Atlanta Crime Report

Version 1

A Data Science and Machine Learning Project

Aleia Knight

Definitions

'Beat' - "The City of Atlanta is divided into six unique geographic areas – known as Zones – for the purposes of allocating APD resources. Each Zone is then divided into 13-14 “beats” assigned to a specific officer for patrol purposes.".

'UCR' - Uniform Crime Reporting Number. This number classifies a crime using a number system. Links to chart attached.

'IBR' - Allows for more specific crime types.

'NPU' - "The City of Atlanta is divided into twenty-five (25) Neighborhood Planning Units (NPUs), which are citizen advisory councils that make recommendations to the Mayor and City Council on zoning, land use, and other planning-related matters. ".

Research

Atlanta Police Beat and Zones

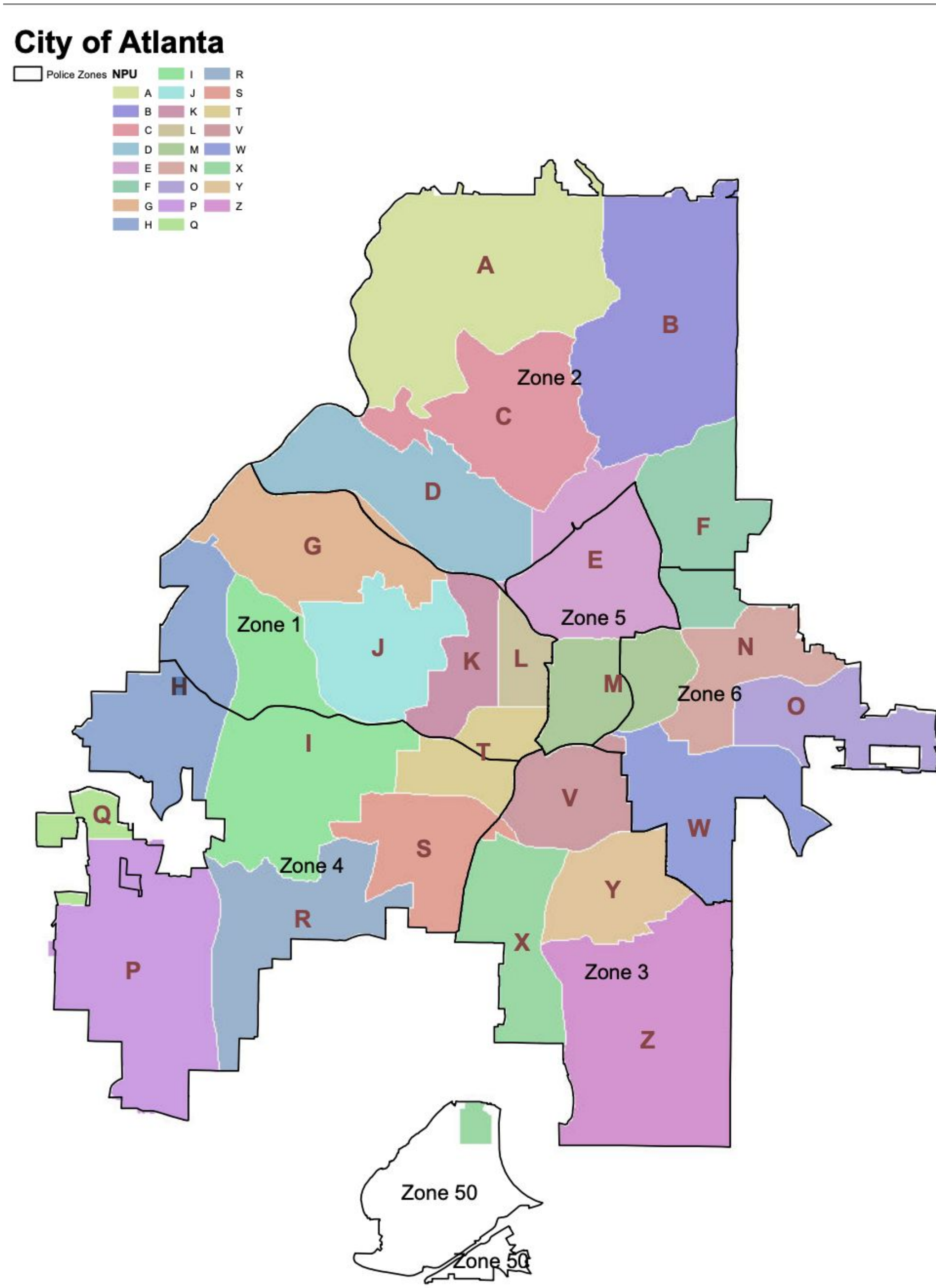
NIBRS

UCR CLASSIFICATION ABBREVIATIONS

Atlanta Police Department Crime Data Downloads

Uniform Crime Reporting Handbook

Zones / NPU


City of Atlanta
Police Zones NPU
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
V
W
X
Y
Z
Zone 1
Zone 2
Zone 3
Zone 4
Zone 5
Zone 6
Zone 50
Zone 50

Imports

I. **Numpy - np**
   - ➢ Python-based ecosystem of open-source software for mathematics, science, and engineering.

II. **Pandas - pd**
   - ➢ “A fast, powerful, flexible and easy to use open source data analysis and manipulation tool”

III. **Matplotlib - plt**
   - ➢ “A plotting library for the Python programming language and its numerical mathematics extension NumPy.”

IV. **Seaborn - sns**
   - ➢ “A data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.”

V. **SciPy - (specifically stats)**
   - ➢ “An ecosystem of open-source software for mathematics, science, and engineering.”

VI. **Plotly - px**
   - ➢ ‘A graphing library that makes interactive, publication-quality graphs.“

VII. **GeoPy - gp - (specifically Nominatim)**
   - ➢ “A client for several popular geocoding web services making it easy for developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.”

VIII. **SkLearn - (specifically accuracy_score, recision_score, recall_score, f1_score, and LogisticRegression)**
   - ➢ “A machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.”

Data

For this project, I use the “COBRA-2009-2019” dataset. It is loaded in as a pandas dataframe.

All data obtained from [Atlanta Police Department Crime Data]()

	Report Number	Report Date	Occur Date	Occur Time	Possible Date	Possible Time	Beat	Apartment Office Prefix	Apartment Number	Location
0	90010930	2009-01-01	2009-01-01	1145	2009-01-01	1148.0	411.0	NaN	NaN	2841 GREENBRIAR PKWY
1	90011083	2009-01-01	2009-01-01	1330	2009-01-01	1330.0	511.0	NaN	NaN	12 BROAD ST SW
2	90011208	2009-01-01	2009-01-01	1500	2009-01-01	1520.0	407.0	NaN	NaN	3500 MARTIN L KING JR DR SW
3	90011218	2009-01-01	2009-01-01	1450	2009-01-01	1510.0	210.0	NaN	NaN	3393 PEACHTREE RD NE
4	90011289	2009-01-01	2009-01-01	1600	2009-01-01	1700.0	411.0	NaN	NaN	2841 GREENBRIAR PKWY SW
5	90011327	2009-01-01	2009-01-01	1645	2009-01-01	1645.0	609.0	NaN	NaN	1217 CAROLINE ST NE
6	90011450	2009-01-01	2009-01-01	1740	2009-01-01	1815.0	408.0	NaN	NaN	2685 METROPOLITAN PARKWAY

Shift Occurence	Location Type	UCR Literal	UCR #	IBR Code	Neighborhood	NPU	Latitude	Longitude
Day Watch	8	LARCENY-NON VEHICLE	630	2303	Greenbriar	R	33.68845	-84.49328
Day Watch	9	LARCENY-NON VEHICLE	630	2303	Downtown	M	33.75320	-84.39201
Unknown	8	LARCENY-NON VEHICLE	630	2303	Adamsville	H	33.75735	-84.50282
Evening Watch	8	LARCENY-NON VEHICLE	630	2303	Lenox	B	33.84676	-84.36212
Unknown	8	LARCENY-NON VEHICLE	630	2303	Greenbriar	R	33.68677	-84.49773
Evening Watch	24	LARCENY-NON VEHICLE	630	2303	Edgewood	O	33.75786	-84.34875
Evening Watch	12	LARCENY-NON VEHICLE	630	2303	Venetian Hills	S	33.70827	-84.45385

Cleaning

- The first step is to clean the data. My steps involved:
    - Dropping unneeded columns

```
dropped_columns = ['Apartment Office Prefix',
                   'Apartment Number',
                   'Location',
                   'Location Type',
                   'Report Number']
atlanta = atlanta.drop(dropped_columns, axis=1)
```

    - Convert UCR #'s to the literal crime names

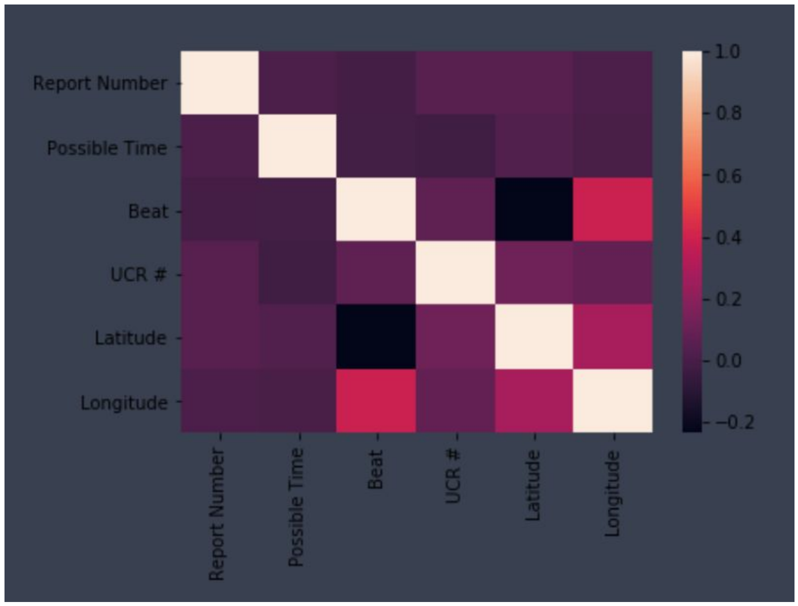
```
def codes_to_crimes(value):
    if value > 100 and value < 199:
        return 'Homicide'
    elif value > 200 and value < 299:
        return 'Rape'
    elif value > 300 and value < 399:
        return 'Robbery'
    elif value > 400 and value < 499:
        return 'Assault'
    elif value > 500 and value < 599:
        return 'Burglary'
    elif value > 600 and value < 699:
        return 'Larceny'
    elif value > 700 and value < 799:
        return 'Motor_theft'
    elif value > 800 and value < 899:
        return 'Arson'
atlanta['Crime'] = pd.Series(atlanta['UCR #']).apply(codes_to_crimes).astype('str')
```

    - Apply data sampling, 1%, while doing testing

```
atlanta = atlanta.sample(frac=0.01)  # 1% sample set
```

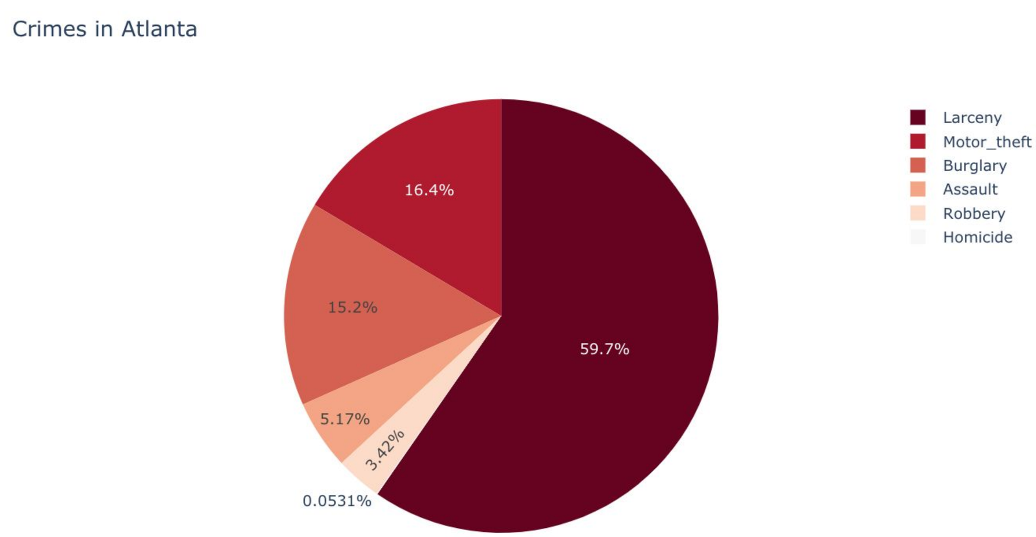
EDA (Exploratory Data Analysis)

- **Plot basic charts to get a general sense of the data**
    - **Heatmap**

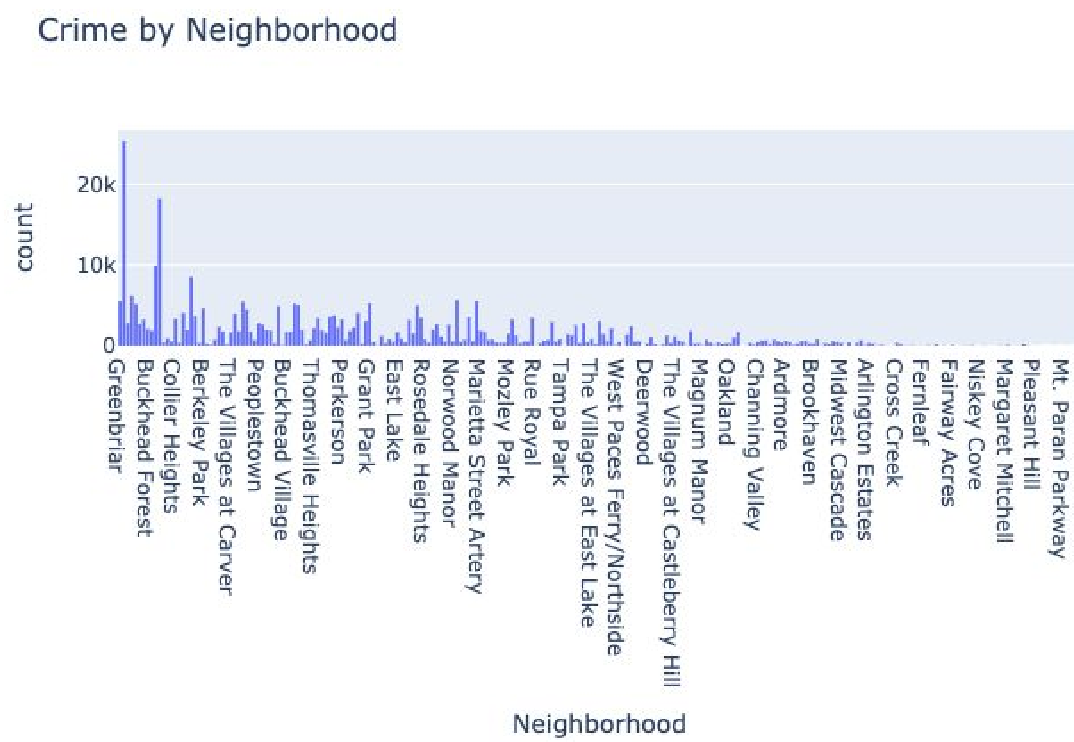


Most can be ignored, but the 'Beat' vs 'UCR #' is interesting. It is not a high correlation but one is there. It does show that there is at least some relation between the Zone of town lived in and the type of crime committed.

- **Pie Chart of Crimes**

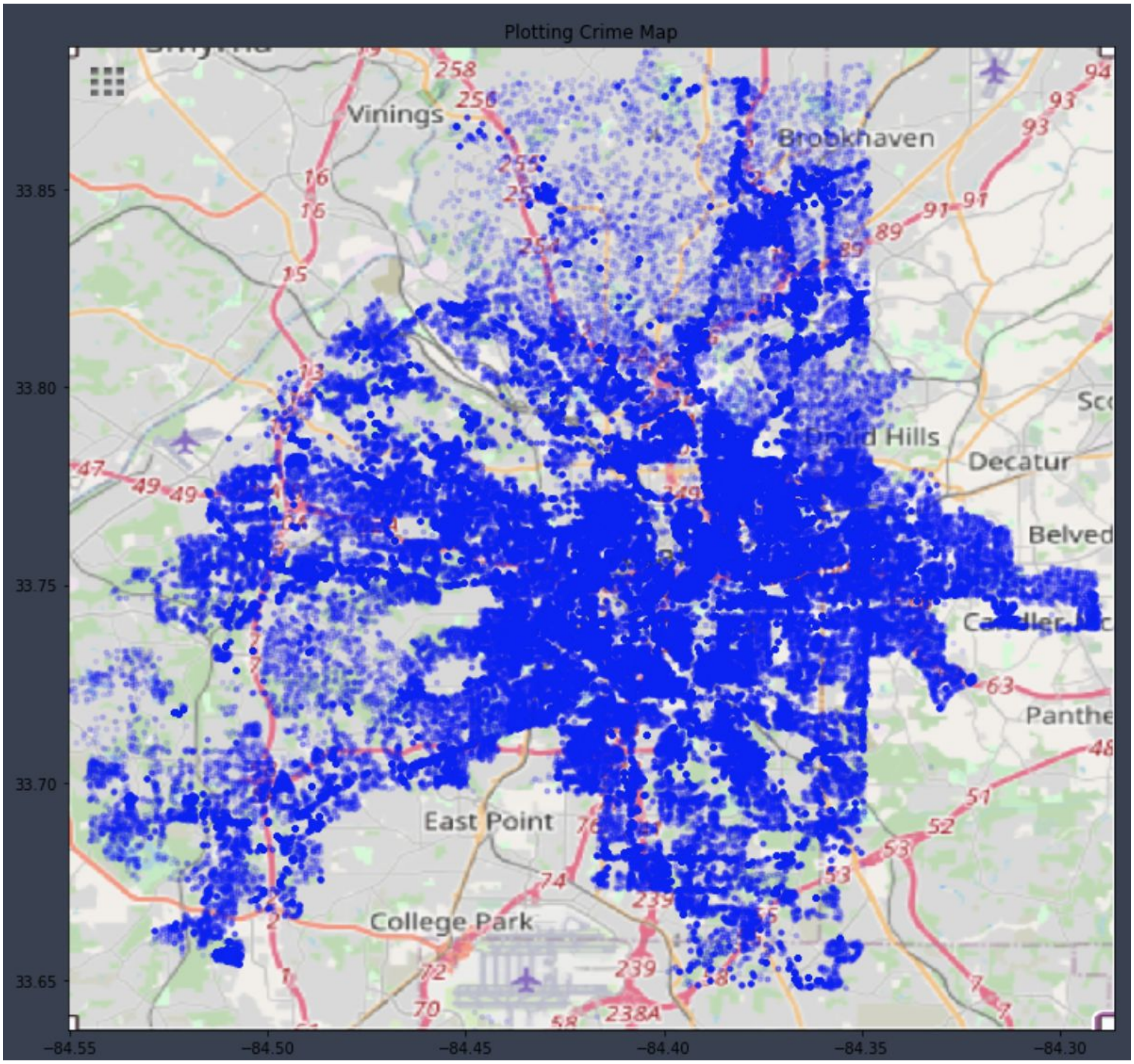


- **Crime Level by Neighborhood**



The neighborhoods with the largest crime rates are Midtown, Downtown, Old Fourth Ward, and West.

- **Geolocation on Map**



There is more crime the closer to the city you are. Also there seems to be less crime around the airport.

Machine Learning

- **Label Encoding**
    - **Convert the Crimes from literal names to numeric values so they can be processed in the model.**

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

atl = atlanta.apply(lambda col: le.fit_transform(col.astype(str)), axis=0, result_type='expand')
atl.head()
```

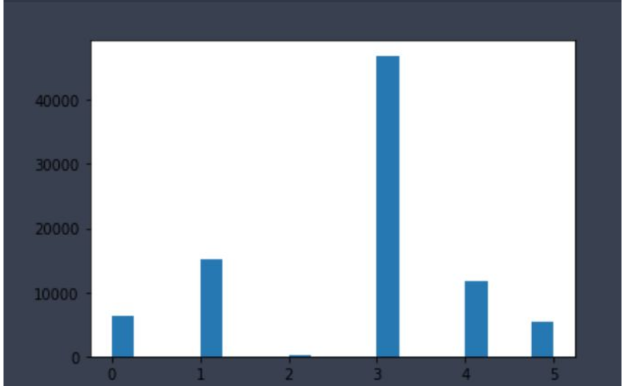
t e	Occur Date	Occur Time	Possible Date	Possible Time	Beat	Shift Occurence	UCR Literal	UCR #	IBR Code	Neighborhood	NPU	Latitude	Longitude	Crime
	169	680	76	127	51	0	6	42	58	107	17	2579	17171	3
	169	806	76	250	66	0	6	42	58	79	12	8905	7609	3
	169	915	76	381	47	3	6	42	58	2	7	9320	17959	3
	169	899	76	370	24	1	6	42	58	136	1	17189	4629	3
	169	985	76	492	51	3	6	42	58	107	17	2433	17543	3

- **Split data into test and train**
    - **Setup data to be split for testing and training. The test size was set to 25% of the data.**

```
from sklearn.model_selection import train_test_split

# Going to base the outcome based on these features
feature_cols = ['Occur Time', 'Neighborhood', 'Beat']

# X is a matrix, access the features we want in feature_cols
X = atl[feature_cols]

# y is a vector, hence we use dot to access 'label'
y = atl['Crime']

# split X and y into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

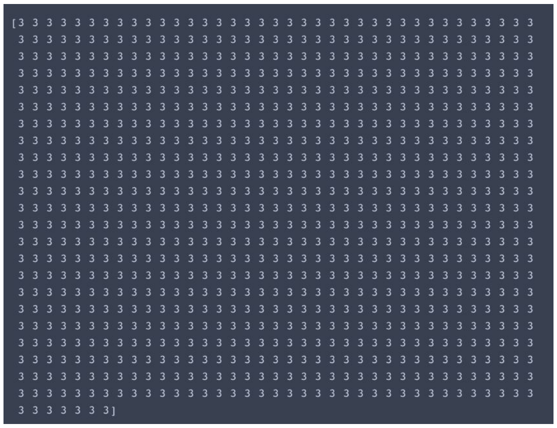
- **Look at distributions**
  - **Plot a histogram**

```
# create and plot a histogram based on y_test
plt.hist(y_test, bins=20)
plt.show()

# count how many of each crime exists
y_test_pd_series = pd.Series(y_test)
y_test_pd_series.value_counts()
```



- **Apply Logistic Regression**
  - **Using sklearn, apply Linear Regression to get y predictions. This resulted in a full array of predictions of ‘3’, which is the crime Larceny.**

```
# LOGISTIC REGRESSION
logreg = LogisticRegression()

# fit model
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)

print(y_pred)
```



- **View Metrics**
    - **Here we can see the Confusion Matrix, Accuracy, Precision, Recall, F1 score, and Classification Report.**

```
[3 3 3 ... 3 3 3]


[[    0     0     0  6288     0     0]
 [    0     0     0 15154     0     0]
 [    0     0     0   285     0     0]
 [    0     0     0 46929     0     0]
 [    0     0     0 11692     0     0]
 [    0     0     0  5381     0     0]]

Accuracy: 0.55

Precision: 0.30

Recall: 0.55

F1-score: 0.39



Classification Report

              precision    recall  f1-score   support

     Class 1       0.00      0.00      0.00      6288
     Class 2       0.00      0.00      0.00     15154
     Class 3       0.00      0.00      0.00       285
     Class 4       0.55      1.00      0.71     46929
     Class 5       0.00      0.00      0.00     11692
     Class 6       0.00      0.00      0.00      5381

    accuracy                           0.55     85729
   macro avg       0.09      0.17      0.12     85729
weighted avg       0.30      0.55      0.39     85729
```