In [1]:

```
#lets start by importing some major libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from statsmodels.tsa.holtwinters import ExponentialSmoothing
from scipy.stats import entropy
from sklearn.cluster import KMeans
from statsmodels.tsa.seasonal import seasonal_decompose
from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import dendrogram,linkage
```

In [2]:

```
#reference:https://moodle4.city.ac.uk/mod/page/view.php?id=379637
crimedata=pd.read_csv('crime.csv')

#in the above code we loaded our dataset
```

In [3]:

```
#reference:https://moodle4.city.ac.uk/mod/page/view.php?id=379637
crimedata.head()

#in the above code we displayed the first few rows of our dataset
```

Out[3]:

	INCIDENT_NUMBER	OFFENSE_CODE	OFFENSE_CODE_GROUP	OFFENSE_DESCRIPTION	DISTRICT	REPORT
0	I182070945	619	Larceny	LARCENY ALL OTHERS	D14	
1	I182070943	1402	Vandalism	VANDALISM	C11	
2	I182070941	3410	Towed	TOWED MOTOR VEHICLE	D4	
3	I182070940	3114	Investigate Property	INVESTIGATE PROPERTY	D4	
4	I182070938	3114	Investigate Property	INVESTIGATE PROPERTY	B3	

In [4]:

```
#firstly we will perform data pre processing
```

In [5]:

```
#reference:https://chartio.com/resources/tutorials/how-to-check-if-any-value-is-nan-in-a
crimedata.isnull().sum()

#in the above code we check for missing values
```

Out[5]:

```
INCIDENT_NUMBER              0
OFFENSE_CODE                 0
OFFENSE_CODE_GROUP           0
OFFENSE_DESCRIPTION          0
DISTRICT                  1765
REPORTING_AREA               0
SHOOTING                318054
OCCURRED_ON_DATE             0
YEAR                         0
MONTH                        0
DAY_OF_WEEK                  0
HOUR                         0
UCR_PART                    90
STREET                   10871
Lat                      19999
Long                     19999
Location                     0
dtype: int64
```

In [6]:

```
#reference:https://www.programiz.com/python-programming/list
droppedcolumns=['INCIDENT_NUMBER','OFFENSE_CODE','OFFENSE_DESCRIPTION','REPORTING_AREA',

#reference:https://www.datacamp.com/tutorial/pandas-drop-column?utm_source=google&utm_me
crimedata.drop(columns=droppedcolumns,inplace=True)

#reference:https://moodle4.city.ac.uk/mod/page/view.php?id=379637
crimedata.head()

#in the above code lines we defined a list with all the columns that we want to remove
#we used the drop function to drop the columns and the inplace=true function to save our
#we use the head function to display the first few rows of our dataset
```

Out[6]:

	OFFENSE_CODE_GROUP	DISTRICT	OCCURRED_ON_DATE	YEAR	MONTH	DAY_OF_WEEK	HOUR	Loc
0	Larceny	D14	02/09/2018 13:00	2018	9	Sunday	13	(42.3577 -71.1393
1	Vandalism	C11	21/08/2018 0:00	2018	8	Tuesday	0	(42.3068 -71.0603
2	Towed	D4	03/09/2018 19:27	2018	9	Monday	19	(42.3465 -71.0724
3	Investigate Property	D4	03/09/2018 21:16	2018	9	Monday	21	(42.3341 -71.0786
4	Investigate Property	B3	03/09/2018 21:05	2018	9	Monday	21	(42.2753 -71.0903

In [7]:

```
#reference:https://www.programiz.com/python-programming/list
wanteddistricts=['A1','A15','A7','B2','B3','C6','C11','D4','D14','E5','E13','E18']

#reference:https://www.w3schools.com/python/pandas/ref_df_isin.asp#:~:text=The%20isin()%
crimedata=crimedata[crimedata['DISTRICT'].isin(wanteddistricts)]

#reference:https://moodle4.city.ac.uk/mod/page/view.php?id=379637
crimedata.head()

#in the above code we defined a list of districts that we want to keep
#we used the isin function to filter the dataset based on if the district columns are in
#we use the head function to display the first few rows of the dataset
```

Out[7]:

	OFFENSE_CODE_GROUP	DISTRICT	OCCURRED_ON_DATE	YEAR	MONTH	DAY_OF_WEEK	HOUR	Loc
0	Larceny	D14	02/09/2018 13:00	2018	9	Sunday	13	(42.3577 -71.1393
1	Vandalism	C11	21/08/2018 0:00	2018	8	Tuesday	0	(42.3068 -71.0603
2	Towed	D4	03/09/2018 19:27	2018	9	Monday	19	(42.3465 -71.0724
3	Investigate Property	D4	03/09/2018 21:16	2018	9	Monday	21	(42.3341 -71.0786
4	Investigate Property	B3	03/09/2018 21:05	2018	9	Monday	21	(42.2753 -71.0903

In [8]:

```
#reference:https://www.geeksforgeeks.org/python-dictionary/
namesfordistricts={
    'A1': 'Downtown',
    'A15': 'Charlestown',
    'A7': 'East Boston',
    'B2': 'Roxbury',
    'B3': 'Mattapan',
    'C6': 'South Boston',
    'C11': 'Dorchester',
    'D4': 'South End',
    'D14': 'Brighton',
    'E5': 'West Roxbury',
    'E13': 'Jamaica Plain',
    'E18': 'Hyde Park'}

#reference:https://www.geeksforgeeks.org/python-pandas-dataframe-replace/
crimedata['DISTRICT']=crimedata['DISTRICT'].replace(namesfordistricts)

#reference:https://moodle4.city.ac.uk/mod/page/view.php?id=379637
crimedata.head()

#in the above code we use curly brackets for dictionaries in order to rename parts of ou
#we use the replace function in order to map the code to the corresponding names set in
#we use the head function to display the first few rows of our dataset
```

Out[8]:

	OFFENSE_CODE_GROUP	DISTRICT	OCCURRED_ON_DATE	YEAR	MONTH	DAY_OF_WEEK	HOUR	Lo
0	Larceny	Brighton	02/09/2018 13:00	2018	9	Sunday	13	(42.357 -71.139
1	Vandalism	Dorchester	21/08/2018 0:00	2018	8	Tuesday	0	(42.306 -71.060
2	Towed	South End	03/09/2018 19:27	2018	9	Monday	19	(42.346 -71.072
3	Investigate Property	South End	03/09/2018 21:16	2018	9	Monday	21	(42.334 -71.078
4	Investigate Property	Mattapan	03/09/2018 21:05	2018	9	Monday	21	(42.275 -71.090

In [9]:

```
#reference:https://www.geeksforgeeks.org/python-dictionary/
namesofthemonths={
    1: 'January',
    2: 'February',
    3: 'March',
    4: 'April',
    5: 'May',
    6: 'June',
```

Loading [MathJax]/extensions/Safe.js

```
    7: 'July',
    8: 'August',
    9: 'September',
    10: 'October',
    11: 'November',
    12: 'December'}

#reference:https://www.geeksforgeeks.org/python-pandas-dataframe-replace/
crimedata['MONTH']=crimedata['MONTH'].replace(namesofthemonths)

crimedata

#in the above code we use curly brackets for dictionaries in order to rename parts of ou
#we use the replace function in order to map the code to the corresponding names set in
```

Out[9]:

	OFFENSE_CODE_GROUP	DISTRICT	OCCURRED_ON_DATE	YEAR	MONTH	DAY_OF_WEEK	HOUR
0	Larceny	Brighton	02/09/2018 13:00	2018	September	Sunday	13
1	Vandalism	Dorchester	21/08/2018 0:00	2018	August	Tuesday	0
2	Towed	South End	03/09/2018 19:27	2018	September	Monday	19
3	Investigate Property	South End	03/09/2018 21:16	2018	September	Monday	21
4	Investigate Property	Mattapan	03/09/2018 21:05	2018	September	Monday	21
...	...	...	...	...	...	...	...
319068	Warrant Arrests	South End	05/06/2016 17:25	2016	June	Sunday	17
319069	Homicide	Hyde Park	09/07/2015 13:38	2015	July	Thursday	13
319070	Warrant Arrests	Hyde Park	09/07/2015 13:38	2015	July	Thursday	13
319071	Warrant Arrests	Jamaica Plain	31/05/2016 19:35	2016	May	Tuesday	19
319072	Warrant Arrests	South End	22/06/2015 0:12	2015	June	Monday	0

317308 rows × 8 columns

In [10]:

```
#reference:https://www.geeksforgeeks.org/how-to-rename-columns-in-pandas-dataframe/
crimedata.rename(columns={
    'OFFENSE_CODE_GROUP':'offense code group',
    'DISTRICT':'district',
    'OCCURRED_ON_DATE':'occurred on date',
    'YEAR':'year',
    'MONTH':'month',
    'DAY_OF_WEEK':'day of week',
    'HOUR':'hour',
    'Location':'location'},inplace=True)

crimedata
#in the above code we renamed the columns of our dataset from upper case to lower case
```

Out[10]:

	offense code group	district	occurred on date	year	month	day of week	hour	location
0	Larceny	Brighton	02/09/2018 13:00	2018	September	Sunday	13	(42.35779134, -71.13937053)
1	Vandalism	Dorchester	21/08/2018 0:00	2018	August	Tuesday	0	(42.30682138, -71.06030035)
2	Towed	South End	03/09/2018 19:27	2018	September	Monday	19	(42.34658879, -71.07242943)
3	Investigate Property	South End	03/09/2018 21:16	2018	September	Monday	21	(42.33418175, -71.07866441)
4	Investigate Property	Mattapan	03/09/2018 21:05	2018	September	Monday	21	(42.27536542, -71.09036101)
...	...	...	...	...	...	...	...	...
319068	Warrant Arrests	South End	05/06/2016 17:25	2016	June	Sunday	17	(42.33695098, -71.08574813)
319069	Homicide	Hyde Park	09/07/2015 13:38	2015	July	Thursday	13	(42.25592648, -71.12317207)
319070	Warrant Arrests	Hyde Park	09/07/2015 13:38	2015	July	Thursday	13	(42.25592648, -71.12317207)
319071	Warrant Arrests	Jamaica Plain	31/05/2016 19:35	2016	May	Tuesday	19	(42.30233307, -71.11156487)
319072	Warrant Arrests	South End	22/06/2015 0:12	2015	June	Monday	0	(42.33383935, -71.08029038)

317308 rows × 8 columns

In [11]:

```
#reference:https://chartio.com/resources/tutorials/how-to-check-if-any-value-is-nan-in-a
crimedata.isnull().sum()

#in the above code we check for null values in our dataset
```

Out[11]:

```
offense code group    0
district              0
occurred on date      0
year                  0
month                 0
day of week           0
hour                  0
location              0
dtype: int64
```

In [12]:

```
#reference:https://www.askpython.com/python-modules/pandas/shape-method#:~:text=To%20use
crimedata.shape

#in the above code we use the shape function to check the shape of our dataset
```

Out[12]:

```
(317308, 8)
```

In [13]:

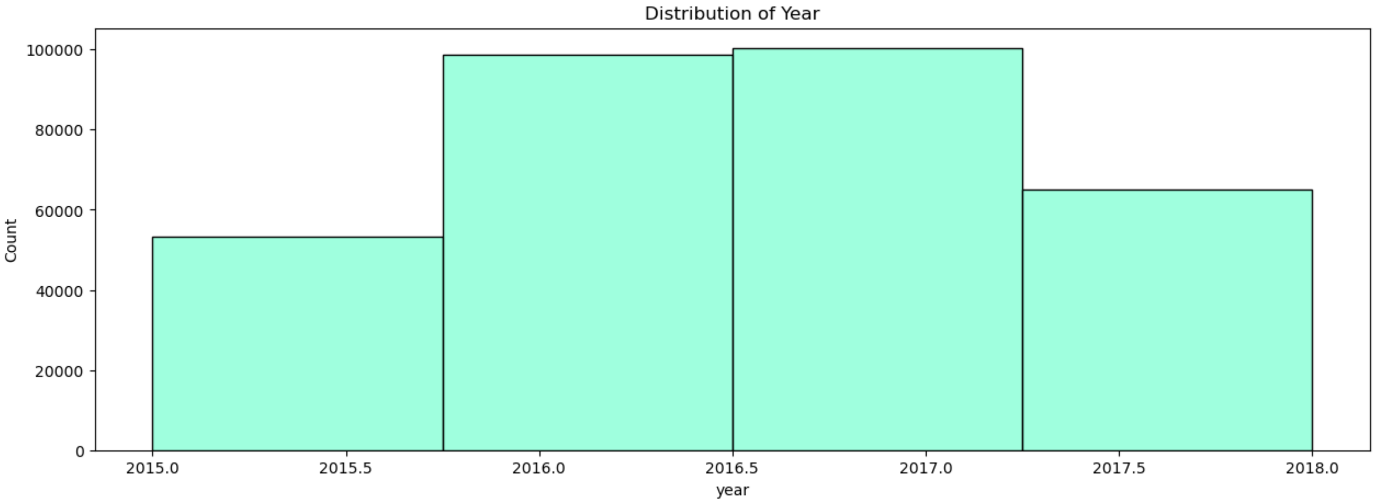
```
#lets continue by performing so exploratory data analysis and vizulization of the variab
```

In [14]:

```
#reference:https://seaborn.pydata.org/tutorial/distributions.html
plt.figure(figsize=(15,5))
sns.histplot(crimedata['year'],bins=4,kde=False,color='aquamarine')
plt.title('Distribution of Year')
plt.show()
```

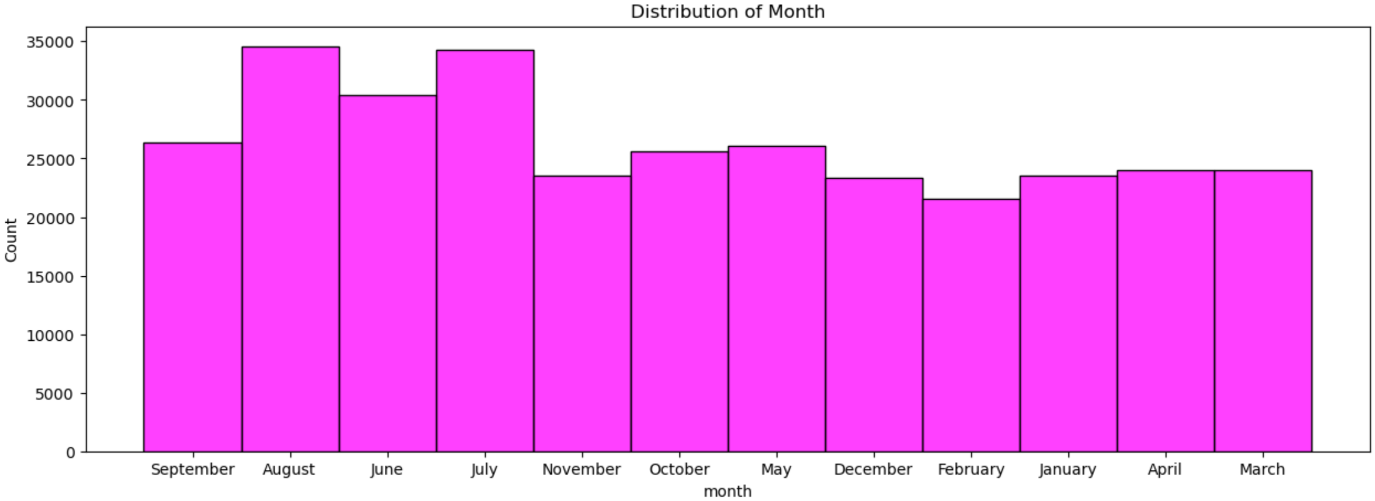
Loading [MathJax]/extensions/Safe.js

```
#in the above code we plot the years of our dataset as a histogram since they are numeri
#we add color and a title to our graph
```



In [15]:

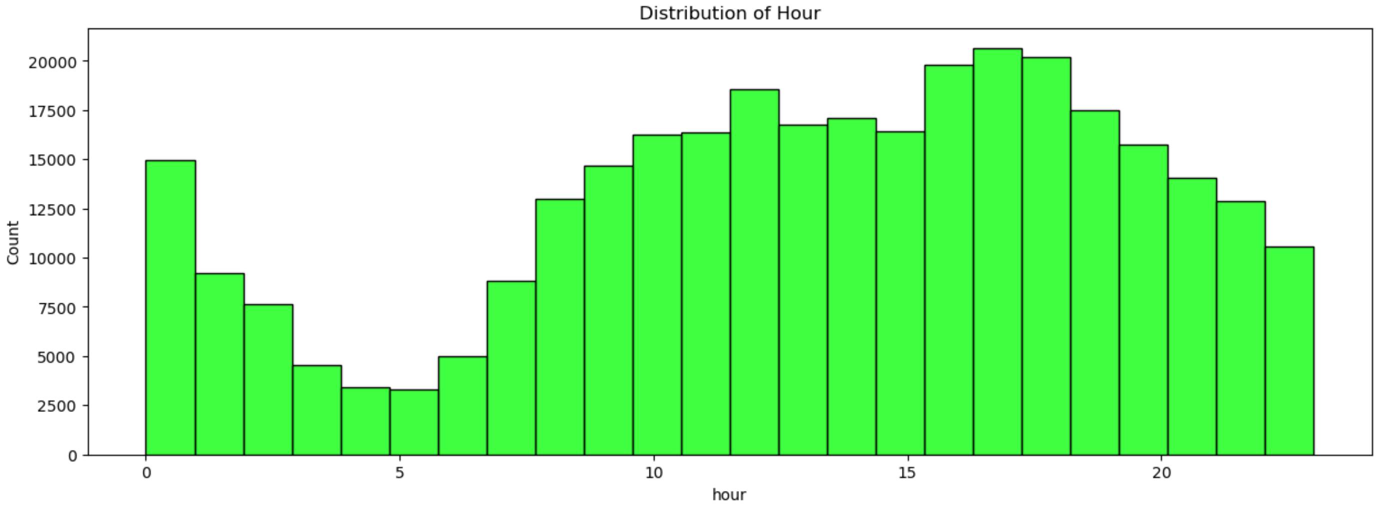
```
#reference:https://seaborn.pydata.org/tutorial/distributions.html
plt.figure(figsize=(15, 5))
sns.histplot(crimedata['month'], bins=12, kde=False,color='magenta')
plt.title('Distribution of Month')
plt.show()

#in the above code we plot the months of our dataset as a histogram since they are numer
#we add color and a title to our graph
```



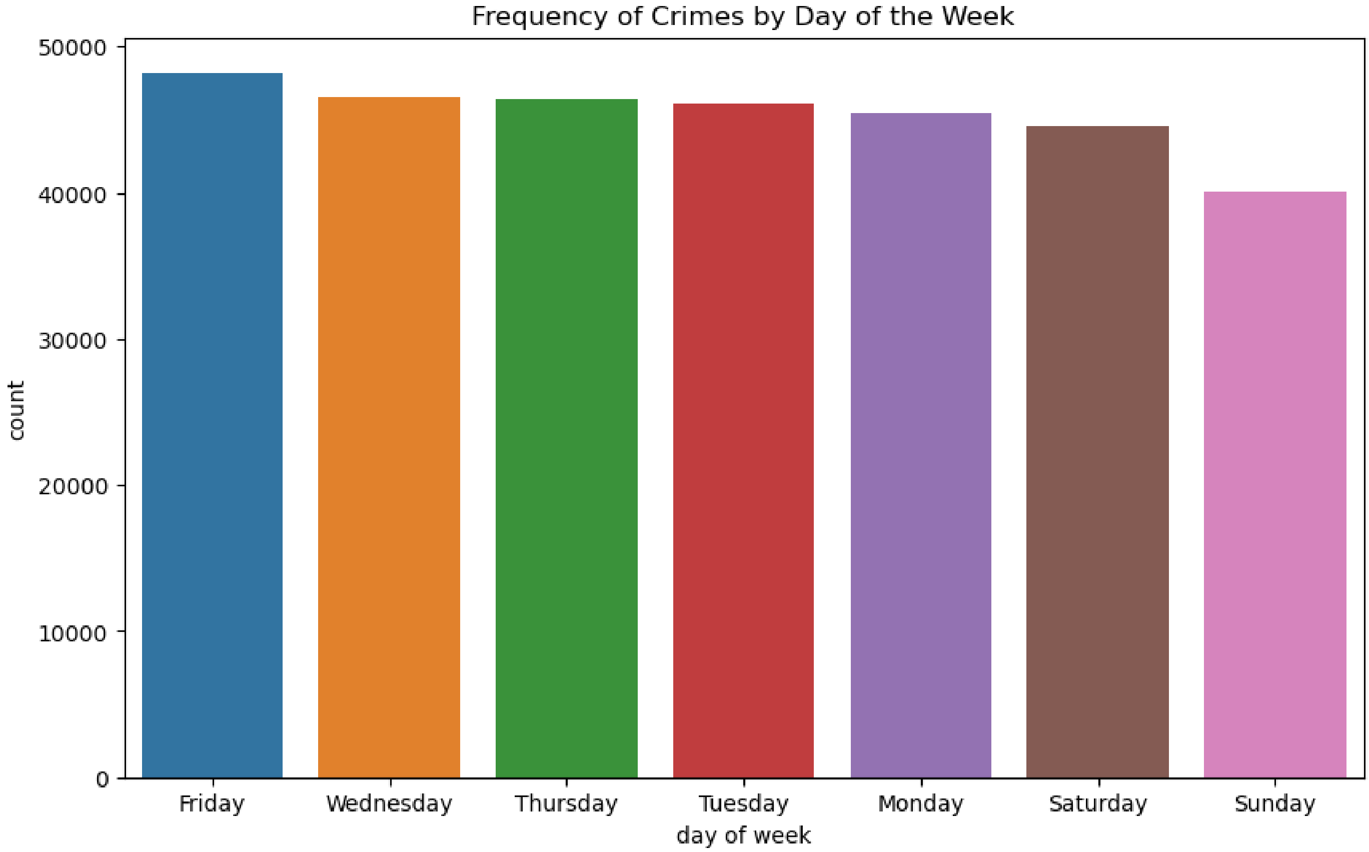
In [16]:

```
#reference:https://seaborn.pydata.org/tutorial/distributions.html
plt.figure(figsize=(15,5))
sns.histplot(crimedata['hour'],bins=24,kde=False,color='lime')
plt.title('Distribution of Hour')
plt.show()

#in the above code we plot the hour column of our dataset as a histogram since they are
#we add color and a title to our graph
```

Loading [MathJax]/extensions/Safe.js



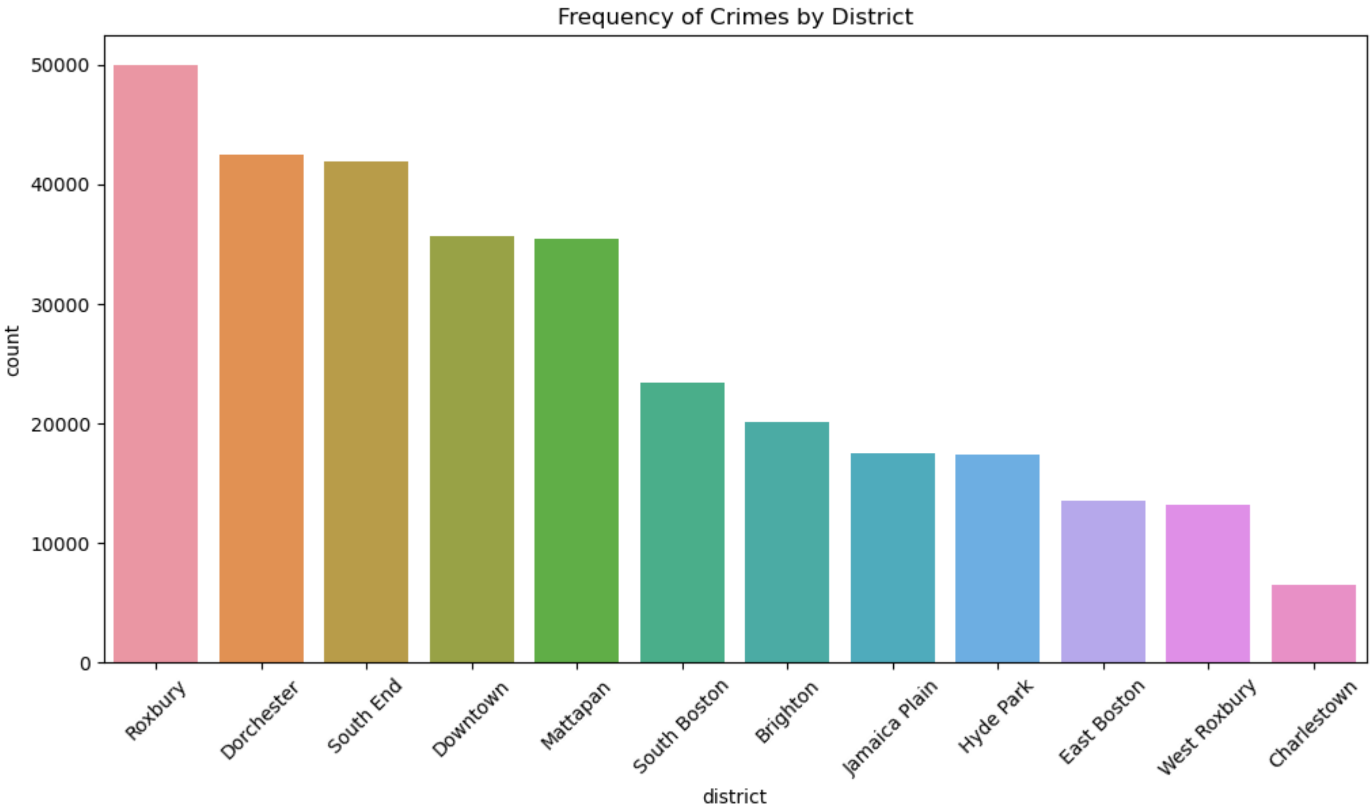
In [17]:
```
#reference:https://seaborn.pydata.org/generated/seaborn.countplot.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
plt.figure(figsize=(10,6))
sns.countplot(x='day of week',data=crimedata,order=crimedata['day of week'].value_counts
plt.title('Frequency of Crimes by Day of the Week')
plt.show()

#in the above code we plot a countplot of the days of the week of our dataset
#we use the value counts function to count how many times each day appears in our datase
#the index command exctracts the unique values from our day of the week column
```



In [18]:
```
#reference:https://seaborn.pydata.org/generated/seaborn.countplot.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
plt.figure(figsize=(12, 6))
sns.countplot(x='district',data=crimedata,order=crimedata['district'].value_counts().ind
plt.title('Frequency of Crimes by District')
plt.xticks(rotation=45)
plt.show()

e code we plot a countplot of the different districts of our dataset
```

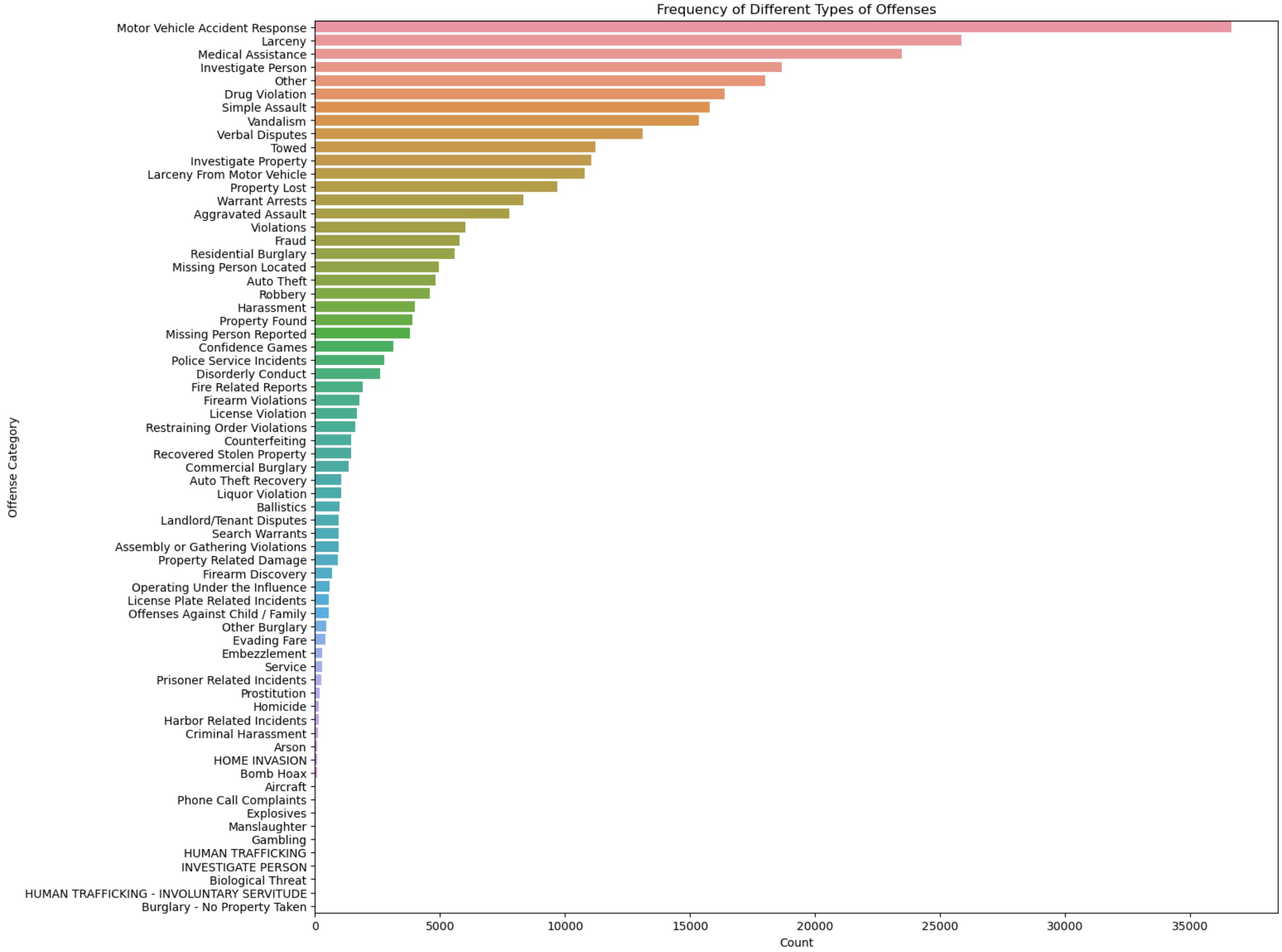
Loading [MathJax]/extensions/Safe.js

```
#we use the value counts function to count how many times each district name  appears in
#the index command exctracts the unique values from district column
#we use the xticks function to rotate our x axis for better visibility
```



In [19]:

```
#reference:https://seaborn.pydata.org/generated/seaborn.countplot.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
plt.figure(figsize=(15, 14))
sns.countplot(y='offense code group',data=crimedata,order=crimedata['offense code group'
plt.title("Frequency of Different Types of Offenses")
plt.xlabel("Count")
plt.ylabel("Offense Category")
plt.show()

#in the above code we plot a countplot of the different types of offences  of our datase
#we use the value counts function to count how many times each offence name  appears in
#the index command exctracts the unique values from the offence column
```



In [22]:

```
#reference:https://www.geeksforgeeks.org/python-pandas-dataframe-describe-method/
crimedata.describe()

#in the above code we get some summary statistics of our numerical variables
```

Out[22]:

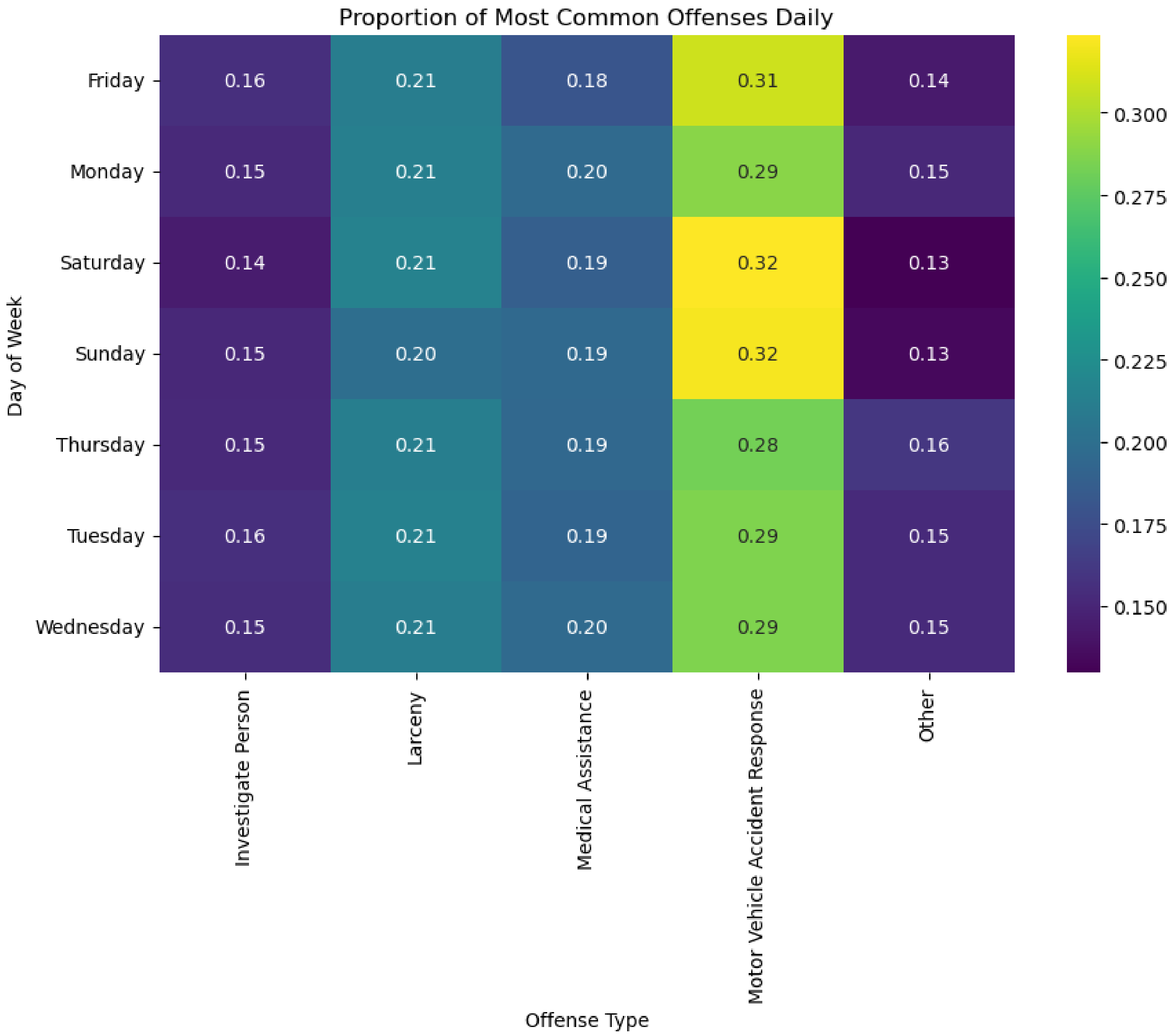
	year	hour
count	317308.000000	317308.000000
mean	2016.558864	13.121409
std	0.996405	6.292247
min	2015.000000	0.000000
25%	2016.000000	9.000000
50%	2017.000000	14.000000
75%	2017.000000	18.000000
max	2018.000000	23.000000

In [23]:

```
#Question 1: Are there any trends in the most common offenses when comparing days of the

#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
commonoffences=crimedata['offense code group'].value_counts().head(5).index

#reference:https://datatofish.com/pivot-table-python/
dailycounts=crimedata[crimedata['offense code group'].isin(commonoffences)].pivot_table(
index='day of week',columns='offense code group',aggfunc='size')

#reference:https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.d
```

Loading [MathJax]/extensions/Safe.js

```
normalized1=dailycounts.div(dailycounts.sum(axis=1),axis=0)

#reference:https://linuxhint.com/seaborn-heatmap-colors/
plt.figure(figsize=(10,6))
sns.heatmap(normalized1,annot=True,fmt=".2f",cmap='viridis')
plt.title('Proportion of Most Common Offenses Daily')
plt.xlabel('Offense Type')
plt.ylabel('Day of Week')
plt.show()

#in the above code we use the value counts function to count how many times the 5 most c
#we create a pivot table with the most commmon offences as the x axis
#we label the y axis as the days of the week
#we normalize by dividing the count of offences by the total offences for each day
#axis = 1 is the total per district
#axis = 0 is dividinb by each row
#we add a title, color and round the correlation answers to 2dp
```

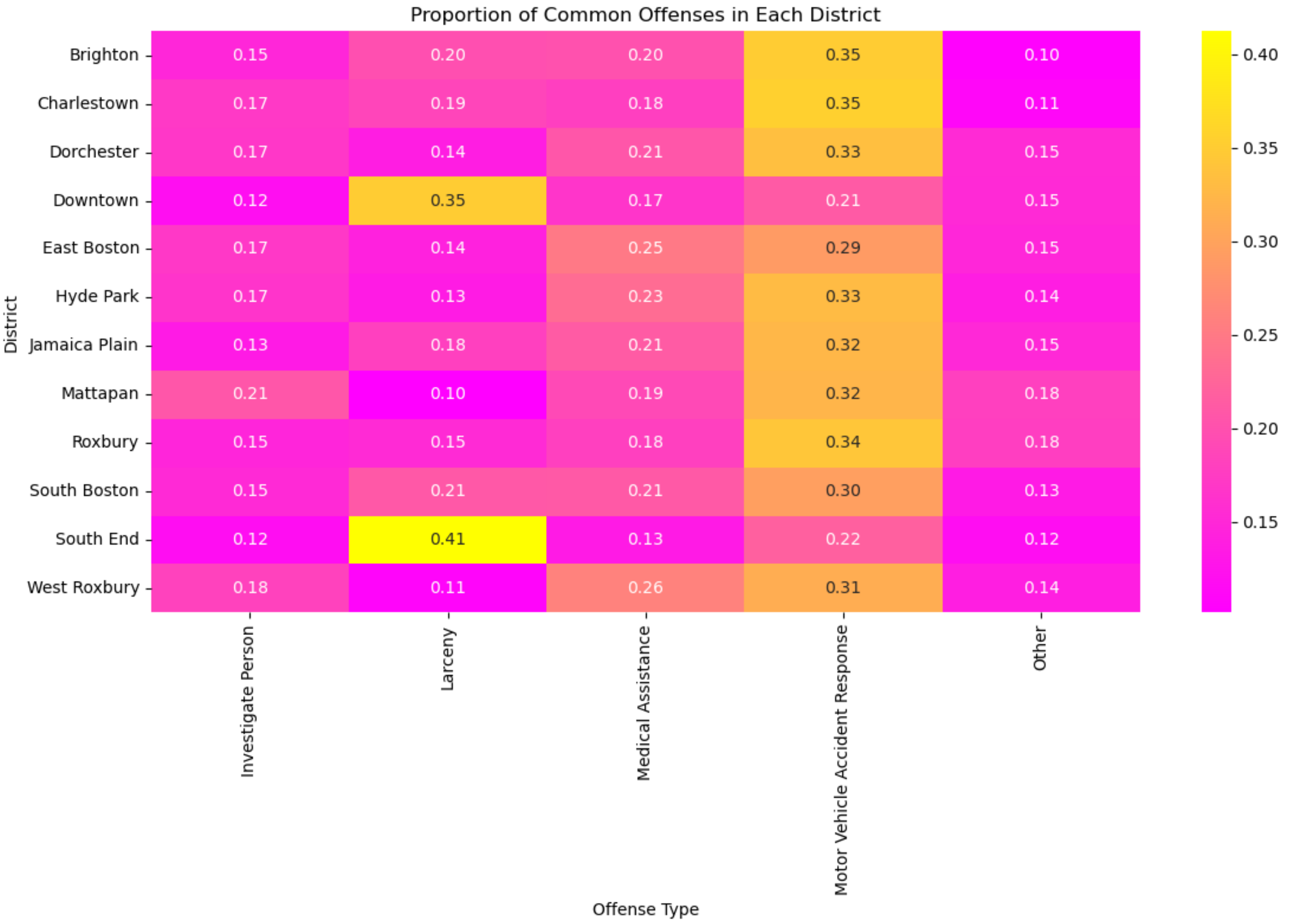


In [24]:

```
#Question 1 continued: Are there any trends in the most common offenses when comparing d

#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
commonoffenses=crimedata['offense code group'].value_counts().head(5).index

#reference:https://datatofish.com/pivot-table-python/
districtcounts=crimedata[crimedata['offense code group'].isin(commonoffenses)].pivot_tab
                   ct',columns='offense code group',aggfunc='size')
```

Loading [MathJax]/extensions/Safe.js

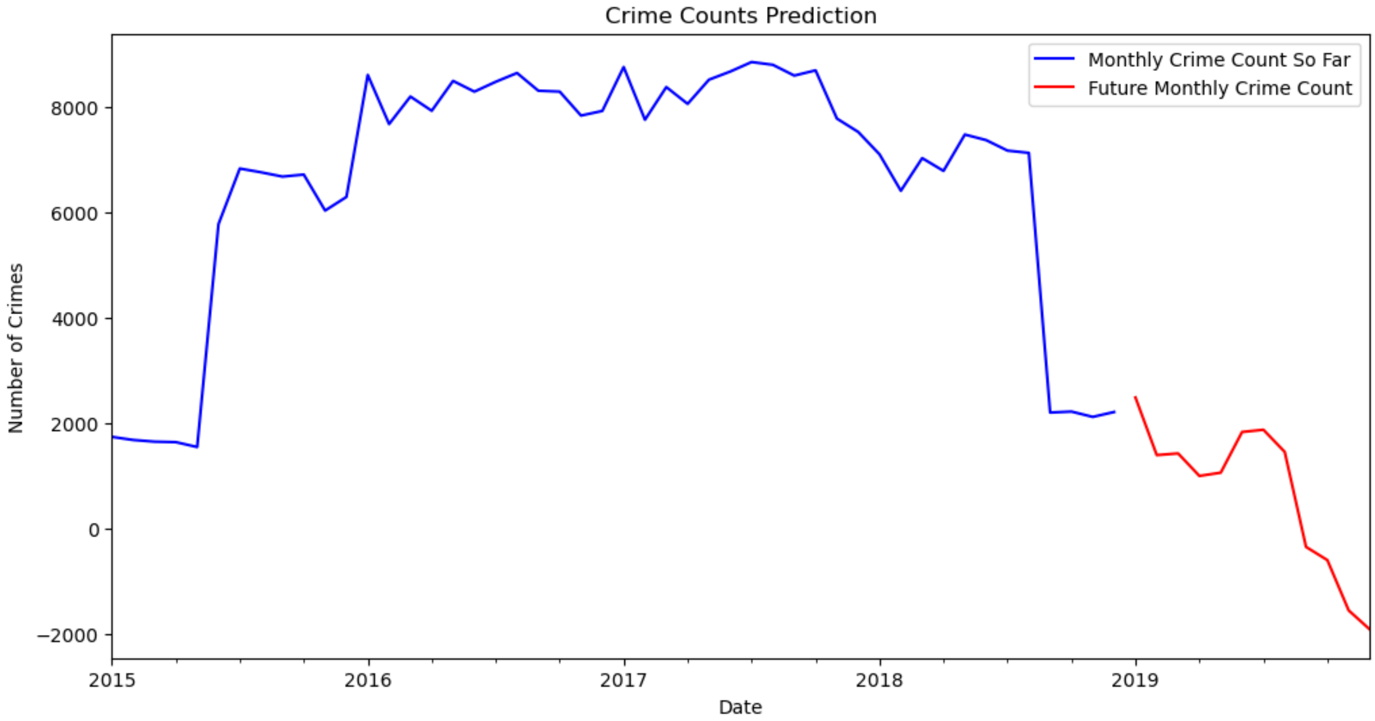
```
#reference:https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.d
normalized2=districtcounts.div(districtcounts.sum(axis=1),axis=0)

#reference:https://linuxhint.com/seaborn-heatmap-colors/
plt.figure(figsize=(12, 8))
sns.heatmap(normalized2,annot=True,fmt=".2f",cmap='spring')
plt.title('Proportion of Common Offenses in Each District')
plt.xlabel('Offense Type')
plt.ylabel('District')
plt.tight_layout()
plt.show()

#in the above code we use the value counts function to count how many times the 5 most c
#we create a pivot table with the most commmon offences as the x axis
#we label the y axis as the different boston districts
#we normalize by dividing each district by the total offences for that district
#axis = 1 is the total per district
#axis = 0 is dividinb by each row
#we add a title, color and round the correlation answers to 2dp
```



In [25]:

```
#Question 2: can we use historical crime data to predict future trends and seasonal crim

#reference:https://saturncloud.io/blog/converting-object-column-in-pandas-dataframe-to-d
crimedata['occurred on date']=pd.to_datetime(crimedata['occurred on date'])

#reference:https://www.geeksforgeeks.org/python-pandas-dataframe-resample/
crimespermonth=crimedata.resample('M',on='occurred on date').size()

#reference:https://www.statsmodels.org/dev/generated/statsmodels.tsa.holtwinters.Exponen
predictivemodel=ExponentialSmoothing(crimespermonth,trend='add',seasonal='add',seasonal_
                                   tps://www.statsmodels.org/dev/generated/statsmodels.tsa.holtwinters.Exponen
```

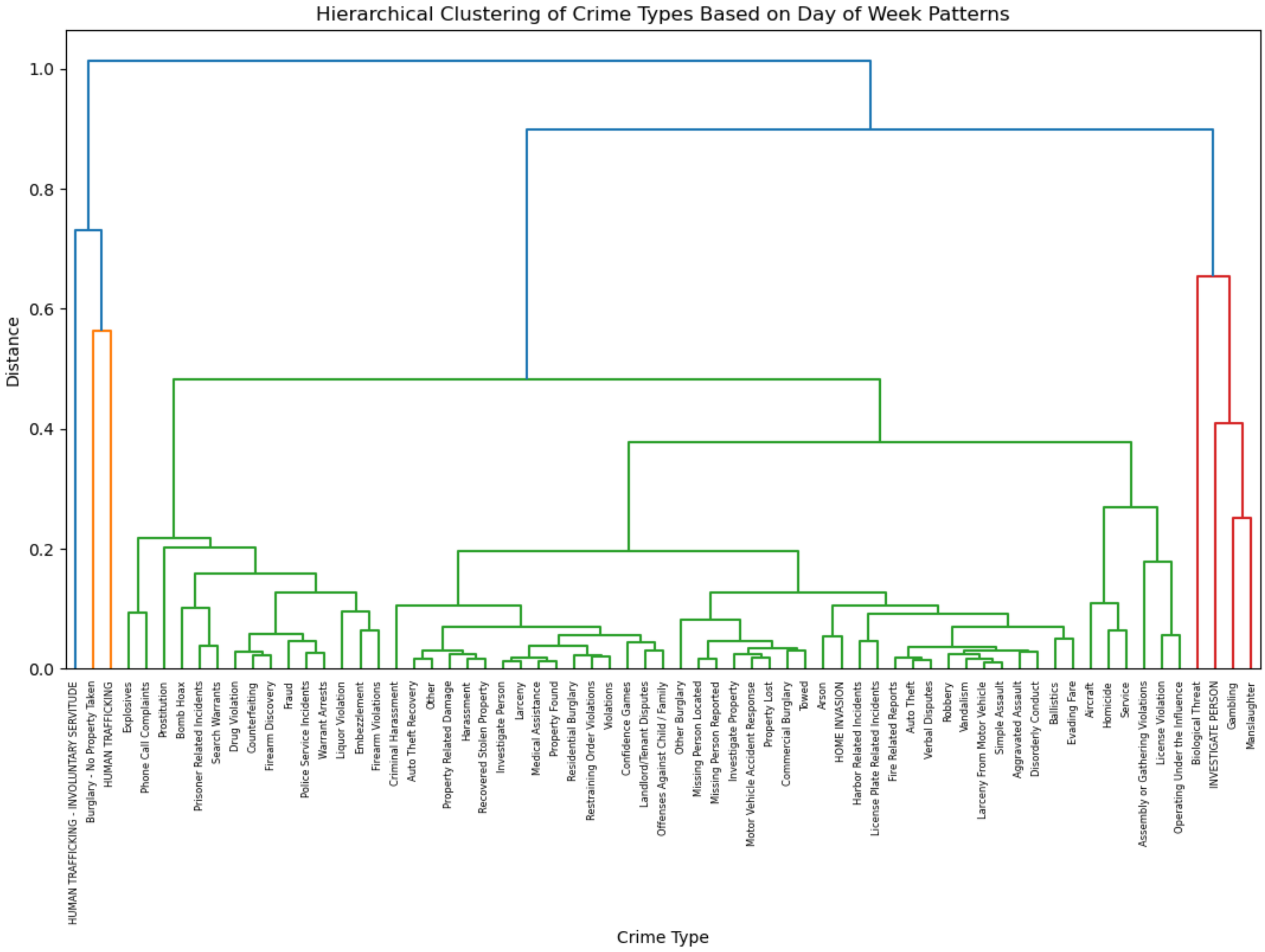
```
fitting=predictivemodel.fit()

#reference:https://www.statsmodels.org/dev/generated/statsmodels.tsa.holtwinters.Exponen
forecasting=fitting.forecast(12)


#reference:https://www.datacamp.com/tutorial/matplotlib-time-series-line-plot
plt.figure(figsize=(12, 6))
crimespermonth.plot(label='Monthly Crime Count So Far', color='blue')
forecasting.plot(label='Future Monthly Crime Count', color='red')
plt.title('Crime Counts Prediction')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.legend()
plt.show()

#in the above code we convert the occured on date column to a date and time format
#we use the resample m code to group our data by months
#we use the size functuion to count the number of crimes per month
#we include trend and seasonality for each month
#we then fit the model and forcast for the following 12 months
#we plot the prediction labeling our graph with a title and axes
```



In [26]:

```
#Question 3: What underlying factors might explain the clustering of crime types by days

#reference:#reference:https://datatofish.com/pivot-table-python/
pivot3=crimedata.pivot_table(index='offense code group',columns='day of week',aggfunc='s

#reference:https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.d
normalized3=pivot3.div(pivot3.sum(axis=1),axis=0)

#reference:https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.
link=linkage(normalized3,method='ward')

#reference:https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.
plt.figure(figsize=(13,7))
dendrogram(link,labels=normalized3.index,orientation='top',leaf_rotation=90)
plt.title('Hierarchical Clustering of Crime Types Based on Day of Week Patterns')
plt.xlabel('Crime Type')
plt.ylabel('Distance')
```

```
#in the above code we create a pivot table  using the different types of crimes
#we normalize by dividing each district by the total offences for that district
#axis = 1 is the total per district
#axis = 0 is dividinb by each row
#the link function is the function helping to plot the dendogram
#we plot the dendogram with a title and labelled axes
#we use leaf rotation for better readability and top orientation to place our roots at t
```



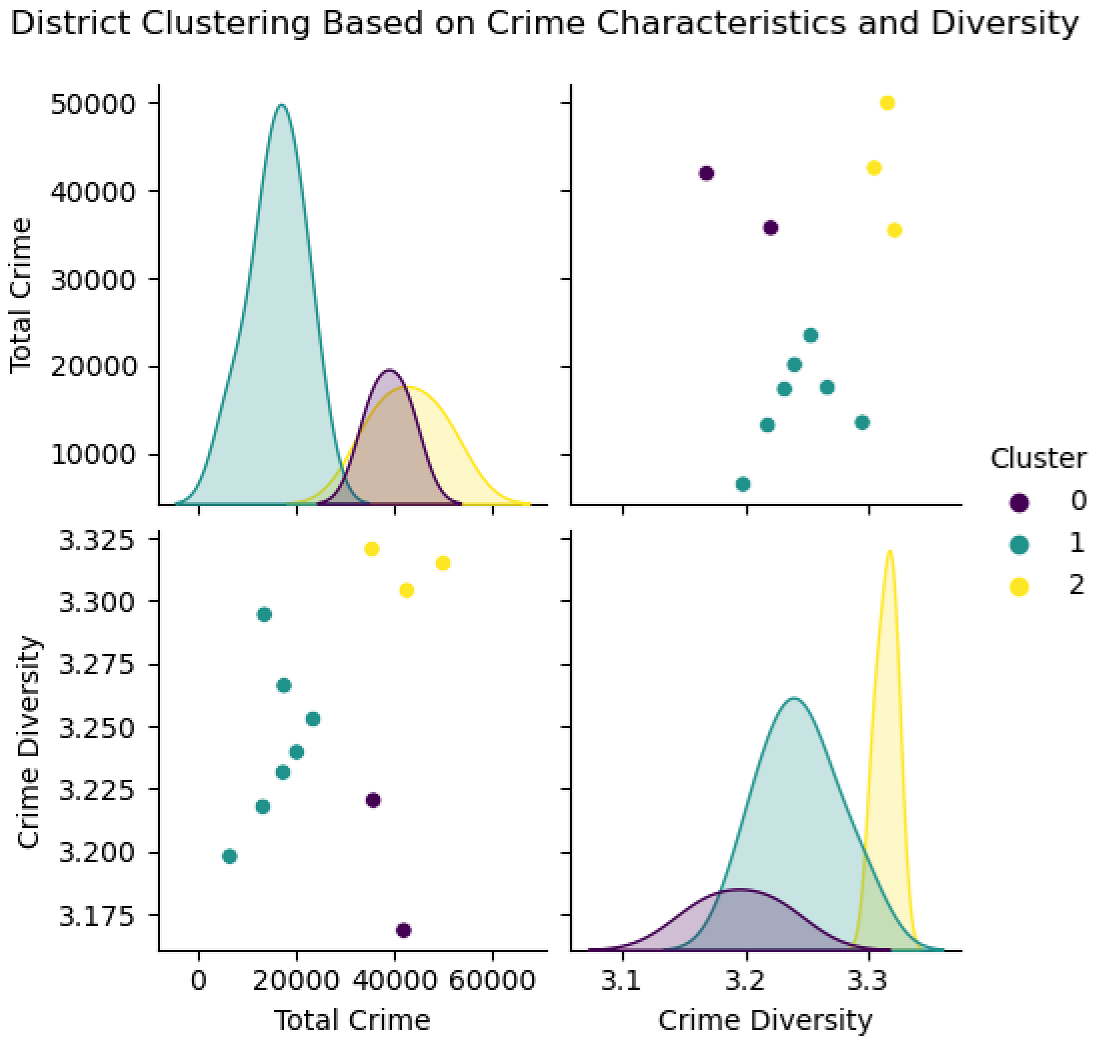
In [27]:

```
#Question 4: Is it possible to identify unique crime pattern clusters among districts by

#reference:https://www.statology.org/pandas-groupby-size/
totalcrimes=crimedata.groupby('district').size()

#reference:https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.core.groupby.DataFrameGro
#reference:https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.entropy.html
diversity=crimedata.groupby('district')['offense code group'].apply(lambda x:entropy(x.v

#reference:https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html
clustering=pd.DataFrame({'Total Crime':totalcrimes,'Crime Diversity':diversity})

#reference:https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning
normalized4=(clustering-clustering.mean())/clustering.std()

#reference:https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
kmeans=KMeans(n_clusters=3,random_state=42)

#reference:https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
clustering['Cluster']=kmeans.fit_predict(normalized4)
```

Loading [MathJax]/extensions/Safe.js

```
#reference:https://www.geeksforgeeks.org/python-seaborn-pairplot-method/
sns.pairplot(clustering,hue='Cluster',palette='viridis')
plt.suptitle("District Clustering Based on Crime Characteristics and Diversity",y=1.05)
plt.show()

#in the above code we use groupby function to group the functions by district
#we use the size function to count the number of crimes
#we then group the data by each district again and we calculate the frequency of each of
#we use the entropy function to calculate entropy and measure the diversity
#we create a new data frame called total crime and crime diversity
#we normalized the data for clustering
#we perform k means clustering to identify 3 clusters
#we use fit predict function to predict clusters from the normalized data
#we use a pair plot with color and title and moved the title upwards for bettter visibil
```

```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of
`n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning:
KMeans is known to have a memory leak on Windows with MKL, when there are less chunks th
an available threads. You can avoid it by setting the environment variable OMP_NUM_THREA
DS=1.
  warnings.warn(
```

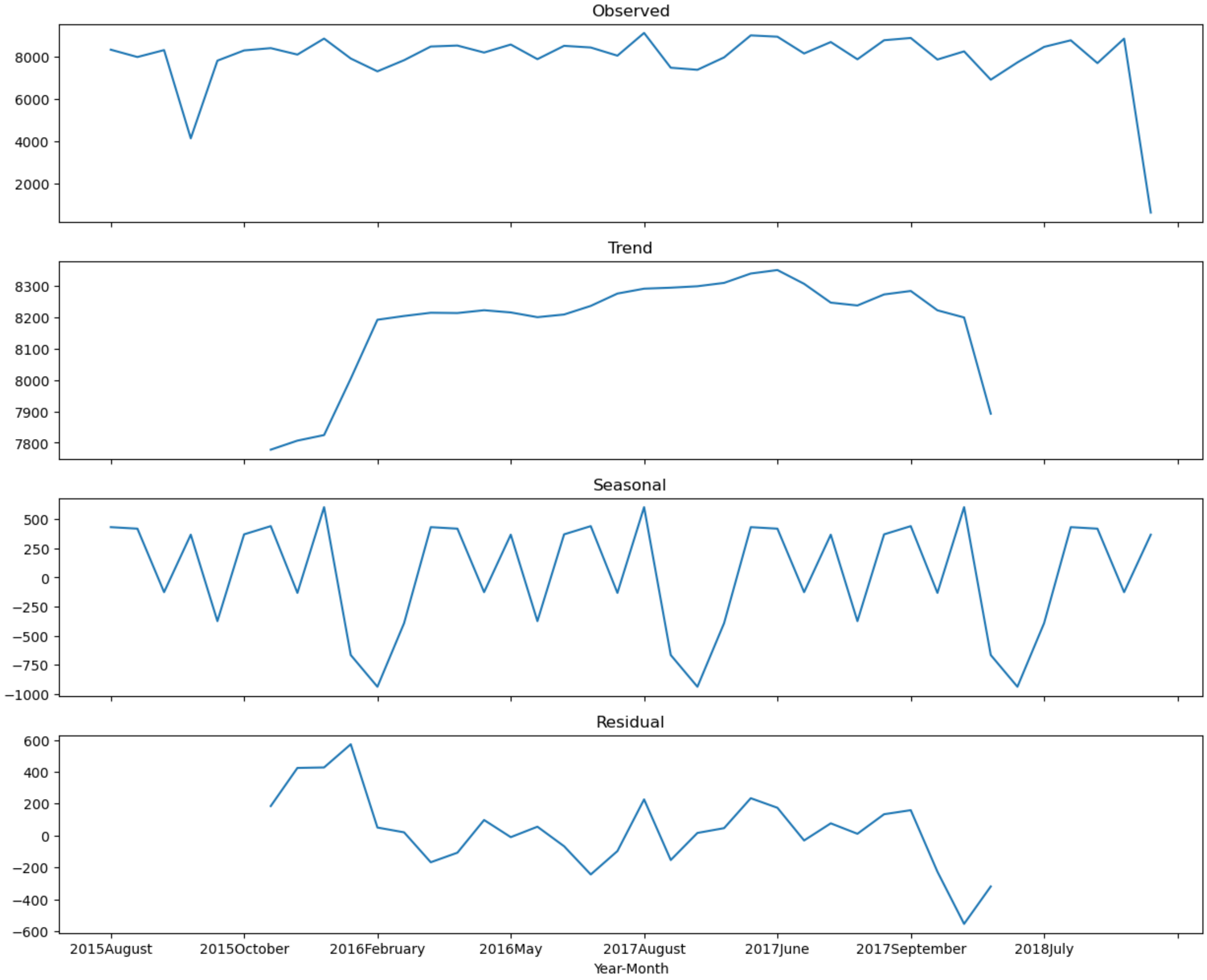


In [28]:

```
#Question 5: What seasonal trends and long-term changes occur in crime rates over time?

#reference:https://pandas.pydata.org/docs/reference/api/pandas.Series.astype.html
crimedata['year_month']=crimedata['year'].astype(str)+""+crimedata['month']
```

Loading [MathJax]/extensions/Safe.js

```
#reference:https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
#reference:https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.count.html
monthlyoffense=crimedata.groupby('year_month')['offense code group'].count()

#reference:https://www.statsmodels.org/dev/generated/statsmodels.tsa.seasonal.seasonal_d
decomposition=seasonal_decompose(monthlyoffense,model='additive',period=12)

#reference:matplotlib subplots documentation
fig,(ax1,ax2,ax3,ax4)=plt.subplots(4,1,figsize=(15,12),sharex=True)

#reference:https://pandas.pydata.org/docs/user_guide/visualization.html
#reference:https://matplotlib.org/stable/api/_as_gen/matplotlib.axes.Axes.set_xlabel.htm
decomposition.observed.plot(ax=ax1, title='Observed')
decomposition.trend.plot(ax=ax2, title='Trend')
decomposition.seasonal.plot(ax=ax3, title='Seasonal')
decomposition.resid.plot(ax=ax4, title='Residual')
ax4.set_xlabel('Year-Month')
plt.show()

#in the above code we assign a new column called year month
#we convert the year column into a string
#we use + " " + to concatinate strings
#in the above code we group or dataset by the year and month
#we count the number of offences per month in the offense code group column
#we decompose the function adding trend and monthly seadonality
#we use sharex = true to allign the x axis
#observed is the original time series
#trend is the trent component of the time series
#seasonal is the sesonality of the time series
#residual is the residual of the time series
```


Observed
8000
6000
4000
2000
Trend
8300
8200
8100
8000
7900
7800
Seasonal
500
250
0
−250
−500
−750
−1000
Residual
600
400
200
0
−200
−400
−600
2015August
2015October
2016February
2016May
2017August
2017June
2017September
2018July
Year-Month