NEW YORK CITY PARKS CRIMES DATA ANALYSIS PROJECT

PROJECT DELIVERABLE 2 – MAY, 2019

Course: Programming for Data Science & Analytics – R and Python

Team: *Charlie's Angels*

TABLE OF CONTENTS

List of Figures

List of Tables

Executive Summary

New York is one of the most densely populated cities in the United States with a population of over 8.6 million as of 2017. A huge part of the population goes to New York City (NYC) parks that covers 14% of the city land, be it individuals or families. There are over 1,700 parks in NYC which have multiple facilities, events and programs that involve a huge public participation, which at the same time also poses a threat of increasing crime incidents. The primary purpose of the project is to provide some valuable insights into investigating the most common crimes occurring in NYC parks, the severity of the crimes, and also the most vulnerable specific locations where most of the crimes are committed. The project's exploratory data analysis found that over the past 4 years from 2015-2018, grand larceny, robbery, and felony assault are the most common crimes occurring in NYC parks. Specifically, in 2018, Burglaries and Grand Larceny of Motor Vehicles experience an increasing trend, whereas the numbers of other crimes tend to decrease. Among seven boroughs, Manhattan and Brooklyn have the highest number of crimes.

Additionally, the project also aims at performing predictive modelling via multiple-linear regression and time series forecast to predict the total number of crimes as well as the number of each type of crime in 2019. Our model findings indicate that Holt-Winter Exponential Smoothing model is selected as the best fit model for our time-ordered crime data since it has lower AIC, BIC and can better account for the seasonality effect in our time series data as compared to Linear Regression model.

Overall, there are two main groups that may benefit from the study. The first group, consisting of the public community who visit the parks, may have an idea knowing which parks/ which facilities of the parks are most prone to crimes so as to avoid or be more cautious whenever visiting these locations. The second group is the NYC police who can gain some insights into classifying which areas might need more security than the others or knowing which quarter/

month in the next year is likely to experience the highest number of crimes. Since parks are considered as one of the most entertaining venues for families and kids most of whom are not always prepared to face a crime, it would be really helpful to know the forecast number of crimes to better prepare and avoid, and we altogether can make NYC parks a safer place than it is right now.

Statement of Scope

The project aims to perform both descriptive and predictive analytics on the seven major crimes (namely *Murder, Rape, Robbery, Assault, Burglary, Grand Larceny, and Grand Larceny of Motor Vehicles*) occurring in NYC parks which operate under the New York City Department of Parks & Recreation. Essentially, the project's primary objectives are listed as following:

1. Identify the top frequent crimes that take place in NYC parks.
2. Explore which locations are most prone towards these crimes.
3. Examine if there exist any significant relationships between the size of the parks, the borough (administrative unit) of the parks, or the types/ facilities of the parks and the number/ type of crime happening there.
4. Analyze the trend of these crimes using Time Series Analysis by comparing the data over four years (2015-2018).
5. Predict the crime situation in NYC parks for the year of 2019 using Decision Tree, Multiple Linear Regression Analysis, and Time Series Forecasting.

Predictor and Target Variable

The original merged data set has about 18,480 observations. After cleaning, we have a sample size of 18,466 rows which will be used for both descriptive analysis and predictive modeling.

The project has multiple ***target variables*** which include the followings:

- The total number of all crimes occurring in a park.
- The number of each type of crimes committed in a park consisting of Murder, Rape, Robbery, Assault, Burglary, Grand Larceny, and Grand Larceny of Motor Vehicle.

The above multiple target variables have the same ***predictors/ independent variables*** as below:

- The size of the park in acres (continuous variable)
- The borough (administrative unit) of the park (categorical variable)
- The types/ categories/ facilities of the park (categorical variable)
- There are also two newly created attributes, one for Year (2015-2018) and one for the Quarter (Q1-Q4) of that particular year, which will also be used as categorical variables in further analysis.

Project Schedule

We completed this project in approximately 105 days. Weekly meetings were scheduled where every week we met for 3 hours to discuss our plans and share our findings. Each team member was assigned with specific individual task based on our discussion and expertise, apart from the coding part which have been agreed to be done together during the meeting to make sure everyone was on the same page. Additionally, the data cleaning and consolidation steps were done in a single coding file so that once these steps were completed, the same file was imported and shared among all of us for further individual work on it. Once the task was allocated, a common google document was maintained to share the work over the week and in the next meeting, we first reviewed everybody's work and then moved on to the next step. This essentially demonstrated our process of completing the project and submitting the final report on time. Below is the GANTT chart for an overview of our progress in the project:

Project Task	January		February				March				April				May
	W4	W5	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1
Project Proposal - Project Idea															
Statement of Scope															
Data Collection															
Data Consolidation															
Data Cleaning															
Data Transformation															
Data Reduction															
Data Description															
Descriptive Statistics															
Executive Summary															
Prepare and Submit First Deliverable															
Adjust Deliverable 1 requirements															
Modeling technique															
Data Splitting & Sub-Samples															
Build Model															
Assess Model															
Combine with First Deliverable															
Make the Final Report															
Formatting and Review of Report															
Presentation															
Record Presentation															
Final Deliverable Presentation															

Data Preparation

1. Data Access

Our data for this project is gathered from New York City government. The link below provides the website address to download the dataset.

https://www1.nyc.gov/site/nypd/stats/crime-statistics/park-crime-stats.page

This dataset is chosen for our project since it has up-to-date statistics with all required attributes spanning across different tables. The dataset also has a considerable large amount of observations which would help us gain more experience in doing analysis on big datasets.

Our dataset provides up-to-date crime-related statistics in the seven major crime categories on the citywide, borough, and precinct levels, as well as historical crime data. We have a total of 16 datasets for separate quarters within the four years' period from 2015 to 2018. The data are in excel/ csv format for 1,154 parks data. We are planning to work on all 1,154 parks by merging all 16 datasets into a single consolidated dataset.

2. Data Dictionary (Preliminary)

Our datasets are comprised of 16 separate data sets for every quarter of each year from 2015 to 2018. The name and embedded link/ source of each data set are provided as below

Year	Dataset Name	Year	Dataset Name
2015	· Fourth Quarter 2015	2017	· Fourth Quarter 2017
	· Third Quarter 2015		· Third Quarter 2017
	· Second Quarter 2015		· Second Quarter 2017
	· First Quarter 2015		· First Quarter 2017
2016	· Fourth Quarter 2016	2018	· Fourth Quarter 2018
	· Third Quarter 2016		· Third Quarter 2018
	· Second Quarter 2016		· Second Quarter 2018
	· First Quarter 2016		· First Quarter 2018

Table 1 - List of Data Sets

The names, data types, and description of all attributes in all of 16 data sets are the same which are provided as below.

	Variable name	Data type	Variable description
1	PARK	Character	Name of the Park
2	BOROUGH	Character	Name of the Administrative Unit that the Park falls under
3	SIZE (ACRES)	Numeric	The size of the park in acres
4	CATEGORY	Character	Type of Park
1	MURDER	Integer	Number of murders in the park
2	RAPE	Integer	Number of rapes in the park
3	ROBBERY	Integer	Number of robberies in the park
4	FELONY ASSAULT	Integer	Number of felony assaults in the park
5	BURGLARY	Integer	Number of burglaries in the park
6	GRAND LARCENY	Integer	Number of grand larcenies in the park
7	GRAND LARCENY OF MOTOR VEHICLE	Integer	Number of grand larcenies of motor vehicles in the park
8	TOTAL	Integer	Total number of all seven crimes in the park

Table 2 - List of Variables

Each of the 16 data sets has the same structure as following:

```
> NYC_Q1_2015 = read.table('nyc-park-crime-stats-q1-2015.csv', sep = ',', header = T, quote = "")
+
> dim(NYC_Q1_2015)
[1] 1155   12
> names(NYC_Q1_2015)
 [1] "PARK"                          "BOROUGH"
 [3] "SIZE..ACRES."                  "CATEGORY"
 [5] "MURDER"                        "RAPE"
 [7] "ROBBERY"                       "FELONY.ASSAULT"
 [9] "BURGLARY"                      "GRAND.LARCENY"
[11] "GRAND.LARCENY.OF.MOTOR.VEHICLE" "TOTAL"
> str(NYC_Q1_2015)
'data.frame':   1155 obs. of  12 variables:
 $ PARK                          : Factor w/ 1155 levels "","\"\"\"UNCLE\"\" VITO E. MARANZANO GL
 $ BOROUGH                       : Factor w/ 7 levels "","BRONX        ",..: 2 2 6 7 6 7 3 4 2 7
 $ SIZE..ACRES.                  : num  2772 1146 1073 913 898 ...
 $ CATEGORY                      : Factor w/ 11 levels "","BASKETBALL & PLAYGROUND LESS THAN ONE
 $ MURDER                        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RAPE                          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ROBBERY                       : int  0 0 1 0 0 0 0 0 0 0 ...
 $ FELONY.ASSAULT                : int  0 0 0 0 1 0 0 0 0 0 ...
 $ BURGLARY                      : int  0 1 0 0 1 0 0 0 0 0 ...
 $ GRAND.LARCENY                 : int  0 0 0 0 1 0 0 0 0 0 ...
 $ GRAND.LARCENY.OF.MOTOR.VEHICLE: int  0 0 0 0 1 0 0 0 0 0 ...
 $ TOTAL                         : int  0 1 1 0 4 0 0 0 0 0 ...
```

3. Data Consolidation

Since all of the 16 datasets have the same number of columns (12) with same names, hence, they can be combined into a single file using the **rbind** function. To prepare for that, we first create two new columns - 'Year' and 'Quarter" - in every dataset to indicate the time of the year when the crime occurs in the park.

```
+ ##2015
+ NYC_Q1_2015$Year = 2015
+
> NYC_Q1_2015$Quarter = 'Q1'
> NYC_Q2_2015$Year = 2015
> NYC_Q2_2015$Quarter = 'Q2'
> NYC_Q3_2015$Year = 2015
> NYC_Q3_2015$Quarter = 'Q3'
> NYC_Q4_2015$Year = 2015
> NYC_Q4_2015$Quarter = 'Q4'
> ##2016
+ NYC_Q1_2016$Year = 2016
+ NYC_Q1_2016$Quarter = 'Q1'
+ NYC_Q2_2016$Year = 2016
+ NYC_Q2_2016$Quarter = 'Q2'
+ NYC_Q3_2016$Year = 2016
+ NYC_Q3_2016$Quarter = 'Q3'
+ NYC_Q4_2016$Year = 2016
+ NYC_Q4_2016$Quarter = 'Q4'
```

```
> ##2017
+ NYC_Q1_2017$Year = 2017
+ NYC_Q1_2017$Quarter = 'Q1'
+ NYC_Q2_2017$Year = 2017
+ NYC_Q2_2017$Quarter = 'Q2'
+ NYC_Q3_2017$Year = 2017
+ NYC_Q3_2017$Quarter = 'Q3'
+ NYC_Q4_2017$Year = 2017
+ NYC_Q4_2017$Quarter = 'Q4'
+
> ##2018
+ NYC_Q1_2018$Year = 2018
+ NYC_Q1_2018$Quarter = 'Q1'
+ NYC_Q2_2018$Year = 2018
+ NYC_Q2_2018$Quarter = 'Q2'
+ NYC_Q3_2018$Year = 2018
+ NYC_Q3_2018$Quarter = 'Q3'
+ NYC_Q4_2018$Year = 2018
+ NYC_Q4_2018$Quarter = 'Q4'
```

Then all columns' names in the data sets are verified to make sure they have exactly matching names for the rbind function to execute successfully. For example, in three data sets of Quarter 4 - 2017, Quarter 1- 2018, and Quarter 3 - 2018, the column "Murder" is in proper format, whereas in all the remaining datasets, this column's name is stored as all uppercase letters ("MURDER"). Its name is then changed accordingly for perfect matching.

```
> #Rename columns for matching
+
> colnames(NYC_Q1_2018)[colnames(NYC_Q1_2018) == "Murder"] = "MURDER"
> colnames(NYC_Q3_2018)[colnames(NYC_Q3_2018) == "Murder"] = "MURDER"
> colnames(NYC_Q4_2017)[colnames(NYC_Q4_2017) == "Murder"] = "MURDER"
```

The process of consolidating is as follow: First, we combine the four quarters data of each year into single dataset for that year. For example, NYC_Q1_2015, NYC_Q2_2015, NYC_Q3_2015, NYC_Q4_2015 are appended together to form NYC_2015 which contains all the crime records for 2015. Similar combinations are performed to create NYC_2016, NYC_2017, and NYC_2018.

```
> #Append datasets for each year
+
> NYC_2015 = do.call("rbind", list(NYC_Q1_2015, NYC_Q2_2015, NYC_Q3_2015, NYC_Q4_2015))
> NYC_2016 = do.call("rbind", list(NYC_Q1_2016, NYC_Q2_2016, NYC_Q3_2016, NYC_Q4_2016))
> NYC_2017 = do.call("rbind", list(NYC_Q1_2017, NYC_Q2_2017, NYC_Q3_2017, NYC_Q4_2017))
> NYC_2018 = do.call("rbind", list(NYC_Q1_2018, NYC_Q2_2018, NYC_Q3_2018, NYC_Q4_2018))
```

In the next step, all the four newly created files are merged together to form the consolidate dataset NYC_Final_Data which consists of 14 columns and 18,466 observations. The details are as following:

```
> # Combina data set of all years into 1 single data set
+ NYC_AllYears = do.call("rbind", list(NYC_2015, NYC_2016, NYC_2017, NYC_2018))
+
> dim(NYC_AllYears)
[1] 18466    14
> names(NYC_AllYears)
 [1] "PARK"                          "BOROUGH"
 [3] "SIZE..ACRES."                  "CATEGORY"
 [5] "MURDER"                        "RAPE"
 [7] "ROBBERY"                       "FELONY.ASSAULT"
 [9] "BURGLARY"                      "GRAND.LARCENY"
[11] "GRAND.LARCENY.OF.MOTOR.VEHICLE" "TOTAL"
[13] "Year"                          "Quarter"
```

```
> head(NYC_AllYears)
                           PARK       BOROUGH SIZE..ACRES.           CATEGORY
1               PELHAM BAY PARK BRONX             2771.747 ONE ACRE OR LARGER
2            VAN CORTLANDT PARK BRONX             1146.430 ONE ACRE OR LARGER
3  ROCKAWAY BEACH AND BOARDWALK QUEENS             1072.564 ONE ACRE OR LARGER
4               FRESHKILLS PARK STATEN ISLAND       913.320 ONE ACRE OR LARGER
5  FLUSHING MEADOWS CORONA PARK QUEENS              897.690 ONE ACRE OR LARGER
6 LATOURETTE PARK & GOLF COURSE STATEN ISLAND       843.970 ONE ACRE OR LARGER
  MURDER RAPE ROBBERY FELONY.ASSAULT BURGLARY GRAND.LARCENY
1      0    0       0              0        0              0
2      0    0       0              0        1              0
3      0    0       1              0        0              0
4      0    0       0              0        0              0
5      0    0       0              1        1              1
6      0    0       0              0        0              0
  GRAND.LARCENY.OF.MOTOR.VEHICLE TOTAL Year Quarter
1                              0     0 2015      Q1
2                              0     1 2015      Q1
3                              0     1 2015      Q1
4                              0     0 2015      Q1
5                              1     4 2015      Q1
6                              0     0 2015      Q1
```

4. Data Cleaning

With the consolidated data set, some preliminary analyses are performed to identify if there are any missing values/ erroneous data or any potential outliers.

a. Column names change

The names of some columns are first changed to meet some naming convention as below:

```
> #Rename columns to match naming conventions
+ colnames(NYC_AllYears)[colnames(NYC_AllYears) == "SIZE..ACRES."] = "SIZE"
+
> colnames(NYC_AllYears)[colnames(NYC_AllYears) == "FELONY.ASSAULT"] = "FELONY_ASSAULT"
> colnames(NYC_AllYears)[colnames(NYC_AllYears) == "GRAND.LARCENY"] = "GRAND_LARCENY"
> colnames(NYC_AllYears)[colnames(NYC_AllYears) == "GRAND.LARCENY.OF.MOTOR.VEHICLE"] = "GRAND_LARCENY__MOTOR"
```

b. Missing values

The whole dataset is then scanned to identify any missing values.

```
> #Data cleaning - remove missing values
+ NYC_AllYears[!complete.cases(NYC_AllYears),]   #list of rows that have missing values
+
     PARK BOROUGH SIZE CATEGORY MURDER RAPE ROBBERY FELONY_ASSAULT BURGLARY
1155               NA       NA     NA   NA      NA             NA       NA
5772               NA       NA     NA   NA      NA             NA       NA
     GRAND_LARCENY GRAND_LARCENY__MOTOR TOTAL Year Quarter
1155            NA                   NA    NA 2015      Q1
5772            NA                   NA    NA 2016      Q1
>
```

As indicated by the above result, out of 18,466 rows, there are two rows that have missing values. Since the number of missing values is much small in relation to the whole data set, thus removing these two rows would not hurt the analysis.

```
> NYC_Final_Data = na.omit(NYC_AllYears)
> NYC_Final_Data[!complete.cases(NYC_Final_Data),]
 [1] PARK                BOROUGH             SIZE                CATEGORY
 [5] MURDER              RAPE                ROBBERY             FELONY_ASSAULT
 [9] BURGLARY            GRAND_LARCENY       GRAND_LARCENY__MOTOR TOTAL
[13] Year                Quarter
<0 rows> (or 0-length row.names)
```

After the removal (omit) of these two rows, the data had zero missing values as can be seen from the above output.

c. Detection of outliers

Outliers can be detected by using the *describe()* function for numeric results or visualized by histograms, box plots, or scatter plots. Histograms and boxplots are excellent for viewing the spread of the data. A histogram provides the frequency in which certain data points appear. This also lets us have an idea about the skewness and kurtosis of the data. Additionally, a box plot shows the mean, median, and any potential outliers of the data.

```
#Check for outliers
par(mfrow = c(1, 1)) #i.e., display 2 graphics with 1 row and 2 columns

hist(NYC_Final_Data$MURDER, main = "Histogram of Murder")
boxplot(NYC_Final_Data$MURDER, main = "Boxplot of Murder")

hist(NYC_Final_Data$RAPE, main = "Histogram of Rape")
boxplot(NYC_Final_Data$RAPE, main = "Boxplot of Rape")

hist(NYC_Final_Data$ROBBERY, main = "Histogram of Robbery")
boxplot(NYC_Final_Data$ROBBERY, main = "Boxplot of Robbery")

hist(NYC_Final_Data$FELONY_ASSAULT, main = "Histogram of Felony Assault")
boxplot(NYC_Final_Data$FELONY_ASSAULT, main = "Boxplot of Felony Assault")

hist(NYC_Final_Data$BURGLARY, main = "Histogram of Burglary")
boxplot(NYC_Final_Data$BURGLARY, main = "Boxplot of Burglary")

hist(NYC_Final_Data$GRAND_LARCENY, main = "Histogram of Grand Larceny")
boxplot(NYC_Final_Data$GRAND_LARCENY, main = "Boxplot of Grand Larceny")

hist(NYC_Final_Data$GRAND_LARCENY__MOTOR, main = "Histogram of Grand Larceny of Motor Vehicle")
boxplot(NYC_Final_Data$GRAND_LARCENY__MOTOR, main = "Boxplot of Grand Larceny of Motor Vehicle")
```

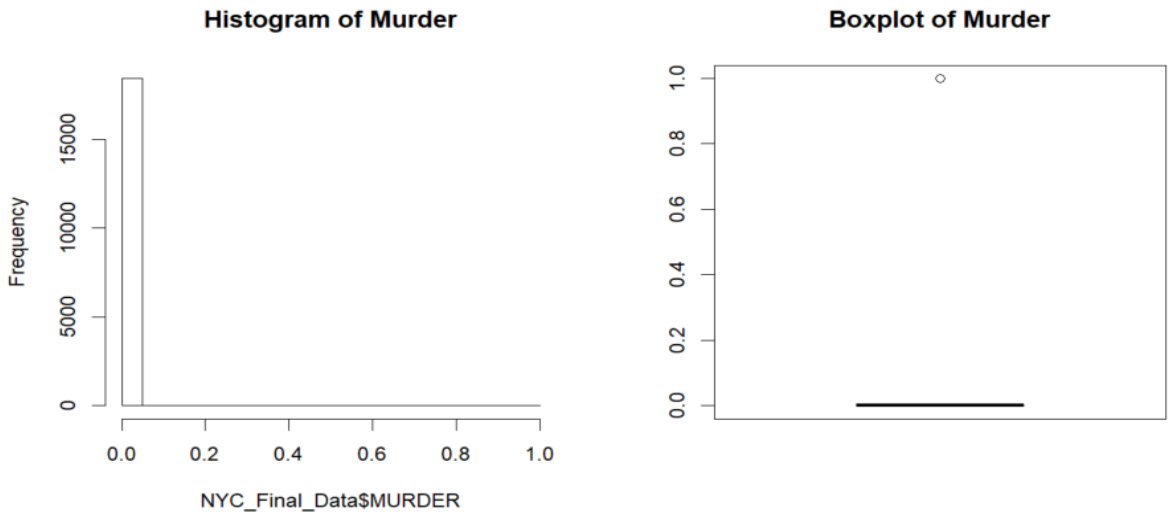


Figure 1 - Distribution of Murder

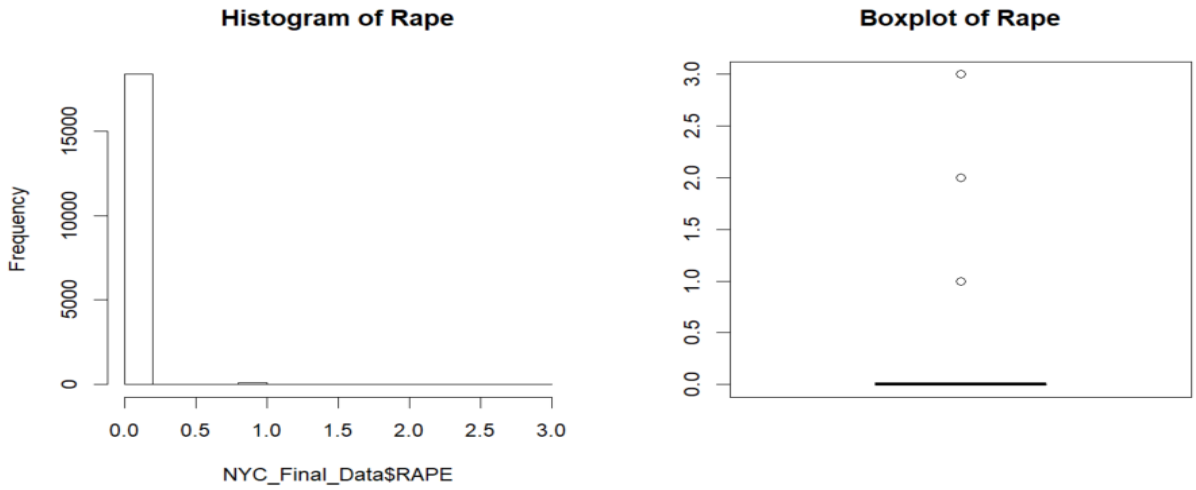


Figure 2 - Distribution of Rape

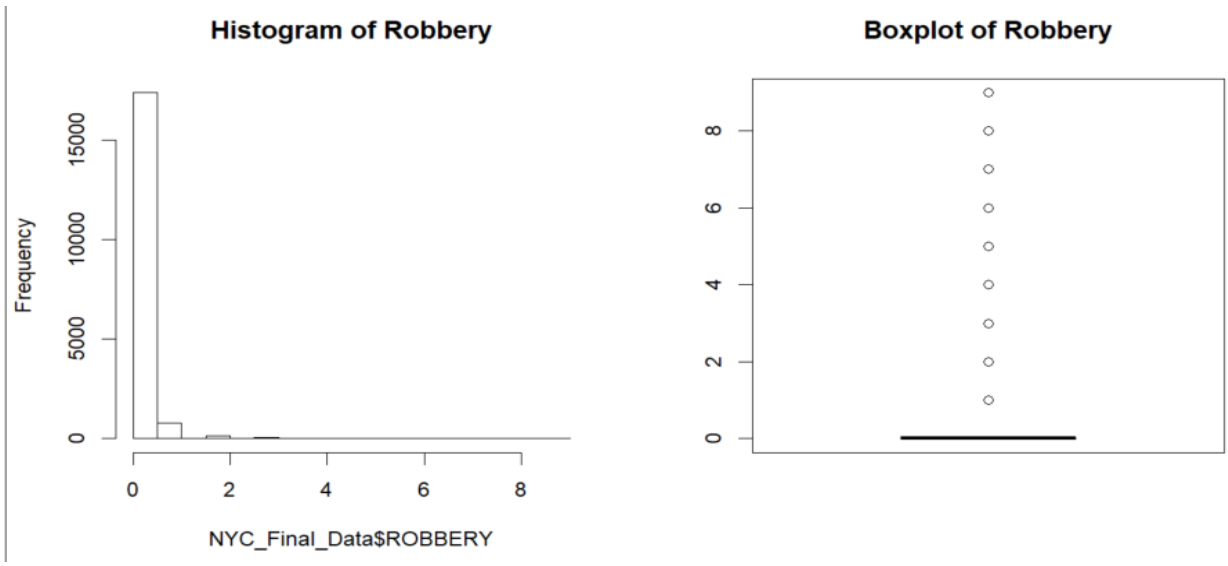


Figure 3 - Distribution of Robbery

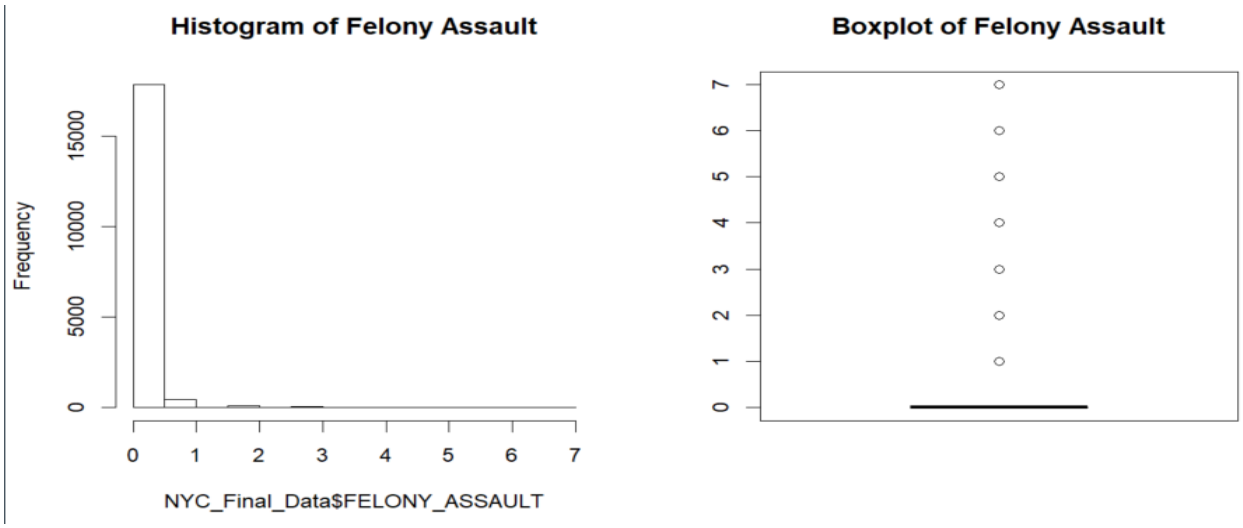


Figure 4 - Distribution of Felony Assault

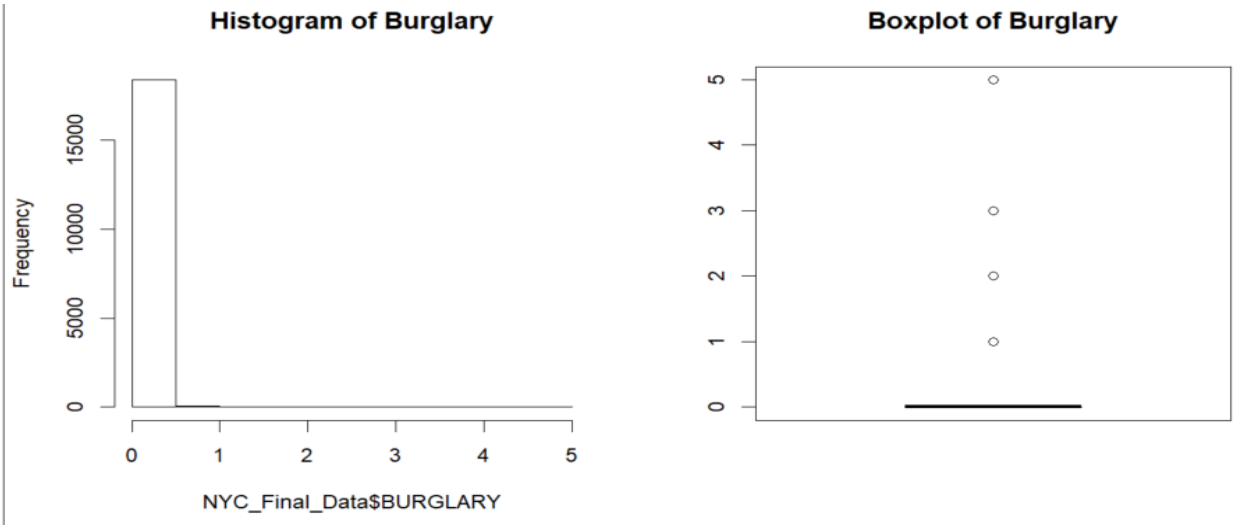


Figure 5 - Distribution of Burglary

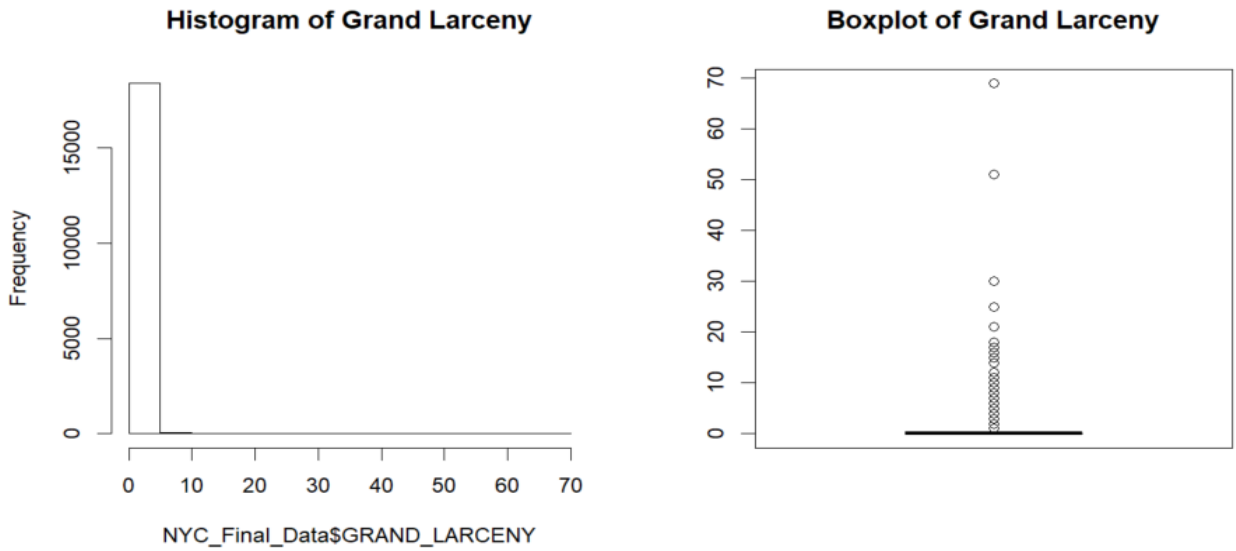


Figure 6 - Distribution of Grand Larceny

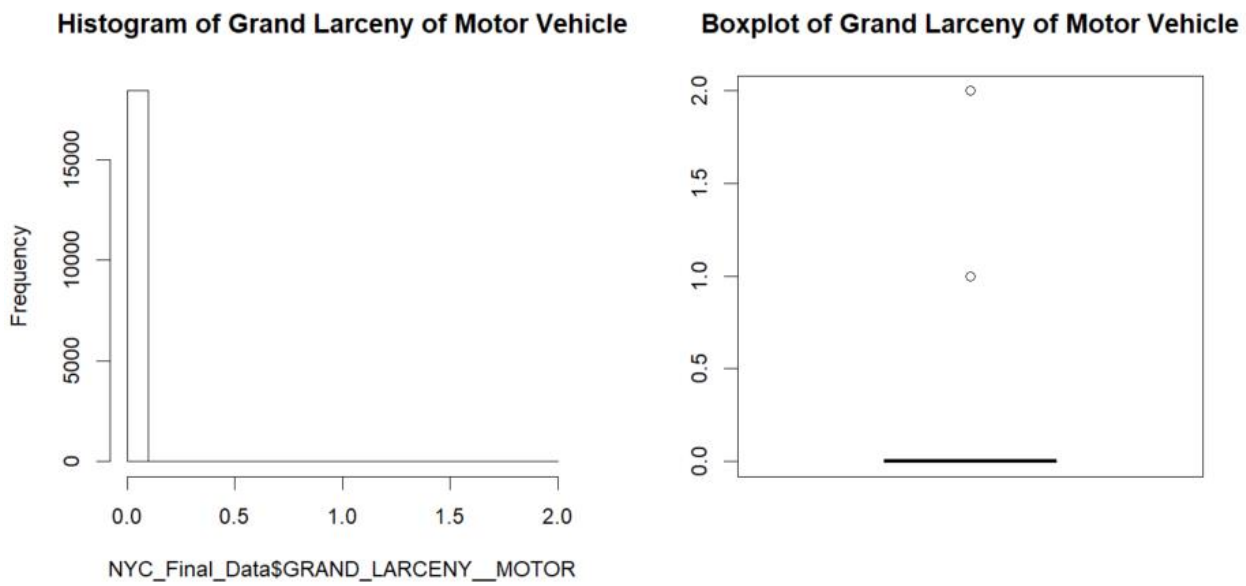


Figure 7 - Distribution of Grand Larceny Motor

From the histograms and box plots of seven types of crimes (*Murder, Rape, Robbery, Felony Assault, Burglary, Grand Larceny, Grand Larceny of Motor Vehicle*), it can be seen that all these crime variables experience a right skewed distribution. There are a couple of potential outliers that lie above the horizontal bar in each distribution.

A further look into the *SIZE* of the park and the *TOTAL* number of crimes happening at the park indicates that either SIZE or TOTAL has a very skewed distribution, as illustrated by the following *describe()* statistics and histograms/box plots of the two variables:

```
> describe(NYC_Final_Data$SIZE)
   vars     n mean     sd median trimmed  mad min     max   range  skew kurtosis
X1    1 18464   25 123.28   1.59    3.58 1.54   0 2771.75 2771.75 12.77   232.64
     se
X1 0.91
> describe(NYC_Final_Data$TOTAL)
   vars     n mean   sd median trimmed mad min max range skew kurtosis   se
X1    1 18464 0.25 1.33      0    0.02   0   0  71    71 20.4   738.15 0.01
```

```
hist(NYC_Final_Data$SIZE, main = "Histogram of Size")
boxplot(NYC_Final_Data$SIZE, main = "Boxplot of Size")

boxplot(NYC_Final_Data$TOTAL, main = "Boxplot of Total Number of Crimes")
hist(NYC_Final_Data$TOTAL, main = "Histogram of Total Number of Crimes")

plot(NYC_Final_Data$SIZE, NYC_Final_Data$TOTAL, main = "Scatterplot
                          of Total Number of Crimes and Park Size")
```

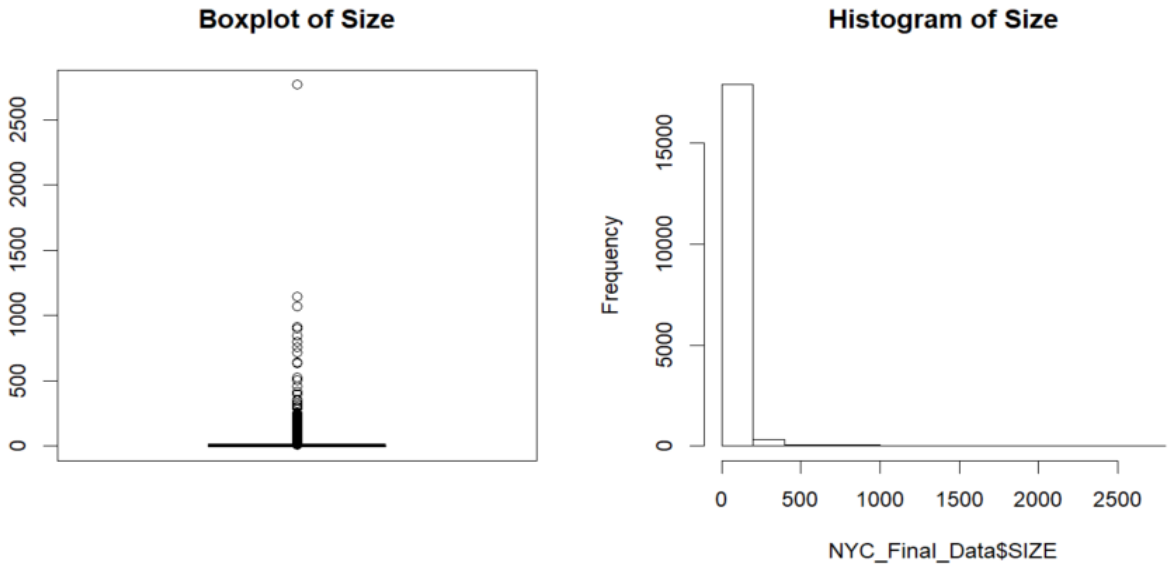


Figure 8 - Distribution of Size

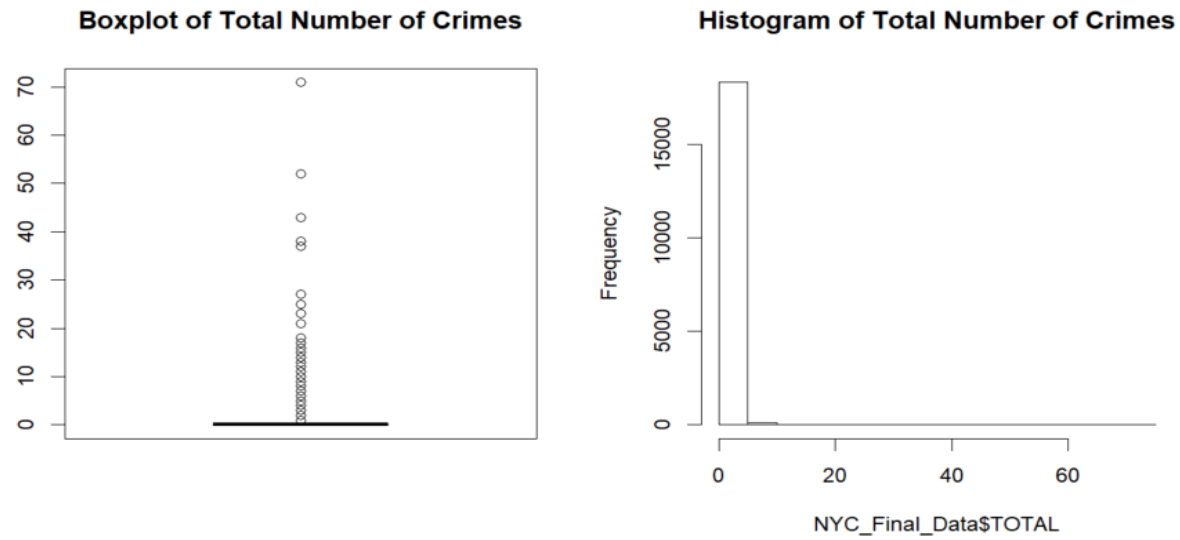


Figure 9 - Distribution of Total number of Crimes

The histograms of both SIZE and TOTAL are quite skewed. Their box plots also indicate a couple of potential outliers which means that there are some certain parks that experience really high number of crimes as compared to other parks.

While a histogram can help visually assess the skewness, it is better to use more objective criteria. The skewness of Size and Total is positive (12.77 and 20.4, respectively) which also suggests the same thing: a right skewed distribution. Kurtosis is another measure of normality that deals with the peak of a distribution. The kurtosis of both variables are very high which indicates a very pointy, or leptokurtic distribution.

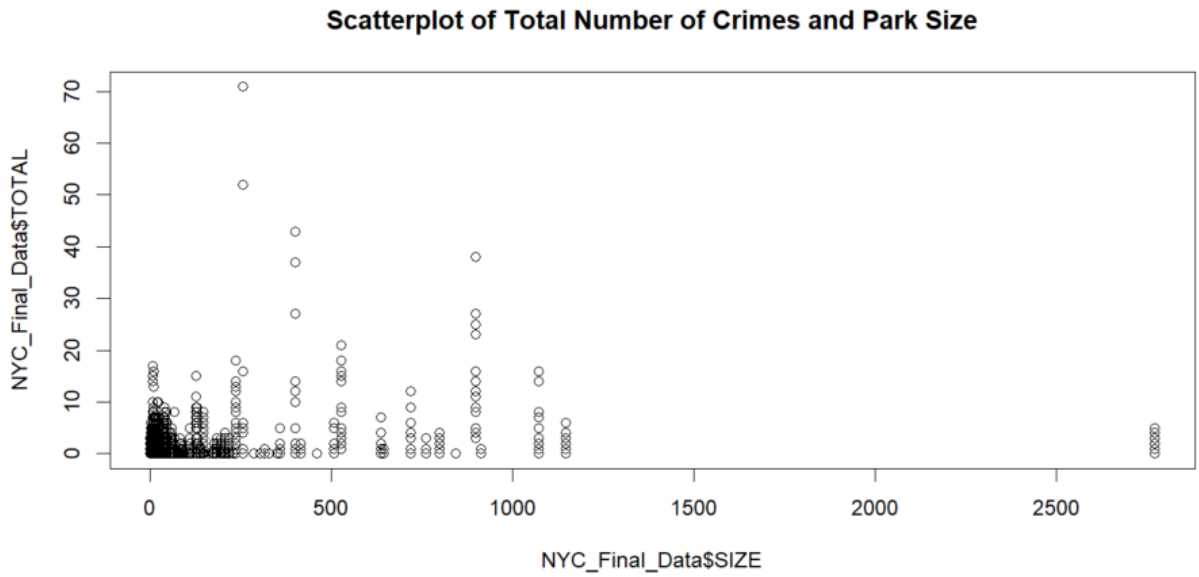


Figure 10 - Scatterplot of Total number of Crimes and Park Size

The above scatter plot between Size and the Total number of crimes illustrates a non-linear relationship between two variables. There are, again, a couple of potential outliers that can be clearly observed from the scatter plot. These extreme data points may be one of the reasons that cause the plot to deviate from linearity.

d. Adjustment to data type when importing the csv file to R

```
> #Adjustment for data types
+ str(NYC_Final_Data$TOTAL)
+
 int [1:18464] 0 1 1 0 4 0 0 0 0 0 ...
> NYC_Final_Data$TOTAL = as.numeric(NYC_Final_Data$TOTAL)
> str(NYC_Final_Data$TOTAL)
 num [1:18464] 0 1 1 0 4 0 0 0 0 0 ...
```

5. Data Transformation

a. Check for normality

```
> #Check for normality
+ qqnorm(NYC_Final_Data$SIZE, main = "Size of the Park QQ-Plot")
+ qqline(NYC_Final_Data$SIZE, lty = 2)
+
> #Anderson-Darling normality test
+ ad.test(NYC_Final_Data$SIZE)$p.value
+
[1] 3.7e-24
```

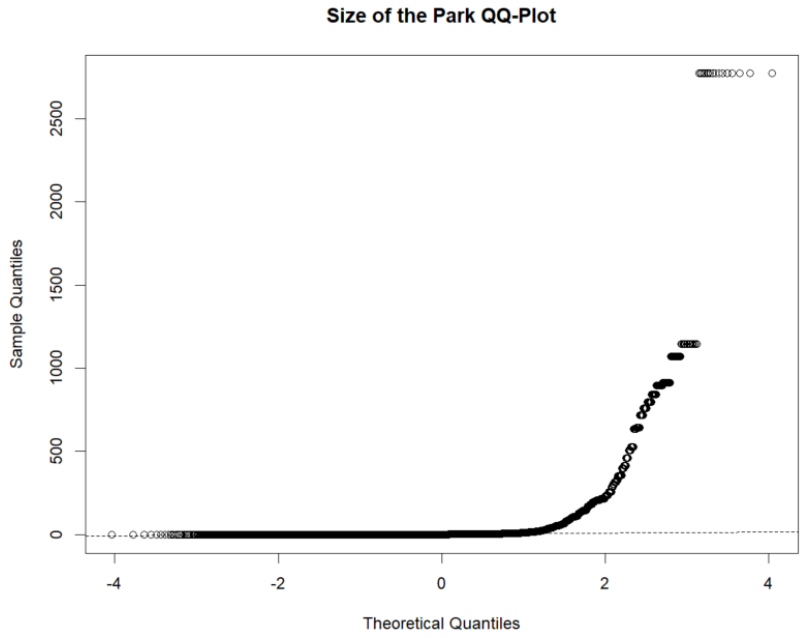


Figure 11 - QQ Plot of Size

The QQ plot of Size shows that a large part of the data seems to fit the line for normal distribution, except for some outliers that lie above the right end. The Shapiro-Wilk test only works with a sample from 3 to 5000, thus it cannot be applied for this data set with more than 18,000 observations. The Anderson-Darling normality test was used instead, resulting in a p-value equal to 3.7e-24, which was much less than an alpha of 0.05. This implied that that the test was highly significant. Hence, Size was likely to follow a non-normal distribution.

b. Constructing new attributes

Besides the two new attributes *Year* and *Quarter* which are created in the data consolidation steps, we also derive two other new variables for analysis. The first derived variable is *Size_thou_acre* which measures the size of the park in thousand acres. The purpose of conversion is to facilitate a better visualization between the Total number of Crimes and the Park size. The second derived variable is Crimes_per_Size which measures the total number of crimes per thousand acres. Since the size of the 1154 parks in the data set span with a wide range, therefore, calculating the number of crime per thousand acres will better reflect the true relationship between the Size and Total variable and also allow for more accurate comparison across all parks with different sizes.

```
#Data transformation
# Derive a new variable name Size_thou_acre with unit= thousand acres
NYC_Final_Data$Size_thou_acre = NYC_Final_Data$SIZE / 1000

# Derive new variable: Crimes_per_Size which calculate
# the total number of crimes per thousand acres
NYC_Final_Data$Crimes_per_Size = NYC_Final_Data$TOTAL / NYC_Final_Data$Size_thou_acre
```

c. Transforming text into categorical variables

The current structure of the data set suggests that the data type of Year and Quarter should be changed to Factor (categorical variables). The results after conversion are as following:

```
> #Transforming text in to categories
+ str(NYC_Final_Data$Year)
+
 num [1:18464] 2015 2015 2015 2015 2015 ...
> str(NYC_Final_Data$Quarter)
 chr [1:18464] "Q1" "Q1" "Q1" "Q1" "Q1" "Q1" "Q1" "Q1" "Q1" "Q1" .
> NYC_Final_Data$Year = as.factor(NYC_Final_Data$Year)
> NYC_Final_Data$Quarter = as.factor(NYC_Final_Data$Quarter)
> str(NYC_Final_Data$Year)
 Factor w/ 4 levels "2015","2016",..: 1 1 1 1 1 1 1 1 1 1 ...
> str(NYC_Final_Data$Quarter)
 Factor w/ 4 levels "Q1","Q2","Q3",..: 1 1 1 1 1 1 1 1 1 1 ...
```

The structure of all the variables in the data set are provided as below:

```
> str(NYC_Final_Data)
'data.frame':   18464 obs. of  14 variables:
 $ PARK                : Factor w/ 1155 levels "","\"\"\"UN
 $ BOROUGH             : Factor w/ 7 levels "","BRONX
 $ SIZE                : num  2772 1146 1073 913 898 ...
 $ CATEGORY            : Factor w/ 11 levels "","BASKETBALL
 $ MURDER              : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RAPE                : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ROBBERY             : int  0 0 1 0 0 0 0 0 0 0 ...
 $ FELONY_ASSAULT      : int  0 0 0 0 1 0 0 0 0 0 ...
 $ BURGLARY            : int  0 1 0 0 1 0 0 0 0 0 ...
 $ GRAND_LARCENY       : int  0 0 0 0 1 0 0 0 0 0 ...
 $ GRAND_LARCENY__MOTOR: int  0 0 0 0 1 0 0 0 0 0 ...
 $ TOTAL               : num  0 1 1 0 4 0 0 0 0 0 ...
 $ Year                : Factor w/ 4 levels "2015","2016",.
 $ Quarter             : Factor w/ 4 levels "Q1","Q2","Q3",
```

6. Data Reduction

In this step, Principal Components Analysis (PCA) and Factor Analysis (FA) are performed to determine if we need to reduce the size or scope of the data set. For either PCA or FA, only the numerical variables are employed for the analysis.

a. Principal Components Analysis (PCA)

```
> # Data Reduction: PCA
+ NYC_data_pca = NYC_Final_Data[c("SIZE", "MURDER", "RAPE","ROBBERY",
+ "FELONY_ASSAULT", "BURGLARY", "GRAND_LARCENY", "GRAND_LARCENY__MOTOR")]
+
> NYC_pcamodel = princomp(NYC_data_pca, cor = TRUE)
> NYC_pcamodel$sdev ^ 2
   Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6
1.9597369 1.0765706 1.0042802 0.9402556 0.8870563 0.7885427
   Comp.7    Comp.8
0.6981093 0.6454484
```

The way the PCA is assessed is that any value greater than 1 is retained, while anything less than 1 is thrown out. Based on the eigenvalues from PCA, we can see that the first three components have values greater than 1. The last five components' values are smaller than 1. This implies that in reality we may be dealing with 3, not 8 variables.

b. Scree plot

A scree plot is also constructed to confirm the findings from PCA as following:

```
> #Create a scree plot
+ plot(NYC_pcamodel, main = "Scree plot of crimes")
```

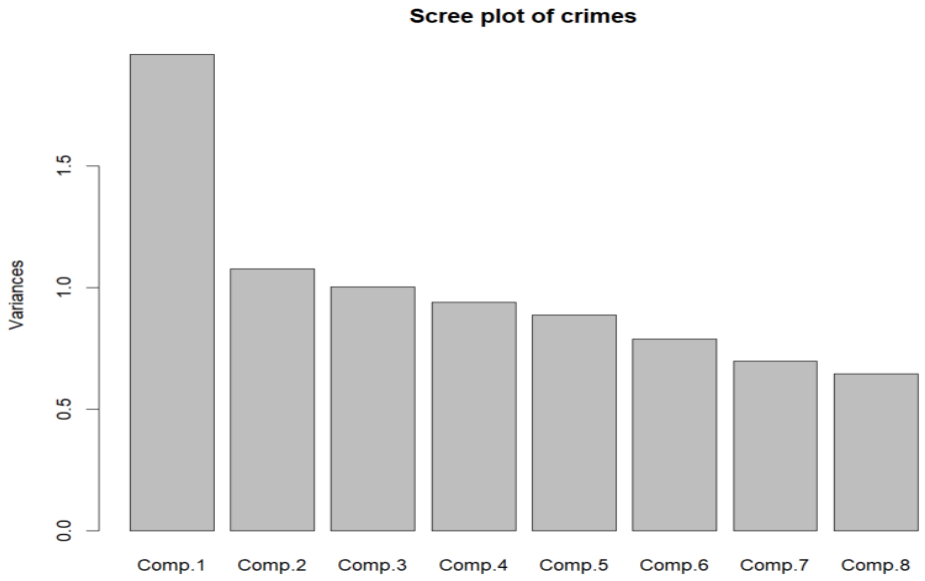


Figure 12- Scree plot of Total number of Crimes

Scree plots are more subjective than assessing eigenvalues. The purpose of using the scree plot is to determine how many components or columns of data we are truly dealing with. The way to read a scree plot is determining where the plot levels off and becomes flat; anything prior to that leveling off is a component that remains.

The scree plot of crimes shows that it never levels off. The variance sharply reduces from component 1 to component 2; then starting from component 2, it does depict a small difference between the following components (but does not really become flat at any component). This implies that based on the scree plot, it is not necessary to combine the components.

While we understand that scree plots are often more subjective than assessing eigenvalues, the results obtained from PCA also indicates a not much impressive result. Only the first component in PCA analysis has a much greater-than-one value, the next two components do not show much difference as compared to the five remaining components. Since the results from both PCA and scree plot are not inherently differentiable, hence, we decide that all variables should be kept and assessed individually for further analysis.

At this point, either PCA or scree plot cannot yield more information. To determine which variables should be kept or removed, a factor analysis must be performed.

c. Factor Analysis (FA)

```
> # Data Reduction: FA
+ NYC_data_FA = factanal(~SIZE + MURDER + RAPE + ROBBERY + FELONY_ASSAULT
+                         + BURGLARY + GRAND_LARCENY + GRAND_LARCENY__MOTOR,
+                factors = 3,
+                rotation = "varimax",
+                scores = "none",
+                data = NYC_Final_Data)
```

```
> NYC_data_FA
+

Call:
factanal(x = ~SIZE + MURDER + RAPE + ROBBERY + FELONY_ASSAULT +     BURGL

Uniquenesses:
                SIZE               MURDER                 RAPE
               0.884                0.996                0.928
            ROBBERY       FELONY_ASSAULT             BURGLARY
               0.720                0.403                0.861
      GRAND_LARCENY GRAND_LARCENY__MOTOR
               0.688                0.005

Loadings:
                   Factor1 Factor2 Factor3
SIZE                0.108   0.308
MURDER
RAPE                        0.255
ROBBERY                     0.401   0.343
FELONY_ASSAULT              0.194   0.744
BURGLARY            0.157   0.338
GRAND_LARCENY               0.502   0.243
GRAND_LARCENY__MOTOR 0.985  0.159

               Factor1 Factor2 Factor3
SS loadings      1.016   0.750   0.749
Proportion Var   0.127   0.094   0.094
Cumulative Var   0.127   0.221   0.314

Test of the hypothesis that 3 factors are sufficient.
The chi square statistic is 94.24 on 7 degrees of freedom.
The p-value is 1.66e-17
>
```

The Factor Loadings depict the relationships between all factors and components. Each of the 8 components has an associated score with each factor. A positive value represents a positive relationship. The closer a value is to 0, the less of a relationship that component has with that factor. It can be seen that not all variables load cleanly to the three factors. For Factor1, only the value of Grand_Larceny_Motor is high, whereas, the values for *Size* and *Burglary are much* smaller. This implies that Factor1 is mainly comprised of Grand_Larceny_Motor. Factor2 is a combination of seven variables, however, no score is greater than the threshold of 0.7. For Factor3, only the score of Felony_Assault is greater than 0.7, whereas the values for the other two factors are not significant.

The above Factor Analysis suggests that out of a total of 8 components in the data set, we are truly dealing with 2. However, since the number of numeric variables are only 8, and the difference between the factor loadings/ components are not much, and given the results obtained from PCA and scree plot in previous analysis, we therefore decide to keep all 8 variables in the data set for the purpose of both descriptive and predictive analyses.

7. Descriptive Statistics

a. Summary statistics for each variable

We used the *describe ()* function to view the statistical measures of all of the variables in the dataset. The results are summarized in the table below.

```
> describe(NYC_Final_Data)
                    vars     n    mean     sd median trimmed    mad min     max   range  skew kurtosis   se
PARK*                  1 18464 578.50 333.14 578.50  578.50 427.73   2 1155.00 1153.00  0.00    -1.20 2.45
BOROUGH*               2 18464   4.33   1.75   5.00    4.28   2.97   2    7.00    5.00  0.04    -1.51 0.01
SIZE                   3 18464  25.00 123.28   1.59    3.58   1.54   0 2771.75 2771.75 12.77   232.64 0.91
CATEGORY*              4 18464   5.67   1.73   6.00    5.82   0.00   2   11.00    9.00 -1.00     0.92 0.01
MURDER                 5 18464   0.00   0.04   0.00    0.00   0.00   0    1.00    1.00 25.17   631.62 0.00
RAPE                   6 18464   0.01   0.08   0.00    0.00   0.00   0    3.00    3.00 18.13   419.52 0.00
ROBBERY                7 18464   0.08   0.42   0.00    0.00   0.00   0    9.00    9.00  8.17    96.28 0.00
FELONY_ASSAULT         8 18464   0.05   0.30   0.00    0.00   0.00   0    7.00    7.00  9.39   119.18 0.00
BURGLARY               9 18464   0.00   0.08   0.00    0.00   0.00   0    5.00    5.00 25.34  1011.54 0.00
GRAND_LARCENY         10 18464   0.11   0.96   0.00    0.00   0.00   0   69.00   69.00 36.27  2055.42 0.01
GRAND_LARCENY__MOTOR  11 18464   0.00   0.04   0.00    0.00   0.00   0    2.00    2.00 33.81  1282.49 0.00
TOTAL                 12 18464   0.25   1.33   0.00    0.02   0.00   0   71.00   71.00 20.40   738.15 0.01
Year*                 13 18464   2.50   1.12   2.50    2.50   1.48   1    4.00    3.00  0.00    -1.36 0.01
Quarter*              14 18464   2.50   1.12   2.50    2.50   1.48   1    4.00    3.00  0.00    -1.36 0.01
```

('' Indicates categorical variables, values in the table are not indicative of distribution)*

The above output from the *describe()* function can be summarized by the following table:

	MEAN	STD. DEV	MEDIAN	MIN	MAX	RANGE	SKEW	KURTOSIS
PARK*	578.5	333.14	578.5	2	1155	1153	0	-1.2
BOROUGH*	4.33	1.75	5	2	7	5	0.04	-1.51
SIZE	25	123.28	1.59	0	2771.75	2771.75	12.77	232.64
CATEGORY*	5.67	1.73	6	2	11	9	-1	0.92
MURDER	0	0.04	0	0	1	1	25.17	631.62
RAPE	0.01	0.08	0	0	3	3	18.13	419.52
ROBBERY	0.08	0.42	0	0	9	9	8.17	96.28
FELONY_ASSAULT	0.05	0.3	0	0	7	7	9.39	119.18
BURGLARY	0	0.08	0	0	5	5	25.34	1011.54
GRAND_LARCENY	0.11	0.96	0	0	69	69	36.27	2055.42
GRAND_LARCENY__MOTOR	0	0.04	0	0	2	2	33.81	1282.49
TOTAL	0.25	1.33	0	0	71	71	20.4	738.15
Year*	2.5	1.12	2.5	1	4	3	0	-1.36
Quarter*	2.5	1.12	2.5	1	4	3	0	-1.36

Table 3 - Summary Statistics of all Variables

The above descriptive statistics indicate that *grand larceny* has the highest mean and standard deviation among all of the crimes. The maximum number of grand larcenies in a particular park has been as high as 69. Overall the distributions have high skewness and kurtosis but these values might have been distorted due to the time-series nature of the data.

b. Frequency distribution of categorical variables

The next part in our descriptive statistics focuses on the distribution of categorical variables. Our two main input categorical variables are BOROUGH and CATEGORY.

```
> ##Summary of categorical variables
+ ### Frequency count of each categorical variable
+ ggplot(NYC_Final_Data, aes(CATEGORY)) +
+     geom_bar(fill = "#FC4E07") +
+     coord_flip() +
+     ggtitle("Frequency of Park Category")
+
+ ggplot(NYC_Final_Data, aes(BOROUGH)) +
+     geom_bar(fill = "#0073C2FF") +
+     ggtitle("Frequency of Park Borough")
```

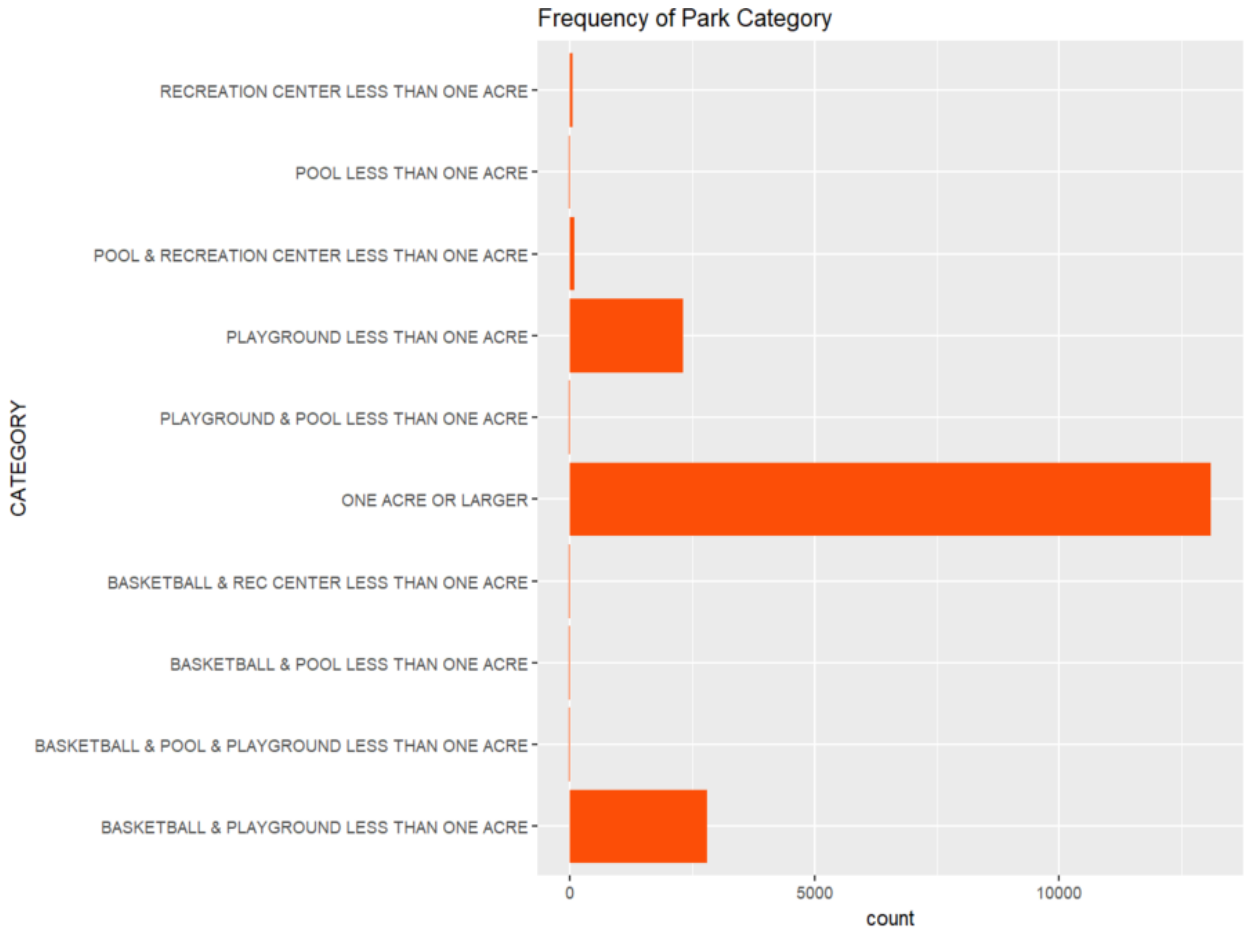


Figure 13 - Frequency of Park Category

As can be seen from the above chart, a majority of the parks in NYC comes under the category of one acre or larger, followed by basketball courts and playgrounds, and then just playgrounds. Regarding BOROUGH, Brooklyn borough leads with the most number of parks followed by Queens, then Bronx, Manhattan and Staten Island. The overall total number of parks in NYC are 1,155. The above count values are high because of the time series nature of the data.

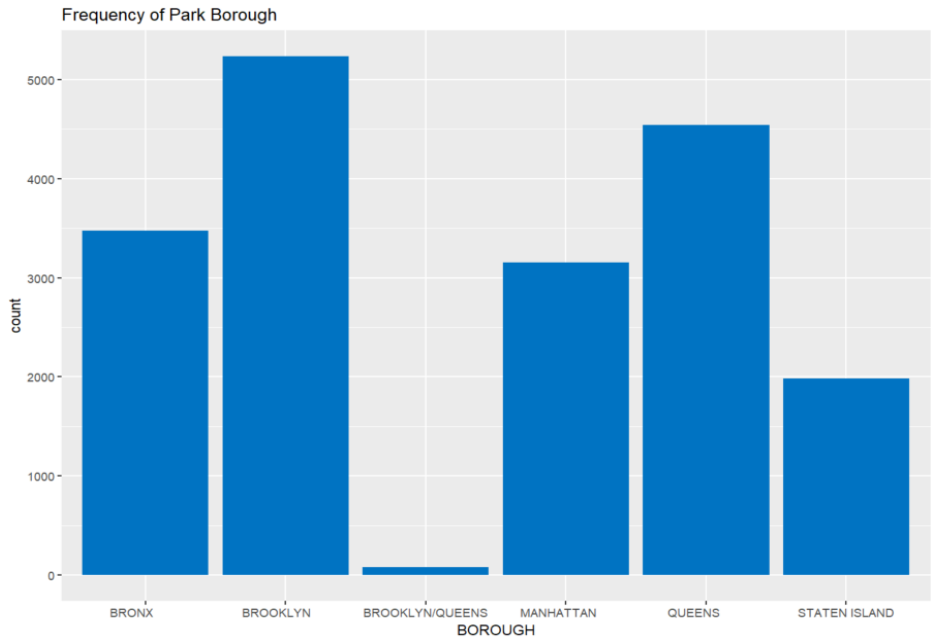


Figure 14 - Frequency of Park Borough

c. Relationship between target variable and categorical variable

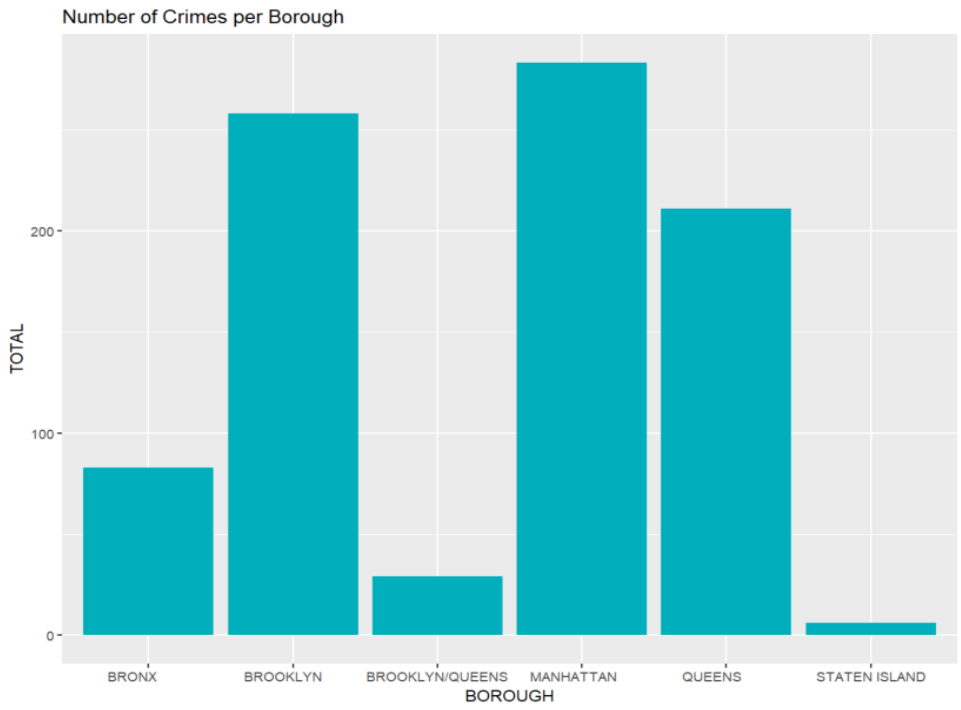


Figure 15 - Number of Crimes per Borough

The above plot shows the distribution of total number of crimes across different units of Borough. It can be seen that Manhattan leads with highest number of total crimes in parks followed by Brooklyn and Queens. Although Manhattan has lower number of parks, it accounts for a higher number of crimes.

d. Relationship between target variable and continuous predictor variable

The only continuous input variable is the size of the park. The relationship between TOTAL number of crimes and park size is analyzed using scatter plot as below.

```
> plot(NYC_Final_Data$SIZE, NYC_Final_Data$TOTAL, col = 4)
+ abline(lm(NYC_Final_Data$TOTAL ~ NYC_Final_Data$SIZE), col = 2) # regression line (y~x)
+
+ sub_NYC <- subset(NYC_Final_Data, NYC_Final_Data$SIZE < 2000)
+
+ plot(sub_NYC$SIZE, sub_NYC$TOTAL, col = 1)
+ abline(lm(sub_NYC$TOTAL ~ sub_NYC$SIZE), col = 2) # regression line (y~x)
```

The initial scatter plot reveals six influential data points that might distort the regression line.

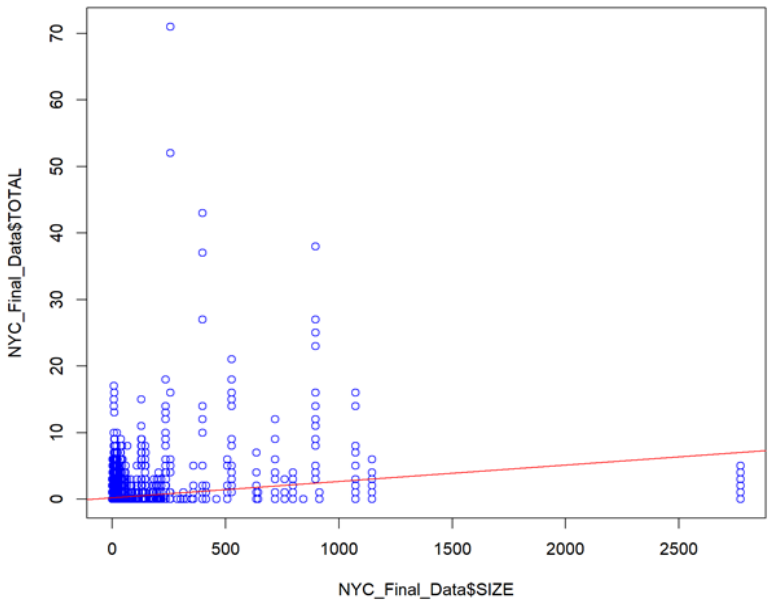


Figure 16 - Scatter plot of TOTAL and SIZE

As can be seen in the plot below, removing influential observations from the plot does not really affect the regression line. The plot also indicates that the relationship between size and total number of crimes is weak (slope is close to zero).

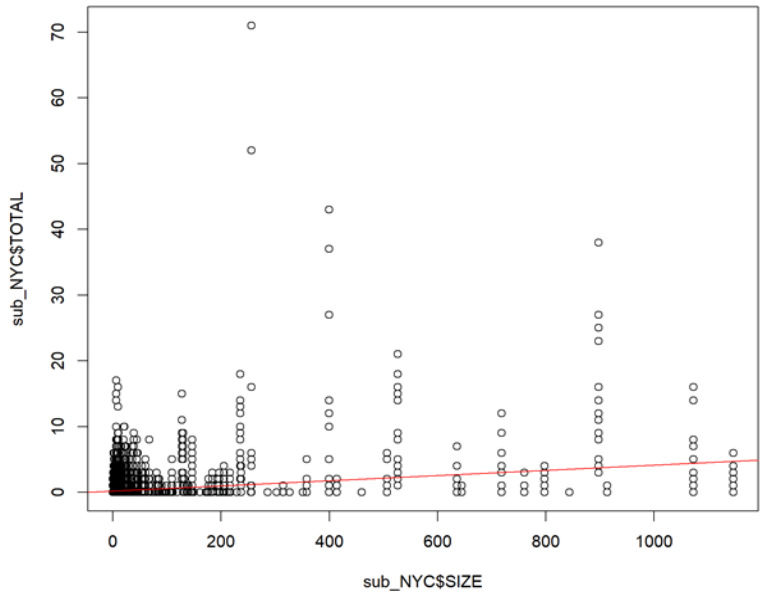


Figure 17 - Scatterplot of TOTAL and SIZE (after removing influentials)

The Pearson correlation analysis also reveals the same thing, as can be seen below:

```
> cor(NYC_Final_Data[,c(3,12)])
             SIZE      TOTAL
SIZE  1.0000000 0.2283219
TOTAL 0.2283219 1.0000000
```

Pearson correlation coefficient is 0.228 which again indicates a week positive relationship between total and size.

e. *Time series analysis*

```
> ## Trend line for Murder
+ Sum_Murder <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(MURDER_TOTAL = sum(MURDER))
+
+ Sum_Murder$Year_Quarter = paste(Sum_Murder$Year, Sum_Murder$Quarter)
+ Sum_Murder$Num = c(1:16)
+ plot(Sum_Murder$Num, Sum_Murder$MURDER_TOTAL, type = 'l', xaxt = "n",
+      main = "Trend of Murders across all quarters in 2015-2018")
+ axis(1, at = Sum_Murder$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                         "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                         "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                         "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

As can be observed from the below plot, the number of *murders* in NYC parks reveals an interesting trend of peak period between Q2-Q4 in a given year. Q1 has had consitently less (close to zero) reported murders in NYC park areas. The years 2015 and 2017 have comparatively lower number of murders reported than those in 2016 and 2018.

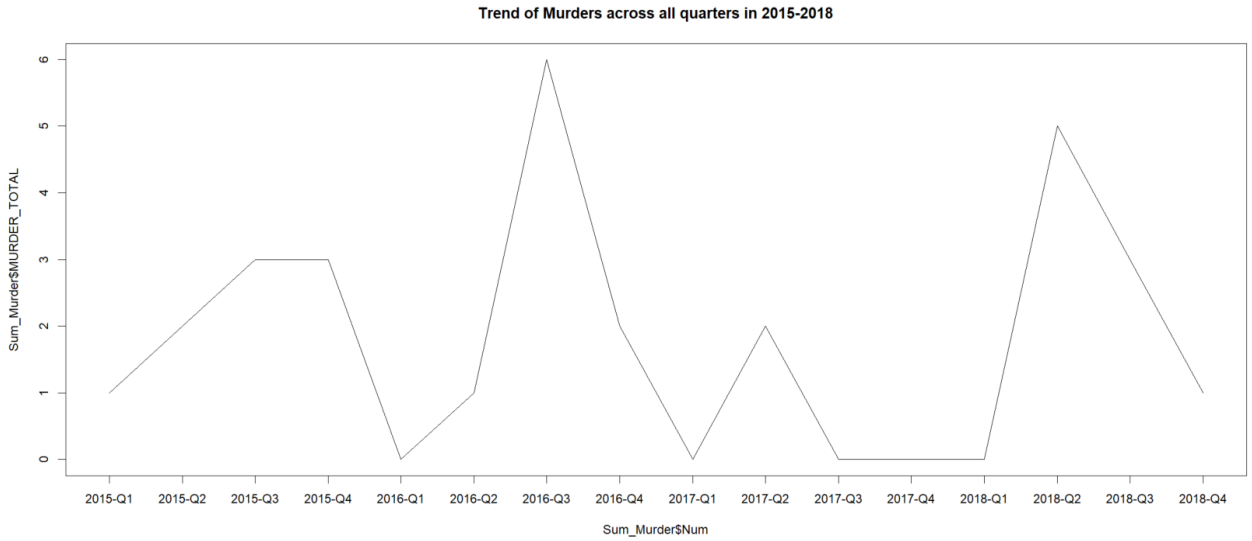


Figure 18 - Trend of MURDER across all quarters in 2015-2018

```
> ## Trend line for Rape
+ Sum_RAPE <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(RAPE_TOTAL = sum(RAPE))
+
+ Sum_RAPE$Year_Quarter = paste(Sum_RAPE$Year, Sum_RAPE$Quarter)
+ Sum_RAPE$Num = c(1:16)
+ plot(Sum_RAPE$Num, Sum_RAPE$RAPE_TOTAL, type = 'l',
+      main = "Trend of Rapes across all quarters in 2015-2018")
+ axis(1, at = Sum_RAPE$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                       "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                       "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                       "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

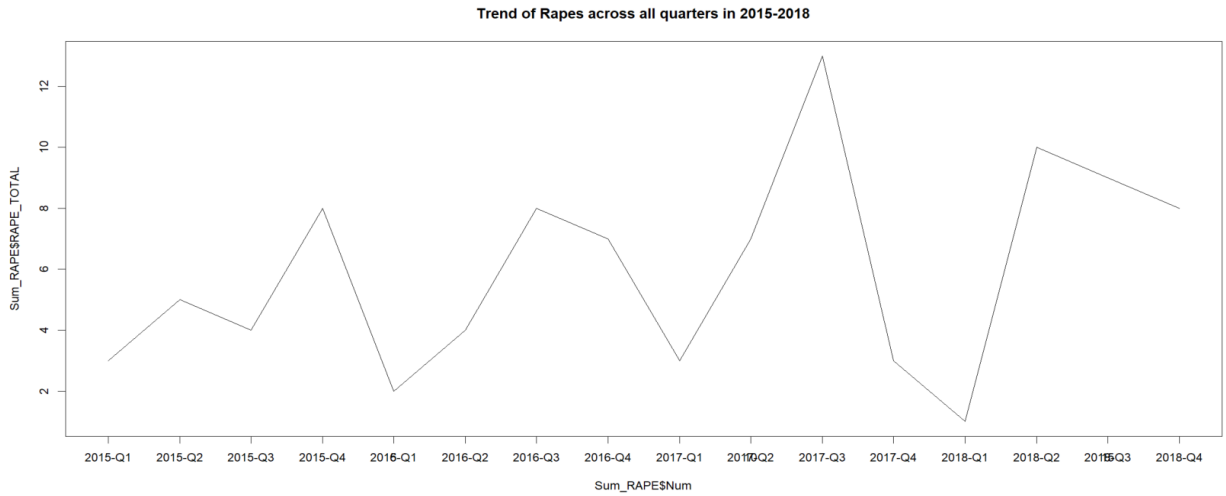


Figure 19 - Trend of RAPE across all quarters in 2015-2018

The number of *rapes* occuring in parks in NYC shows a similar trend of peak period between Q2-Q4 in a given year. Q1 has had consistently comparatively lesser reported rapes. Overall throughout the years there seems to be an increasing trend which needs to be further looked into.

```
> ## Trend line for Robbery
+ Sum_ROBBERY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(ROBBERY_TOTAL = sum(ROBBERY))
+
+ Sum_ROBBERY$Year_Quarter = paste(Sum_ROBBERY$Year, Sum_ROBBERY$Quarter)
+ Sum_ROBBERY$Num = c(1:16)
+ plot(Sum_ROBBERY$Num, Sum_ROBBERY$ROBBERY_TOTAL, type = 'l',
+      main = "Trend of Robberies across all quarters in 2015-2018")
+ axis(1, at = Sum_ROBBERY$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                          "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                          "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                          "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

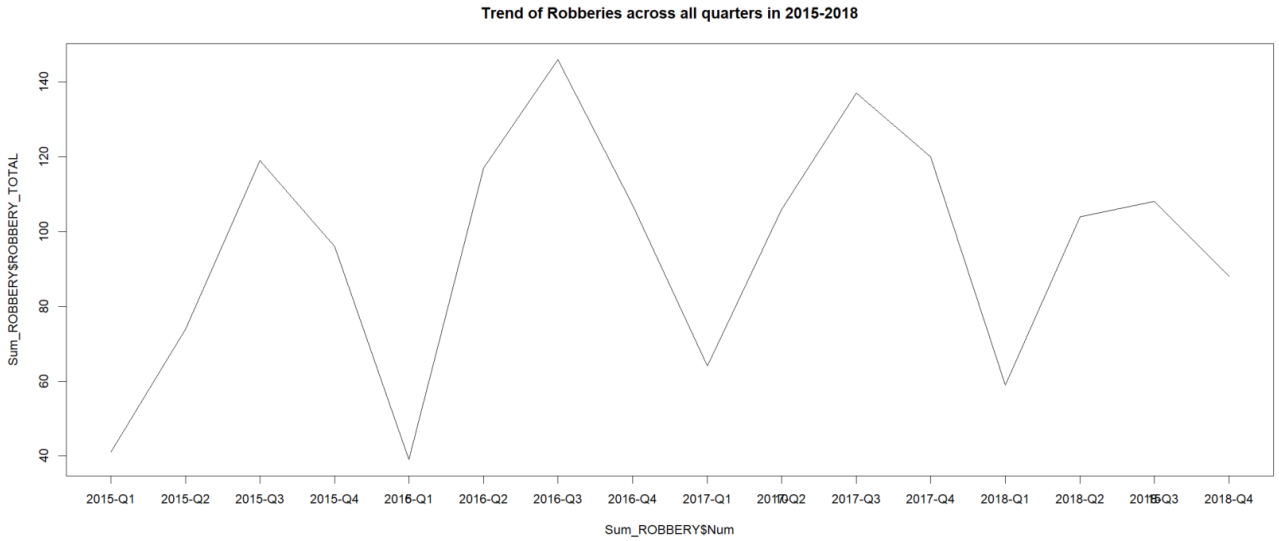


Figure 20 - Trend of ROBBERY across all quarters in 2015-2018

The number of *robberies* in NYC parks also depicts the same trend of peak period between Q2-Q4 in a given year with Q1 having relatively lower (between 40 to 60) robberies across the years. It is at a peak during Q3 for all the years. The numbers seem to go down in 2018 which is a favourable trend.

```
> ## Trend line for Felony Assault
+ Sum_FELONY_ASSAULT <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(FELONY_ASSAULT_TOTAL = sum(FELONY_ASSAULT))
+
+ Sum_FELONY_ASSAULT$Year_Quarter = paste(Sum_FELONY_ASSAULT$Year, Sum_FELONY_ASSAULT$Quarter)
+ Sum_FELONY_ASSAULT$Num = c(1:16)
+ plot(Sum_FELONY_ASSAULT$Num, Sum_FELONY_ASSAULT$FELONY_ASSAULT_TOTAL, type = 'l',
+      main = "Trend of Felony Assaults across all quarters in 2015-2018")
+ axis(1, at = Sum_FELONY_ASSAULT$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                          "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                          "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                          "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

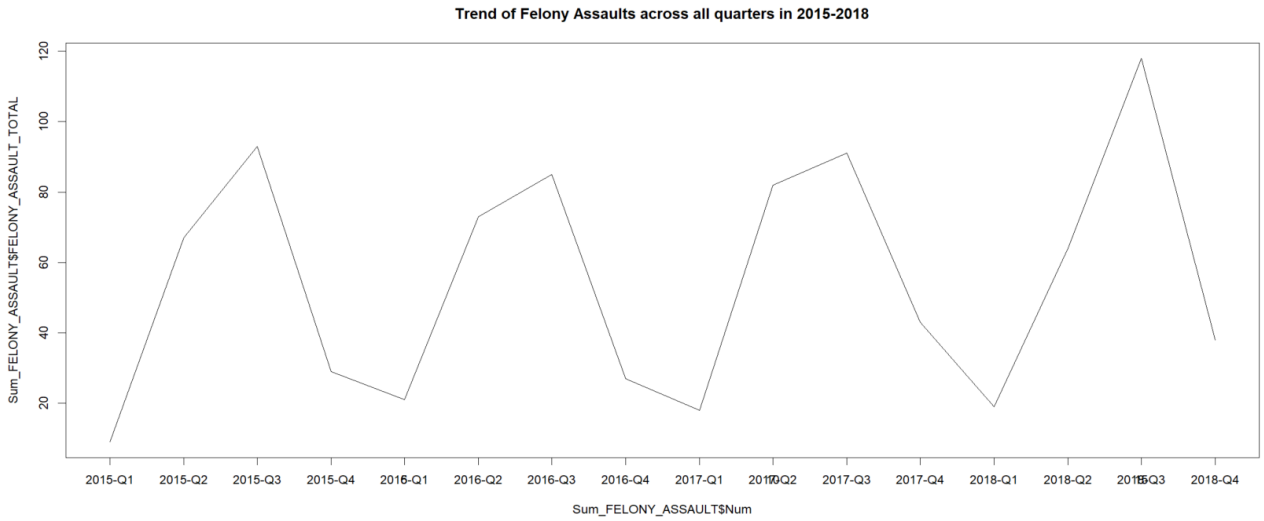


Figure 21 - Trend of FELONY ASSAULT across all quarters in 2015-2018

The number of *felony assaults* in NYC parks show a trend of peak period between Q2-Q3 in a given year . There is an overall increase in number of felony assaults and this needs to be looked at further.

```
> ## Trend line for Burglary
+ Sum_BURGLARY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(BURGLARY_TOTAL = sum(BURGLARY))
+
+ Sum_BURGLARY$Year_Quarter = paste(Sum_BURGLARY$Year, Sum_BURGLARY$Quarter)
+ Sum_BURGLARY$Num = c(1:16)
+ plot(Sum_BURGLARY$Num, Sum_BURGLARY$BURGLARY_TOTAL, type = 'l',
+      main = "Trend of Burglaries across all quarters in 2015-2018")
+ axis(1, at = Sum_BURGLARY$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                           "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                           "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                           "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

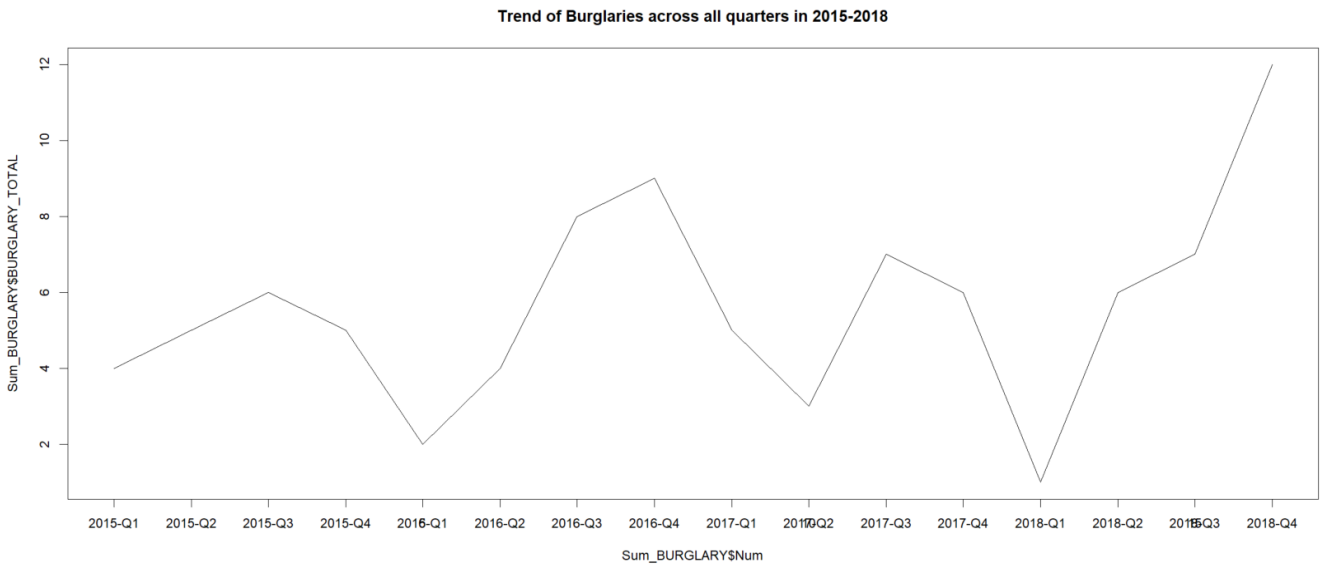


Figure 22 - Trend of BURGLARY across all quarters in 2015-2018

The number of *Burglaries* does not display a specific trend after 2016, however, it experiences a steady increase in the year 2018 (Q1-Q4) which may be worrying.

```
> ## Trend line for Grand Larceny
+ Sum_GRAND_LARCENY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(GRAND_LARCENY_TOTAL = sum(GRAND_LARCENY))
+
+ Sum_GRAND_LARCENY$Year_Quarter = paste(Sum_GRAND_LARCENY$Year, Sum_GRAND_LARCENY$Quarter)
+ Sum_GRAND_LARCENY$Num = c(1:16)
+ plot(Sum_GRAND_LARCENY$Num, Sum_GRAND_LARCENY$GRAND_LARCENY_TOTAL, type = 'l',
+      main = "Trend of Grand Larceny across all quarters in 2015-2018")
+ axis(1, at = Sum_GRAND_LARCENY$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                                "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                                "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                                "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

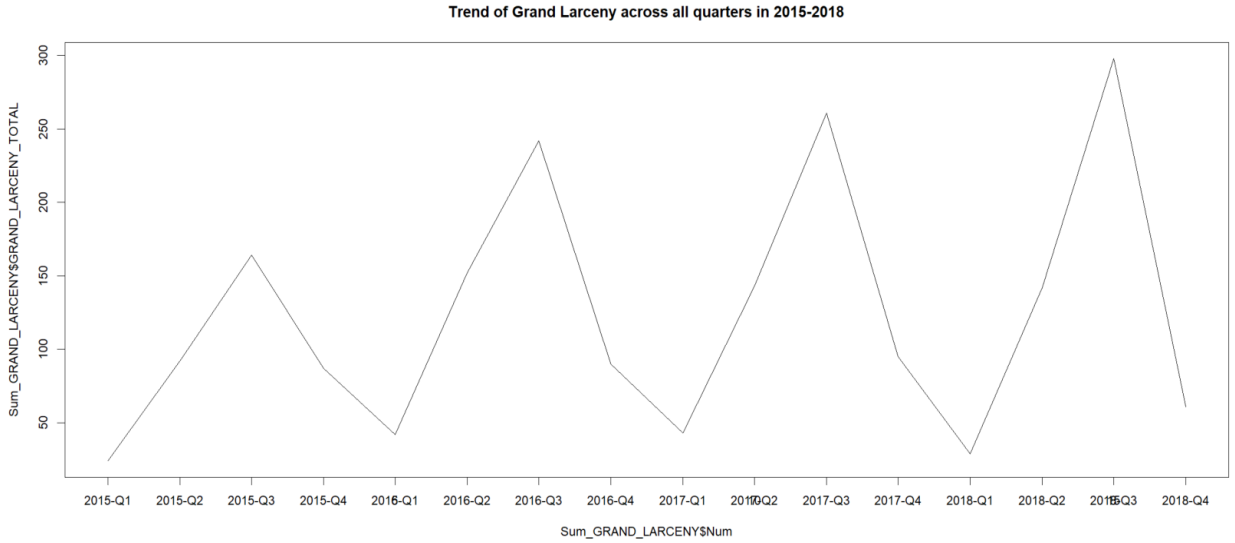


Figure 23 - Trend of GRAND LARCENY across all quarters in 2015-2018

The number of *grand larceny* in NYC parks indicates a trend of peak period between Q2-Q4 in a given year with the highest number in Q3's of each year. The overall trend is an increase in the number of grand larcenies.

```
> ## Trend line for Grand Larceny of Motor Vehicle
+ Sum_GRAND_LARCENY__MOTOR <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(GRAND_LARCENY__MOTOR_TOTAL = sum(GRAND_LARCENY__MOTOR))
+
+ Sum_GRAND_LARCENY__MOTOR$Year_Quarter = paste(Sum_GRAND_LARCENY__MOTOR$Year,
+                                               Sum_GRAND_LARCENY__MOTOR$Quarter)
+ Sum_GRAND_LARCENY__MOTOR$Num = c(1:16)
+ plot(Sum_GRAND_LARCENY__MOTOR$Num, Sum_GRAND_LARCENY__MOTOR$GRAND_LARCENY__MOTOR_TOTAL,
+      type = 'l', main = "Trend of Grand Larceny of Motor Vehicles across all quarters in 2015-2018")
+ axis(1, at = Sum_GRAND_LARCENY__MOTOR$Num, labels = c("2015-Q1", "2015-Q2", "2015-Q3", "2015-Q4",
+                                                       "2016-Q1", "2016-Q2", "2016-Q3", "2016-Q4",
+                                                       "2017-Q1", "2017-Q2", "2017-Q3", "2017-Q4",
+                                                       "2018-Q1", "2018-Q2", "2018-Q3", "2018-Q4"))
```

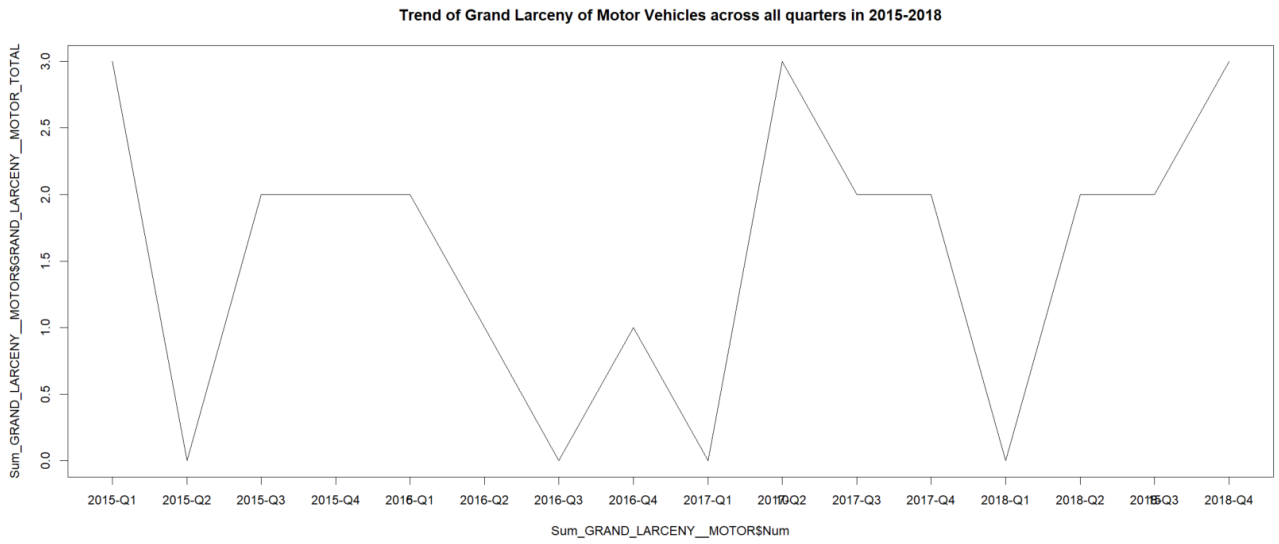


Figure 24 - Trend of GRAND LARCENY MOTOR across all quarters in 2015-2018

There is no particular trend in the number of Motor Grand Larcenies till 2016. In 2017 and 2018, the trend of increasing numbers between Q2-Q4 is observed which needs to be investigated further.

Selecting Modeling Technique

1. Justify the choice of models

Once the descriptive analysis is completed, it's time to decide which kind of models we should go forward with. Based on our previous analysis, it can be seen that our target variable is a continuous variable - the number of crimes. With a continuous target variable, multiple linear regression model was our first choice. Also, from the above analysis, we can see that the variables are normally distributed with no sign of collinearity.

Another modeling technique which was chosen was time series analysis. Since we have time-ordered data, it made sense to predict the trend and to forecast the number of crimes till quarter 4 of the year 2020.

First the assumptions of the models will be checked and then both models will be run separately. Based on different statistics, comparison will be done for both models in order to select the optimal model for the project.

2. Explain the assumption of each modeling technique

a. Linear Regression Model

Our dataset has continuous target variable which is the number of crimes in each park, thus we performed linear regression to check if our continuous and categorical inputs can be helpful to predict our target. The assumptions for this model are:

- The inputs such as the size, park, borough, and different crimes are linearly correlated with the target variable - Total number of crimes.
- All the different crimes are not correlated with each other.
- Residuals have zero mean, equal variance, normally distributed and have no autocorelation among them.

The above linear regression model assumptions are checked for our dataset and the results are shown below.

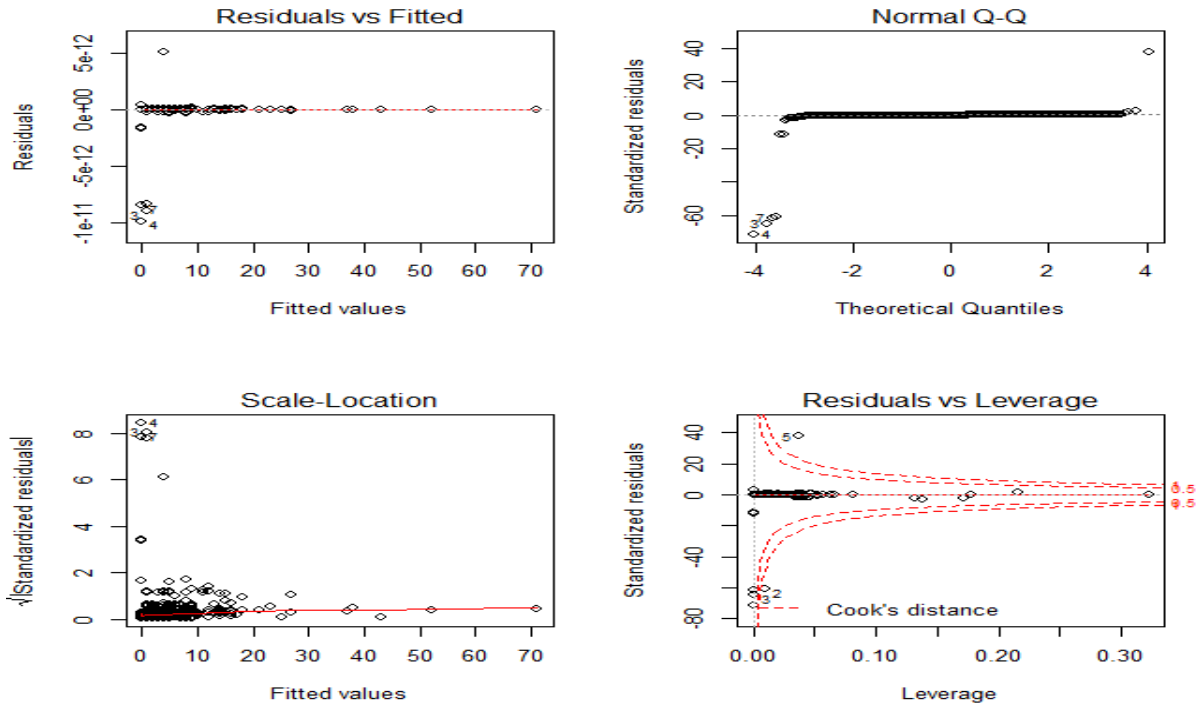


Figure 25 - Residual Plot & QQ Plot of Linear Regression

Residuals vs Fitted: Used to check the linear relationship assumptions. A horizontal line, without distinct patterns is an indication of a linear relationship, which is good.

Normal Q-Q: Used to examine whether the residuals are normally distributed. It's good if residuals points follow the straight dashed line.

Scale-Location (or Spread-Location): Used to check the homogeneity of variance of the residuals (homoscedasticity). Horizontal line with equally spread points is a good indication of homoscedasticity.

Residuals vs Leverage: Used to identify influential cases, which are extreme values that might influence the regression results when included or excluded from the analysis. There are three most extreme data points labeled with the row numbers of the data in the data set. They might be potentially problematic.

Also, upon checking the Multicollinearity with VIF function, the values are less than 4. So, the information in the independent variables is not explained by other independent variables present in the given model, which means, there is likely no redundancy.

```
> VIF(lm(reg_data$MURDER~reg_data$RAPE+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.002841
> VIF(lm(reg_data$MURDER~reg_data$RAPE+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.002841
> VIF(lm(reg_data$RAPE~reg_data$MURDER+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.03362
> VIF(lm(reg_data$ROBBERY~reg_data$MURDER+reg_data$RAPE+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.19464
> VIF(lm(reg_data$FELONY_ASSAULT~reg_data$MURDER+reg_data$RAPE+reg_data$ROBBERY+reg_data$BURGLARY+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.188867
> VIF(lm(reg_data$BURGLARY~reg_data$MURDER+reg_data$RAPE+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$GRAND_LARCENY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.086644
> VIF(lm(reg_data$GRAND_LARCENY~reg_data$MURDER+reg_data$RAPE+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY__MOTOR))
[1] 1.179261
> VIF(lm(reg_data$GRAND_LARCENY__MOTOR~reg_data$MURDER+reg_data$RAPE+reg_data$ROBBERY+reg_data$FELONY_ASSAULT+reg_data$BURGLARY+reg_data$GRAND_LARCENY))
[1] 1.062616
```

b. Time Series Model

The assumptions for Time Series Models (Holt-Winters model and ARMA model) are explained and tested for each single model and for each target variable in the model building part of this report.

Data Splitting and Sub-Sampling (for Linear Regression Model)

We have around 18,464 records of data. Given that our dataset has sufficient records, we decided to do 80/20 split, since for sufficiently large datasets, 70/30 or 80/20 split does not seem to make any difference. After splitting the dataset into 80% training and 20% validation, the structures of resulting training and validation dataset are shown as below:

```
> str(dfTraining)
'data.frame':   14771 obs. of  25 variables:
 $ X                                          : int  1 4 5 6 8 9 10 11 12 15 ...
 $ PARK                                       : Factor w/ 1154 levels "\"\"\"UNCLE"
 $ BOROUGH                                    : Factor w/ 6 levels "BRONX        ",
 $ SIZE                                       : num  2772 913 898 844 760 ...
 $ CATEGORY                                   : Factor w/ 10 levels "BASKETBALL & 1
 $ MURDER                                     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RAPE                                       : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ROBBERY                                    : int  0 0 0 0 0 0 0 0 1 0 ...
 $ FELONY_ASSAULT                             : int  0 0 1 0 0 0 0 0 0 0 ...
 $ BURGLARY                                   : int  0 0 1 0 0 0 0 0 0 0 ...
 $ GRAND_LARCENY                              : int  0 0 1 0 0 0 0 0 1 0 ...
 $ GRAND_LARCENY__MOTOR                       : int  0 0 1 0 0 0 0 0 0 0 ...
 $ TOTAL                                      : int  0 0 4 0 0 0 0 0 2 0 ...
 $ Year                                       : int  2015 2015 2015 2015 2015 2015
 $ Quarter                                    : Factor w/ 4 levels "Q1","Q2","Q3",

> str(dfValidation)
'data.frame':   3692 obs. of  25 variables:
 $ X                                          : int  2 3 7 13 14 19 24 33 34 35 ..
 $ PARK                                       : Factor w/ 1154 levels "\"\"\"UNCLE"
 $ BOROUGH                                    : Factor w/ 6 levels "BRONX        "
 $ SIZE                                       : num  1146 1073 798 507 460 ...
 $ CATEGORY                                   : Factor w/ 10 levels "BASKETBALL & 1
 $ MURDER                                     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RAPE                                       : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ROBBERY                                    : int  0 1 0 1 0 0 0 0 0 0 ...
 $ FELONY_ASSAULT                             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ BURGLARY                                   : int  1 0 0 0 0 0 0 0 0 0 ...
 $ GRAND_LARCENY                              : int  0 0 0 1 0 0 0 0 0 0 ...
 $ GRAND_LARCENY__MOTOR                       : int  0 0 0 0 0 0 0 0 0 0 ...
 $ TOTAL                                      : int  1 1 0 2 0 0 0 0 0 0 ...
 $ Year                                       : int  2015 2015 2015 2015 2015 2015
 $ Quarter                                    : Factor w/ 4 levels "Q1","Q2","Q3",
```

```
> mean(data_train$TOTAL)
[1] 0.2461005
> sd(data_train$TOTAL)
[1] 1.236154
> mean(data_validation$TOTAL)
[1] 0.2595321
> sd(data_validation$TOTAL)
[1] 1.579876
```

There is not much difference between the mean of TOTAL in Training dataset vs that in the Validation data set (0.25 vs 0.26). The standard deviation of TOTAL in Validation data set is higher than that in Training data (1.58 vs 1.24).

Overall, both the training and validation datasets have roughly similar means and standard deviation, which implies that this split is fair for prediction.

Models Building and Assessment

1. Regression Model

1.1. Build the Model

```
Residuals:
       Min         1Q     Median         3Q        Max
-9.813e-12 -9.000e-16  1.600e-15  4.300e-15  5.069e-12

Coefficients: (1 not defined because of singularities)
                                                                 Estimate Std. Error    t value Pr(>|t|)
(Intercept)                                                    -4.364e-12  1.822e-12 -2.395e+00   0.0166 *
reg_data$MURDER                                                 1.000e+00  2.555e-14  3.914e+13   <2e-16 ***
reg_data$RAPE                                                   1.000e+00  1.316e-14  7.600e+13   <2e-16 ***
reg_data$ROBBERY                                                1.000e+00  2.667e-15  3.749e+14   <2e-16 ***
reg_data$FELONY_ASSAULT                                         1.000e+00  3.647e-15  2.742e+14   <2e-16 ***
reg_data$BURGLARY                                               1.000e+00  1.287e-14  7.768e+13   <2e-16 ***
reg_data$GRAND_LARCENY                                          1.000e+00  1.147e-15  8.720e+14   <2e-16 ***
reg_data$GRAND_LARCENY__MOTOR                                   1.000e+00  2.543e-14  3.933e+13   <2e-16 ***
reg_data$Year                                                   2.150e-15  9.035e-16  2.379e+00   0.0174 *
reg_data$QuarterQ2                                              7.189e-15  2.862e-15  2.511e+00   0.0120 *
reg_data$QuarterQ3                                              7.270e-15  2.871e-15  2.532e+00   0.0113 *
reg_data$QuarterQ4                                              7.345e-15  2.860e-15  2.569e+00   0.0102 *
reg_data$CATEGORY_BASKETBALL...PLAYGROUND.LESS.THAN.ONE.ACRE    1.270e-16  1.735e-14  7.000e-03   0.9942
reg_data$CATEGORY_BASKETBALL...POOL...PLAYGROUND.LESS.THAN.ONE.ACRE 3.204e-17  3.836e-14  1.000e-03   0.9993
reg_data$CATEGORY_BASKETBALL...POOL.LESS.THAN.ONE.ACRE          2.712e-16  3.836e-14  7.000e-03   0.9944
reg_data$CATEGORY_BASKETBALL...REC.CENTER.LESS.THAN.ONE.ACRE    3.204e-17  3.836e-14  1.000e-03   0.9993
reg_data$CATEGORY_ONE.ACRE.OR.LARGER                           -2.394e-15  1.720e-14 -1.390e-01   0.8893
reg_data$CATEGORY_PLAYGROUND...POOL.LESS.THAN.ONE.ACRE          3.204e-17  3.836e-14  1.000e-03   0.9993
reg_data$CATEGORY_PLAYGROUND.LESS.THAN.ONE.ACRE                 4.192e-17  1.739e-14  2.000e-03   0.9981
reg_data$CATEGORY_POOL...RECREATION.CENTER.LESS.THAN.ONE.ACRE   6.313e-17  2.215e-14  3.000e-03   0.9977
reg_data$CATEGORY_POOL.LESS.THAN.ONE.ACRE                       3.204e-17  3.836e-14  1.000e-03   0.9993
reg_data$CATEGORY_RECREATION.CENTER.LESS.THAN.ONE.ACRE                 NA         NA         NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The above linear regression model is significant with p-value less than 0.05, implying that the model is better than baseline model. The most important variables in determining the target variable are Murder, Rape, Robbery, Felony Assault, Burglary, Grand Larceny, and Grand Larceny Assault. Other significant variables are Year, Quarter. The other dummy categorical variables are not statistically significant in explaining the target variable.

```
Residual standard error: 1.034 on 18460 degrees of freedom
Multiple R-squared:  0.3961,    Adjusted R-squared:  0.396
F-statistic:  4037 on 3 and 18460 DF,  p-value: < 2.2e-16
```

The adjusted R-square is 0.39 which means 39% of the Target variance is explained by this model.

The beside screenshot shows the model's prediction for the TOTAL variable. These are the first five records in the data.

```
> head(actuals_preds)
  actuals predicteds
1       0 0.07845634
2       1 0.07845634
3       1 1.97271881
4       0 0.07845634
5       4 0.07845634
6       0 0.07845634
```

On calculating the overall accuracy of this model, we found that the model has 62.9% accuracy. This means that the model is able to predict 63% of the records accurately.

```
> correlation_accuracy <- cor(actuals_preds)
> correlation_accuracy
             actuals predicteds
actuals    1.0000000  0.6293938
predicteds 0.6293938  1.0000000
```

1.2. Assess the Model

One advantage of linear regression model is that it is fairly easy to explain and interpret. The assumptions of the linear regression are satisfied, which means the results from the model are deemed to be accurate. Overall the model can explain 36% of the variance in the target variable and predict 63% records accurately.

The AIC & BIC values and the Residual Sum of Squares (RSS), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) of the model are calculated as beside.

```
> AIC(reg)
[1] 47936.08
> BIC(reg)
[1] 47990.84
```

```
> RSS
[1] 14488.84
> MSE
[1] 0.7847078
> RMSE
[1] 0.8858373
```

While the model results can be accurate, the linear regression model also possess potential drawbacks such as:

- It assumes the relationship between the target and dependent variables to be linear only. However, there may be some other relationships between theses variables which are not considered in this model.
- Also, the range of the target variable's predicted values is restricted between 0-71 values because values greater than 25% of the target range are not predicted properly by linear regression model.

```
> min(reg_data$TOTAL)
[1] 0
> max(reg_data$TOTAL)
[1] 71
```

Since for our dataset, the time period of the event does matter and because the linear regression model does not take the YEAR and QUARTER as the most important variables, using this model for prediction is not likely to be optimal for our project goal.

2. Time Series Model

Since our data consists of time series of the crime variables from first quarter of 2015 to the last quarter of 2018, therefore we decide to analyze all of our crime variables separately in different time series models. We also include the variable TOTAL which is the total number of all crimes.

To generate the time series, we first summed up the variables across different parks grouped by the combination of year and quarter (using summarise() and group_by() functions). We then used the ts() function to generate a time series. This was done for each of the seven crime variables as well as the TOTAL variable. When converting to time series object, the dataset was reduced to 16 rows, thus we did not do a training and testing split when running time series models (since splitting the data resulted in bad models).

All of the variables in our data had seasonal components, therefore we ran Holt-Winters model along with ARMA/ARIMA models to identify the best model that would fit our time series. Below is a summary table of the models that we recommend for respective variables.

Variable	Recommended Model	Model Parameters
TOTAL	Holt-Winters	(α=0.37, β=0.37, γ=1)
MURDER	Holt-Winters	(α=0, β=0, γ=0.47)
RAPE	Holt-Winters	(α=0, β=0, γ=0.69)
ROBBERY	ARMA	(AR=1,MA=0)
FELONY_ASSAULT	Holt-Winters	(α=0.06, β=1, γ=0)
BURGLARY	Seasonal Random Walk with AR(1)	(1,0,0)(1,0,0)[4]
GRAND_LARCENY	Holt-Winters	(α=0.27, β=0.34, γ=1)
GRAND_LARCENY__MOTOR	Holt-Winters	(α=0.18, β=0.36, γ=0.38)

Table 4 - Summary of Predictive Model for each Variable

Note: A seasonal random walk model is a special case of an ARIMA model in which there is one order of seasonal differencing, a constant term, and no other parameters. In this case the model has one AR component.

2.1. Target variable = TOTAL

a. Building time series model for TOTAL

```
> Sum_TOTAL <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(TOTAL = sum(TOTAL))
+
+ TOTAL_ts = ts(Sum_TOTAL$TOTAL,frequency = 4,start=2015)
+
+ plot(TOTAL_ts, main = 'Time Series of Total variable')
```

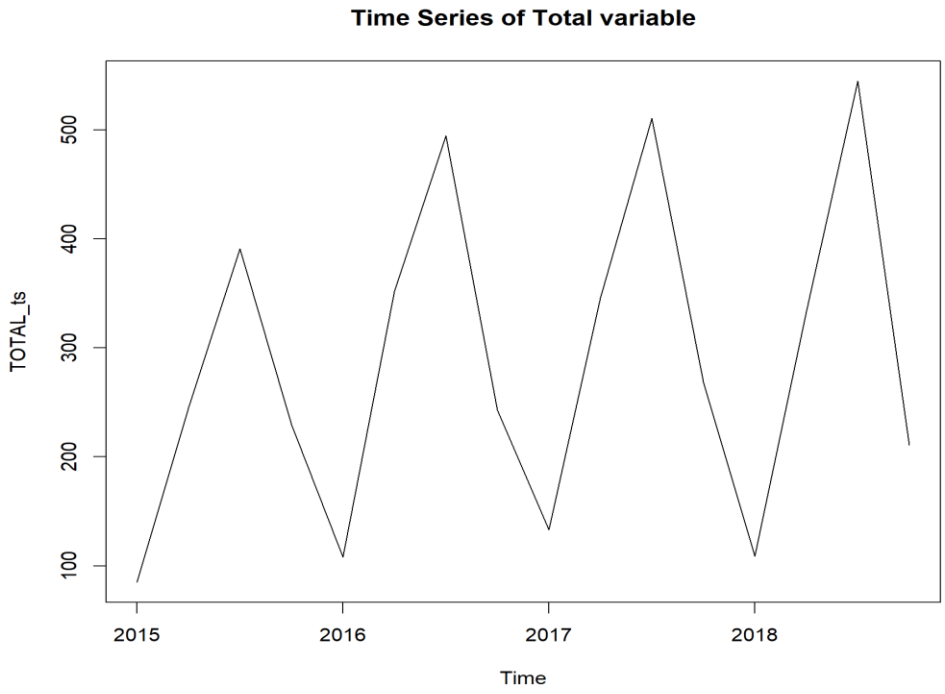


Figure 26 - Time Series of TOTAL

❖ Identifying seasonal patterns

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(TOTAL_ts ~ cycle(TOTAL_ts), main = 'Seasonality of Total variable')
+
+ TOTAL_ts_dc = decompose(TOTAL_ts)
+ plot(TOTAL_ts_dc)
```

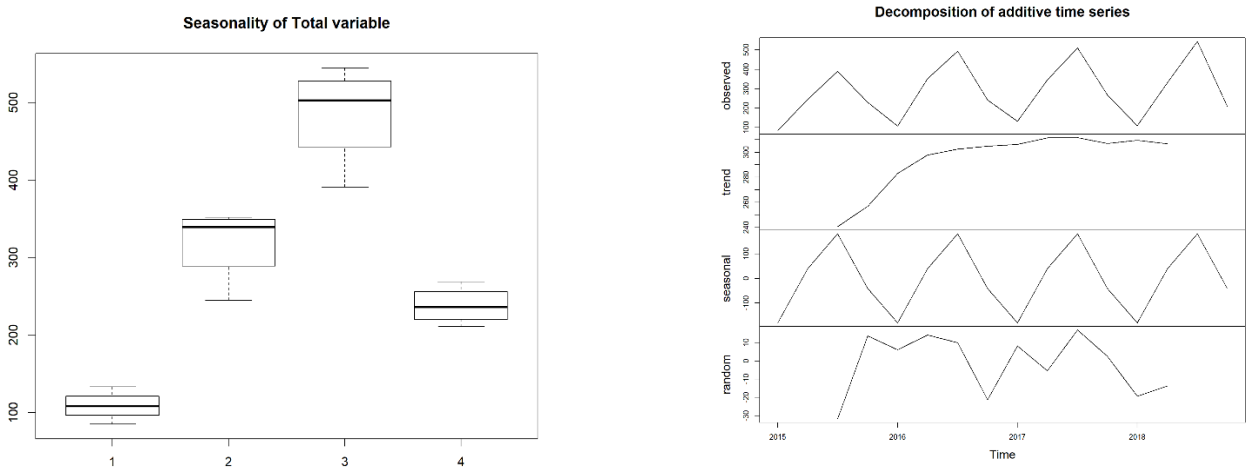


Figure 27 - Time Series Decompose for TOTAL

Both the box-plot and the decomposed time-series plot of the TOTAL variable indicate that there is a strong seasonal component along with a trend component in the time series. Hence, the Holt-Winters Exponential Smoothing is an appropriate model. We use the *HoltWinters()* function to generate the alpha, beta and gamma values. This model was then used to forecast the data up to quarter 4 of 2020. Below are the code and results.

❖ *Holt-Winters Model and forecast*

As expected the gamma value is high (1) indicating there was a strong seasonal component.

```
> TOTAL_hw = HoltWinters(TOTAL_ts)
+ TOTAL_hw
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = TOTAL_ts)

Smoothing parameters:
 alpha: 0.3787764
 beta : 0.3289754
 gamma: 1

Coefficients:
          [,1]
a   319.729278
b    -2.749565
s1 -222.648186
s2   10.843650
s3  208.052454
s4 -108.729278
```

Looking at the model plotted, it is apparent that the model is extremely responsive to the actual data.

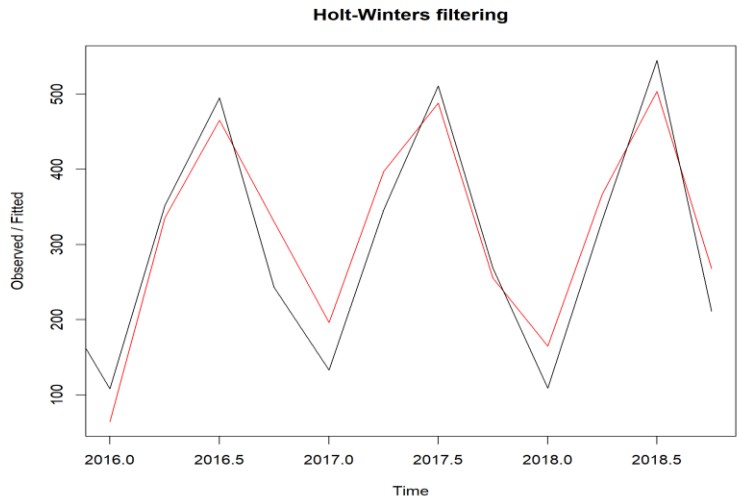


Figure 28 - Holt-Winters model plot for TOTAL

The forecasted values for TOTAL crimes in New York City's parks are shown below.

```
+ TOTAL_hw_fore = forecast:::forecast.HoltWinters(TOTAL_hw)
+ TOTAL_hw_fore
+ #Look at forecasted values
+ plot(TOTAL_hw_fore)
        Point Forecast     Lo 80    Hi 80       Lo 95    Hi 95
2019 Q1       94.33153  33.79740 154.8657    1.752571 186.9105
2019 Q2      325.07380 257.30272 392.8449  221.426875 428.7207
2019 Q3      519.53304 441.82810 597.2380  400.693584 638.3725
2019 Q4      200.00174 109.92627 290.0772   62.243189 337.7603
2020 Q1       83.33327 -44.49953 211.1661 -112.170119 278.8367
2020 Q2      314.07554 172.58727 455.5638   97.687903 530.4632
2020 Q3      508.53478 351.47295 665.5966  268.329441 748.7401
2020 Q4      189.00348  14.63755 363.3694  -77.666204 455.6732
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are not very wide, which indicates the model is good at forecasting.

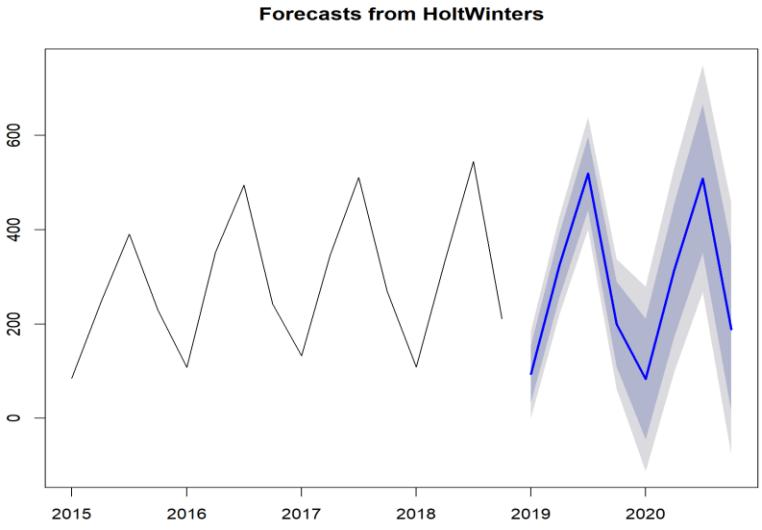


Figure 29 - Holt-Winters Forecast for TOTAL

❖ *Testing Assumptions - Holt-Winters forecast*

To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

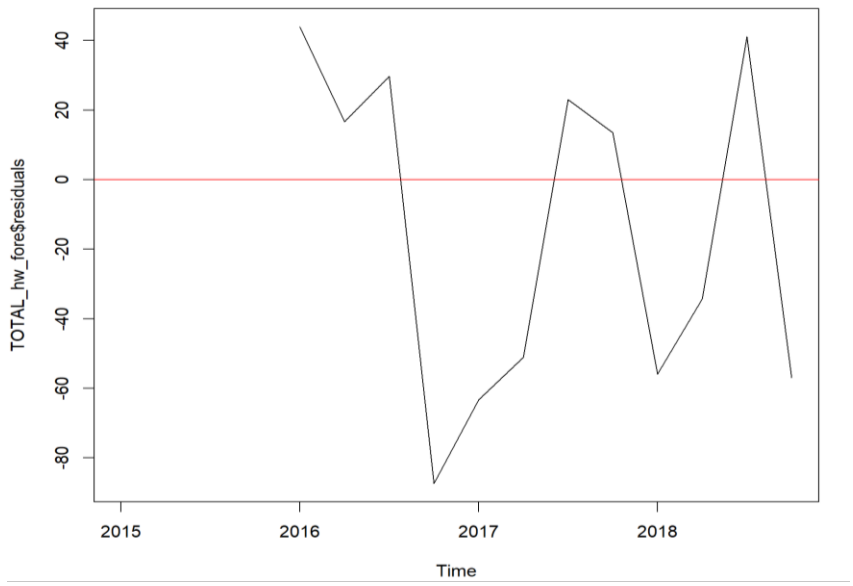


Figure 30 - Residual plot from Holt-Winters Forecast for TOTAL

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

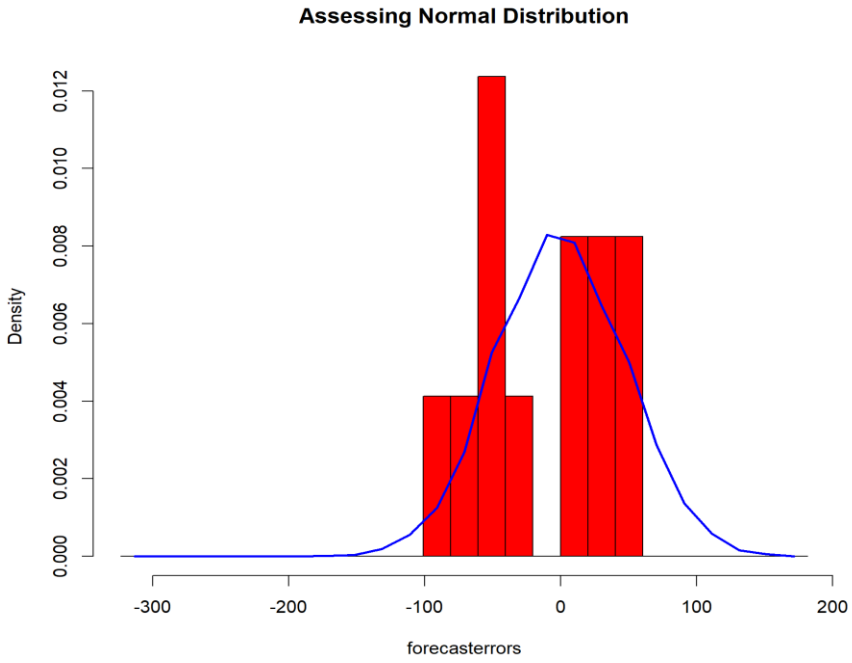


Figure 31 - Distribution of Forecast Errors from Holt-Winters model for TOTAL

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> TOTAL_trendcomp = TOTAL_ts_dc$trend
+
+ TOTAL_trend_data = data.frame(trend = c(TOTAL_trendcomp), time = c(time(TOTAL_trendcomp)))
+
+ TOTAL_trend_reg = lm(TOTAL_trend_data$trend ~ TOTAL_trend_data$time)
+ summary(TOTAL_trend_reg)

Call:
lm(formula = TOTAL_trend_data$trend ~ TOTAL_trend_data$time)

Residuals:
    Min      1Q  Median      3Q     Max
-26.099  -9.963   5.118   9.796  15.756

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)          -41103.880   9848.352  -4.174  0.00191 **
TOTAL_trend_data$time     20.526      4.883   4.204  0.00182 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.6 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.6386,    Adjusted R-squared:  0.6025
F-statistic: 17.67 on 1 and 10 DF,  p-value: 0.001818
```

The p-value for the variable *time* is significant, indicating the trend component heavily influences the time series (i.e. time series is not stationary).

We also performed Augmented Dickey-Fuller (ADF) test and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test to check stationarity. The output is shown below.

```
> adf.test(TOTAL_ts_trend, k = 1, alternative = "stationary")
+ kpss.test(TOTAL_ts_trend)

    Augmented Dickey-Fuller Test

data:  TOTAL_ts_trend
Dickey-Fuller = -1.7848, Lag order = 1, p-value = 0.6544
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  TOTAL_ts_trend
KPSS Level = 0.59744, Truncation lag parameter = 0, p-value = 0.02287
```

The p-value of the ADF test is not significant and indicates the time series is non-stationary. The p-value of the KPSS test is significant, indicating the time series data is non-stationary. To make the time series stationary, differencing was performed. Below are the results of ADF and KPSS test after differencing.

```
> TOTAL_ts_diff = diff(TOTAL_ts_trend, differences = 4)
+ adf.test(TOTAL_ts_diff,k=0, alternative = "stationary")
+ kpss.test(TOTAL_ts_diff)

    Augmented Dickey-Fuller Test

data:  TOTAL_ts_diff
Dickey-Fuller = -6.6286, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(TOTAL_ts_diff, k = 0, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  TOTAL_ts_diff
KPSS Level = 0.10996, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(TOTAL_ts_diff) : p-value greater than printed p-value
```

Now the p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant, indicating the time series data is stationary.

It is interesting to note that the lag order (k) had to be 0 in order for this data to be stationary according to the ADF test (p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. Only one lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows no lag effect above 95% confidence interval indicating there are no partial correlations in this time series.

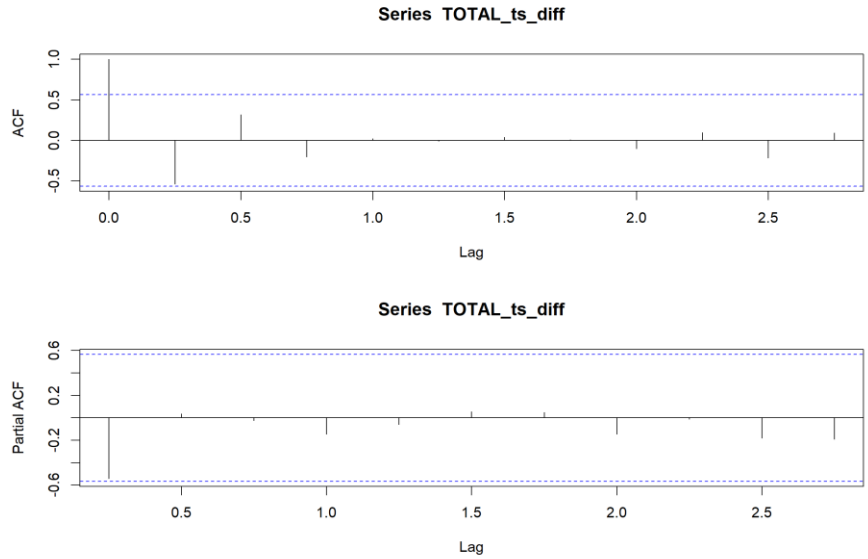


Figure 32 - ACF and PACF plots of ARMA model for TOTAL

❖ *ARMA model*

We chose to run auto.arima() to get the model. The final model was ARMA(1,0) with zero mean with an AIC of 149.37.

```
> TOTAL_arima = auto.arima(TOTAL_ts_diff)
+ TOTAL_arima
Series: TOTAL_ts_diff
ARIMA(1,0,0) with zero mean

Coefficients:
          ar1
      -0.7759
s.e.   0.2079

sigma^2 estimated as 10792:  log likelihood=-72.69
AIC=149.37   AICc=150.7   BIC=150.34
```

The forecast using this model is given below. Looking the values, we conclude the model tends to under-forecast the value of TOTAL crime in Ney York City's parks.

```
> TOTAL_arima_fore = forecast(TOTAL_arima)
+ TOTAL_arima_fore
+ plot(TOTAL_arima_fore)
        Point Forecast      Lo 80      Hi 80      Lo 95     Hi 95
2019 Q1      227.14586   94.01003 360.281695   23.53218 430.75955
2019 Q2     -176.24336 -344.75498  -7.731736 -433.95964  81.47293
2019 Q3      136.74790  -49.85436 323.350165 -148.63563 422.13144
2019 Q4     -106.10323 -302.79580  90.589342 -406.91856 194.71210
2020 Q1       82.32591 -120.19902 284.850831 -227.40923 392.06105
2020 Q2      -63.87699 -269.83352 142.079534 -378.86032 251.10633
2020 Q3       49.56241 -158.43274 257.557546 -268.53871 367.66352
2020 Q4      -38.45566 -247.66853 170.757200 -358.41913 281.50780
```

The 80% and 95% prediction interval are also wider for this model.

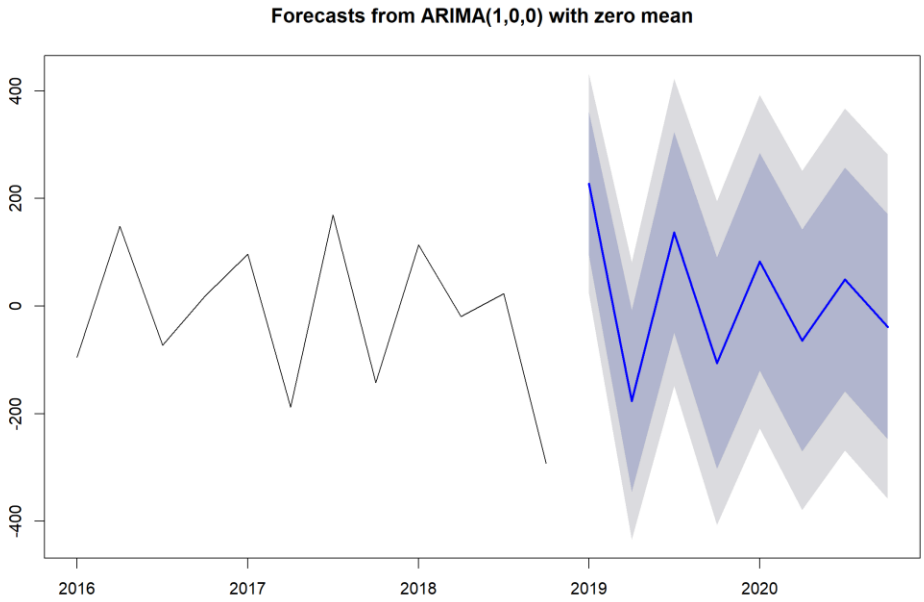


Figure 33 - Forecasts from ARIMA(1,0,0) for TOTAL

b. Assess the Models for TOTAL

We now had two time series models the Hot Winters model (α=0.37, β=0.37, γ=1) and the ARMA(1,0) model. To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(TOTAL_arima_fore)
                    ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set -13.81482 99.4637 69.15778 -15.52437 89.13589 0.3912746 -0.1885793

> accuracy(TOTAL_hw_fore)
                    ME     RMSE      MAE       MPE     MAPE     MASE       ACF1
Training set -15.08868 47.67481 43.05333 -9.872317 21.27604 1.150646 0.0481207
```

The values have been summarized in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARMA	-13.81	99.46	69.16	-15.52	89.14	0.39	-0.19
Holt-Winters	-15.09	47.67	43.05	-9.87	21.28	1.15	0.05

Table 5 - Model Comparison for TOTAL

Based on ME, RMSE, MAE and MAPE, the Holt-Winters model is a better fit for TOTAL data.

2.2. Target variable = MURDER

a. Building time series model for MURDER

```
> #time series murder
+ Sum_MURDER <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(MURDER_TOTAL = sum(MURDER))
+
+ MURDER_ts = ts(Sum_MURDER$MURDER_TOTAL, frequency = 4, start = 2015)
+ plot(MURDER_ts, main = 'Time Series of MURDER variable')
```

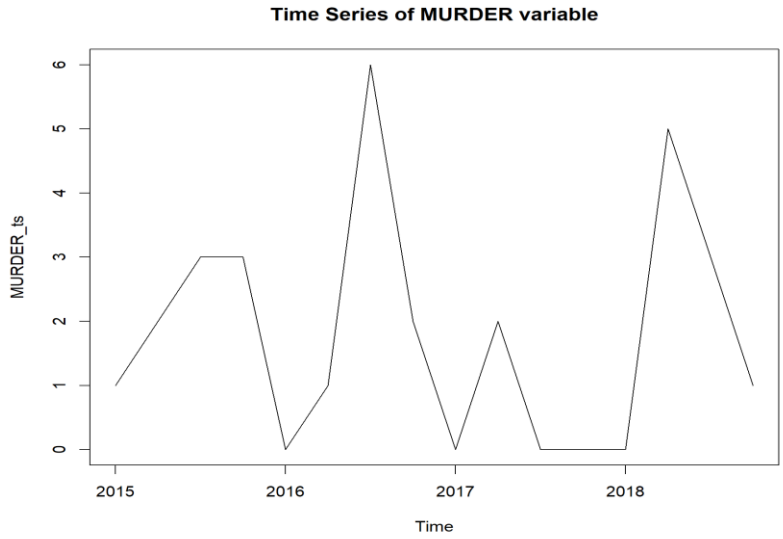


Figure 34 - Time Series of MURDER

❖ Identifying seasonal patterns

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(TOTAL_ts ~ cycle(TOTAL_ts), main = 'Seasonality of Total variable')
+
+ TOTAL_ts_dc = decompose(TOTAL_ts)
+ plot(TOTAL_ts_dc)
```

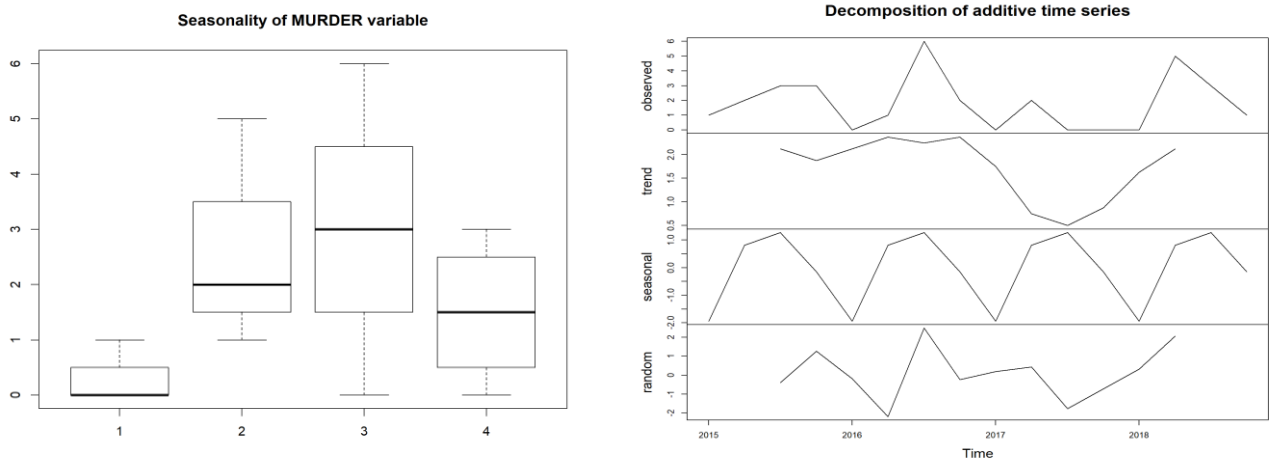


Figure 35 - Time Series Decompose for MURDER

The box-plot and the decomposed time-series plot of the MURDER variable indicate that there is a strong seasonal component along. The trend component is not that strong. Because of the strong seasonal component, we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

As expected the alpha and beta values are 0 and the gamma value is higher (0.47) indicating there was a stronger seasonal component.

```
> MURDER_hw = HoltWinters(MURDER_ts)
+ MURDER_hw
+ plot(MURDER_hw)
+
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = MURDER_ts)

Smoothing parameters:
 alpha: 0
 beta : 0
 gamma: 0.4752555

Coefficients:
          [,1]
a    3.07500000
b    0.10000000
s1 -2.42244883
s2  0.60628724
s3  0.05088089
s4 -1.47251729
```

Looking at the model plotted, it is apparent that the model is not very responsive to the actual data.

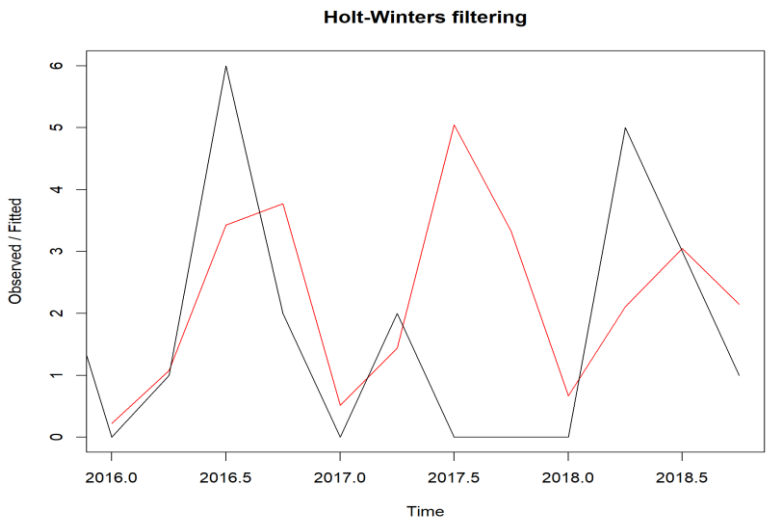


Figure 36 - Holt-Winters model plot for MURDER

❖ *Forecast*

The forecasted values for MURDER crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ MURDER_hw_fore = forecast:::forecast.HoltWinters(MURDER_hw)
+ MURDER_hw_fore
+ #Look at forecasted values
+ plot(MURDER_hw_fore)
         Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
2019 Q1      0.7525512 -2.0681704 3.573273 -3.5613702 5.066473
2019 Q2      3.8812872  1.0605657 6.702009 -0.4326342 8.195209
2019 Q3      3.4258809  0.6051593 6.246602 -0.8880405 7.739802
2019 Q4      2.0024827 -0.8182389 4.823204 -2.3114387 6.316404
2020 Q1      1.1525512 -1.9705212 4.275624 -3.6237758 5.928878
2020 Q2      4.2812872  1.1582149 7.404360 -0.4950397 9.057614
2020 Q3      3.8258809  0.7028085 6.948953 -0.9504461 8.602208
2020 Q4      2.4024827 -0.7205896 5.525555 -2.3738443 7.178810
```

The plot of the forecast reveals that the 80% and 95% prediction intervals wide which indicates the model might not be good at forecasting.

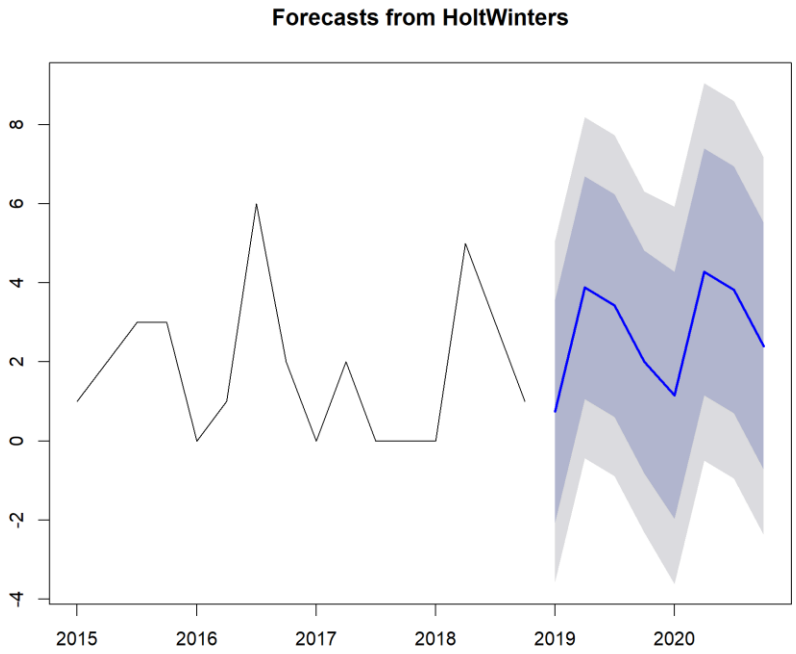


Figure 37 - Holt-Winters Forecast for MURDER

❖ *Testing Assumptions - Holt-Winters forecast*

To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

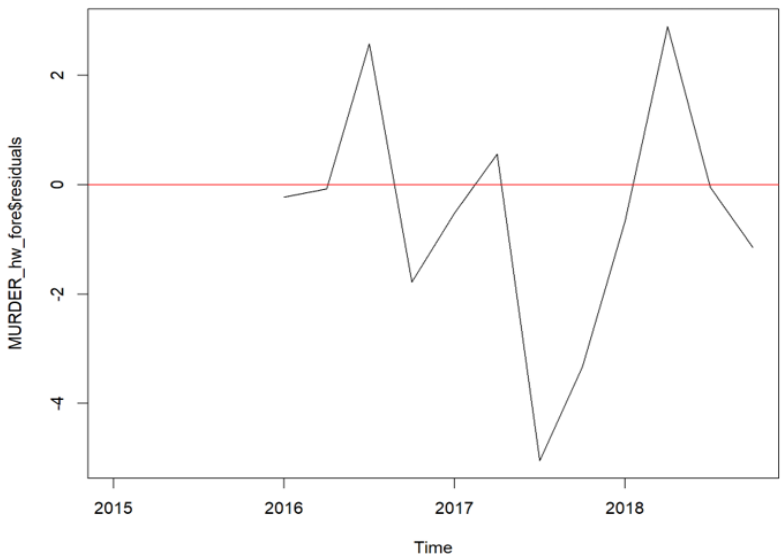


Figure 38 - Residual plot from Holt-Winters Forecast for MURDER

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

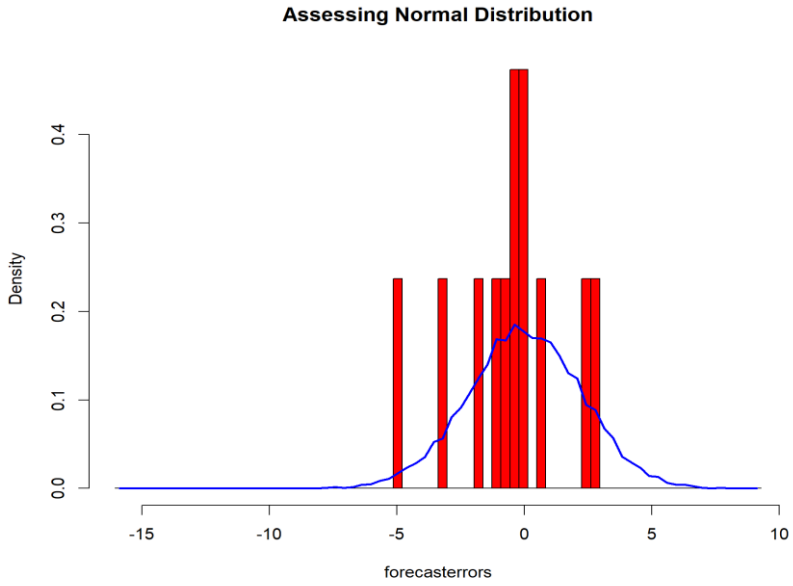


Figure 39 - Distribution of Forecast Errors from Holt-Winters model for MURDER

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> MURDER_trendcomp = MURDER_ts_dc$trend
+ MURDER_trend_data = data.frame(trend = c(MURDER_trendcomp), time = c(time(MURDER_
+ MURDER_trend_reg = lm(MURDER_trend_data$trend ~ MURDER_trend_data$time)
+ summary(MURDER_trend_reg)

Call:
lm(formula = MURDER_trend_data$trend ~ MURDER_trend_data$time)

Residuals:
     Min       1Q   Median       3Q      Max
-1.00626 -0.32707  0.07459  0.39605  0.88622

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            721.0342   408.7268   1.764    0.108
MURDER_trend_data$time  -0.3566     0.2027  -1.760    0.109

Residual standard error: 0.6058 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.2365,    Adjusted R-squared:  0.1601
F-statistic: 3.097 on 1 and 10 DF,  p-value: 0.1089
```

The p-value for the variable time is not significant, indicating the trend component does not influence the time series (i.e. time series is stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> MURDER_ts_trend = MURDER_ts - MURDER_ts_dc$seasonal
+ adf.test(MURDER_ts_trend, k = 0, alternative = "stationary")
+ kpss.test(MURDER_ts_trend)

    Augmented Dickey-Fuller Test

data:  MURDER_ts_trend
Dickey-Fuller = -3.277, Lag order = 0, p-value = 0.09485
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  MURDER_ts_trend
KPSS Level = 0.1109, Truncation lag parameter = 0, p-value = 0.1
```

The p-value of the ADF test is not significant and indicates the time series is not stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Below are the results of ADF and KPSS test after differencing.

```
> MURDER_ts_diff = diff(MURDER_ts_trend, differences = 4)
+ adf.test(MURDER_ts_diff, k = 0, alternative = "stationary")
+ kpss.test(MURDER_ts_diff)

    Augmented Dickey-Fuller Test

data:  MURDER_ts_diff
Dickey-Fuller = -7.0022, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(MURDER_ts_diff, k = 0, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  MURDER_ts_diff
KPSS Level = 0.033945, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(MURDER_ts_diff) : p-value greater than printed p-value
```

❖ *ACF & PACF*

The ACF of the time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. Only one lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows no lag effect above 95% confidence interval indicating there are no partial correlations in this time series.

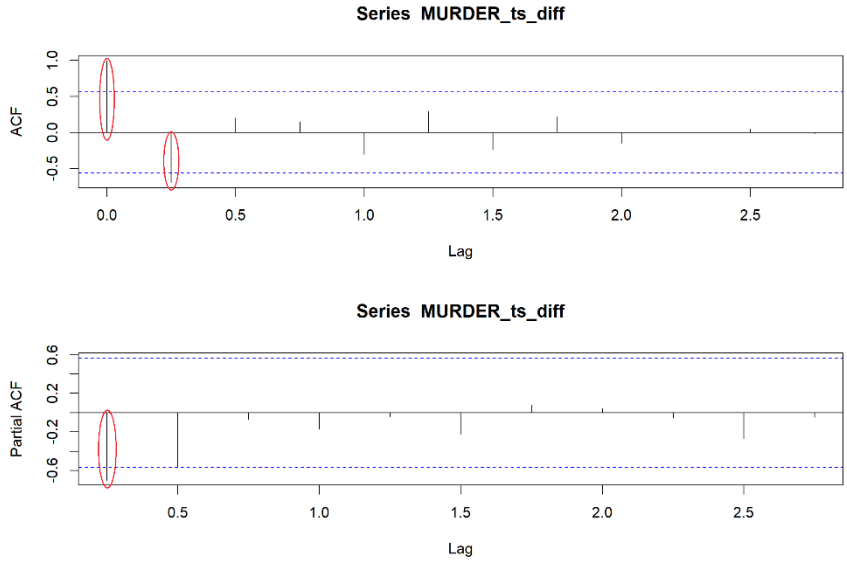


Figure 40 - ACF and PACF plots of ARMA model for MURDER

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARMA(2,0).

```
> MURDER_arima = auto.arima(MURDER_ts_diff)
+ MURDER_arima
Series: MURDER_ts_diff
ARIMA(2,0,0) with zero mean

Coefficients:
          ar1      ar2
      -1.2045  -0.6364
s.e.   0.2047   0.1937

sigma^2 estimated as 35.88:  log likelihood=-38.32
AIC=82.65   AICc=85.65   BIC=84.1
```

❖ *Forecast*

The forecast using this model is given below. The results suggest that the model tends to under-forecast the values of MURDER crime in Ney York City's parks.

```
> MURDER_arima_fore = forecast(MURDER_arima)
+ MURDER_arima_fore
        Point Forecast      Lo 80     Hi 80     Lo 95      Hi 95
2019 Q1    -10.8123072 -18.488657 -3.135957 -22.55227  0.9276568
2019 Q2      5.5463653  -6.471244 17.563974 -12.83298 23.9257120
2019 Q3      0.1999215 -13.346957 13.746800 -20.51824 20.9180846
2019 Q4     -3.7703979 -17.417059  9.876263 -24.64116 17.1003687
2020 Q1      4.4143525  -9.377340 18.206044 -16.67822 25.5069252
2020 Q2     -2.9178456 -17.134776 11.299085 -24.66076 18.8250737
2020 Q3      0.7054514 -13.801754 15.212657 -21.48140 22.8923077
2020 Q4      1.0071117 -13.556621 15.570845 -21.26620 23.2804195
```

The 80% and 95% prediction intervals are wide for this model.

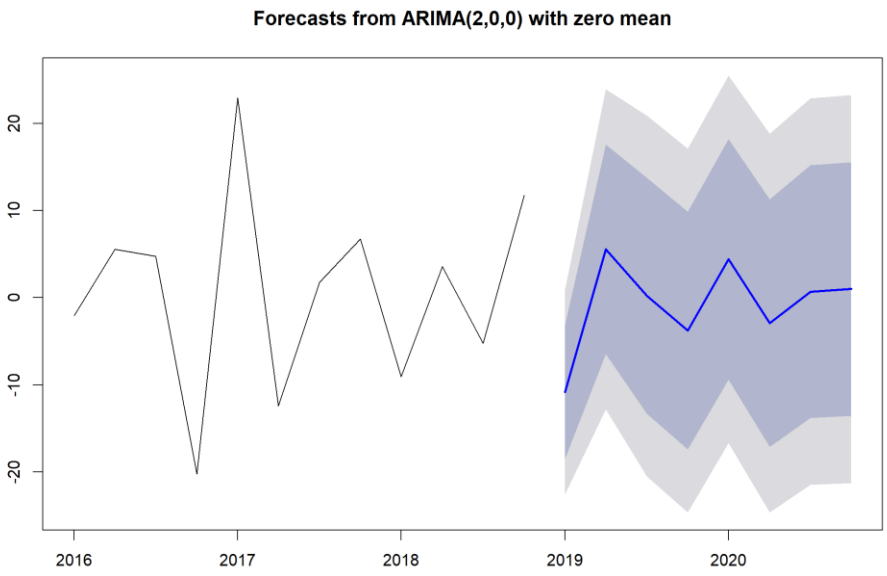


Figure 41 - Forecasts from ARIMA(2,0,0) for MURDER

b. Assess the Models for MURDER

We now had two time series models the Hot Winters model (α=0, β=0, γ=0.47) and the ARMA (2,0) model. To compare the two models, we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(MURDER_arima_fore)
+ accuracy(MURDER_hw_fore)
                     ME     RMSE      MAE      MPE     MAPE      MASE       ACF1
Training set 0.4569305 5.467994 4.096864 46.94826 64.57495 0.2464279 -0.3458967
                      ME     RMSE      MAE  MPE MAPE      MASE      ACF1
Training set -0.5677293 2.182453 1.572702 -Inf  Inf 0.8578377 0.1251354
```

The values have been summarized in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA	0.46	5.47	4.10	46.95	64.57	0.25	-0.3459
Holt-Winters	-0.57	2.18	1.57	-Inf	Inf	0.86	0.13

Table 6 - Model Comparison for MURDER

Based on ME, RMSE, MAE and MPE values, Holt-Winters is the best model for the MURDER data.

2.3. Target variable = RAPE

a. Building time series model for RAPE

```
+ Sum_RAPE <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(RAPE_TOTAL = sum(RAPE))
+
+ RAPE_ts = ts(Sum_RAPE$RAPE_TOTAL, frequency = 4, start = 2015)
+ plot(RAPE_ts, main = 'Time Series of RAPE variable')
```

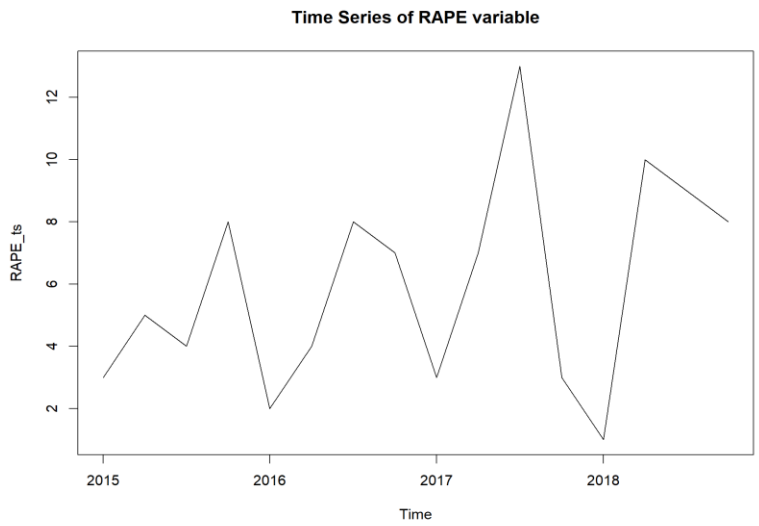


Figure 42 - Time Series of RAPE

❖ *Identifying seasonal patterns*

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(RAPE_ts ~ cycle(RAPE_ts), main = 'Seasonality of RAPE variable')
+
+ RAPE_ts_dc = decompose(RAPE_ts)
+ plot(RAPE_ts_dc)
```

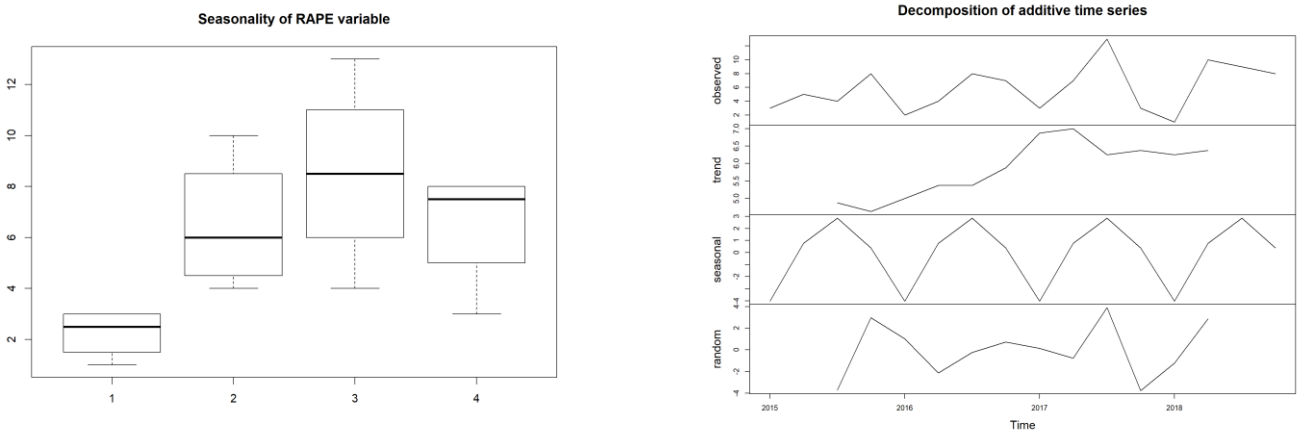


Figure 43 - Time Series Decompose for RAPE

The box-plot and the plot of decomposed time-series of the RAPE variable indicate that there is a strong seasonal component. The trend component is not that strong. Because of the strong seasonal component we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

```
> RAPE_hw = HoltWinters(RAPE_ts)
+ RAPE_hw
+ plot(RAPE_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = RAPE_ts)

Smoothing parameters:
 alpha: 0
 beta : 0
 gamma: 0.6907595

Coefficients:
          [,1]
a    6.7500000
b    0.1875000
s1 -4.3563753
s2  2.7131179
s3  3.4010829
s4  0.4518864
```

As expected the alpha and beta values are 0 and the gamma value is higher (0.69) indicating there was a stronger seasonal component.

Looking at the model plotted, it is very apparent that the model is not very responsive to the actual data.

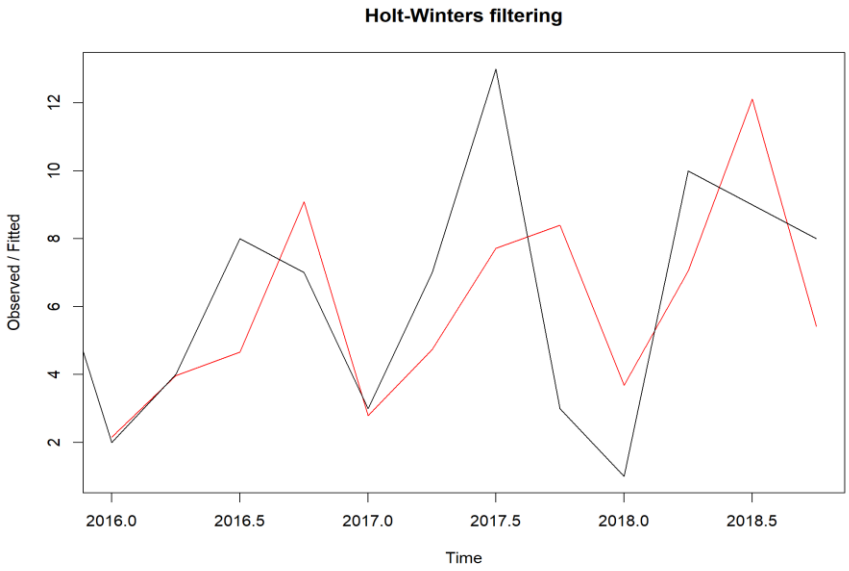


Figure 44 - Holt-Winters model plot for RAPE

❖ *Forecast*

The forecasted values for RAPE crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ RAPE_hw_fore = forecast:::forecast.HoltWinters(RAPE_hw)
+ RAPE_hw_fore
+ #Look at forecasted values
+ plot(RAPE_hw_fore)
+
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
2019 Q1       2.581125 -1.457220  6.619469 -3.594990  8.757239
2019 Q2       9.838118  5.799774 13.876462  3.662003 16.014233
2019 Q3      10.713583  6.675239 14.751927  4.537468 16.889698
2019 Q4       7.951886  3.913542 11.990231  1.775772 14.128001
2020 Q1       3.331125 -1.576998  8.239248 -4.175202 10.837451
2020 Q2      10.588118  5.679995 15.496241  3.081791 18.094444
2020 Q3      11.463583  6.555460 16.371706  3.957256 18.969909
2020 Q4       8.701886  3.793763 13.610010  1.195560 16.208213
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are wide which indicates the model might not be good at forecasting.

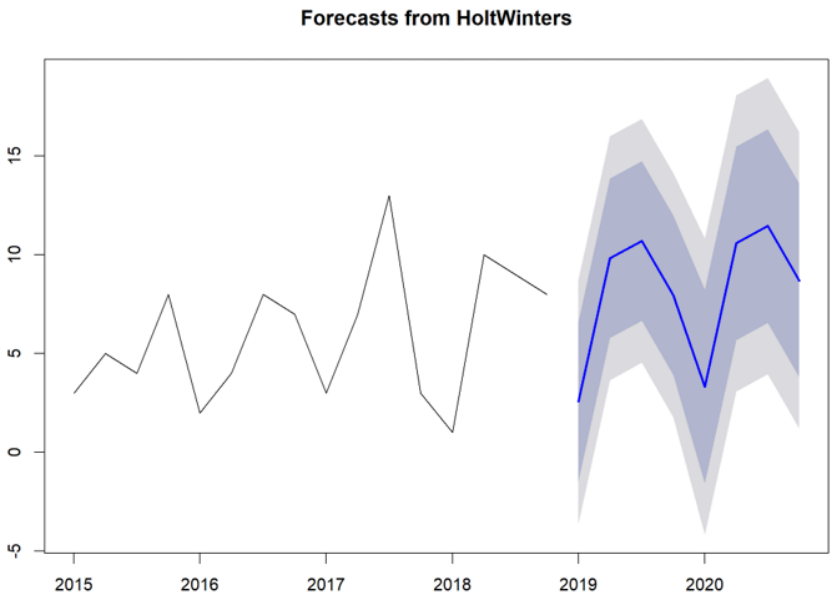


Figure 45 - Holt-Winters Forecast for RAPE

❖ *Testing Assumptions - Holt-Winters forecast*

To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

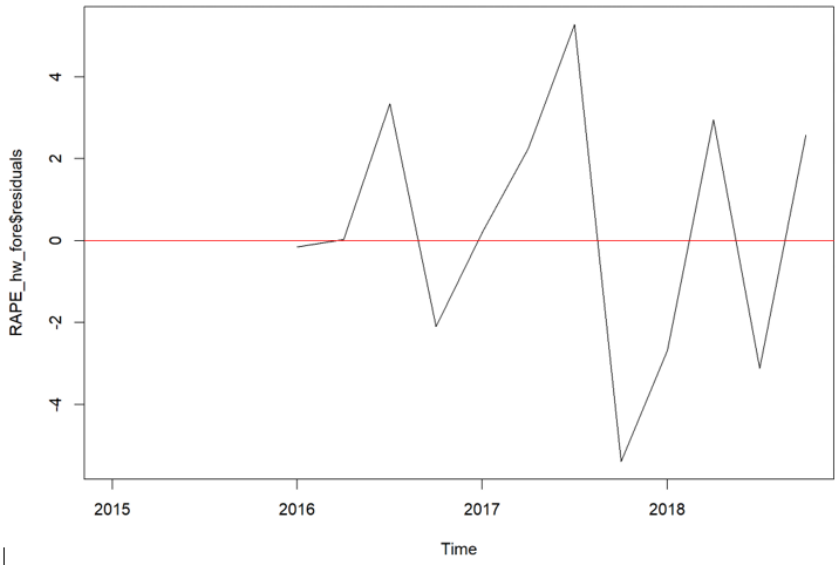


Figure 46 - Residual plot from Holt-Winters Forecast for RAPE

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

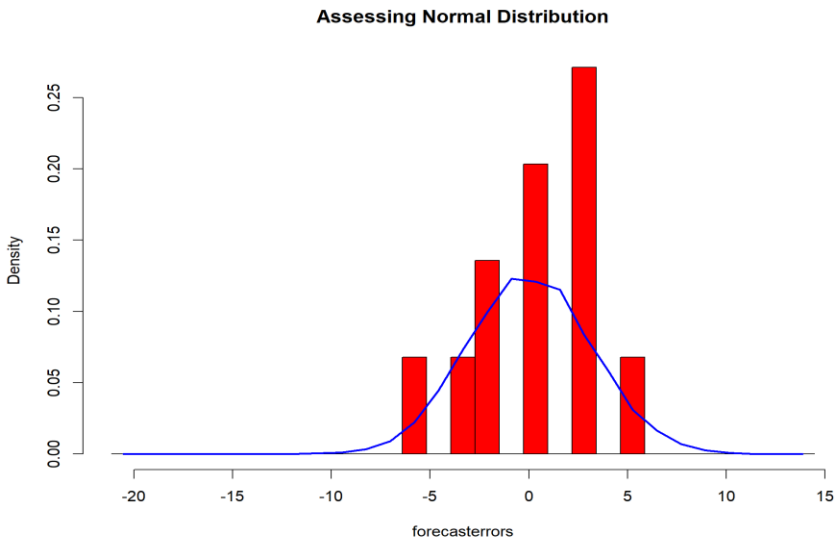


Figure 47 - Distribution of Forecast Errors from Holt-Winters model for RAPE

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
+ RAPE_trendcomp = RAPE_ts_dc$trend
+ RAPE_trend_data = data.frame(trend = c(RAPE_trendcomp), time = c(time(RAPE_trendcomp)))
+ RAPE_trend_reg = lm(RAPE_trend_data$trend ~ RAPE_trend_data$time)
+ summary(RAPE_trend_reg)

Call:
lm(formula = RAPE_trend_data$trend ~ RAPE_trend_data$time)

Residuals:
     Min       1Q   Median       3Q      Max
-0.45994 -0.27418 -0.07663  0.02870  0.93167

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          -1432.7560   325.1301  -4.407  0.00132 **
RAPE_trend_data$time     0.7133     0.1612   4.425  0.00128 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4819 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.6619,    Adjusted R-squared:  0.6281
F-statistic: 19.58 on 1 and 10 DF,  p-value: 0.001285
```

The p-value for the variable *time* is significant, indicating the *trend* component influences the time series (i.e. time series is not stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> RAPE_ts_trend = RAPE_ts - RAPE_ts_dc$seasonal
+ adf.test(RAPE_ts_trend, k = 1, alternative = "stationary")
+ kpss.test(RAPE_ts_trend)
+

    Augmented Dickey-Fuller Test

data:  RAPE_ts_trend
Dickey-Fuller = -5.2284, Lag order = 1, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(RAPE_ts_trend, k = 1, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  RAPE_ts_trend
KPSS Level = 0.22652, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(RAPE_ts_trend) : p-value greater than printed p-value
> acf(RAPE_ts_trend, lag.max = 20)
+ pacf(RAPE_ts_trend, lag.max = 20)
```

The p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Although the ADF and KPSS tests state that the time series is stationary, the regression indicated that it was not hence differencing was performed. Below are the results of ADF and KPSS test after differencing.

```
> RAPE_ts_diff = diff(RAPE_ts_trend, differences = 1)
+ adf.test(RAPE_ts_diff, k = 1, alternative = "stationary")
+ kpss.test(RAPE_ts_diff)

    Augmented Dickey-Fuller Test

data:  RAPE_ts_diff
Dickey-Fuller = -6.7124, Lag order = 1, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(RAPE_ts_diff, k = 1, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  RAPE_ts_diff
KPSS Level = 0.039705, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(RAPE_ts_diff) : p-value greater than printed p-value
```

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. Only one lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows one lag effect above 95% confidence interval indicating there are partial correlations in this time series.

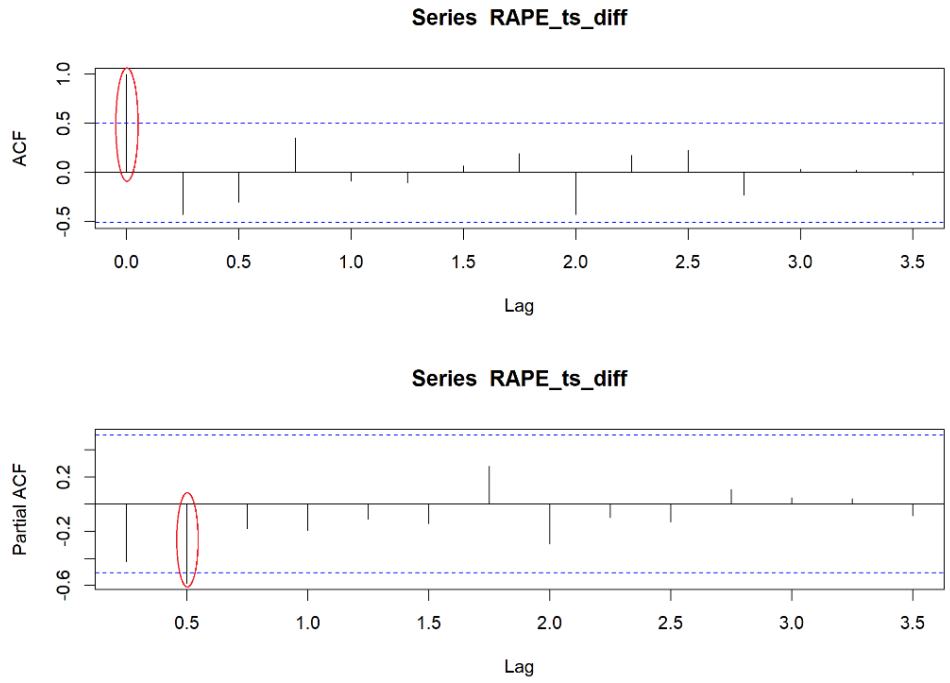


Figure 48 - ACF and PACF plots of ARMA model for RAPE

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARIMA(0,0,0)(0,1,0)[4] which is a seasonal random walk model with 4 seasons (Q1-Q4). This model with an AIC of 69.65 was used as the final model.

```
> RAPE_arima = auto.arima(RAPE_ts_diff)
+ RAPE_arima
Series: RAPE_ts_diff
ARIMA(0,0,0)(0,1,0)[4]

sigma^2 estimated as 27.45:  log likelihood=-33.83
AIC=69.65   AICc=70.1   BIC=70.05
```

❖ *Forecast*

The forecast using the seasonal random walk model is given below. Looking the values, we conclude the model tends to forecast the same the values of RAPE crime for all subsequent years in Ney York City's parks.

```
> RAPE_arima_fore = forecast(RAPE_arima)
+ RAPE_arima_fore
+ layout(1:1)
+ plot(RAPE_arima_fore)
        Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
2019 Q1       2.416667  -4.298291  9.131624  -7.852974 12.686308
2019 Q2       4.208333  -2.506624 10.923291  -6.061308 14.477974
2019 Q3      -3.083333  -9.798291  3.631624 -13.352974  7.186308
2019 Q4       1.458333  -5.256624  8.173291  -8.811308 11.727974
2020 Q1       2.416667  -7.079717 11.913050 -12.106799 16.940132
2020 Q2       4.208333  -5.288050 13.704717 -10.315132 18.731799
2020 Q3      -3.083333 -12.579717  6.413050 -17.606799 11.440132
2020 Q4       1.458333  -8.038050 10.954717 -13.065132 15.981799
```

The 80% and 95% prediction interval is also wider for this model.

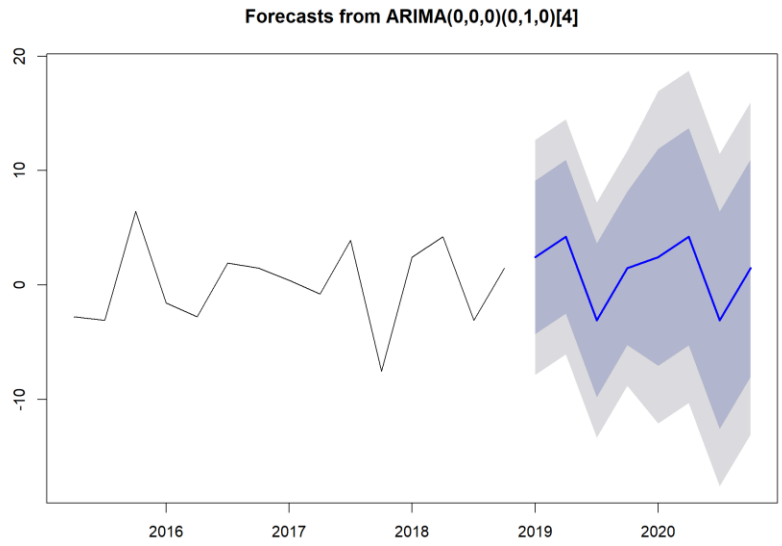


Figure 49 - Forecasts from ARIMA(0,0,0) (0,1,0)[4] for RAPE

b. Assess the Models for RAPE

We now had two time series models the Hot Winters model (α=0, β=0, γ=0.69) and the ARIMA(0,0,0)(0,1,0)[4] model. To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(RAPE_arima_fore)
+ accuracy(RAPE_hw_fore)
                     ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 0.3999333 4.487019 3.200928 90.79481 170.1933 0.7335459 -0.5005356
                    ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
Training set 0.26658 3.028738 2.508425 -28.08726 58.75008 0.8853265 -0.3147108
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA	0.40	4.49	3.20	90.795	170.19	0.73	-0.50
Holt-Winters	0.27	3.03	2.51	-28.087	58.75	0.89	-0.31

Table 7 - Model Comparison for RAPE

Based on ME,RMSE,MAE, MAPE and MPE values the Holt-Winters model is a better fit for the RAPE data.

2.4. Target variable = ROBBERY

a. Building time series model for ROBBERY

```
> #time series ROBBERY
+ Sum_ROBBERY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(ROBBERY_TOTAL = sum(ROBBERY))
+
+ ROBBERY_ts = ts(Sum_ROBBERY$ROBBERY_TOTAL, frequency = 4, start = 2015)
+ plot(ROBBERY_ts, main = 'Time Series of ROBBERY variable')
```

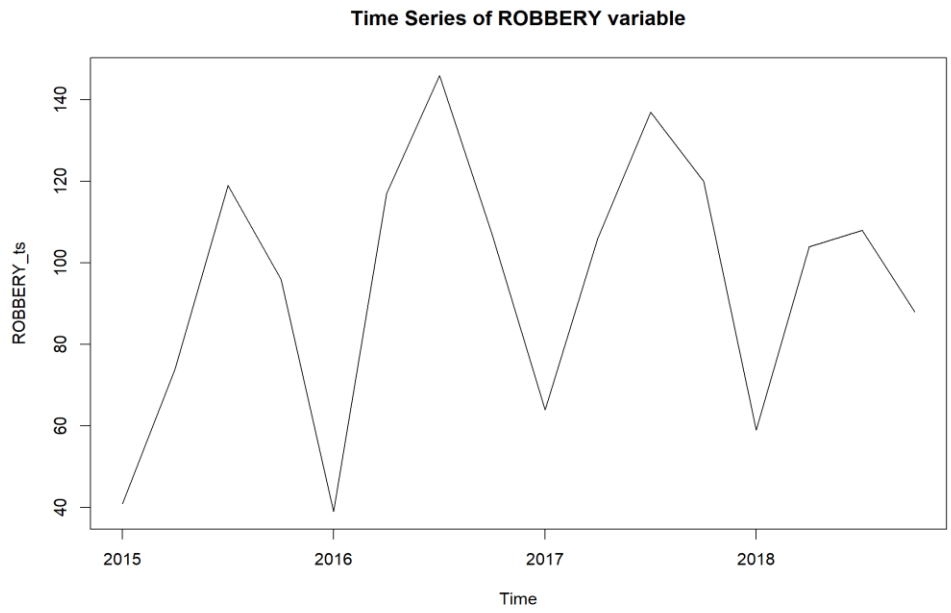


Figure 50 - Time Series of ROBBERY

❖ *Identifying seasonal patterns*

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(ROBBERY_ts ~ cycle(ROBBERY_ts), main = 'Seasonality of ROBBERY variable')
+
+ ROBBERY_ts_dc = decompose(ROBBERY_ts)
+ plot(ROBBERY_ts_dc)
```

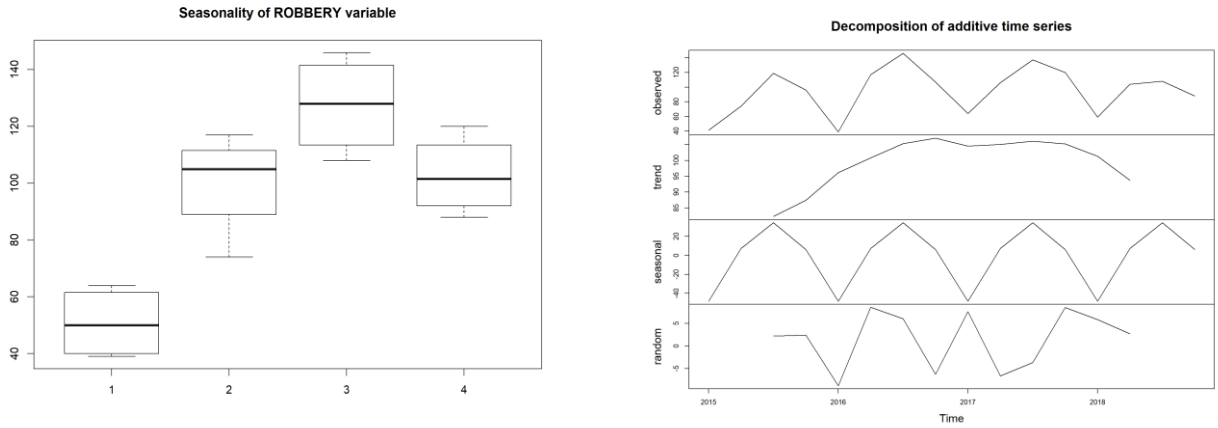


Figure 51 - Time Series Decompose for ROBBERY

The box-plot and the plot of decomposed time-series of the ROBBERY variable indicate that there is a strong seasonal component along with a trend component. Given the strong seasonal component, we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

```
> ROBBERY_hw = HoltWinters(ROBBERY_ts)
+ ROBBERY_hw
+ plot(ROBBERY_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = ROBBERY_ts)

Smoothing parameters:
 alpha: 0.64717
 beta : 0.3067567
 gamma: 0.7999003

Coefficients:
         [,1]
a   79.799893
b   -5.974225
s1 -53.568797
s2   1.327966
s3  26.735568
s4   7.824748
`
```

The beta value is lower than the alpha and gamma value indicating the recent effects (previous as well seasonal) have a stronger influence on the model than the overall or trend effects.

Looking at the model plotted, it is apparent that the model is slightly responsive to the actual data.

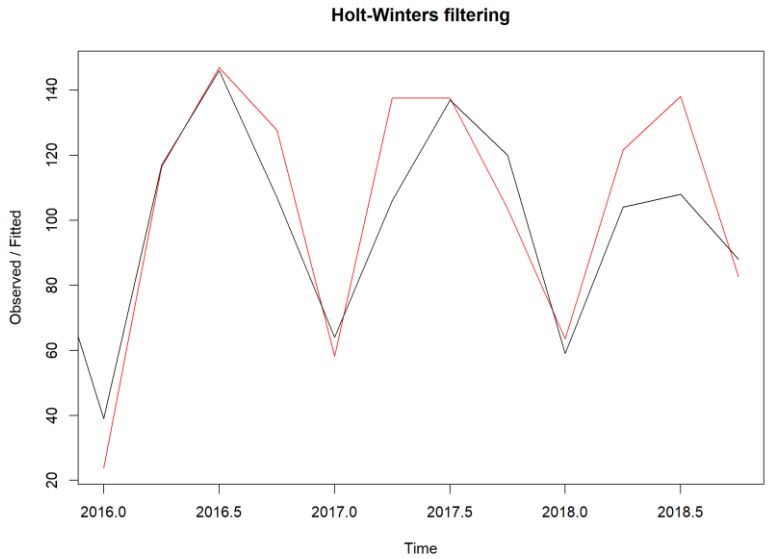


Figure 52 - Holt-Winters model plot for ROBBERY

❖ *Forecast*

The forecasted values for ROBBERY crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ ROBBERY_hw_fore = forecast:::forecast.HoltWinters(ROBBERY_hw)
+ ROBBERY_hw_fore
+ #Look at forecasted values
+ plot(ROBBERY_hw_fore)
        Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
2019 Q1      20.256871  -0.5430731  41.05682 -11.553898  52.06764
2019 Q2      69.179409  41.9386448  96.42017  27.518255 110.84056
2019 Q3      88.612786  53.7731078 123.45246  35.330096 141.89548
2019 Q4      63.727742  20.3460479 107.10944  -2.618833 130.07432
2020 Q1      -3.640029 -59.9169492  52.63689 -89.708151  82.42809
2020 Q2      45.282509 -20.5233744 111.08839 -55.358905 145.92392
2020 Q3      64.715886 -11.3923733 140.82414 -51.681652 181.11342
2020 Q4      39.830841 -47.2745655 126.93625 -93.385383 173.04707
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are wide which indicates the model might not be good at forecasting.

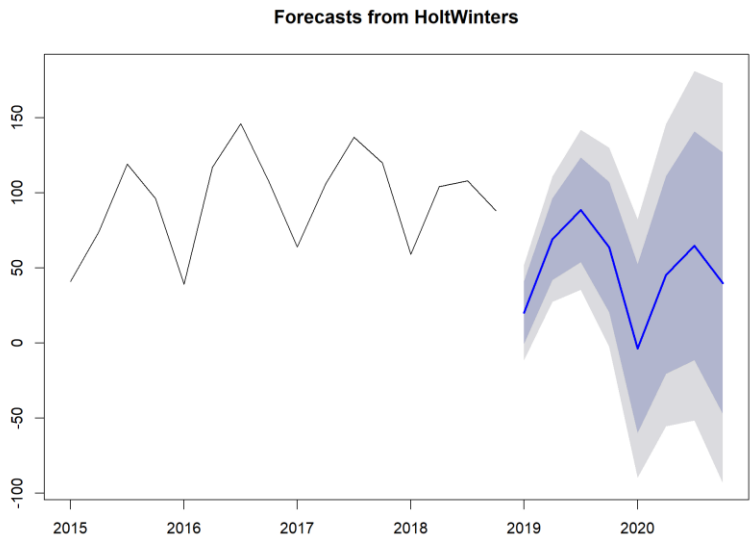


Figure 53 - Holt-Winters Forecast for ROBBERY

❖ *Testing Assumptions - Holt-Winters forecast*

To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

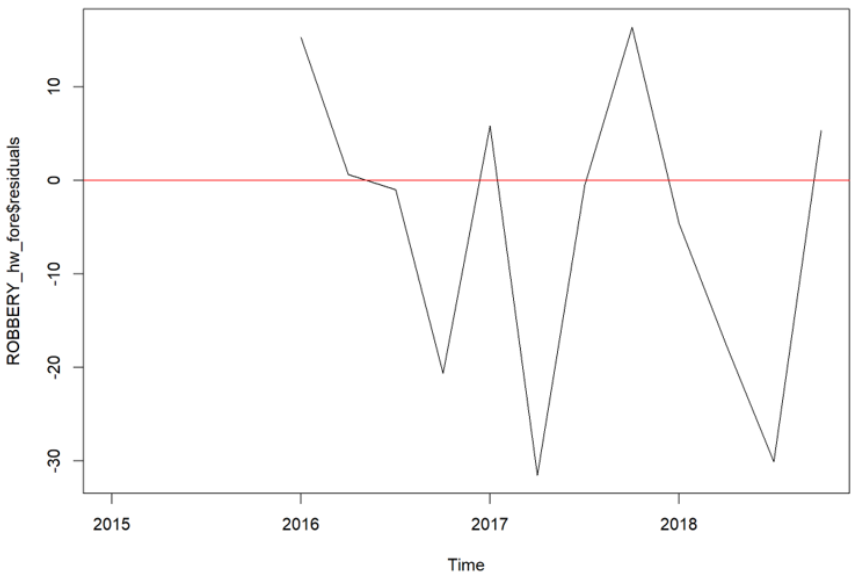


Figure 54 - Residual plot from Holt-Winters Forecast for ROBBERY

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

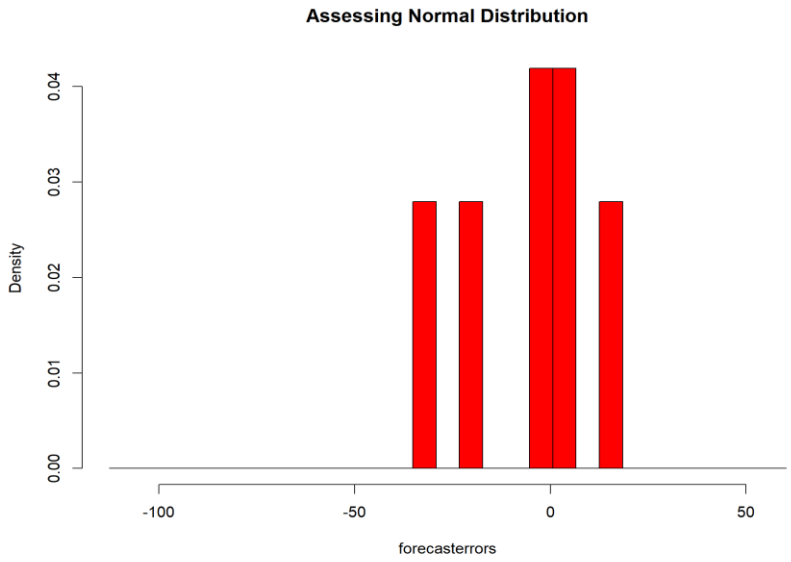


Figure 55 - Distribution of Forecast Errors from Holt-Winters model for ROBBERY

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> ROBBERY_trendcomp = ROBBERY_ts_dc$trend
+ ROBBERY_trend_data = data.frame(trend = c(ROBBERY_trendcomp), time = c(time(ROBBERY
+ ROBBERY_trend_reg = lm(ROBBERY_trend_data$trend ~ ROBBERY_trend_data$time)
+ summary(ROBBERY_trend_reg)

Call:
lm(formula = ROBBERY_trend_data$trend ~ ROBBERY_trend_data$time)

Residuals:
    Min      1Q  Median      3Q     Max
-12.391  -4.410   2.513   4.274   8.104

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            -9473.490   4866.944  -1.946   0.0802 .
ROBBERY_trend_data$time     4.747      2.413   1.967   0.0775 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.214 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.279, Adjusted R-squared:  0.2069
F-statistic: 3.869 on 1 and 10 DF,  p-value: 0.07753
```

The p-value for the variable *time* is not significant, indicating the *trend* component does not influence the time series (i.e. time series is stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> ROBBERY_ts_trend = ROBBERY_ts - ROBBERY_ts_dc$seasonal
+ adf.test(ROBBERY_ts_trend, k =0 , alternative = "stationary")
+ kpss.test(ROBBERY_ts_trend)

    Augmented Dickey-Fuller Test

data:  ROBBERY_ts_trend
Dickey-Fuller = -1.644, Lag order = 0, p-value = 0.708
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  ROBBERY_ts_trend
KPSS Level = 0.32781, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(ROBBERY_ts_trend) : p-value greater than printed p-value
```

The p-value of the ADF test is not significant and indicates the time series is non-stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. To make the time series stationary differencing was performed. Below are the results of ADF and KPSS test after differencing.

```
> ROBBERY_ts_diff = diff(ROBBERY_ts_trend, differences = 1)
+ adf.test(ROBBERY_ts_diff, k = 1, alternative = "stationary")
+ kpss.test(ROBBERY_ts_diff)

    Augmented Dickey-Fuller Test

data:  ROBBERY_ts_diff
Dickey-Fuller = -4.0575, Lag order = 1, p-value = 0.02125
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  ROBBERY_ts_diff
KPSS Level = 0.11999, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(ROBBERY_ts_diff) : p-value greater than printed p-value
```

Now the p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary.

Note that the lag order (k) had to be 1 in order for this data to be stationary according to the ADF test(p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. One lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows no lag effect above 95% confidence interval indicating there are no partial correlations in this time series.

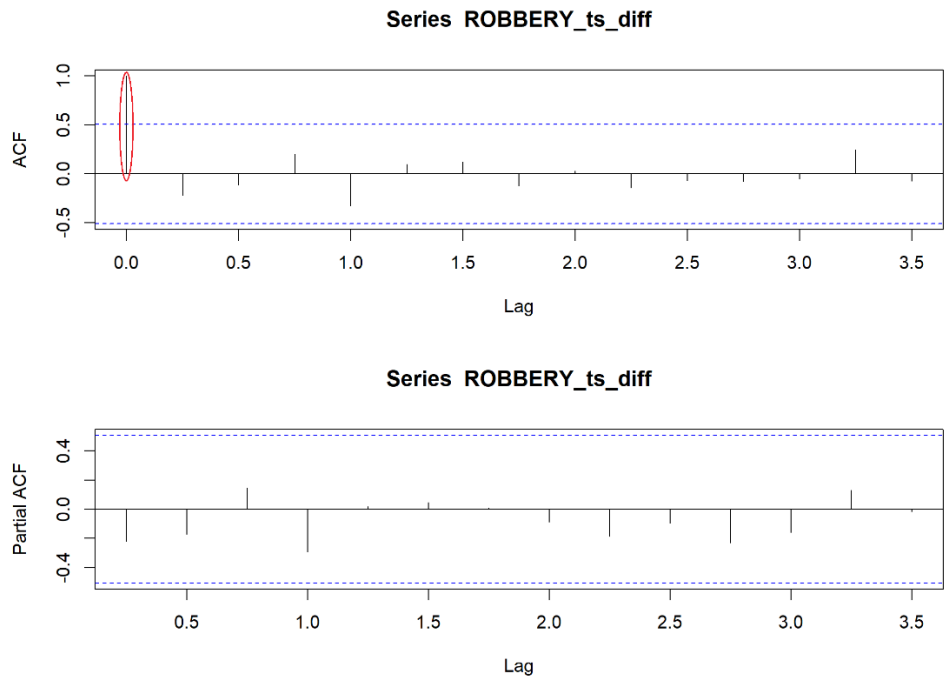


Figure 56 - ACF and PACF plots of ARMA model for ROBBERY

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARIMA(0,0,0) which means that the data is white noise. This is not a good model.

```
> ROBBERY_arima = auto.arima(ROBBERY_ts_diff)
+ ROBBERY_arima
Series: ROBBERY_ts_diff
ARIMA(0,0,0) with zero mean

sigma^2 estimated as 180.9:  log likelihood=-60.27
AIC=122.53   AICc=122.84   BIC=123.24
```

We then chose to not use the differenced data. This resulted in ARMA(1,0) with an AIC of 130.25.

```
> ROBBERY_arima = auto.arima(ROBBERY_ts_trend)
+ ROBBERY_arima
Series: ROBBERY_ts_trend
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1     mean
      0.5305  94.1021
s.e.  0.2065   5.8629

sigma^2 estimated as 154.6:  log likelihood=-62.13
AIC=130.25   AICc=132.25   BIC=132.57
```

❖ *Forecast*

The forecast using this model is given below. The results show that the model forecasts ROBBERY count to be in the range 87 to 94.

```
> ROBBERY_arima_fore = forecast(ROBBERY_arima)
+ ROBBERY_arima_fore
+ layout(1:1)
+ plot(ROBBERY_arima_fore)
        Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
2019 Q1       87.57717 71.64409 103.5103 63.20962 111.9447
2019 Q2       90.64074 72.60460 108.6769 63.05684 118.2246
2019 Q3       92.26590 73.68080 110.8510 63.84244 120.6894
2019 Q4       93.12801 74.39133 111.8647 64.47272 121.7833
2020 Q1       93.58534 74.80622 112.3645 64.86516 122.3055
2020 Q2       93.82795 75.03690 112.6190 65.08953 122.5664
2020 Q3       93.95665 75.16225 112.7510 65.21309 122.7002
2020 Q4       94.02492 75.22957 112.8203 65.27992 122.7699
```

The 80% and 95% prediction intervals are wide for this model.

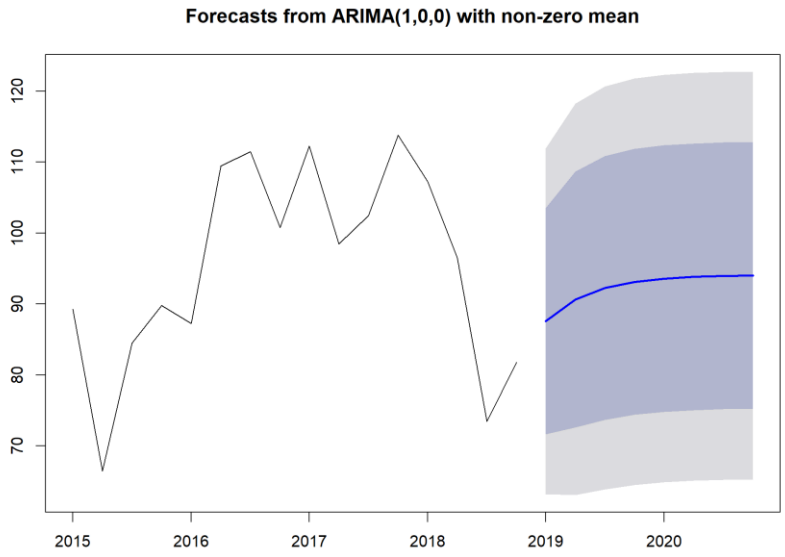


Figure 57 - Forecasts from ARIMA(1,0,0) for ROBBERY

b. Assess the Models for ROBBERY

We now had two time series models the Hot Winters model (α=0.64, β=0.30, γ=0.79) and the ARMA(1, 0) model. To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(ROBBERY_arima_fore)
+ accuracy(ROBBERY_hw_fore)
                     ME     RMSE      MAE       MPE     MAPE      MASE        ACF1
Training set 0.2066125 11.62968 8.877248 -1.517813 9.932906 0.5096985 0.06796711
                     ME    RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set -5.220502 16.3928 12.45875 -2.858914 14.27228 0.7153346 -0.1213589
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARMA	0.21	11.63	8.88	-1.5178	9.93	0.51	0.07
Holt-Winters	-5.22	16.39	12.46	-2.8589	14.27	0.72	-0.12

Table 8 - Model Comparison for ROBBERY

Based on RMSE, MAE, MAPE and MASE values the ARMA model is a better fit for ROBBERY data.

2.5. Target variable = FELLONY ASSAULT

a. Building time series model for FELLONY ASSAULT

```
> #time series FELONY_ASSAULT
+ Sum_FELONY_ASSAULT <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(FELONY_ASSAULT_TOTAL = sum(FELONY_ASSAULT))
+
+ FELONY_ASSAULT_ts = ts(Sum_FELONY_ASSAULT$FELONY_ASSAULT_TOTAL, frequency = 4, start = 2015)
+ plot(FELONY_ASSAULT_ts, main = 'Time Series of FELONY_ASSAULT variable')
```

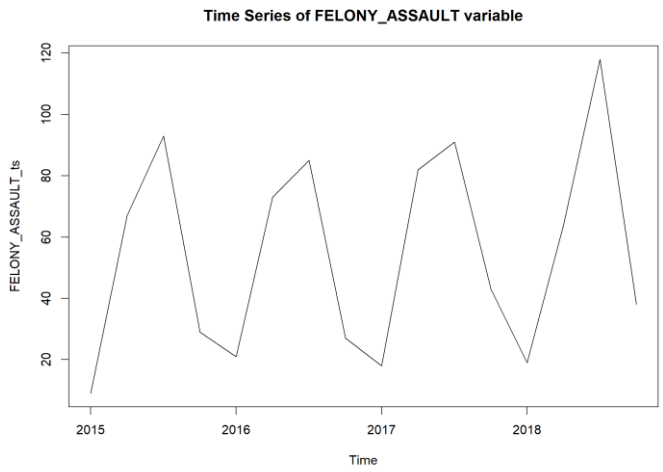


Figure 58 - Time Series of FELONY ASSAULT

❖ *Identifying seasonal patterns*

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(ROBBERY_ts ~ cycle(ROBBERY_ts), main = 'Seasonality of ROBBERY variable')
+
+ ROBBERY_ts_dc = decompose(ROBBERY_ts)
+ plot(ROBBERY_ts_dc)
```

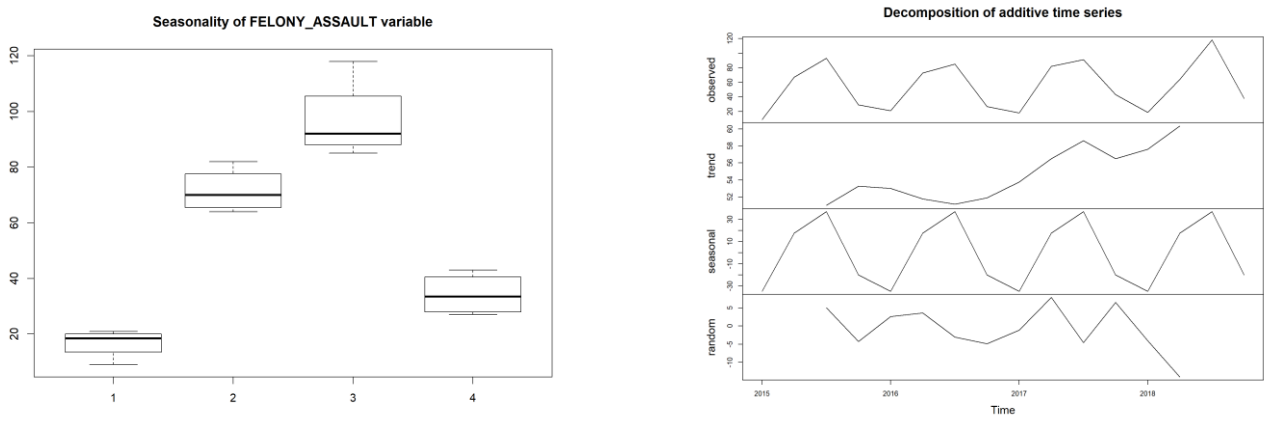


Figure 59 - Time Series Decompose for FELONY ASSAULT

The box-plot and the plot of decomposed time-series of the FELLONY_ASSAULT variable indicate that there is a strong seasonal component along with a trend component. Because of the strong seasonal component we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

```
> FELONY_ASSAULT_hw = HoltWinters(FELONY_ASSAULT_ts)
+ FELONY_ASSAULT_hw
+ plot(FELONY_ASSAULT_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = FELONY_ASSAULT_ts)

Smoothing parameters:
 alpha: 0.06483632
 beta : 1
 gamma: 0

Coefficients:
         [,1]
a   59.963066
b    2.010068
s1 -33.750000
s2  19.500000
s3  40.250000
s4 -26.000000
```

Surprisingly the beta value is higher than the alpha and gamma value indicating the recent effects (previous as well seasonal) do not have much influence on the model but the overall or trend effects do.

Looking at the model plotted, it is apparent that the model is very responsive to the actual data.

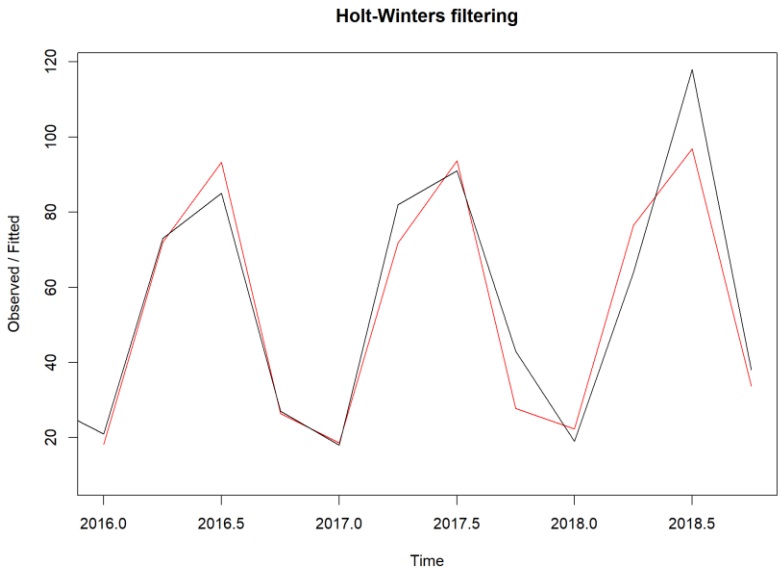


Figure 60 - Holt-Winters model plot for FELONY ASSAULT

❖ *Forecast*

The forecasted values for FELLONY_ASSAULT crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ FELONY_ASSAULT_hw_fore = forecast:::forecast.HoltWinters(FELONY_ASSAULT_hw)
+ FELONY_ASSAULT_hw_fore
+ #Look at forecasted values
+ plot(FELONY_ASSAULT_hw_fore)
        Point Forecast    Lo 80     Hi 80     Lo 95     Hi 95
2019 Q1       28.22313 16.03675  40.40952  9.585663  46.86060
2019 Q2       83.48320 71.19479  95.77162 64.689690 102.27671
2019 Q3      106.24327 93.72833 118.75821 87.103316 125.38322
2019 Q4       42.00334 29.09549  54.91118 22.262494  61.74418
2020 Q1       36.26341 22.76453  49.76228 15.618654  56.90816
2020 Q2       91.52347 77.21634 105.83061 69.642597 113.40435
2020 Q3      114.28354 98.94456 129.62253 90.824593 137.74249
2020 Q4       50.04361 33.45328  66.63394 24.670895  75.41633
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are narrow which indicates the model is good at forecasting.

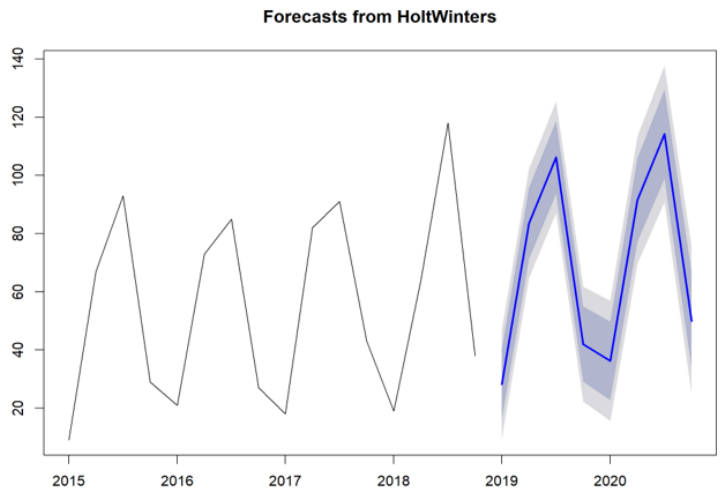
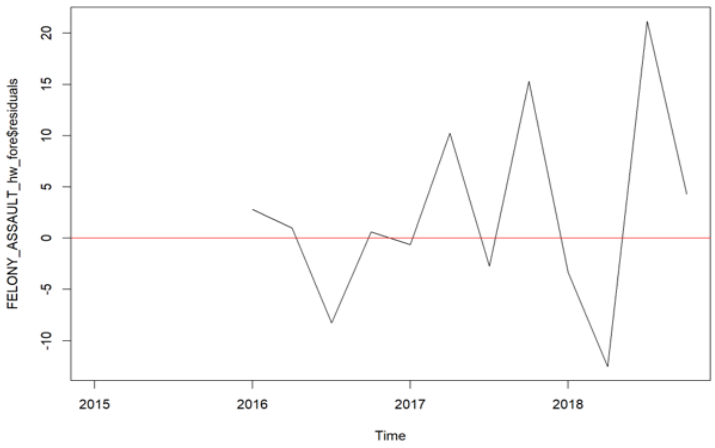


Figure 61 - Holt-Winters Forecast for FELONY ASSAULT

❖ *Testing Assumptions - Holt-Winters forecast*



To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

Figure 62 - Residual plot from Holt-Winters Forecast for ROFELONY ASSAULT

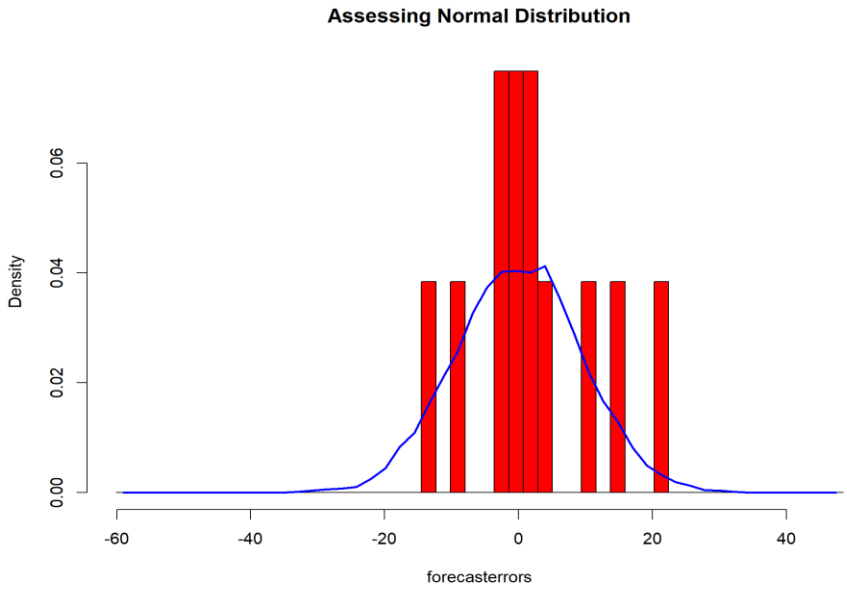


Figure 63 - Distribution of Forecast Errors from Holt-Winters model for FELONY ASSAULT

The above histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> FELONY_ASSAULT_trendcomp = FELONY_ASSAULT_ts_dc$trend
+ FELONY_ASSAULT_trend_data = data.frame(trend = c(FELONY_ASSAULT_trendcomp), time = c(time(FELONY_ASSAULT_trendcomp)))
+ FELONY_ASSAULT_trend_reg = lm(FELONY_ASSAULT_trend_data$trend ~ FELONY_ASSAULT_trend_data$time)
+ summary(FELONY_ASSAULT_trend_reg)

Call:
lm(formula = FELONY_ASSAULT_trend_data$trend ~ FELONY_ASSAULT_trend_data$time)

Residuals:
     Min       1Q   Median       3Q      Max
-2.35606 -1.02225  0.08144  1.18797  2.09280

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                  -6133.5246  1118.2071  -5.485 0.000267 ***
FELONY_ASSAULT_trend_data$time     3.0682     0.5544   5.534 0.000250 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.657 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.7538,    Adjusted R-squared:  0.7292
F-statistic: 30.62 on 1 and 10 DF,  p-value: 0.0002496
```

The p-value for the variable *time* is significant, indicating the *trend* component influences the time series (i.e. time series is not stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> FELONY_ASSAULT_ts_trend = FELONY_ASSAULT_ts - FELONY_ASSAULT_ts_dc$seasonal
+ adf.test(FELONY_ASSAULT_ts_trend, k = 1, alternative = "stationary")
+ kpss.test(FELONY_ASSAULT_ts_trend)

    Augmented Dickey-Fuller Test

data:  FELONY_ASSAULT_ts_trend
Dickey-Fuller = -3.8455, Lag order = 1, p-value = 0.03247
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  FELONY_ASSAULT_ts_trend
KPSS Level = 0.42492, Truncation lag parameter = 0, p-value = 0.06642
```

The p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Although the ADF and KPSS tests state that the time series is stationary, the regression indicated that it was not hence differencing was performed. Below are the results of ADF and KPSS test after differencing.

```
> FELONY_ASSAULT_ts_diff = diff(FELONY_ASSAULT_ts_trend, differences = 1)
+ adf.test(FELONY_ASSAULT_ts_diff, k = 1, alternative = "stationary")
+ kpss.test(FELONY_ASSAULT_ts_diff)

    Augmented Dickey-Fuller Test

data:  FELONY_ASSAULT_ts_diff
Dickey-Fuller = -4.7283, Lag order = 1, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(FELONY_ASSAULT_ts_diff, k = 1, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  FELONY_ASSAULT_ts_diff
KPSS Level = 0.031872, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(FELONY_ASSAULT_ts_diff) : p-value greater than printed p-value
```

Note that the lag order (k) had to be 1 in order for this data to be stationary according to the ADF test(p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. Two lag effects are above 95% confidence interval which indicates at least two autocorrelation components. The PACF shows one lag effects above 95% confidence interval indicating there are partial correlations in this time series.

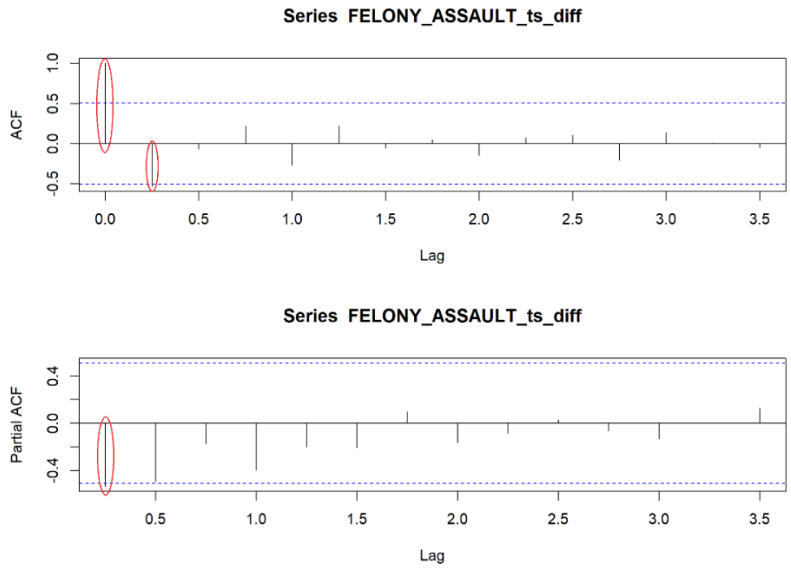


Figure 64 - ACF and PACF plots of ARMA model for FELONY ASSAULT

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARMA(2,0) with an AIC of 115.1.

```
> FELONY_ASSAULT_arima = auto.arima(FELONY_ASSAULT_ts_diff)
+ FELONY_ASSAULT_arima
Series: FELONY_ASSAULT_ts_diff
ARIMA(2,0,0) with zero mean

Coefficients:
          ar1     ar2
      -0.8318  -0.666
s.e.   0.2114   0.255

sigma^2 estimated as 88.36:  log likelihood=-54.55
AIC=115.1   AICc=117.28   BIC=117.23
```

❖ *Forecast*

The forecast using this model is given below. Looking the values, it seems model under forecasts FELLONY_ASSAULT.

```
> FELONY_ASSAULT_arima_fore = forecast(FELONY_ASSAULT_arima)
+ FELONY_ASSAULT_arima_fore
        Point Forecast      Lo 80     Hi 80      Lo 95    Hi 95
2019 Q1     -3.9499249 -15.996472  8.096622 -22.373529 14.47368
2019 Q2     18.6311233   2.961984 34.300263  -5.332758 42.59500
2019 Q3    -12.8664181 -28.538656  2.805820 -36.835037 11.10220
2019 Q4     -1.7061753 -18.640177 15.227826 -27.604495 24.19214
2020 Q1      9.9881423  -7.829885 27.806170 -17.262179 37.23846
2020 Q2     -7.1716612 -24.992904 10.649582 -34.426901 20.08358
2020 Q3     -0.6867914 -18.831313 17.457730 -28.436443 27.06286
2020 Q4      5.3475545 -13.053470 23.748579 -22.794384 33.48949
```

The 80% and 95% prediction intervals are wide for this model.

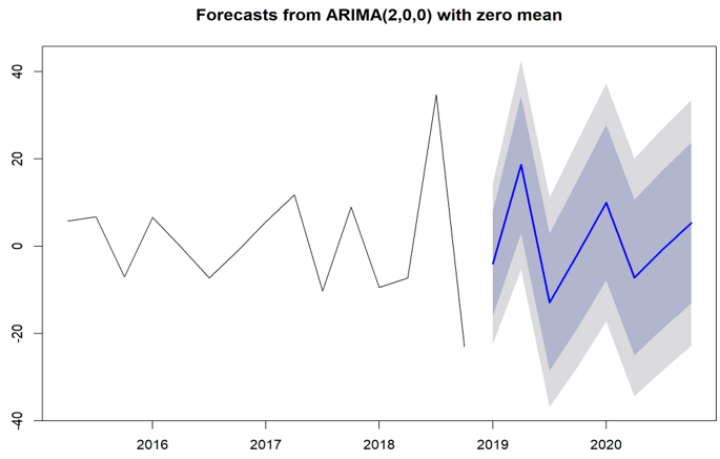


Figure 65 - Forecasts from ARIMA(2,0,0) for FELONY ASSAULT

b. Assess the Models for FELLONY ASSAULT

We now had two time series models the Hot Winters model (α=0.06, β=1, γ=0) and the ARMA(2,0) model. To compare the two models we chose to look at their accuracy measures.

```
> accuracy(FELONY_ASSAULT_arima_fore)
+ accuracy(FELONY_ASSAULT_hw_fore)
                     ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
Training set 2.735086 8.750896 6.530234 80.95642 119.0935 0.4406906 -0.1214241
                     ME     RMSE      MAE      MPE     MAPE      MASE        ACF1
Training set 2.326459 9.396805 6.911148 3.410462 12.30624 0.7339272 -0.3259017
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARMA	2.74	8.75	6.53	80.956	119.09	0.44	-0.12
Holt-Winters	2.33	9.40	6.91	3.4105	12.31	0.73	-0.33

Table 9 - Model Comparison for FELONY ASSAULT

Based on ME, MPE, MAPE and ACF1 values, Holt-Winters model is a better fit for FELONY ASSAULT data.

2.6. Target variable = BURGLARY

a. Building time series model for BURGLARY

```
> #time series BURGLARY
+ Sum_BURGLARY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(BURGLARY_TOTAL = sum(BURGLARY))
+
+ BURGLARY_ts = ts(Sum_BURGLARY$BURGLARY_TOTAL, frequency = 4, start = 2015)
+ plot(BURGLARY_ts, main = 'Time Series of BURGLARY variable')
```

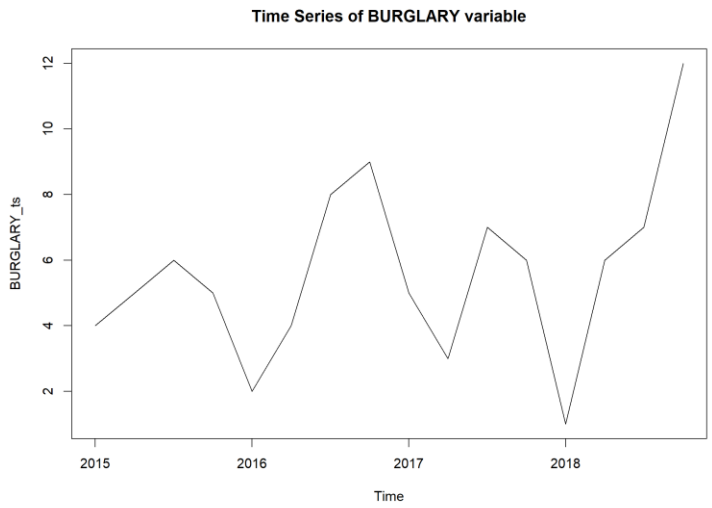


Figure 66 - Time Series of BURGLARY

❖ *Identifying seasonal patterns*

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(BURGLARY_ts ~ cycle(BURGLARY_ts), main = 'Seasonality of BURGLARY variable')
+
+ BURGLARY_ts_dc = decompose(BURGLARY_ts)
+ plot(BURGLARY_ts_dc)
```

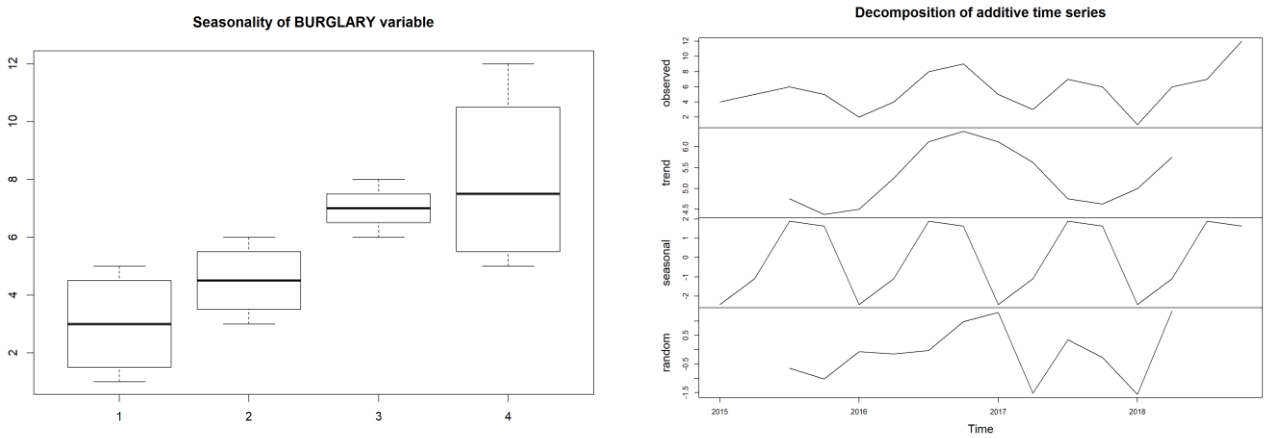


Figure 67 - Time Series Decompose for BURGLARY

The box-plot and the decomposed time-series plot of the BURGLARY variable indicate that there is a strong seasonal component. The trend component is not that significant. Because of the strong seasonal component we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

As expected the beta and alpha values are 0 but the gamma value is very small indicating the recent effects (seasonal) has some influence on the model.

```
> BURGLARY_hw = HoltWinters(BURGLARY_ts)
+ BURGLARY_hw
+ plot(BURGLARY_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = BURGLARY_ts)

Smoothing parameters:
 alpha: 0
 beta : 0
 gamma: 0.01553882

Coefficients:
         [,1]
a   6.2625000
b   0.1625000
s1 -2.0514068
s2 -0.7890159
s3  1.7257533
s4  1.1994365
```

Looking at the model plotted, it is apparent that the model is not very responsive to the actual data.

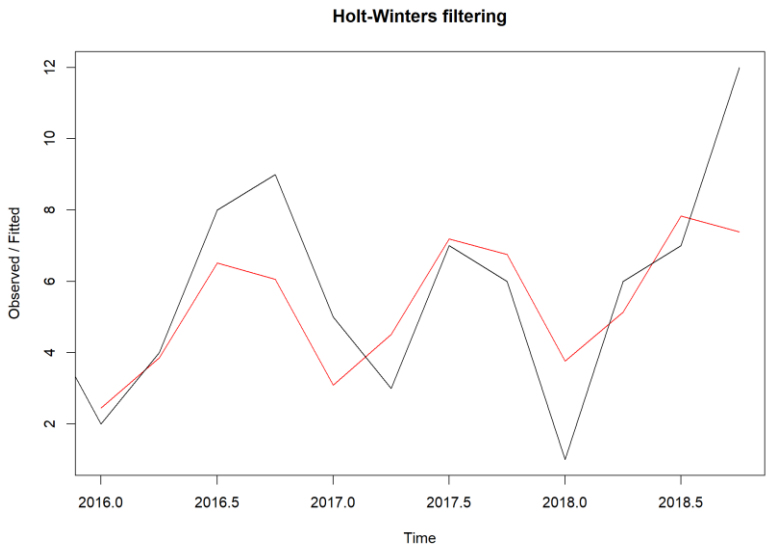


Figure 68 - Holt-Winters model plot for BURGLARY

❖ *Forecast*

The forecasted values for BURGLARY crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ BURGLARY_hw_fore = forecast:::forecast.HoltWinters(BURGLARY_hw)
+ BURGLARY_hw_fore
+ #Look at forecasted values
+ plot(BURGLARY_hw_fore)
+
        Point Forecast    Lo 80     Hi 80     Lo 95     Hi 95
2019 Q1       4.373593 1.767445  6.979742 0.3878332  8.359353
2019 Q2       5.798484 3.192336  8.404632 1.8127240  9.784244
2019 Q3       8.475753 5.869605 11.081902 4.4899932 12.461513
2019 Q4       8.111936 5.505788 10.718085 4.1261764 12.097697
2020 Q1       5.023593 2.417130  7.630056 1.0373520  9.009835
2020 Q2       6.448484 3.842021  9.054947 2.4622428 10.434725
2020 Q3       9.125753 6.519290 11.732216 5.1395120 13.111995
2020 Q4       8.761936 6.155473 11.368400 4.7756952 12.748178
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are wide which indicates the model is not good at forecasting.

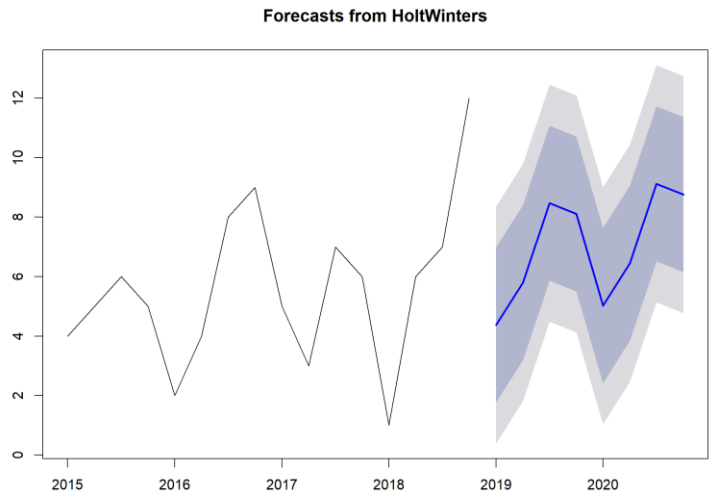
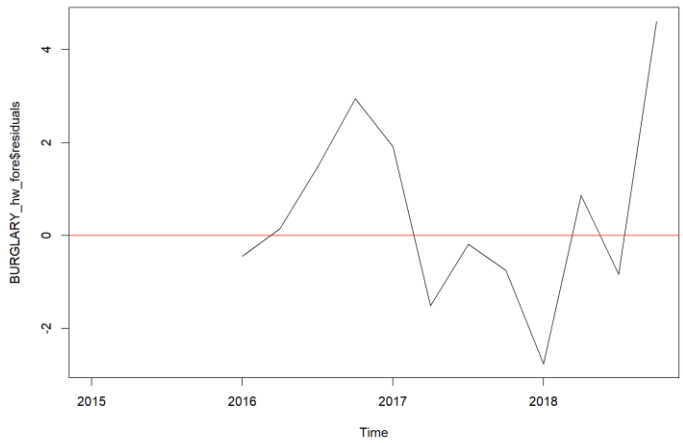


Figure 69 - Holt-Winters Forecast for BURGLARY

❖ *Testing Assumptions - Holt-Winters forecast*



To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

Figure 70 - Residual plot from Holt-Winters Forecast for BURGLARY

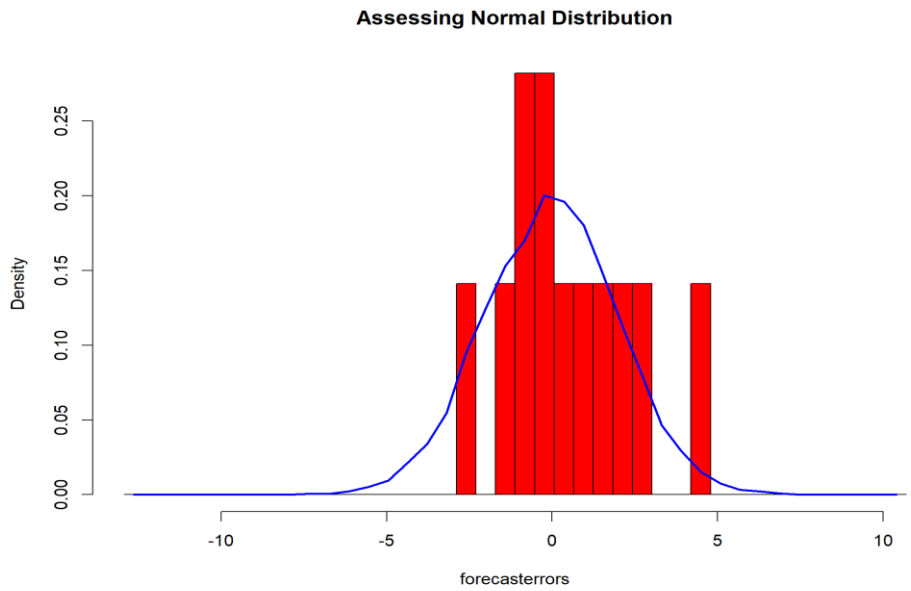


Figure 71 - Distribution of Forecast Errors from Holt-Winters model for BURGLARY

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> BURGLARY_trendcomp = BURGLARY_ts_dc$trend
+ BURGLARY_trend_data = data.frame(trend = c(BURGLARY_trendcomp), time = c(time(BURGLARY_trendcomp
+ BURGLARY_trend_reg = lm(BURGLARY_trend_data$trend ~ BURGLARY_trend_data$time)
+ summary(BURGLARY_trend_reg)

Call:
lm(formula = BURGLARY_trend_data$trend ~ BURGLARY_trend_data$time)

Residuals:
     Min       1Q   Median       3Q      Max
-0.80798 -0.61568 -0.08552  0.42126  1.12733

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              -368.4857   484.4549  -0.761    0.464
BURGLARY_trend_data$time    0.1853     0.2402   0.771    0.458

Residual standard error: 0.7181 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.05618,   Adjusted R-squared:  -0.0382
F-statistic: 0.5952 on 1 and 10 DF,  p-value: 0.4583
```

The p-value for the variable *time* is not significant, indicating the *trend* component does not influence the time series (i.e. time series is stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> BURGLARY_ts_trend = BURGLARY_ts - BURGLARY_ts_dc$seasonal
+ adf.test(BURGLARY_ts_trend, k = 5, alternative = "stationary")
+ kpss.test(BURGLARY_ts_trend)

    Augmented Dickey-Fuller Test

data:  BURGLARY_ts_trend
Dickey-Fuller = -9.0736, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(BURGLARY_ts_trend, k = 5, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  BURGLARY_ts_trend
KPSS Level = 0.18673, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(BURGLARY_ts_trend) : p-value greater than printed p-value
```

The p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Differencing is not required.

Note that the lag order (k) had to be 5 in order for this data to be stationary according to the ADF test(p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. One lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows no effects above 95% confidence interval indicating there are no partial correlations in this time series.

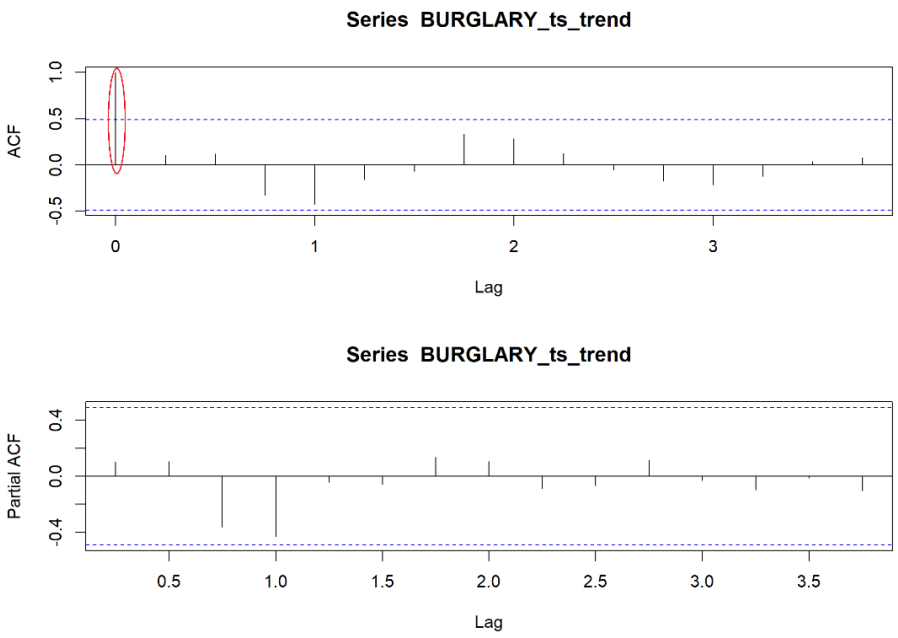


Figure 72 - ACF and PACF plots of ARMA model for BURGLARY

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARIMA(1,0,0)(1,0,0)[4], which is a seasonal random walk model with one autoregressive(AR) component , with an AIC of 62.67.

```
> BURGLARY_arima = auto.arima(BURGLARY_ts_trend)
+ BURGLARY_arima
+
Series: BURGLARY_ts_trend
ARIMA(1,0,0)(1,0,0)[4] with non-zero mean

Coefficients:
          ar1     sar1    mean
      -0.4859  -0.8583  5.5335
s.e.   0.4873   0.1212  0.1249

sigma^2 estimated as 1.556:  log likelihood=-27.33
AIC=62.67   AICc=66.31   BIC=65.76
```

❖ *Forecast*

The forecast using this model is given below. The results show that the model forecasts BURGLARY count in the Ney York City's Parks to be mostly 5 with most variability in quarter 4(1-8).

```
> BURGLARY_arima_fore = forecast(BURGLARY_arima)
+ BURGLARY_arima_fore
        Point Forecast      Lo 80     Hi 80     Lo 95     Hi 95
2019 Q1       5.482173  3.8836984  7.080647  3.037517  7.926828
2019 Q2       5.084531  3.3073513  6.861710  2.366569  7.802492
2019 Q3       5.465192  3.6483851  7.281999  2.686626  8.243759
2019 Q4       1.608103 -0.2179347  3.434140 -1.184581  4.400786
2020 Q1       5.474472  3.2428126  7.706132  2.061443  8.887502
2020 Q2       5.969043  3.6519564  8.286129  2.425365  9.512721
2020 Q3       5.567840  3.2310403  7.904639  1.994013  9.141666
2020 Q4       8.914689  6.5732595 11.256118  5.333782 12.495596
```

The 80% and 95% prediction intervals are wide for this model.

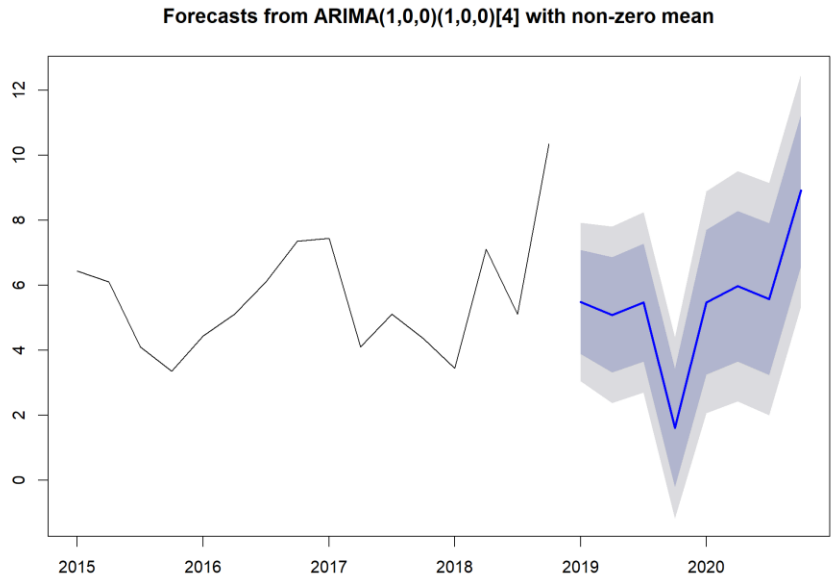


Figure 73 - Forecasts from ARIMA(1,0,0)(1,0,0)[4] for BURGLARY

b. Assess the Models for BURGLARY

We now had two time series models the Hot Winters model (α=0, β=0, γ=0.01) and the ARIMA(1,0,0)(1,0,0)[4] seasonal random walk model with AR(1). To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(BURGLARY_arima_fore)
+ accuracy(BURGLARY_hw_fore)
                      ME      RMSE       MAE        MPE     MAPE      MASE        ACF1
Training set -0.07650117 1.124297 0.8015304 -5.803997 14.7219 0.3206122 -0.0650499
                    ME      RMSE       MAE        MPE      MAPE      MASE        ACF1
Training set 0.4545985 1.999379 1.538159 -19.20611 43.52455 0.6152636 0.04378061
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA	-0.08	1.12	0.80	-5.804	14.72	0.32	-0.07
Holt-Winters	0.45	2.00	1.54	-19.206	43.52	0.62	0.04

Table 10 - Model Comparison for BURGLARY

Based on ME, RMSE, MAE, MAPE, MASE and ACF1 values, seasonal random walk model is a better fit for BURGLARY data.

2.7. Target variable = GRAND_LARCENY

a. Building time series model for GRAND_LARCENY

```
> #time series GRAND_LARCENY
+ Sum_GRAND_LARCENY <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(GRAND_LARCENY_TOTAL = sum(GRAND_LARCENY))
+
+ GRAND_LARCENY_ts = ts(Sum_GRAND_LARCENY$GRAND_LARCENY_TOTAL, frequency = 4, start = 2015)
+ plot(GRAND_LARCENY_ts, main = 'Time Series of GRAND_LARCENY variable')
```

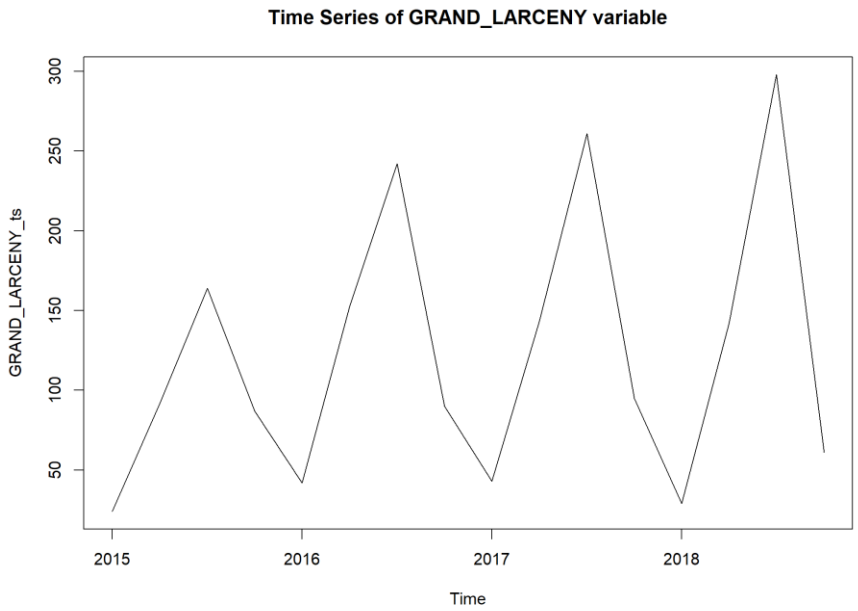


Figure 74 - Time Series of GRAND LARCENY

❖ Identifying seasonal patterns

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(GRAND_LARCENY_ts ~ cycle(GRAND_LARCENY_ts), main = 'Seasonality of GRAND_LARCENY variable')
+
+ GRAND_LARCENY_ts_dc = decompose(GRAND_LARCENY_ts)
+ plot(GRAND_LARCENY_ts_dc)
```

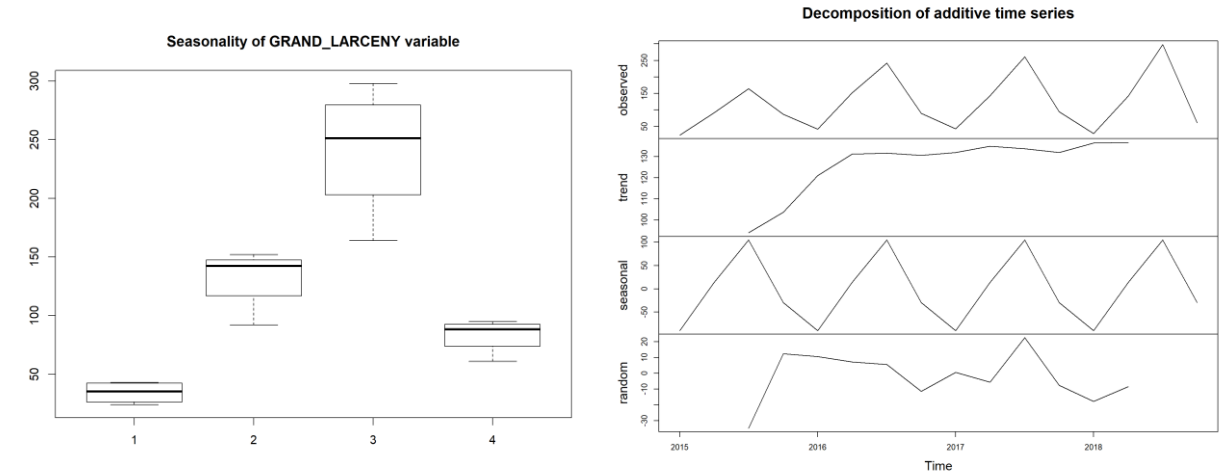


Figure 75 - Time Series Decompose for GRAND LARCENY

The box-plot and the decomposed time-series plot of the GRAND_LARCENY variable indicate that there is a strong seasonal component. The trend component also looks significant. Because of the strong seasonal component we used Holt-Winters Exponential Smoothing. Below are the code and results.

❖ *Holt-Winters Model and forecast*

As expected the beta and alpha values are small and the gamma value is very high indicating the recent effects (seasonal) has a lot of influence on the model.

```
> GRAND_LARCENY_hw = HoltWinters(GRAND_LARCENY_ts)
+ GRAND_LARCENY_hw
+ plot(GRAND_LARCENY_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = GRAND_LARCENY_ts)

Smoothing parameters:
 alpha: 0.2717869
 beta : 0.3400896
 gamma: 1

Coefficients:
          [,1]
a   156.2699724
b     0.4177355
s1 -125.5257726
s2  -11.2176753
s3  133.9688950
s4  -95.2699724
```

The plot shows that the model is slightly responsive to the actual data.

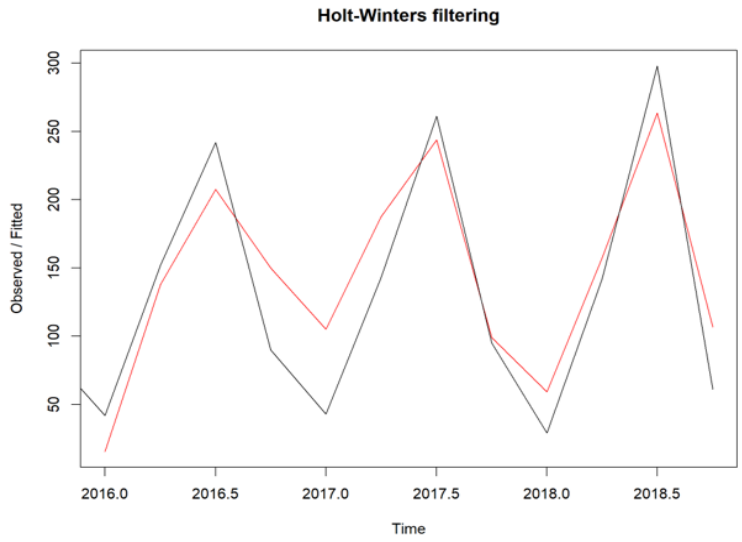


Figure 76 - Holt-Winters model plot for GRAND LARCENY

❖ *Forecast*

The forecasted values for GRAND_LARCENY crimes in New York City's parks are shown below.

```
> #Forecast the model beyond the known range of data
+ GRAND_LARCENY_hw_fore = forecast:::forecast.HoltWinters(GRAND_LARCENY_hw)
+ GRAND_LARCENY_hw_fore
+ #Look at forecasted values
+ plot(GRAND_LARCENY_hw_fore)
        Point Forecast      Lo 80     Hi 80       Lo 95    Hi 95
2019 Q1       31.16194 -15.745553  78.06942  -40.576878 102.9007
2019 Q2      145.88777  95.965873 195.80966   69.538818 222.2367
2019 Q3      291.49207 237.168723 345.81542  208.411679 374.5725
2019 Q4       62.67094   2.551043 122.79084  -29.274509 154.6164
2020 Q1       32.83288 -55.158750 120.82450 -101.738703 167.4045
2020 Q2      147.55871  53.071755 242.04566    3.053383 292.0640
2020 Q3      293.16302 191.033783 395.29225  136.969833 449.3562
2020 Q4       64.34188 -46.508963 175.19273 -105.189857 233.8736
```

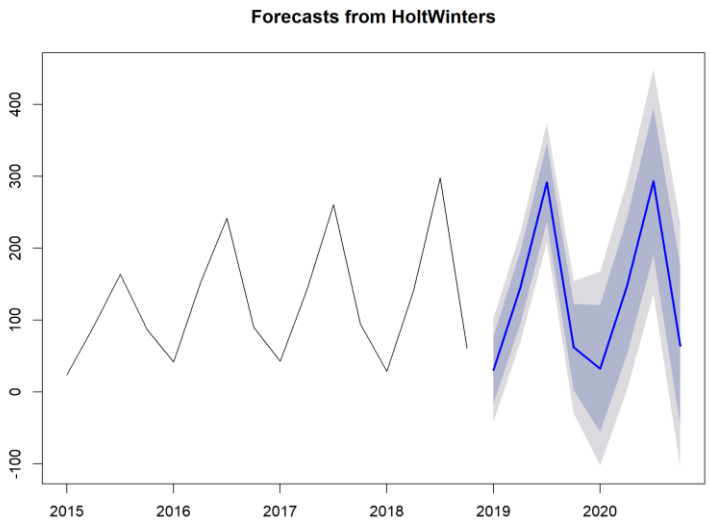


Figure 77 - Holt-Winters Forecast for GRAND LARCENY

The plot of the forecast reveals that the 80% and 95% prediction intervals are narrow which indicates the model is good at forecasting.

❖ *Testing Assumptions - Holt-Winters forecast*

To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

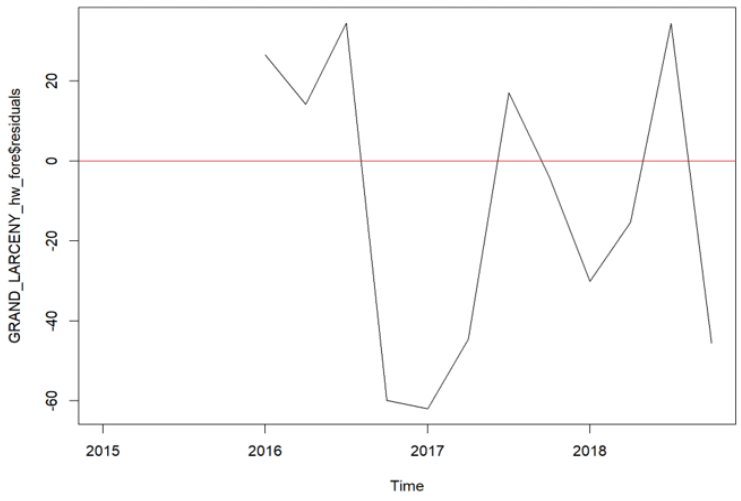


Figure 78 - Residual plot from Holt-Winters Forecast for GRAND LARCENY

Histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be chosen.

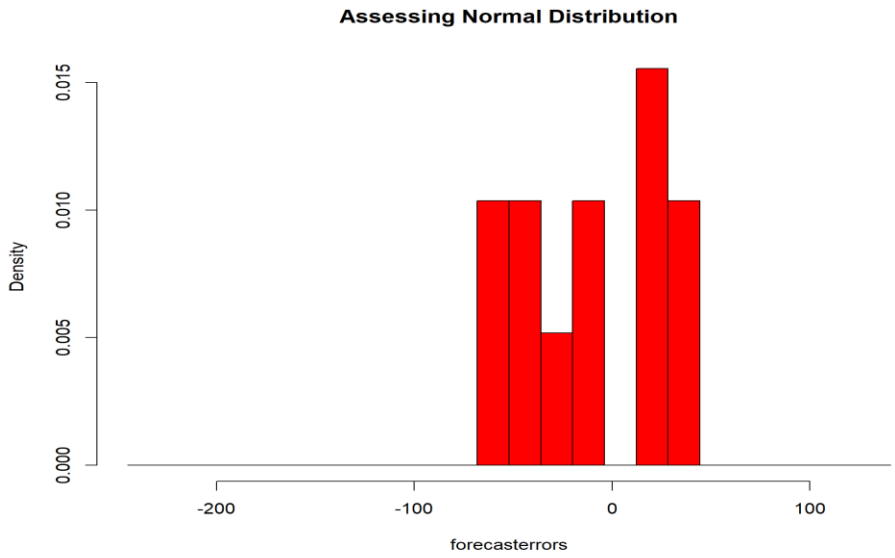


Figure 79 - Distribution of Forecast Errors from Holt-Winters model for GRAND LARCENY

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> GRAND_LARCENY_trendcomp = GRAND_LARCENY_ts_dc$trend
+ GRAND_LARCENY_trend_data = data.frame(trend = c(GRAND_LARCENY_trendcomp), time = c(time(GRAND_LARCENY_trendcomp)))
+ GRAND_LARCENY_trend_reg = lm(GRAND_LARCENY_trend_data$trend ~ GRAND_LARCENY_trend_data$time)
+ summary(GRAND_LARCENY_trend_reg)

Call:
lm(formula = GRAND_LARCENY_trend_data$trend ~ GRAND_LARCENY_trend_data$time)

Residuals:
    Min      1Q  Median      3Q     Max
-15.851  -5.461   1.801   5.246  12.210

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                 -24248.81    5828.28  -4.161  0.00195 **
GRAND_LARCENY_trend_data$time     12.09       2.89   4.182  0.00188 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.639 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.6362,    Adjusted R-squared:  0.5999
F-statistic: 17.49 on 1 and 10 DF,  p-value: 0.001881
```

The p-value for the variable *time* is significant, indicating the *trend* component influences the time series (i.e. time series is not stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> GRAND_LARCENY_ts_trend = GRAND_LARCENY_ts - GRAND_LARCENY_ts_dc$seasonal
+ adf.test(GRAND_LARCENY_ts_trend, k = 5, alternative = "stationary")
+ kpss.test(GRAND_LARCENY_ts_trend)
+

    Augmented Dickey-Fuller Test

data:  GRAND_LARCENY_ts_trend
Dickey-Fuller = -3.7095, Lag order = 5, p-value = 0.04218
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  GRAND_LARCENY_ts_trend
KPSS Level = 0.40238, Truncation lag parameter = 0, p-value = 0.07613
```

The p-value of the ADF test is significant and indicates the time series is stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Although the ADF

and KPSS tests state that the time series is stationary, the regression indicated that it was not hence differencing was performed. Below are the results of ADF and KPSS test after differencing.

```
> GRAND_LARCENY_ts_diff = diff(GRAND_LARCENY_ts_trend, differences = 4)
+ adf.test(GRAND_LARCENY_ts_diff, k = 0, alternative = "stationary")
+ kpss.test(GRAND_LARCENY_ts_diff)

    Augmented Dickey-Fuller Test

data:  GRAND_LARCENY_ts_diff
Dickey-Fuller = -7.9122, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(GRAND_LARCENY_ts_diff, k = 0, alternative = "stationary") :
  p-value smaller than printed p-value

    KPSS Test for Level Stationarity

data:  GRAND_LARCENY_ts_diff
KPSS Level = 0.084543, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(GRAND_LARCENY_ts_diff) : p-value greater than printed p-value
```

Note that the lag order (k) had to be 0 in order for this data to be stationary according to the ADF test(p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

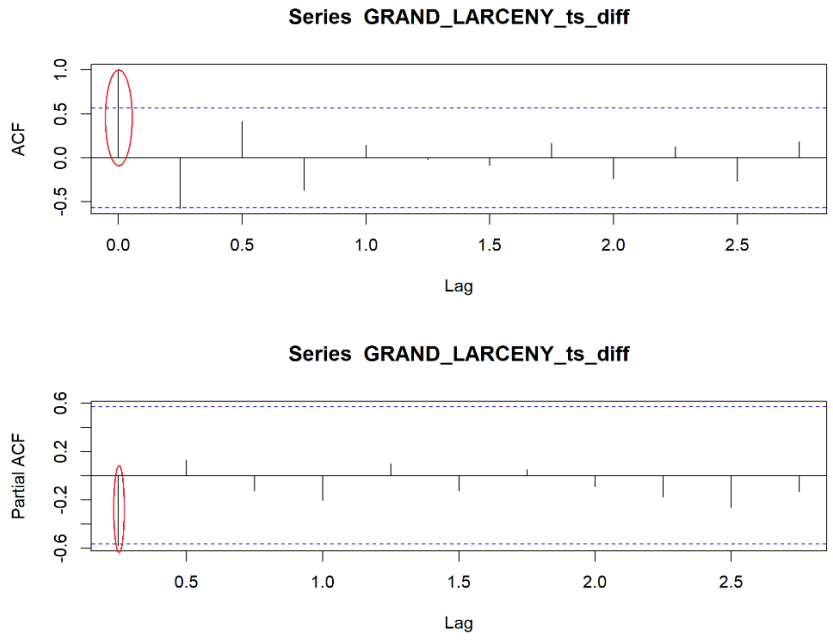


Figure 80 - ACF and PACF plots of ARMA model for GRAND LARCENY

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. One lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows one lag effect above 95% confidence interval indicating there are partial correlations in this time series.

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARMA(1,0) , with an AIC of 145.35.

```
> GRAND_LARCENY_arima = auto.arima(GRAND_LARCENY_ts_diff)
+ GRAND_LARCENY_arima
Series: GRAND_LARCENY_ts_diff
ARIMA(1,0,0) with zero mean

Coefficients:
          ar1
      -0.8746
s.e.   0.1488

sigma^2 estimated as 7390:  log likelihood=-70.68
AIC=145.35   AICc=146.69   BIC=146.32
```

❖ *Forecast*

The forecast using this model is given below. The results show that the model forecasts GRAND_LARCENY count in the Ney York City's Parks to be mostly 5 with most variability in quarter 4(1-8).

```
> GRAND_LARCENY_arima_fore = forecast(GRAND_LARCENY_arima)
+ GRAND_LARCENY_arima_fore
+ layout(1:1)
+ plot(GRAND_LARCENY_arima_fore)
        Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
2019 Q1      229.57111  119.399243 339.74298   61.07778 398.06444
2019 Q2     -200.77293 -347.133642 -54.41222 -424.61232  23.06646
2019 Q3      175.58729    6.702638 344.47195  -82.69950 433.87408
2019 Q4     -153.56103 -337.824128  30.70208 -435.36712 128.24507
2020 Q1      134.29781  -60.911523 329.50714 -164.24910 432.84472
2020 Q2     -117.45104 -320.634964  85.73289 -428.19404 193.29197
2020 Q3      102.71758 -106.360525 311.79568 -217.03979 422.47495
2020 Q4      -89.83234 -303.308790 123.64411 -416.31640 236.65172
```

The 80% and 95% prediction intervals are wide for this model.

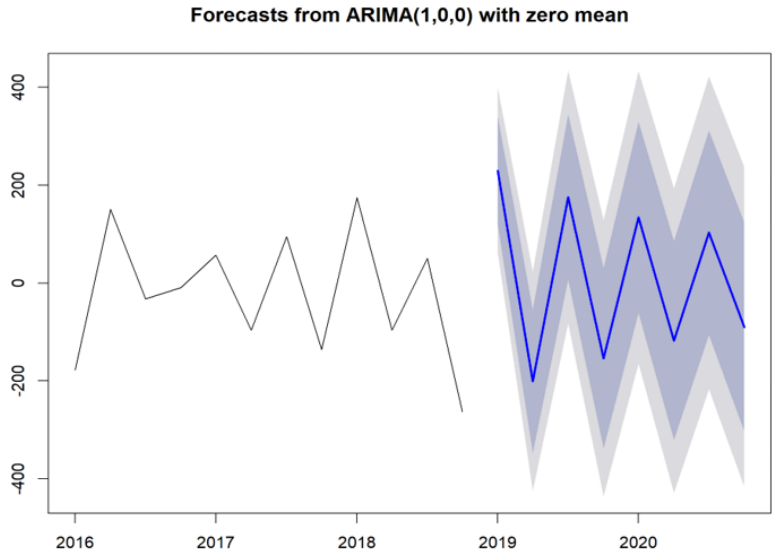


Figure 81 - Forecasts from ARIMA(1,0,0) for GRAND LARCENY

b. Assess the Models for GRAND_LARCENY

We now had two time series models the Hot Winters model (α=0.27, β=0.34, γ=1) and the ARMA(1,0). To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(GRAND_LARCENY_arima_fore)
+ accuracy(GRAND_LARCENY_hw_fore)
                     ME     RMSE      MAE      MPE     MAPE      MASE         ACF1
Training set -17.4478 82.30768 62.54055 26.15524 98.35366 0.4890757 -0.02534023
                     ME     RMSE      MAE      MPE     MAPE     MASE      ACF1
Training set -11.21976 36.79612 32.37885 -27.5488 45.06411 1.392639 0.1020618
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA	-17.45	82.31	62.54	26.155	98.35	0.49	-0.03
Holt-Winters	-11.22	36.80	32.38	-27.549	45.06	1.39	0.10

Table 11 - Model Comparison for GRAND LARCENY

Based on RMSE, MAE, MAPE and MPE values, Holt-Winters model is a better fit for GRAND LARCENY data.

2.8. Target variable = GRAND_LARCENY_MOTOR

a. Building time series model for GRAND_LARCENY_MOTOR

```
> #time series GRAND_LARCENY__MOTOR
+ Sum_GRAND_LARCENY__MOTOR <- NYC_Final_Data %>%
+     group_by(Year, Quarter) %>%
+     summarise(GRAND_LARCENY__MOTOR_TOTAL = sum(GRAND_LARCENY__MOTOR))
+
+ GRAND_LARCENY__MOTOR_ts = ts(Sum_GRAND_LARCENY__MOTOR$GRAND_LARCENY__MOTOR_TOTAL, frequency = 4, start = 2015)
+ plot(GRAND_LARCENY__MOTOR_ts, main = 'Time Series of GRAND_LARCENY__MOTOR variable')
```

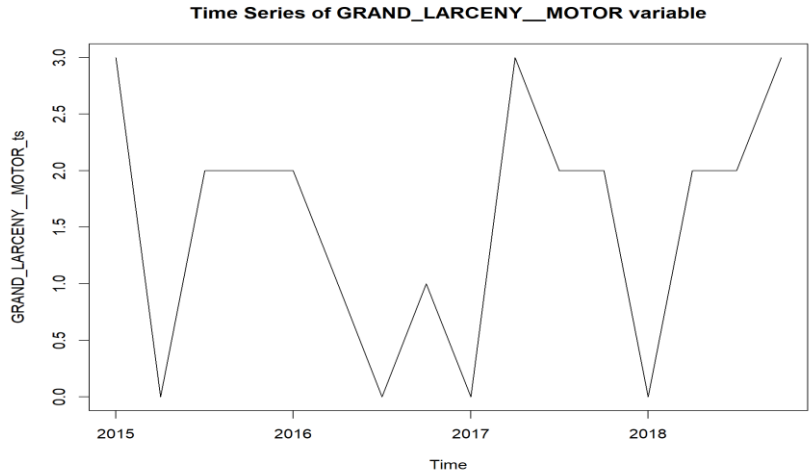
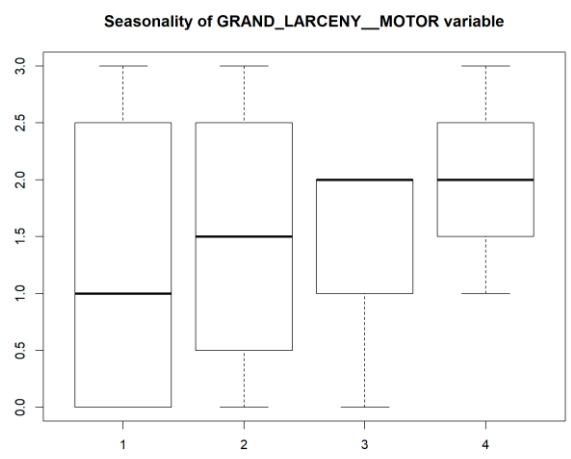


Figure 82 - Time Series of GRAND LARCENY MOTOR

❖ Identifying seasonal patterns

To look at the seasonality and trend we used a boxplot and decompose() function. The code and results are shown below.

```
> boxplot(GRAND_LARCENY__MOTOR_ts ~ cycle(GRAND_LARCENY__MOTOR_ts), main = 'Seasonality of GRAND_LARCENY__MOTOR variable')
+
+ GRAND_LARCENY__MOTOR_ts_dc = decompose(GRAND_LARCENY__MOTOR_ts)
+ plot(GRAND_LARCENY__MOTOR_ts_dc)
```



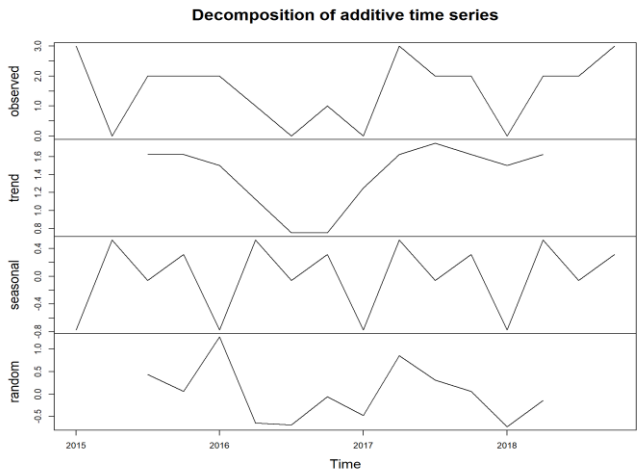


Figure 83 - Time Series Decompose for GRAND LARCENY MOTOR

The box-plot and the decomposed time-series PLOT of the GRAND_LARCENY__MOTOR variable indicate that there is a weak seasonal component. The trend component also does not look significant.

❖ *Holt-Winters Model and forecast*

As expected alpha, beta and gamma values is very weak indicating the recent effects and trend effects do not have much influence on the model.

```
> GRAND_LARCENY__MOTOR_hw = HoltWinters(GRAND_LARCENY__MOTOR_ts)
+ GRAND_LARCENY__MOTOR_hw
+ plot(GRAND_LARCENY__MOTOR_hw)
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = GRAND_LARCENY__MOTOR_ts)

Smoothing parameters:
 alpha: 0.180953
 beta : 0.3674035
 gamma: 0.383604

Coefficients:
         [,1]
a   1.5430064
b   0.1626438
s1 -0.3952283
s2  0.7193943
s3  0.4045632
s4  0.8078628
```

Looking at the model plotted, it is apparent that the model is not responsive to the actual data.

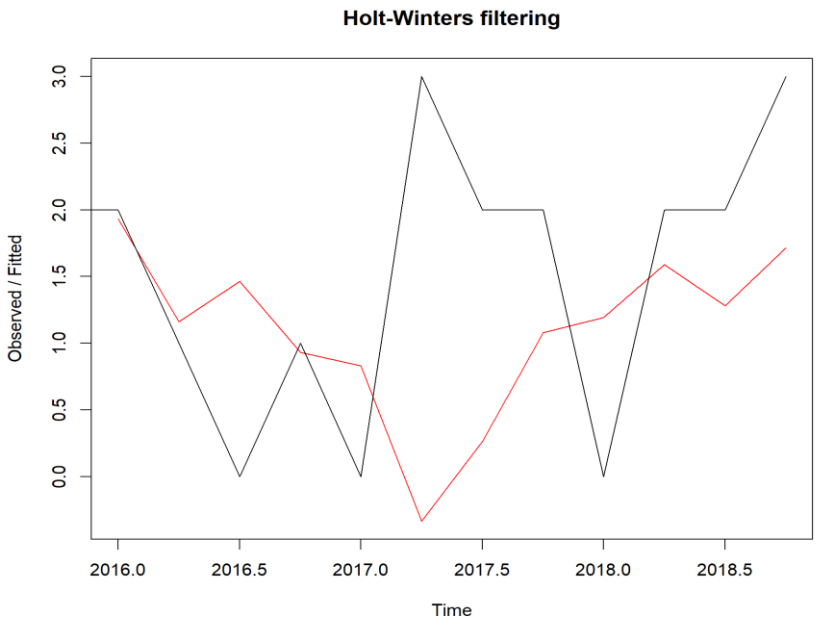


Figure 84 - Holt-Winters model plot for GRAND LARCENY MOTOR

❖ *Forecast*

The forecasted values for GRAND_LARCENY__MOTOR crimes in New York City's parks are shown below.

```
+ GRAND_LARCENY__MOTOR_hw_fore = forecast:::forecast.HoltWinters(GRAND_LARCENY__MOTOR_hw)
+ GRAND_LARCENY__MOTOR_hw_fore
+ #Look at forecasted values
+ plot(GRAND_LARCENY__MOTOR_hw_fore)
        Point Forecast      Lo 80    Hi 80       Lo 95    Hi 95
2019 Q1       1.310422 -0.4002801 3.021124 -1.30587104 3.926715
2019 Q2       2.587688  0.8253959 4.349981 -0.10750535 5.282882
2019 Q3       2.435501  0.5932019 4.277800 -0.38205233 5.253054
2019 Q4       3.001445  1.0475905 4.955299  0.01328264 5.989606
2020 Q1       1.960997 -0.3869107 4.308905 -1.62981817 5.551813
2020 Q2       3.238264  0.7314854 5.745042 -0.59552294 7.072050
2020 Q3       3.086076  0.3901718 5.781981 -1.03695388 7.209107
2020 Q4       3.652020  0.7381850 6.565855 -0.80430598 8.108346
```

The plot of the forecast reveals that the 80% and 95% prediction intervals are wide which indicates the model is not good at forecasting.

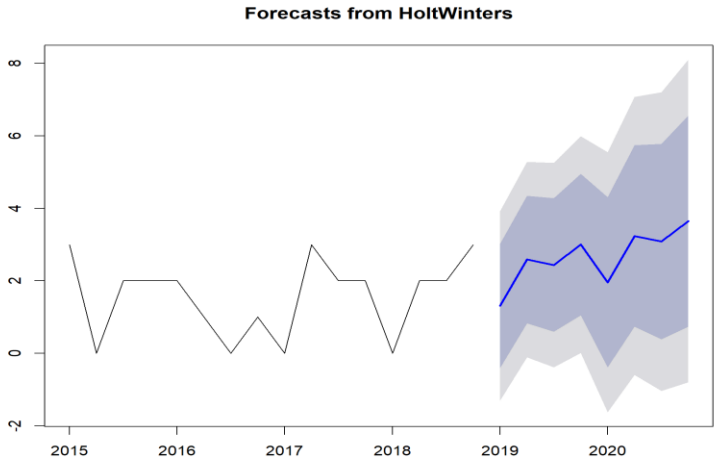
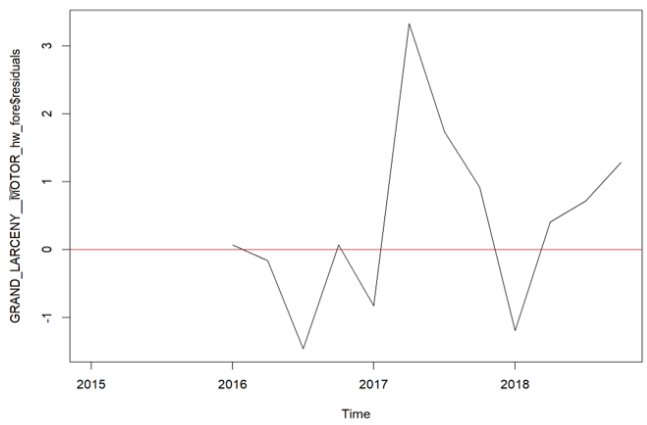


Figure 85 - Holt-Winters Forecast for GRAND LARCENY MOTOR

❖ *Testing Assumptions - Holt-Winters forecast*



To check for constant variance, we plotted the residuals from the forecasted model. The data does not exhibit homoscedasticity, but reveals heteroscedasticity, or non-constant variance. The data is not spread consistently about the mean (red line).

Figure 86 - Residual plot from Holt-Winters Forecast for GRAND LARCENY MOTOR

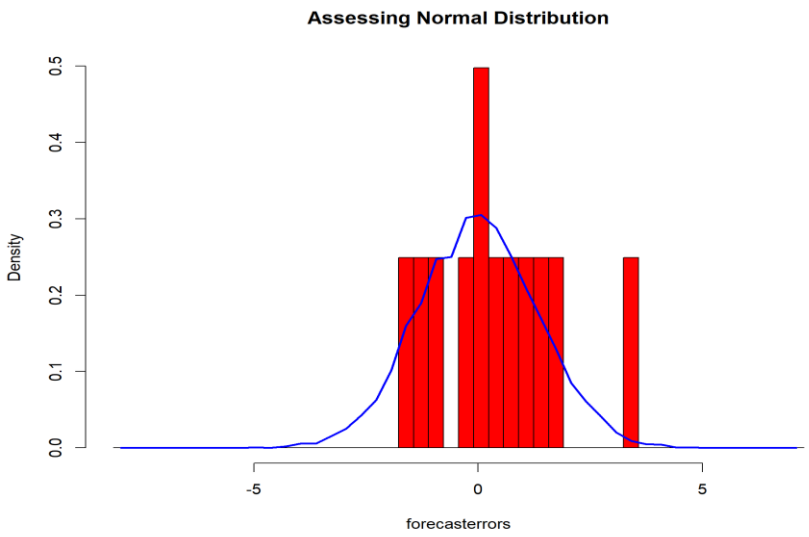


Figure 87 - Distribution of Forecast Errors from Holt-Winters model for GRAND LARCENY MOTOR

The histogram of the residual distribution reveals that the errors do not come close to forming a normal distribution. Based on these two assessments perhaps a different model should be selected.

❖ *ARMA model and forecast*

To generate the ARMA model we first need to check whether the time series is stationary. To check for stationarity we converted the time series into a data frame and then used the function lm() to create the regression function. Below are the results of the regression.

```
> GRAND_LARCENY__MOTOR_trendcomp = GRAND_LARCENY__MOTOR_ts_dc$trend
+ GRAND_LARCENY__MOTOR_trend_data = data.frame(trend = c(GRAND_LARCENY__MOTOR_trendcomp), time = c(time(GRAND_LARCENY__MOTOR_trendcomp)))
+ GRAND_LARCENY__MOTOR_trend_reg = lm(GRAND_LARCENY__MOTOR_trend_data$trend ~ GRAND_LARCENY__MOTOR_trend_data$time)
+ summary(GRAND_LARCENY__MOTOR_trend_reg)
+

Call:
lm(formula = GRAND_LARCENY__MOTOR_trend_data$trend ~ GRAND_LARCENY__MOTOR_trend_data$time)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6353 -0.1718  0.1348  0.2237  0.3446

Coefficients:
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                      -167.85242  240.95117  -0.697    0.502
GRAND_LARCENY__MOTOR_trend_data$time    0.08392    0.11947   0.702    0.498

Residual standard error: 0.3572 on 10 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.04702,   Adjusted R-squared:  -0.04828
F-statistic: 0.4934 on 1 and 10 DF,  p-value: 0.4984
```

The p-value for the variable *time* is not significant, indicating the *trend* component does not influence the time series (i.e. time series is stationary).

We also performed ADF and KPSS test to check stationarity. The output is shown below.

```
> GRAND_LARCENY__MOTOR_ts_trend = GRAND_LARCENY__MOTOR_ts - GRAND_LARCENY__MOTOR_ts_dc$seasonal
+ adf.test(GRAND_LARCENY__MOTOR_ts_trend, k = 3, alternative = "stationary")
+ kpss.test(GRAND_LARCENY__MOTOR_ts_trend)

    Augmented Dickey-Fuller Test

data:  GRAND_LARCENY__MOTOR_ts_trend
Dickey-Fuller = -2.3677, Lag order = 3, p-value = 0.4323
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  GRAND_LARCENY__MOTOR_ts_trend
KPSS Level = 0.091513, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(GRAND_LARCENY__MOTOR_ts_trend) :
  p-value greater than printed p-value
```

The p-value of the ADF test is not significant and indicates the time series is not stationary. The p-value of the KPSS test is not significant indicating the time series data is stationary. Below are the results of ADF and KPSS test after differencing.

```
> GRAND_LARCENY__MOTOR_ts_diff =
+     diff(GRAND_LARCENY__MOTOR_ts_trend, differences = 2)
+ adf.test(GRAND_LARCENY__MOTOR_ts_diff, k = 3, alternative = "stationary")
+ kpss.test(GRAND_LARCENY__MOTOR_ts_diff)

    Augmented Dickey-Fuller Test

data:  GRAND_LARCENY__MOTOR_ts_diff
Dickey-Fuller = -3.957, Lag order = 3, p-value = 0.02475
alternative hypothesis: stationary


    KPSS Test for Level Stationarity

data:  GRAND_LARCENY__MOTOR_ts_diff
KPSS Level = 0.086067, Truncation lag parameter = 0, p-value = 0.1

Warning message:
In kpss.test(GRAND_LARCENY__MOTOR_ts_diff) :
  p-value greater than printed p-value
```

Note that the lag order (k) had to be at most 3 in order for this data to be stationary according to the ADF test(p-value significant). All other lag orders resulted in an error or a non-significant p-value.

❖ *ACF & PACF*

The ACF of the differenced time series presents an abrupt end to the lag effects greater than the 95% confidence interval, which may indicate the model has at least a single moving average component. One lag effect is above 95% confidence interval which indicates at least one autocorrelation component. The PACF shows no lag effects above 95% confidence interval indicating there are no partial correlations in this time series.

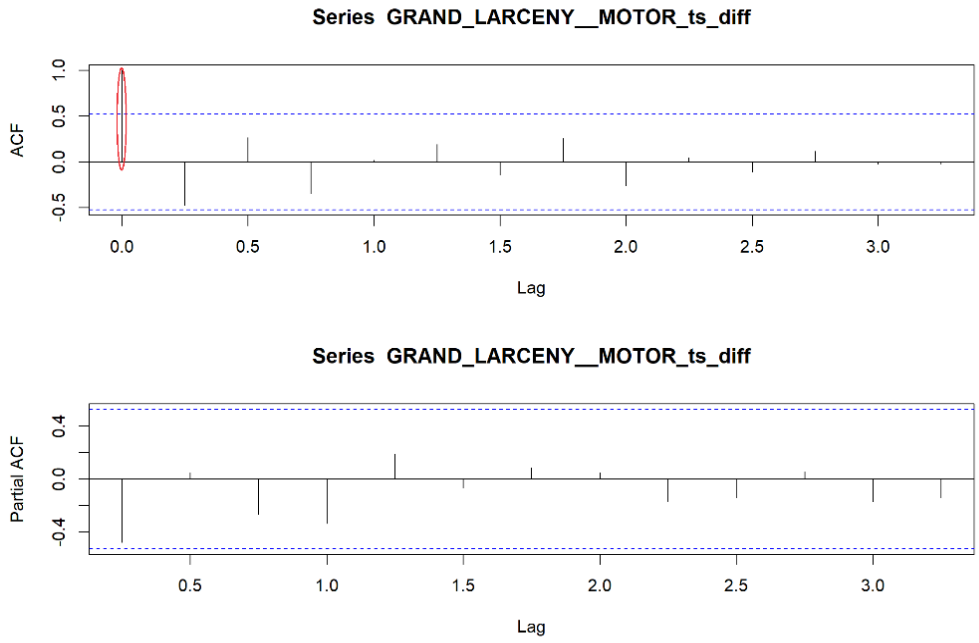


Figure 88 - ACF and PACF plots of ARMA model for GRAND LARCENY MOTOR

❖ *ARMA model*

We chose to run auto.arima() to get the model. This resulted in ARMA(1,0) , with an AIC of 60.84.

```
> GRAND_LARCENY__MOTOR_arima = auto.arima(GRAND_LARCENY__MOTOR_ts_diff)
+ GRAND_LARCENY__MOTOR_arima
+
Series: GRAND_LARCENY__MOTOR_ts_diff
ARIMA(1,0,0) with zero mean

Coefficients:
          ar1
      -0.8385
s.e.   0.1823

sigma^2 estimated as 3.353:  log likelihood=-28.42
AIC=60.84   AICc=61.93   BIC=62.12
```

❖ *Forecast*

The forecast using this model is given below.

```
> GRAND_LARCENY__MOTOR_arima_fore = forecast(GRAND_LARCENY__MOTOR_arima)
+ GRAND_LARCENY__MOTOR_arima_fore
+ layout(1:1)
+ plot(GRAND_LARCENY__MOTOR_arima_fore)
        Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
2019 Q1    -0.03493695 -2.381484 2.311610 -3.623672 3.553798
2019 Q2     0.02929418 -3.032982 3.091570 -4.654053 4.712641
2019 Q3    -0.02456278 -3.502961 3.453836 -5.344314 5.295189
2019 Q4     0.02059557 -3.722771 3.763962 -5.704389 5.745580
2020 Q1    -0.01726912 -3.936213 3.901675 -6.010777 5.976238
2020 Q2     0.01447993 -4.023337 4.052297 -6.160829 6.189789
2020 Q3    -0.01214123 -4.131481 4.107198 -6.312127 6.287845
2020 Q4     0.01018026 -4.165521 4.185882 -6.376004 6.396365
```

The 80% and 95% prediction intervals are wide for this model, which indicates it is not a good model.

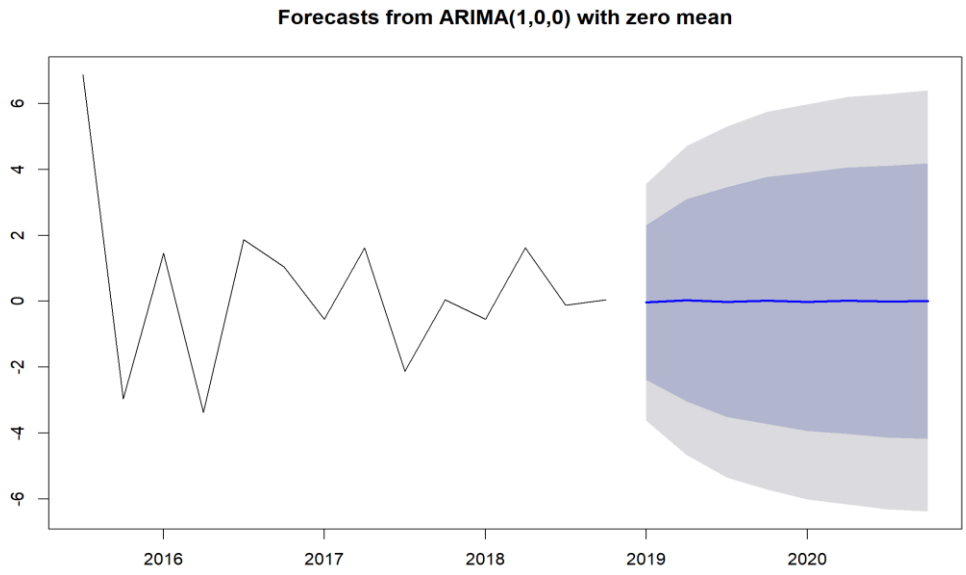


Figure 89 - Forecasts from ARIMA(1,0,0) for GRAND LARCENY MOTOR

b. Assess the Models for GRAND_LARCENY_MOTOR

We now had two time series models for GRAND_LARCENY_MOTOR: the Hot Winters model (α=0.18, β=0.36, γ=0.38) and the ARMA(1,0). To compare the two models we chose to look at their accuracy measures. The code and results are shown below.

```
> accuracy(GRAND_LARCENY__MOTOR_arima_fore)
+ accuracy(GRAND_LARCENY__MOTOR_hw_fore)
                     ME     RMSE      MAE       MPE     MAPE     MASE      ACF1
Training set 0.4196877 1.764416 1.448508 -353.7335 445.5507 0.629786 0.2834863
                     ME     RMSE      MAE  MPE MAPE      MASE      ACF1
Training set 0.4075542 1.341448 1.015342 -Inf  Inf 0.8702935 0.1379996
```

The values have been summarised in the table below.

Model	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA	0.42	1.76	1.45	-353.73	445.55	0.63	0.28
Holt-Winters	0.41	1.34	1.02	-Inf	Inf	0.87	0.14

Table 12 - Model Comparison for GRAND LARCENY MOTOR

Based on ME, RMSE, MAE, MPE and ACF1 values, Holt-Winters model is a better fit for GRAND LARCENY MOTOR data.

Conclusion

The final step is to compare and decide which model should be selected to predict the primary target variable **TOTAL** based on different modeling techniques' results and given the project's goals and objectives.

Some of the quantifiable means to compare a Linear Regression model to a Time Series model could be either AIC, BIC, or Root Mean Square Error (RMSE) since time series model, unlike regression model, does not provide us with an adjusted R-square. A comparison table for these above statistics (calculated in the previous modeling part) is summarized as below:

	AIC	BIC	RMSE
Linear Regression Model	47,936	47,991	0.886
Time Series Model (Holt-Winters)	163	168	47.67

It can be seen that Time Series Model is likely to subject to more errors as compared to Linear Regression Model in predicting TOTAL. This is understandable since Linear Regression has more records to train in building the model as compared to Time Series model.

Regarding AIC and BIC, however, Time Series Model appears to have both lower AIC and BIC, thus, overall Holt-Winters model is likely to be better performing than Linear Regression model.

Additionally, as previously mentioned, one of the main drawbacks of Linear Regression is that it assumes the relationship between the target and dependent variables to be linear only. Meanwhile, there could exist some non-linear relationship between these variables as well which are not considered in Linear Regression model. Besides, the range of the target variable's predicted values are restricted between 0-71 since values greater than 25% of the target range are not predicted properly by Linear Regression. However, there is no such restriction in Time Series Model.

Moreover, given our project's primary objective of predicting the number of crimes in 2019 based on historical time-ordered data in 2014-2018, a time series model would be reasonably appropriate to better reflect the past trend of crimes as well as more accurately forecast the expected number of crimes in the coming years. More importantly, our exploratory analysis indicates that trend and seasonality are heavily apparent in our crime data, hence, a Holt-Winters Exponential Smoothing model is a better fit model since it takes into account the seasonality effect, which is not considered by Linear Regression model.

To sum up, based on the above model assessments and given the project's ultimate goals and objectives, Holt-Winters Exponential Smoothing Model is selected as the best fit model to predict the number of crimes in New York City's Parks.