Term Project

Final Project Submission

DSC 550

Prof. Brett Werner

Carlos Cano

Table of Contents

Abstract

This proposed research & data exploration paper culminating in various Milestones will focus on crime statistics within Los Angeles, California from 2020 to present. With respect with areas within this metropolitan a heighten focus will be that of microcosms within Los Angeles. Secondary focus will fall within victim sexes within the observed data, and their relationship to varying other reported variables as it relates to each variable. The hopes of this will be to gain further insight that will aid in continued extrapolation and possible refinement of models that are created.

Introduction

Issue At Hand

The issue at the heart of this research report is that of an understanding and translations of crime statistics within Los Angeles and how the information reported can be mined and explored in a fashion that can both make use of the data through to use of sourcing, preparing, building, and evaluating the models created.

Why It Matters?

The validity and efficacy of this data is of significant importance as the data not solely on the merit of metrics and variables that are quantifiable, but also because it also contains a human element. Each reported crimes contains a victim, lives are lost and sometimes forever changed there exists a desire to understand this data and to search for meaning that may not be apparent.

Stakeholder Interests

This type of data will continue to source possibly into perpetuity, as such there exist many stakeholders who can profit in one way another as a result, from students learning data mining to global conglomerates bent on profiting with this information t o those ambitious enough to hopefully try to predict incidents of crime.

Data Source

The Los Angeles Crime Data was sourced from Kaggle as they serve as the prime basis for initial research projects and data exploration. This information is originally sourced from data.gov which houses various Local, State and Federal reported information. The Los Angeles Police Department also known as LAPD has their written reports transcribed digitally. This dataset is updated weekly from data.lacity.org for continued reporting and transparency as required by law.

Detail Summary of Milestones

Milestone 1:

Milestone 1 proved to be the most challenging issue as continued scouring of various data sources which contained both a purposeful goal that aligned with learned skills within this course exhausted previously sourced data sources. It was ultimately decided upon for it's widely used research application that is able to be sourced from various government databases.

Once sourced this data was imported into a Jupyter Notebook that was processed with Python. Several common libraries were used, including but not limited to Numpy, Pandas, Matplotlib, and Seaborn. After which an initial data processing and exploration was conducted, and a further thesis was established based on area and victim sex was solidified.

Milestone 2:

This milestone's focused primarily on data preparation, as such further exploration was conducted with an initial focus on removal of data variables or columns. Of this data that was removed many were dropped as they were descriptor data for other columns within the dataset.

The next issue that was tackled was that of adjusting the variable of "Vict Sex", within the data frame. The information within this column was stored as an object and was converted into numeric values for data frame unification under a single type structure, in this case int64.

The removal of NaN data within this processed data frame was quantified and observed that over 100,000 were solely within the "Vict Sex" column. Fortunately there was a coded variable of 'X' which was mutated to 2 with accounts for unknown. As a result, NaN data was filled which resulted in no loss of the data in that aspect. The remaining 9 NaN variables fell within "Premis Cd" and were dropped as they didn't hold much if any weight on the remaining data frame.

Milestone 3:

The final milestone's focus was on that of training and evaluating a model with the now processed data frame. KNN & Linear Regression Model were utilized using a training and test set of data. Once applied this information was processed and garnered 85.77% & 99.9% accuracy respectfully. A quick look at the residual data was tested as the 99.9% seemed highly suspect in its nature. A third model, that of a Linear Regression was created with a focus on Time OCC. This model proved less accurate and returned a 73.3% accuracy, which intuitively would be harder to imagine as a time variable in crime would be harder to express within this function and model.

Milestone 3 Expanded:

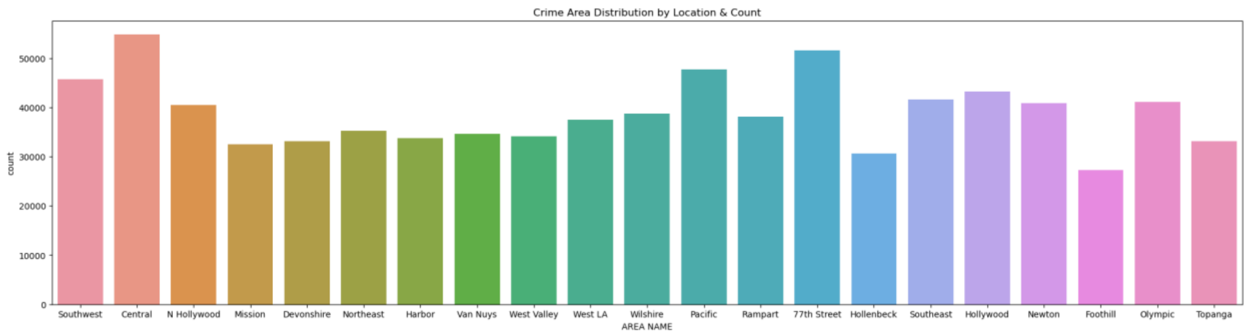
After further thought on Milestone 3, a secondary look was taken on the dataset and the initial focus of this report. It warranted a dive into the 'Vict Sex' variable of the dataset within this course as there was an emphasis on dummy variables and the merits of how they are used and incorporated within modeling train and test sets data. The eventual creation of new models facilitated the use of a modified data frame which used a similar structure within previous coding.

In revisiting this issue two new models were created to elaborate on the data revisited, both a logistic regression & a linear regression which produced an accuracy of 66.2% & 61.3% respectively was produced. It's telling as in the case of the 'Time OCC' variable this also produced a much lower than was initially reported with other models and variables produced previously within Milestone 3.

Further extrapolation of this data with other models proved insightful to say the least as it led to further questions that could be conceived from further data exploration.

Exploratory Data Analysis:

Within the initial milestone's that was conducted the thesis that emerged was that of an in depth look at crime statistics within Los Angeles and how the microcosm within this variable played an effect as a whole. It was initially observed that of the 21 Community Police Stations which represent the geographic areas within the data frames had a fairly level distribution with no great outliers. This is represented in the graph below with Central and 77th St being the two top places with reported incidents of crimes.



Conclusion

What does the model explain?

Through this data exploration and extrapolation there has been garnered several models, through this process it can be confidently stated that there exist a high degree of confidence that there is a high accuracy of the variable 'Area' contributing and to the efficacy of the model. In expanding on Milestone 3 different variables with the implementation of other models demonstrated less merit.

Can it be deployed?

The model as it stands currently can be deployed as is, although rudimentary in its function, there is much that has been derived from this exploratory analysis.

Recommendations

With a new found understanding within my expansion from Milestone 3 I believe a complete revisit to the original source data might be warranted. Also as the data is updated on a weekly business and is sourced throughout various website there could be an api coding that be created to continuously pull updated data.

Challenges

As stated initially the biggest hurdle was the concept creation and data sourcing for this EDA. That being said, understanding how to create test/train data splits and manipulate them accordingly while proving to be counter intuitive also posed a conceptual challenge within this project.

Opportunities

Expansion, working through this data there was an inspiration to revisit the original data which contained a time variable that could be further studied and investigated.

References

Setu, S. (2023, October 11). *Crime Data in Los Angeles (2020 to present)*. Kaggle. https://www.kaggle.com/datasets/sahityasetu/crime-data-in-los-angeles-2020-to-present/data

Term Project Final Submission
DSC 550

Carlos Cano

[1]:
```
## DSC 550
## Carlos Cano
## Activity 10.2
```

[2]:
```
## ** ---------- **
```

[3]:
```
print('** ----------- **')
print('** Milestone 1 **')
print('** ----------- **')
```

```
** ----------- **
** Milestone 1 **
** ----------- **
```

[4]:
```
## ** ---------- **
```

[5]:
```
## Begin Milestone 1 with a 250-500-word narrative describing your original idea␣
↪for the analysis/model building business problem.
## Clearly identify the problem you will address and the target for your model.
```

[6]:
```
## Then, do a graphical analysis creating a minimum of four graphs.
## Label your graphs appropriately and explain/analyze the information provided␣
↪by each graph.
## Your analysis should begin to answer the question(s) you are addressing.
```

Import Libraries

[7]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt

from scipy.stats import norm

import seaborn as sns
```

[8]:
```python
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
```

[9]:
```python
from sklearn import linear_model
import statsmodels.api as sm
```

[10]:
```python
df = pd.read_csv("Crime_Data_from_2020_to_Present.csv")
```

[11]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 815882 entries, 0 to 815881
Data columns (total 28 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DR_NO           815882 non-null  int64
 1   Date Rptd       815882 non-null  object
 2   DATE OCC        815882 non-null  object
 3   TIME OCC        815882 non-null  int64
 4   AREA            815882 non-null  int64
 5   AREA NAME       815882 non-null  object
 6   Rpt Dist No     815882 non-null  int64
 7   Part 1-2        815882 non-null  int64
 8   Crm Cd          815882 non-null  int64
 9   Crm Cd Desc     815882 non-null  object
 10  Mocodes         703120 non-null  object
 11  Vict Age        815882 non-null  int64
 12  Vict Sex        708690 non-null  object
 13  Vict Descent    708682 non-null  object
 14  Premis Cd       815873 non-null  float64
 15  Premis Desc     815402 non-null  object
 16  Weapon Used Cd  284434 non-null  float64
 17  Weapon Desc     284434 non-null  object
 18  Status          815882 non-null  object
 19  Status Desc     815882 non-null  object
 20  Crm Cd 1        815872 non-null  float64
 21  Crm Cd 2        60117 non-null   float64
 22  Crm Cd 3        2013 non-null    float64
 23  Crm Cd 4        59 non-null      float64
 24  LOCATION        815882 non-null  object
 25  Cross Street    130521 non-null  object
 26  LAT             815882 non-null  float64
 27  LON             815882 non-null  float64
```

```
dtypes: float64(8), int64(7), object(13)
memory usage: 174.3+ MB
```

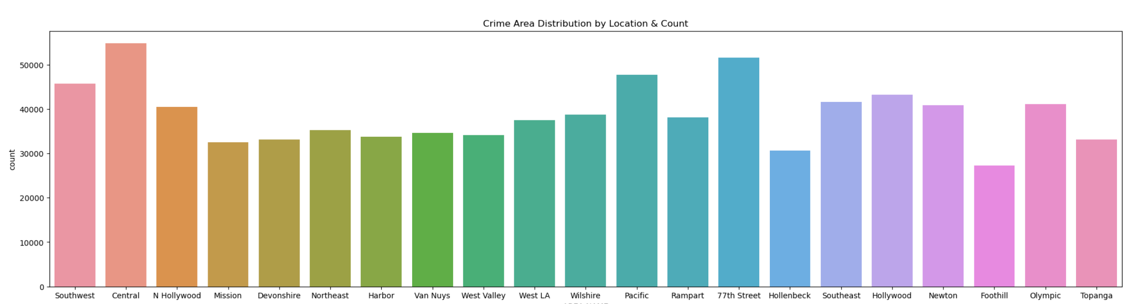
[12]:
```python
df.head(1)
```

[12]:
```
      DR_NO               Date Rptd               DATE OCC  TIME OCC  AREA  \
0  10304468  01/08/2020 12:00:00 AM  01/08/2020 12:00:00 AM      2230     3

   AREA NAME  Rpt Dist No  Part 1-2  Crm Cd              Crm Cd Desc  ...  \
0  Southwest          377         2     624  BATTERY - SIMPLE ASSAULT  ...

  Status   Status Desc Crm Cd 1 Crm Cd 2  Crm Cd 3 Crm Cd 4  \
0     AO  Adult Other    624.0      NaN       NaN      NaN

                                  LOCATION Cross Street      LAT       LON
0  1100 W  39TH                         PL          NaN  34.0141 -118.2978

[1 rows x 28 columns]
```

[13]:
```python
df.columns
```

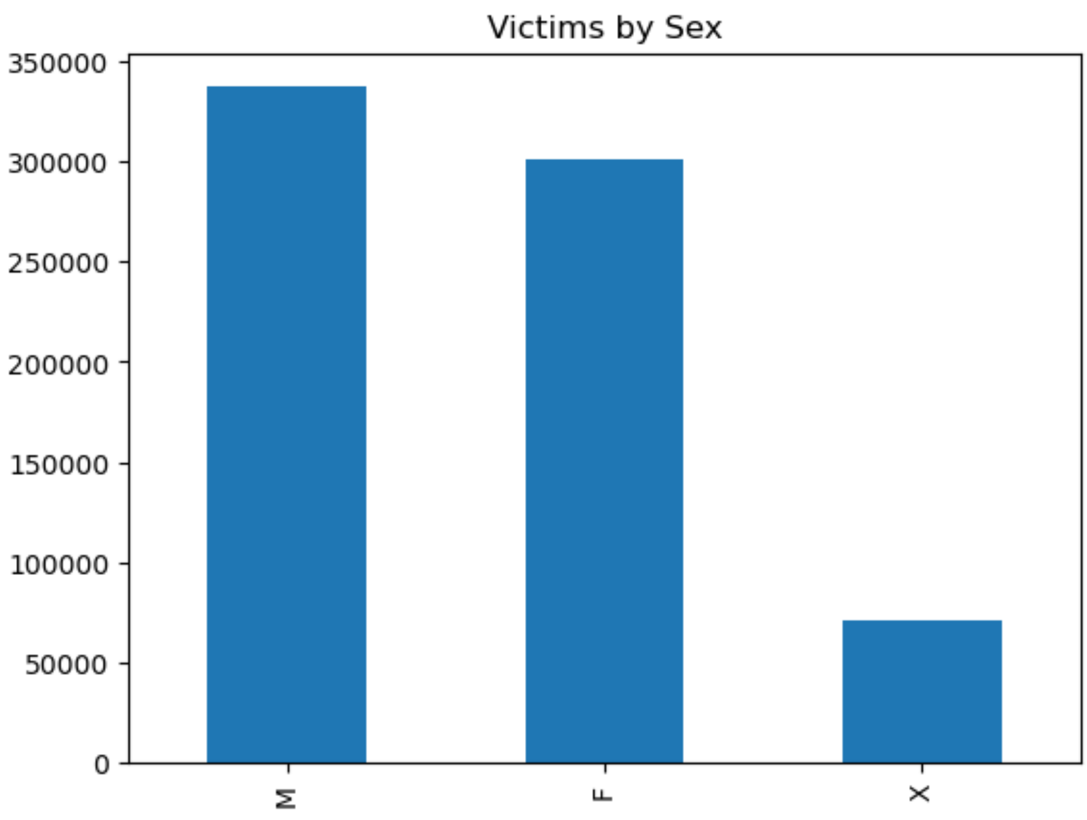
[13]:
```
Index(['DR_NO', 'Date Rptd', 'DATE OCC', 'TIME OCC', 'AREA', 'AREA NAME',
       'Rpt Dist No', 'Part 1-2', 'Crm Cd', 'Crm Cd Desc', 'Mocodes',
       'Vict Age', 'Vict Sex', 'Vict Descent', 'Premis Cd', 'Premis Desc',
       'Weapon Used Cd', 'Weapon Desc', 'Status', 'Status Desc', 'Crm Cd 1',
       'Crm Cd 2', 'Crm Cd 3', 'Crm Cd 4', 'LOCATION', 'Cross Street', 'LAT',
       'LON'],
      dtype='object')
```

[14]:
```python
plt.figure(figsize=(25,6))
sns.countplot(x='AREA NAME', data=df)
plt.title('Crime Area Distribution by Location & Count')
plt.show()


Size = df[['Crm Cd Desc','AREA NAME']].groupby(['Crm Cd Desc'], as_index=False).
 ↪sum()
#Size.sort_values(by=['Crm Cd Desc'],ascending=False).head(5)
```



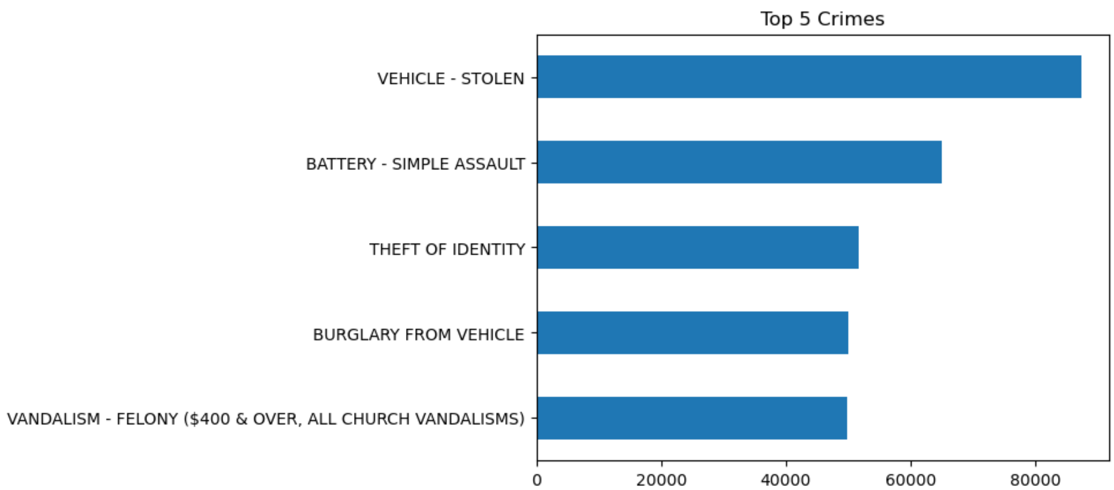
[15]:
```
## This graph showcases the various areas within the reported data from Los␣
↪Angeles Crime Dataset
```

[16]:
```
df['Vict Sex'].value_counts().iloc[:3].plot(kind='bar', title='Victims by Sex');
```



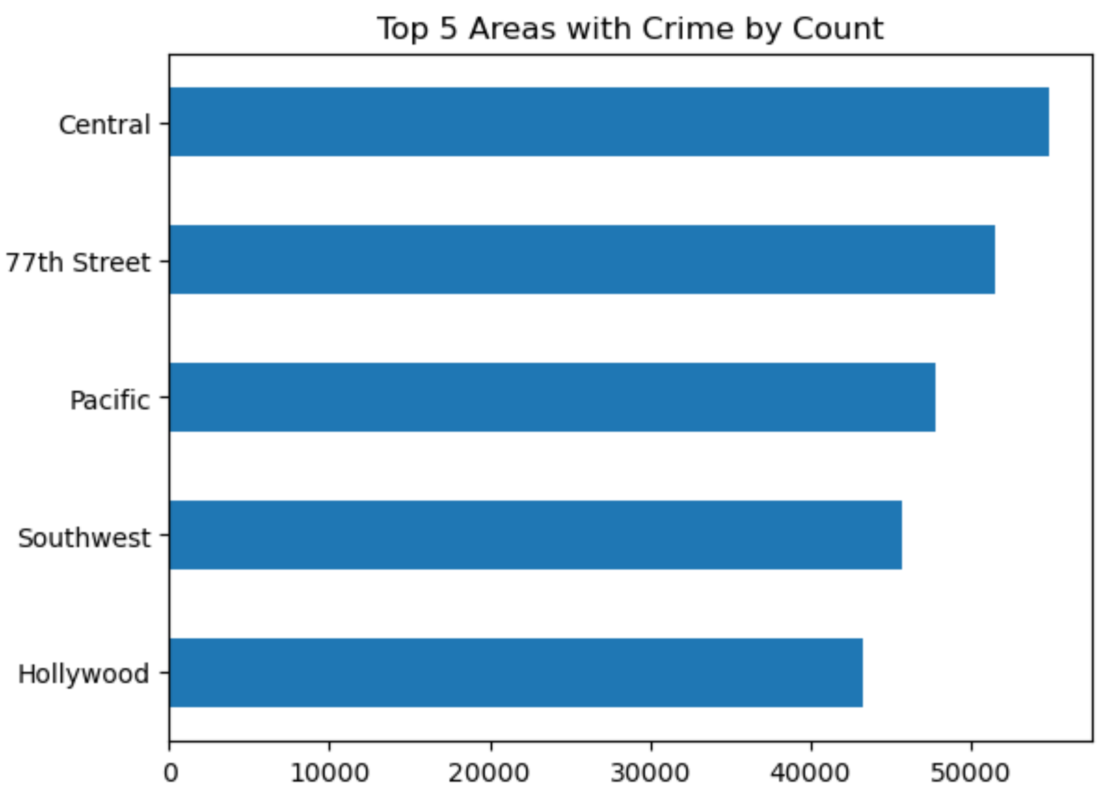
[17]:
```
## This graph explains the distribution of Victims by Sex within the Dataset,␣
↪there seems to be a fairly level distribution from the reported cases with the␣
↪exception of non M / F.
```

[18]:
```
df['Crm Cd Desc'].value_counts().iloc[:5].sort_values().
↪plot(kind='barh',title="Top 5 Crimes");
```



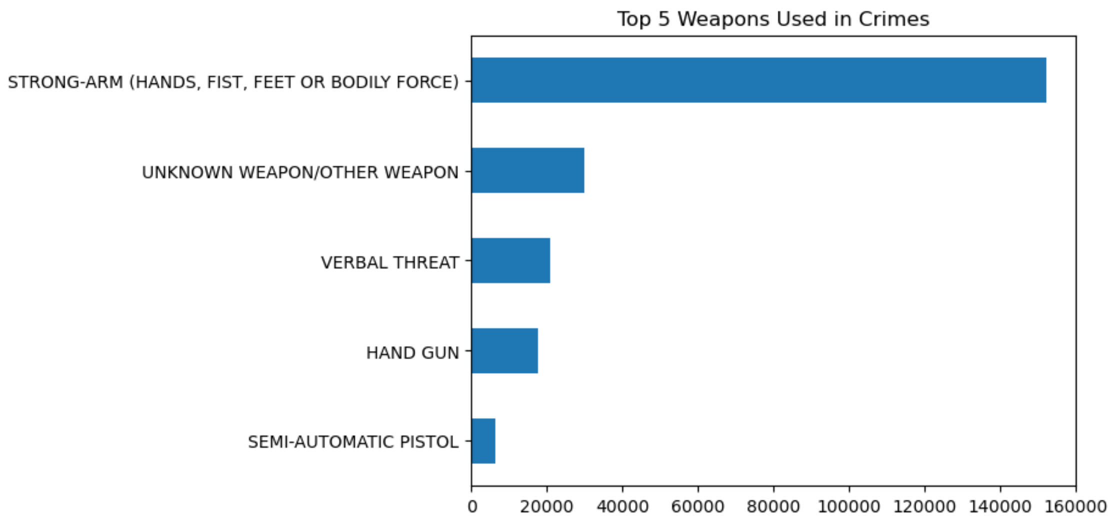
[19]:
```
## This graph highlights the top 5 crimes within Los Angeles from 2020-Current␣
↪with a dataset of about 900,000 stolen vehicle appears to be about 10% of␣
↪crimes.
```

[20]:
```
df['AREA NAME'].value_counts().iloc[:5].sort_values().
↪plot(kind='barh',title="Top 5 Areas with Crime by Count");
```



[21]:
```
## This further elaborates on area, with a deeper look at the top 5 areas of
↪reported crime. As noted in the global graph there was a fairly even
↪distribution. Central Los Angeles, formally known as South Central alongside
↪77th Street make up over 10% of the reported crime for this dataset.
```

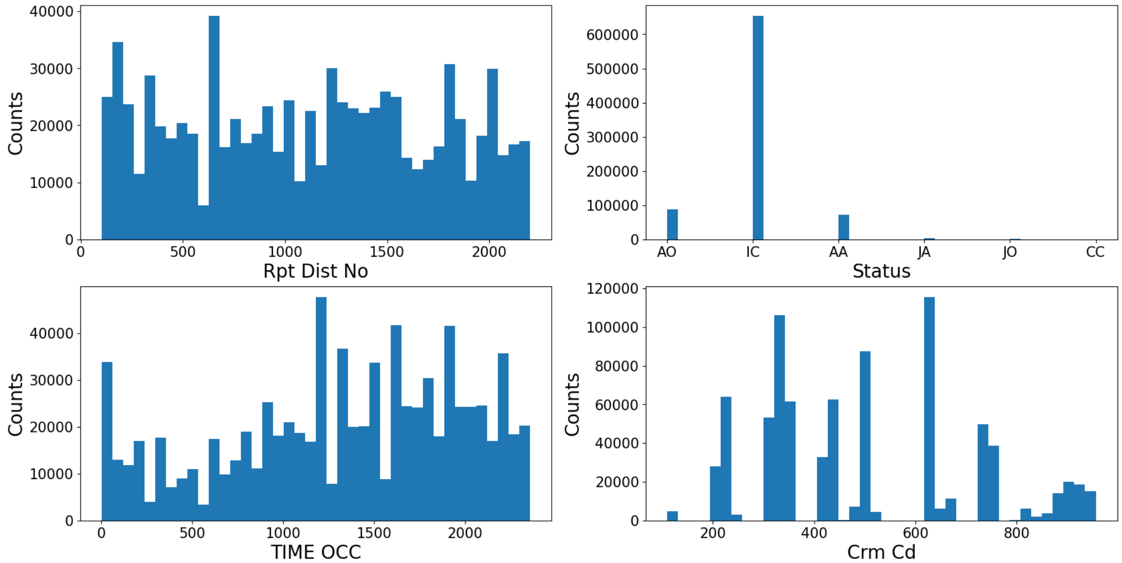
[22]:
```
df['Weapon Desc'].value_counts().iloc[:5].sort_values().
↪plot(kind='barh',title="Top 5 Weapons Used in Crimes");
```



[23]:
```
## This graph highlights the Top 5 Weapons reportedly used in crimes, with an
↪outlier of Strong-Arm accounting for nearly 20% of reported crime. This
↪quickly falls off for the other weapons.
```

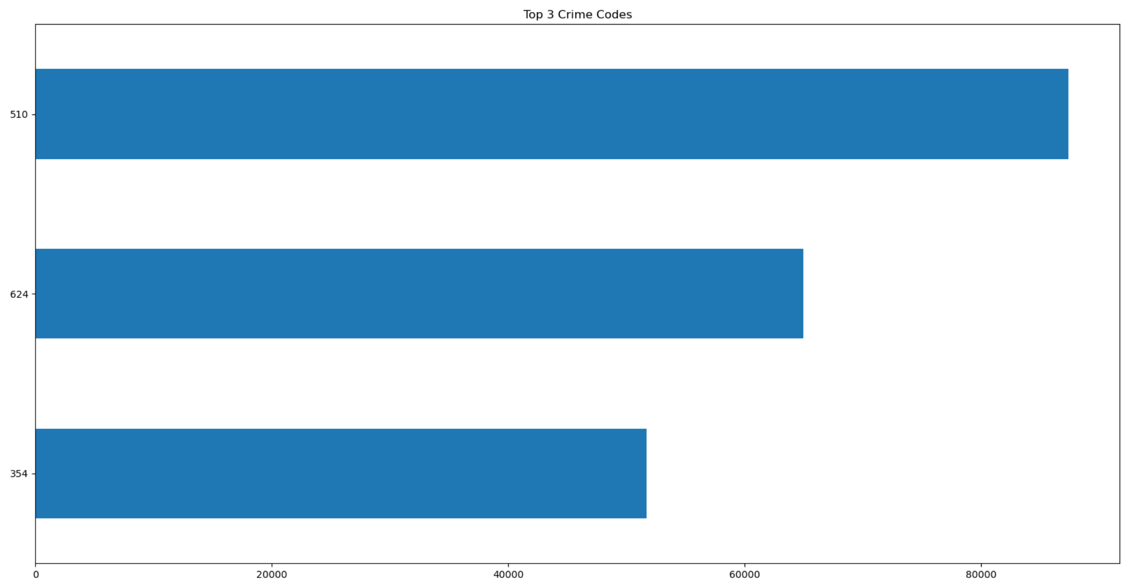
[24]:
```
plt.rcParams['figure.figsize'] = (20, 10)

fig, axes = plt.subplots(nrows = 2, ncols = 2)


num_features = ['Rpt Dist No', 'Status', 'TIME OCC', 'Crm Cd']

xaxes = num_features
yaxes = ['Counts', 'Counts', 'Counts', 'Counts']

axes = axes.ravel()
for idx, ax in enumerate(axes):
    ax.hist(df[num_features[idx]].dropna(), bins=40)
    ax.set_xlabel(xaxes[idx], fontsize=20)
    ax.set_ylabel(yaxes[idx], fontsize=20)
```

```
    ax.tick_params(axis='both', labelsize=15)
plt.show()
```



[25]:
```
## These graphs were used to gain insight to other integer type variables and
their distributions, of particular interest is increases in reported crime at
midnight and 12 o clock pm.
```

[26]:
```
df['Crm Cd'].value_counts().iloc[:3].sort_values().plot(kind='barh',title="Top 3
Crime Codes");
```



[27]:
```
# This graph highlights the Top 3 reported crimes by their code. 510: Speeding␣
↪or Racing | 624: Battery Simmple Assault | 354: Theft of ID, this was a␣
↪offshoot from the multi-variable graphset previous to this.
```

[ ]:
```

```

[28]:
```
## Write a short overview/conclusion of the insights gained from your graphical␣
↪analysis.
```

[29]:
```
# There was some definite insight garnered through this graphical analysis, as␣
↪noted in each section. Area, Time OCC, and Crm Cd stood out as data of␣
↪particular note from this graphical analysis.
```

[30]:
```
## ** ---------- **
```

[31]:
```
print('** ----------- **')
print('** Milestone 2 **')
print('** ----------- **')
```

```
** ----------- **
** Milestone 2 **
** ----------- **
```

[32]:
```
## ** ---------- **
```

[33]:
```
## Drop any features that are not useful for your model building and explain why␣
↪they are not useful.
```

[34]:
```
df.columns
```

[34]:
```
Index(['DR_NO', 'Date Rptd', 'DATE OCC', 'TIME OCC', 'AREA', 'AREA NAME',
       'Rpt Dist No', 'Part 1-2', 'Crm Cd', 'Crm Cd Desc', 'Mocodes',
       'Vict Age', 'Vict Sex', 'Vict Descent', 'Premis Cd', 'Premis Desc',
       'Weapon Used Cd', 'Weapon Desc', 'Status', 'Status Desc', 'Crm Cd 1',
       'Crm Cd 2', 'Crm Cd 3', 'Crm Cd 4', 'LOCATION', 'Cross Street', 'LAT',
       'LON'],
      dtype='object')
```

[35]:
```
df2=df.drop(columns=['Crm Cd 1', 'Crm Cd 2', 'Crm Cd 3', 'Crm Cd 4'], axis=1)
```

[36]:
```
# Crm Cd 1, Crm Cd 2, Crm Cd 3 and Crm Cd 4 are redundant as specific Crm Cd␣
↪variable is preserved.
```

[37]:
```
df2=df2.drop(columns=['LAT', 'LON' , 'Cross Street', 'DR_NO', 'Date Rptd',␣
↪'LOCATION', 'DATE OCC'], axis=1)
```

[38]:
```
# LOCATION, LAT, LON, and Cross Street are all very specific geographic␣
↪locations that encompassed in AREA variable that is preserved.
```

[39]:
```
# Date Rptd and DATE OCC are similar in nature but unnecessary in relation to␣
↪the focus of this research.
```

[40]:
```
# DR_NO is just the specific incident number, which can also be represented as a␣
↪row for the purposes of this EDA.
```

[41]:
```
df2=df2.drop(columns=['Status', 'Status Desc'], axis=1)
```

[42]:
```
# Status and Status Desc explain whether the case is ongoing or closed, again␣
↪for this research is also irrelavent.
```

[43]:
```
df2=df2.drop(columns=['AREA NAME', 'Crm Cd Desc', 'Premis Desc'], axis=1)
```

[44]:
```
# AREA NAME, Crm Cd Desc and Premis Desc are redudant as serve only as␣
↪references to their numeric value which are preserved in the modified dataset.
```

[45]:
```
df2=df2.drop(columns=['Weapon Desc', 'Vict Descent', 'Mocodes', 'Part 1-2'],␣
↪axis=1)
```

[46]:
```
# Mocodes has to do with Modus Operandi and describes the activity the suspect␣
↪was engaging in. Part 1-2 has no description or associated identifying labels/
↪descriptors.
```

[47]:
```
# Weapon Desc and Vict Descent are descriptive data that are redundant on␣
↪superfluous in nature.
```

[48]:
```
df2=df2.drop(columns=['Weapon Used Cd'], axis=1)
```

[49]:
```
# This data was to be used, but with over 400,000 Nans in this column, the␣
↪decision to remove it was made to preserve more overall data. It's telling as␣
↪a lot of crimes are committed without a weapon.
```

[50]:
```
df2.head()
```

[50]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd	Vict Age	Vict Sex	Premis Cd
0	2230	3	377	624	36	F	501.0
1	330	1	163	624	25	M	102.0
2	1200	1	155	845	0	X	726.0
3	1730	15	1543	745	76	F	502.0
4	415	19	1998	740	31	X	409.0

[51]:
```
df2.describe()
```

[51]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd \
count	815882.000000	815882.000000	815882.000000	815882.000000
mean	1335.614658	10.711521	1117.576886	500.777800
std	654.102822	6.092813	609.276287	207.816937
min	1.000000	1.000000	101.000000	110.000000

```
25%       900.000000       6.000000     621.000000     331.000000
50%      1415.000000      11.000000    1142.000000     442.000000
75%      1900.000000      16.000000    1617.000000     626.000000
max      2359.000000      21.000000    2199.000000     956.000000

            Vict Age      Premis Cd
count  815882.000000  815873.000000
mean       29.818963     305.776683
std        21.772828     216.646998
min        -3.000000     101.000000
25%         8.000000     101.000000
50%        31.000000     203.000000
75%        45.000000     501.000000
max       120.000000     976.000000
```

[52]:
```
## Perform any data extraction/selection steps.
```

[53]:
```
## Transform features if necessary.
```

[54]:
```
df2["Vict Sex"]
```

[54]:
```
0         F
1         M
2         X
3         F
4         X
         ..
815877    M
815878    F
815879    M
815880    F
815881    F
Name: Vict Sex, Length: 815882, dtype: object
```

[55]:
```
df2['Vict Sex'].nunique()
```

[55]:
```
5
```

[56]:
```
df2.replace({'F':0,'M':1, 'X':2, 'H':3, "-":4}, inplace=True)
```

[57]:
```
print(df2['Vict Sex'].value_counts())
```

```
1.0    337050
0.0    300602
2.0     70947
3.0        90
4.0         1
Name: Vict Sex, dtype: int64
```

[58]:
```
## Remove Nans
```

[59]:
```
df2['Vict Sex'] = df2['Vict Sex'].replace(np.nan, 2)
```

[60]:
```
# After processing, data in df2 Vict Sex was adjust for unknown values. NaNs␣
↪converted to unknown labeled as value 2 as missing values are unknown.
```

[61]:
```
df2 = df2.drop(df2[df2['Vict Sex'] == 3].index)
```

[62]:
```
df2 = df2.drop(df2[df2['Vict Sex'] == 4].index)
```

[63]:
```
# Variables 3 & 4 previous labeled H & - were dropped as erroneously reported␣
↪data.
```

[64]:
```
df2['Vict Sex'] = pd.to_numeric(df2['Vict Sex'])
```

[65]:
```
df2['Vict Sex'] = df2['Vict Sex'].astype('int64')
```

[66]:
```
print(df2['Vict Sex'].value_counts())
```

```
1    337050
0    300602
2    178139
Name: Vict Sex, dtype: int64
```

[67]:
```
## Engineer new useful features.
```

[68]:
```
df2['Premis Cd'] = df2['Premis Cd'].replace(np.nan, 710)
```

[69]:
```
## NaNs replaced with code 710 which is labeled as other.
```

[70]:
```
df2['Premis Cd'] = df2['Premis Cd'].astype('int64')
```

[71]:
```
## Deal with missing data (do not just drop rows or columns without justifying␣
↪this).
```

[72]:
```
NaN_Missing_Count = df2.isnull().sum(axis = 0)

NaN_Missing_Count
```

[72]:
```
TIME OCC       0
AREA           0
Rpt Dist No    0
Crm Cd         0
Vict Age       0
Vict Sex       0
Premis Cd      0
dtype: int64
```

[73]:
```
df2.head()
```

[73]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd	Vict Age	Vict Sex	Premis Cd
0	2230	3	377	624	36	0	501
1	330	1	163	624	25	1	102
2	1200	1	155	845	0	2	726
3	1730	15	1543	745	76	0	502
4	415	19	1998	740	31	2	409

[74]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 815791 entries, 0 to 815881
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   TIME OCC     815791 non-null  int64
 1   AREA         815791 non-null  int64
 2   Rpt Dist No  815791 non-null  int64
 3   Crm Cd       815791 non-null  int64
 4   Vict Age     815791 non-null  int64
 5   Vict Sex     815791 non-null  int64
 6   Premis Cd    815791 non-null  int64
dtypes: int64(7)
memory usage: 49.8 MB
```

[75]:
```
df4 = df2
```

[ ]:
```
```

[76]:
```
## Create dummy variables if necessary.
```

[77]:
```
# df3 = pd.get_dummies(df2)
```

[78]:
```
# As data was filtered and descriptions removed the need for dummy variables was␣
↪unwarranted. Coding was placed to showcase how to do so in the event it was␣
↪needed.
```

[ ]:
```
```

[79]:
```
## ** ---------- **
```

[80]:
```
print('** ----------- **')
print('** Milestone 3 **')
print('** ----------- **')
```

```
** ----------- **
** Milestone 3 **
** ----------- **
```

[81]:
```
## ** ---------- **
```

[ ]:
```
```

[82]:
```
## You are required to train and evaluate at least one model in this milestone.
```

[83]:
```
print('** ------------------------------ **')
print('** AREA K Nearest Neighbors Model **')
print('** ------------------------------ **')
```

```
** ------------------------------ **
** AREA K Nearest Neighbors Model **
** ------------------------------ **
```

[84]:
```
## As the data has been cleaned, and there are a finite amount of variables,␣
↪clustering of the data for AREA will be the focus.
```

[85]:
```
X = df2.drop('AREA', axis=1)
y = df2['AREA']
```

[86]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[87]:
```
pipeline = Pipeline([('scaler', MinMaxScaler()),('classifier',␣
↪KNeighborsClassifier())])
```

[88]:
```
pipeline.fit(X_train, y_train)
```

[88]:
```
Pipeline(steps=[('scaler', MinMaxScaler()),
                ('classifier', KNeighborsClassifier())])
```

[89]:
```
accuracy = pipeline.score(X_test, y_test)

print("The accuracy of the KNN classifier from the test set is: {:.2f}".
↪format(accuracy*100),"%")
```

```
The accuracy of the KNN classifier from the test set is: 85.77 %
```

[ ]:
```
```

[90]:
```
print('** ---------------------------- **')
print('** AREA Linear Regression Model **')
print('** ---------------------------- **')
```

```
** ---------------------------- **
** AREA Linear Regression Model **
** ---------------------------- **
```

[91]:
```
## Taking a practical approach for dependent variable we will look at the
↪relationship of this model other variables in relation to AREA.
```

[92]:
```
LinearModel = linear_model.LinearRegression()
```

[93]:
```
LinearModel.fit(X,y)
```

[93]:
```
LinearRegression()
```

[94]:
```
model = sm.OLS(y, X).fit()
```

[95]:
```
prediction = model.predict(X)
```

[96]:
```
print(model.summary())
```

```
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:                   AREA   R-squared (uncentered):
0.999
Model:                            OLS   Adj. R-squared (uncentered):
0.999
Method:                 Least Squares   F-statistic:
2.614e+08
Date:                Sat, 18 Nov 2023   Prob (F-statistic):
0.00
Time:                        16:49:42   Log-Likelihood:
-1.2185e+05
No. Observations:              815791   AIC:
2.437e+05
Df Residuals:                  815785   BIC:
2.438e+05
Df Model:                           6
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
TIME OCC    -5.821e-05   4.34e-07   -134.118      0.000   -5.91e-05   -5.74e-05
Rpt Dist No     0.0099   4.74e-07     2.1e+04      0.000       0.010       0.010
Crm Cd         -0.0002   1.32e-06   -159.733      0.000      -0.000      -0.000
Vict Age       -0.0026   1.51e-05   -173.442      0.000      -0.003      -0.003
Vict Sex       -0.0851      0.000   -197.399      0.000      -0.086      -0.084
Premis Cd      -0.0001   1.43e-06    -92.581      0.000      -0.000      -0.000
==============================================================================
Omnibus:                    66639.184   Durbin-Watson:                   1.988
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            22358.649
Skew:                          -0.097   Prob(JB):                         0.00
Kurtosis:                       2.213   Cond. No.                     2.68e+03
```

```
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not
contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[3] The condition number is large, 2.68e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[97]:
```
print("The accuracy of the Linear Regression Model of AREA from the test set is:␣
↪99.9%")
```

```
The accuracy of the Linear Regression Model of AREA from the test set is: 99.9%
```

[98]:
```
prediction.head(3)
```

[98]:
```
0    3.323915
1    1.304809
2    1.026085
dtype: float64
```

[99]:
```
y.head(3)
```

[99]:
```
0    3
1    1
2    1
Name: AREA, dtype: int64
```

[100]:
```
xyz = pd.concat([y, prediction], axis = 1, join = "inner")
xyz = xyz.rename(columns={'AREA':'y', 0:'prediction'})
xyz['residual'] = xyz['y'] - xyz['prediction']
```

[101]:
```
# xyz[['residual']].head(7).style.hide(axis='index')
```

[ ]:
```

```

[102]:
```
print('** -------------------------------- **')
print('** TIME OCC Linear Regression Model **')
print('** -------------------------------- **')
```

```
** -------------------------------- **
** TIME OCC Linear Regression Model **
** -------------------------------- **
```

[ ]:
```

```

[103]:
```
X2 = df2.drop('TIME OCC', axis=1)
y2 = df2['TIME OCC']
```

[104]:
```
X2.head()
```

[104]:

	AREA	Rpt Dist No	Crm Cd	Vict Age	Vict Sex	Premis Cd
0	3	377	624	36	0	501
1	1	163	624	25	1	102
2	1	155	845	0	2	726
3	15	1543	745	76	0	502
4	19	1998	740	31	2	409

[105]:
```
LinearModel2 = linear_model.LinearRegression()
```

[106]:
```
X_train, X_test, y_train, y_test = train_test_split(X2, y2, test_size=0.2,␣
↪random_state=42)
```

[107]:
```
LinearModel2.fit(X2,y2)
```

[107]: LinearRegression()

[108]:
```
model2 = sm.OLS(y2, X2).fit()
```

[109]:
```
print(model2.summary())
```

```
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:               TIME OCC   R-squared (uncentered):
0.773
Model:                            OLS   Adj. R-squared (uncentered):
0.773
Method:                 Least Squares   F-statistic:
4.624e+05
Date:                Sat, 18 Nov 2023   Prob (F-statistic):
0.00
Time:                        16:49:42   Log-Likelihood:
-6.5122e+06
No. Observations:              815791   AIC:
1.302e+07
Df Residuals:                  815785   BIC:
1.302e+07
Df Model:                           6
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
AREA         -370.6093      2.763   -134.118      0.000    -376.025    -365.193
Rpt Dist No     3.8784      0.027    141.302      0.000       3.825       3.932
Crm Cd          0.7640      0.003    233.572      0.000       0.758       0.770
Vict Age        7.7272      0.038    204.138      0.000       7.653       7.801
```

```
Vict Sex      280.2741      1.069     262.218      0.000      278.179      282.369
Premis Cd       0.2042      0.004      56.508      0.000        0.197        0.211
==============================================================================
Omnibus:                    39994.000   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            24707.348
Skew:                          -0.295   Prob(JB):                         0.00
Kurtosis:                       2.384   Cond. No.                     4.89e+03
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not
contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[3] The condition number is large, 4.89e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[110]:
```
print("The accuracy of the Linear Regression Model of TIME OCC from the test set␣
↪is: 73.3%")
```

```
The accuracy of the Linear Regression Model of TIME OCC from the test set is:
73.3%
```

[111]:
```
##  Write step-by-step for performing each of these steps.
## You can use any methods/tools you think are most appropriate, but you should␣
↪explain/justify why you are selecting the model(s) and evaluation metric(s)␣
↪you choose.
```

[ ]:
```

```

[112]:
```
## Write a short overview/conclusion of the insights gained from your model␣
↪building/evaluation.
```

[113]:
```
# Looking at the various models, it's clear that when accounting for the test␣
↪model of AREA there is a high factor of 85.77% using the KNN model, while␣
↪satisfactory a separate look was taken using a linear regression model with␣
↪garnered a near perfect model. As such, when accounting for TIME OCC there was␣
↪a significant drop in a linear regression model's explanatory data of a␣
↪suggested 77.3%. This would suggest given the data imported and cleaned as␣
↪well as tested shows a strong correlation as well as explained tested data.
```

[114]:
```
print('** ------------- **')
print('** Data Revision **')
print('** ------------- **')
```

```
** ------------- **
** Data Revision **
** ------------- **
```

[115]:
```
## Separation of data from Vict Sex was revisited as originally data was␣
↪converted into int64 from object. It occurred to me that the data could have␣
↪been preserved and factored into the model within its original format.
```

[116]:
```
## This would allow for the creation of dummie variables and explore the data␣
↪similar to that of previous assignments.
```

[117]:
```
df3 = df2
```

[118]:
```
df3['Vict Sex'].replace({0:'F', 1:'M', 2: 'X'}, inplace=True)
```

[119]:
```
df3.head(2)
```

[119]:
```
   TIME OCC  AREA  Rpt Dist No  Crm Cd  Vict Age Vict Sex  Premis Cd
0      2230     3          377     624        36        F        501
1       330     1          163     624        25        M        102
```

[120]:
```
df3['Vict Sex']
```

[120]:
```
0         F
1         M
2         X
3         F
4         X
         ..
815877    M
815878    F
815879    M
815880    F
815881    F
Name: Vict Sex, Length: 815791, dtype: object
```

[121]:
```
df_cat = df3['Vict Sex']
df_int = df3.drop('Vict Sex', axis=1)
```

[122]:
```
cat_enc = pd.get_dummies(df_cat, drop_first=True)
hot_var_list = cat_enc.columns.tolist()
```

[123]:
```
df_enc = cat_enc.merge(df_int, left_index=True, right_index=True)
enc_var_list = df_enc.columns.tolist()
```

[124]:
```
x_columns = df_enc
x_columns = x_columns.drop('M', axis=1)
```

[125]:
```
y_column = df_enc[('M')]
```

[126]:
```
X_train, X_test, y_train, y_test = train_test_split( x_columns, y_column,␣
↪test_size=0.2, random_state=12345, stratify= y_column)
```

[127]:
```
y_t = pd.DataFrame(y_train)
y_t.head(3)
```

[127]:

	M
627454	1
496051	0
455606	0

[128]:
```
import sys, traceback
class Suppressor(object):
    def __enter__(self):
        self.stdout = sys.stdout
        sys.stdout = self
    def __exit__(self, type, value, traceback):
        sys.stdout = self.stdout
        if type is not None:
            def write(self, x): pass
```

[129]:
```
cat_var_list = ['Vict Sex']
cat_var_list
```

[129]: ['Vict Sex']

[130]:
```
int_var_list = df_int.columns.tolist()
```

[131]:
```
int_var_list
```

[131]: ['TIME OCC', 'AREA', 'Rpt Dist No', 'Crm Cd', 'Vict Age', 'Premis Cd']

[ ]:
```

```

[132]:
```
print('** ---------------------------------- **')
print('** Vict Sex Logistic Regression Model **')
print('** ---------------------------------- **')
```

```
** ---------------------------------- **
** Vict Sex Logistic Regression Model **
** ---------------------------------- **
```

[ ]:
```

```

[133]:
```
from sklearn.compose import ColumnTransformer
```

[134]:
```
from sklearn.feature_selection import SelectKBest, f_classif
```

[135]:
```
from sklearn.preprocessing import StandardScaler
```

[136]:
```
column_transformer = ColumnTransformer([("scaler", StandardScaler(),␣
↪int_var_list)], remainder="passthrough")
```

[137]:
```
logistic_pipeline = Pipeline([('datafeed', column_transformer),('selector',␣
↪SelectKBest(f_classif, k='all')),('classifier', LogisticRegression())])
```

[138]:
```
logistic_pipeline.fit(X_train, y_train)
```

[138]:
```
Pipeline(steps=[('datafeed',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('scaler', StandardScaler(),
                                                  ['TIME OCC', 'AREA',
                                                   'Rpt Dist No', 'Crm Cd',
                                                   'Vict Age',
                                                   'Premis Cd'])])),
                ('selector', SelectKBest(k='all')),
                ('classifier', LogisticRegression())])
```

[139]:
```
y_test_pred = logistic_pipeline.predict(X_test)
```

[140]:
```
from sklearn.metrics import roc_auc_score as rocauc
```

[141]:
```
rocscore = rocauc(y_test, y_test_pred)
```

[142]:
```
print(f'Vict Sex Model Accuracy: {100*logistic_pipeline.score(X_test, y_test)}%')
```

```
Vict Sex Model Accuracy: 66.2298739266605%
```

[143]:
```
print(f' Vict Sex ROC AUC Score: {100*rocscore}%')
```

```
 Vict Sex ROC AUC Score: 67.09404516294767%
```

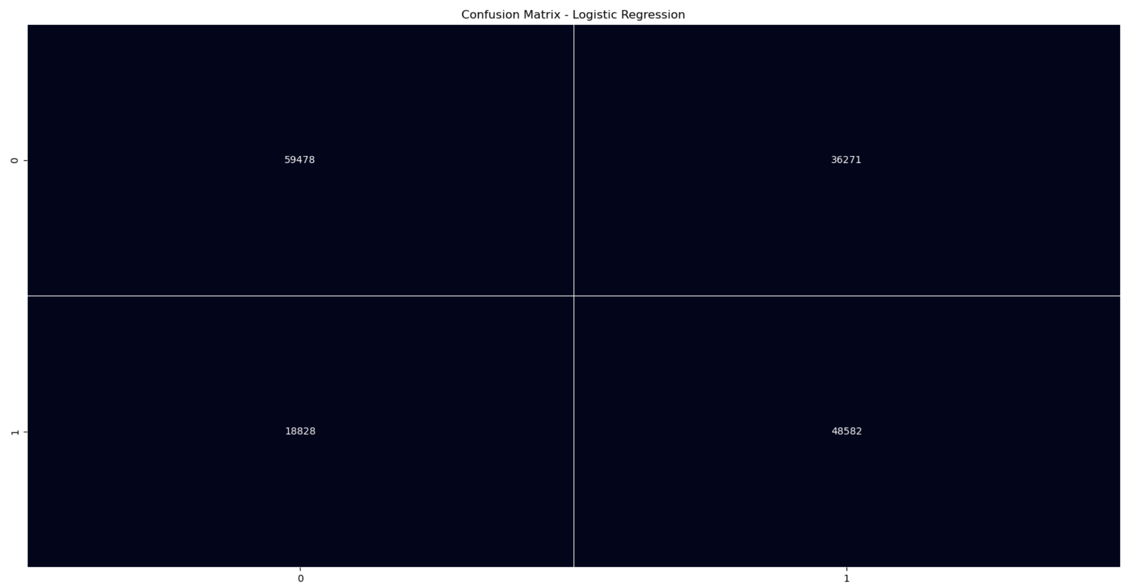
[144]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

[145]:
```
print(classification_report(y_test, y_test_pred))
```

```
              precision    recall  f1-score   support

           0       0.76      0.62      0.68     95749
           1       0.57      0.72      0.64     67410

    accuracy                           0.66    163159
   macro avg       0.67      0.67      0.66    163159
weighted avg       0.68      0.66      0.66    163159
```

[ ]:
```
```

```
[146]: with Suppressor():
           cf_matrix = confusion_matrix(y_test, y_test_pred)
           sns.heatmap(cf_matrix, annot=True, fmt='g',
                   vmin=9999999, vmax=9999999, linewidths=.5,
                         cbar=False).set(
               title="Confusion Matrix - Logistic Regression")
```




```
[147]: ## Regression's confusion matrix highlights the classication's report summary,␣
        ↪and we can further understand the nature of the True / False Positives and␣
        ↪True / False Neagtives within the model.
```

```
[148]: from imblearn.under_sampling import RandomUnderSampler
```

```
[149]: rus = RandomUnderSampler(random_state=12345)
```

```
[150]: X_rus, y_rus = rus.fit_resample(X_train, y_train)
```

```
[151]: logistic_pipeline.fit(X_rus, y_rus)
       y_test_pred = logistic_pipeline.predict(X_test)
```

```
[152]: rocscore = rocauc(y_test, y_test_pred)
```

```
[ ]:
```

```
[153]: print(f'Overall Accuracy: {100*logistic_pipeline.score(X_test, y_test)}%')
       print(f'ROC AUC Score: {100*rocscore}%')
       print(classification_report(y_test, y_test_pred))
```

```
Overall Accuracy: 64.27227428459355%
ROC AUC Score: 68.44096251422485%
              precision    recall  f1-score   support

           0       0.89      0.44      0.59     95749
           1       0.54      0.92      0.68     67410

    accuracy                           0.64    163159
   macro avg       0.72      0.68      0.64    163159
weighted avg       0.75      0.64      0.63    163159
```

[ ]:

[154]:
```python
print('** -------------------------------- **')
print('** Vict Sex Linear Regression Model **')
print('** -------------------------------- **')
```

```
** -------------------------------- **
** Vict Sex Linear Regression Model **
** -------------------------------- **
```

[155]:
```python
## Upon revisiting the data, a comparison of this information was created to␣
↪account for the previous adjustment.
```

[156]:
```python
df4.head(2)
```

[156]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd	Vict Age	Vict Sex	Premis Cd
0	2230	3	377	624	36	F	501
1	330	1	163	624	25	M	102

[157]:
```python
print(df4['Vict Sex'].value_counts())
```

```
M    337050
F    300602
X    178139
Name: Vict Sex, dtype: int64
```

[158]:
```python
df4.replace({'F':0,'M':1, 'X':2}, inplace=True)
```

[159]:
```python
df4.head(2)
```

[159]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd	Vict Age	Vict Sex	Premis Cd
0	2230	3	377	624	36	0	501
1	330	1	163	624	25	1	102

[160]:
```python
X3 = df4.drop('Vict Sex', axis=1)
y3 = df4['Vict Sex']
```

[161]:
```
X3.head(2)
```

[161]:

	TIME OCC	AREA	Rpt Dist No	Crm Cd	Vict Age	Premis Cd
0	2230	3	377	624	36	501
1	330	1	163	624	25	102

[162]:
```
LinearModel3 = linear_model.LinearRegression()
```

[163]:
```
X_train, X_test, y_train, y_test = train_test_split(X3, y3, test_size=0.2,␣
↪random_state=42)
```

[164]:
```
LinearModel2.fit(X3,y3)
```

[164]: LinearRegression()

[165]:
```
model3 = sm.OLS(y3, X3).fit()
```

[166]:
```
print(model3.summary())
```

```
                            OLS Regression Results
================================================================================
=======
Dep. Variable:               Vict Sex   R-squared (uncentered):
0.613
Model:                            OLS   Adj. R-squared (uncentered):
0.613
Method:                 Least Squares   F-statistic:
2.158e+05
Date:                Sat, 18 Nov 2023   Prob (F-statistic):
0.00
Time:                        16:49:49   Log-Likelihood:
-8.7261e+05
No. Observations:              815791   AIC:
1.745e+06
Df Residuals:                  815785   BIC:
1.745e+06
Df Model:                           6
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
TIME OCC        0.0003   1.06e-06    262.218      0.000       0.000       0.000
AREA           -0.5359      0.003   -197.399      0.000      -0.541      -0.531
Rpt Dist No     0.0056   2.69e-05    207.203      0.000       0.006       0.006
Crm Cd          0.0007   3.27e-06    219.587      0.000       0.001       0.001
Vict Age       -0.0143   3.52e-05   -404.685      0.000      -0.014      -0.014
Premis Cd      -0.0001    3.6e-06    -29.985      0.000      -0.000      -0.000
==============================================================================
```

```
Omnibus:                   108306.608   Durbin-Watson:                   1.955
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            33975.673
Skew:                          -0.239   Prob(JB):                         0.00
Kurtosis:                       2.122   Cond. No.                     6.73e+03
==============================================================================
```

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 6.73e+03. This might indicate that there are strong multicollinearity or other numerical problems.

[ ]:

```
[167]: ## *** Milestone 3 Revisit and Revision ***
```

```
[168]: ## Write a short overview/conclusion of the insights gained from your model
       ↪building/evaluation.
```

```
[169]: ## By revisiting this data and accounting for dummie variables, a logistic
       ↪regression analysis demonsrated a signicant decrease in the accuracy of this
       ↪model. It is highly suspect and may need further investigation when accounting
       ↪for the differences in variables.
```

```
[170]: ## When comparing the Linear vs Logistic regression of the models they are quite
       ↪close in their Accuracy when accounting for the model's relationship within
       ↪the variation and test set data.
```