy for this model is **60.4** and auc score is **56.5** which is not suitable for this model

```
#Based on model predictors, we count the number of values obtained for
each crime code
p1 = pred_y_rf.tolist()

count = dict()
for i in p1:
    count[i] = count.get(i,0)+1
count
```

```
{6: 99827, 5: 42302, 4: 98}
```

```
data = count
pd.DataFrame.from_dict(data,orient = 'index', dtype = None,columns =
['CRM CD LOG'])
```

```
   CRM CD LOG
6       99827
5       42302
4          98
```

Using this model, we observed that Rape, Assaults and violation of court order are the crimes that ere reported the highest in the past 20 years.

DECISION TREE

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

```
from sklearn import tree
# Decision trees for classification, using entropy criterion
dt = tree.DecisionTreeClassifier(criterion='entropy')
dt.fit(X_train, y_train)
```

```
DecisionTreeClassifier(criterion='entropy')
```

```
#Fitting the model
pred_y_dt = dt.predict(X_test)

pred_prob_dt = dt.predict_proba(X_test)

from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
#Classification report
```

```
result1=classification_report(y_test,pred_y_dt)
print("Classification report")
print(result1)

#Accuracy Score of the model
result2=accuracy_score(y_test,pred_y_dt)
print("accuracy score")
print(result2)
```

```
Classification report
              precision    recall  f1-score   support

           4       0.08      0.10      0.09      2847
           5       0.59      0.60      0.59     48292
           6       0.78      0.76      0.77     91088

    accuracy                           0.69    142227
   macro avg       0.48      0.49      0.48    142227
weighted avg       0.70      0.69      0.70    142227

accuracy score
0.6933563950586035
```

```
#AUC Score of the model
result3 = metrics.roc_auc_score(y_test, pred_prob_dt,multi_class =
'ovr')
print(result3)
```

```
0.6374793015673733
```

We can see that the accuracy score for this model is **69.5** and the auc score for this model is **63.7**

```
#Based on model predictors, we count the number of values obtained for
each crime code
p2 = pred_y_rf.tolist()

count = dict()
for i in p2:
    count[i] = count.get(i,0)+1
count
```

```
{6: 99827, 5: 42302, 4: 98}
```

```
data = count
pd.DataFrame.from_dict(data,orient = 'index', dtype = None,columns =
['CRM CD LOG'])
```

```
   CRM CD LOG
6       99827
5       42302
4          98
```

Using this model, we observed that Rape, Assaults and violation of court order are the crimes that ere reported the highest in the past 20 years.

Gaussian Naive Bayes

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian distribution. An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions.

```
from sklearn.naive_bayes import GaussianNB
model_GNB= GaussianNB()
model_GNB.fit(X_train,y_train)# fitting the training data in the model
```

```
GaussianNB()
```

```
pred_y_gnb=model_GNB.predict(X_test)
```

```
pred_prob_gnb = dt.predict_proba(X_test)
```

```
result1=classification_report(y_test,pred_y_gnb)
print("Classification report")
print(result1)
result2=accuracy_score(y_test,pred_y_gnb)
print("Accuracy score")
print(result2)
```

```
Classification report
              precision    recall  f1-score   support

           4       0.00      0.00      0.00      2847
           5       0.77      0.38      0.51     48292
           6       0.73      0.95      0.82     91088

    accuracy                           0.74    142227
   macro avg       0.50      0.44      0.45    142227
weighted avg       0.73      0.74      0.70    142227

Accuracy score
0.7356971601735254
```

```
result3 = metrics.roc_auc_score(y_test,
pred_prob_gnb,multi_class="ovr")
print("AUC Score:",result3)
```

```
AUC Score: 0.6374793015673733
```

We can see that the accuracy score for this model is **73.7** and the auc score for this model is **72.7**

Artificial Neural Network

A neural network is a collection of algorithms that recognize underlying relationships in a set of data using a method that replicates how the human brain works. The artificial neural network (ANN) integrates information in the same manner that the human brain processes.

```
from sklearn import neural_network
# Specify an ANN model, use 1 hidden layer with 20 nodes
ann1 = neural_network.MLPRegressor(alpha=1e-5,
                                   hidden_layer_sizes=(20),
                                   random_state=1)
ann1.fit(X_train, y_train)
# Predict on test set
pred_y1 = ann1.predict(X_test)
pd.Series(pred_y1).describe()
```

```
count    142227.000000
mean          5.590623
std           0.287885
min           4.476281
25%           5.399191
50%           5.674505
75%           5.826425
max           6.076525
dtype: float64
```

```
# Calculate MAE
metrics.mean_absolute_error(y_test, pred_y1)
```

```
0.4141179571168786
```

```
# Caculate R squared
metrics.r2_score(y_test, pred_y1)
```

```
0.041545702220777914
```

```
# Calculate MSE
metrics.mean_squared_error(y_test, pred_y1)
```

```
0.26408552708278793
```

```
# Calculate RMSE
metrics.mean_squared_error(y_test, pred_y1,squared=False)
```

```
0.5138925248364564
```

After considering the above R2 score, we found that although AUC is scale independent, due to low R2 score the AUC score can be higher.

Results

One of our research questions which day in a week has the highest crime rate. Upon plotting we could observe that Sunday is the day which has the highest crime rates.

The next question was to find out highest crime rate recorded in between 2010 and 2020. We plotted a line plot for the variable YEAR. The year 2018 reported the highest for crime rate from 2010 to 2020.

The next research question addresses the analysis of crime as per area in Los Angeles. In all 21 areas in Los Angeles, we observed the crime Battery Assault was the highest reported.

In this project we employed a variety of data mining approaches, including those discussed in class as well as methods like Gaussian Naïve Bayes classifier and Random Forest which are not used in class.

We implemented various Data mining models by splitting the dataset as training and testing datasets. We split the data as 80-20% split. Among all the models that we used to predict the performance; Random Forest gave the highest AUC score of 79.6%.

References

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

https://towardsdatascience.com/understanding-random-forest-58381e0602d2

https://scikit-learn.org/stable/modules/naive_bayes.html

https://scikit-learn.org/stable/modules/tree.html

https://towardsdatascience.com/data-visualization-using-matplotlib-16f1aae5ce70

https://www.getsmarter.com/blog/career-advice/how-artificial-neural-networks-can-be-used-for-data-mining/