Time Series Forecasting: Buffalo Crime Data

Andrew Bialy, Ryan Glasser, & Rishabh Verma

University at Buffalo
School of Engineering and Applied Sciences

Data Acquisition

- Open Data Buffalo repository of datasets: Can be accessed at https://data.buffalony.gov/
    - For this presentation, the specific dataset used is “Crime Incidents”, which contains a record of crimes dating back decades
        - Updated daily as crime occurs

Updated daily as crime occurs

Column Names:

```
 [1] "Case Number"           "Incident Datetime"          "Incident ID"              "Incident Type Primary"     "Incident Description"      "Parent Incident Type"
 [7] "Hour of Day"           "Day of Week"                "Address"                  "City"                      "State"                     "Location"
[13] "Latitude"              "Longitude"                  "Created At"               "updated_at"                "2010 Census Tract"         "2010 Census Block Group"
[19] "2010 Census Block"     "Census Tract"               "Census Block"             "Census Block Group"        "Neighborhood"              "Police District"
[25] "Council District"      "TRACTCE20"                  "GEOID20_tract"            "GEOID20_blockgroup"        "GEOID20_block"             "Zip Codes"
[31] "Tracts"                "Block Groups"               "Blocks"                   "Neighborhoods"             "Council Districts"         "Police Districts"
[37] "Census Tracts 2020"    "Council Districts (2011)"   "Police Districts A-E"     "Census Block Groups 2020"  "Opportunity Zones 2021"
```

Data Cleaning

```
[1] "Incident Datetime"     "Parent Incident Type" "Hour of Day"          "Day of Week"
[5] "Neighborhood"          "Police District"      "Council District"     "Neighborhoods"
```

1. Select columns of the data set that are of particular interest for analyses

1. Convert dates to a format of interest

1. Limit data to crime incidents taking place between November 2012 - October 2022

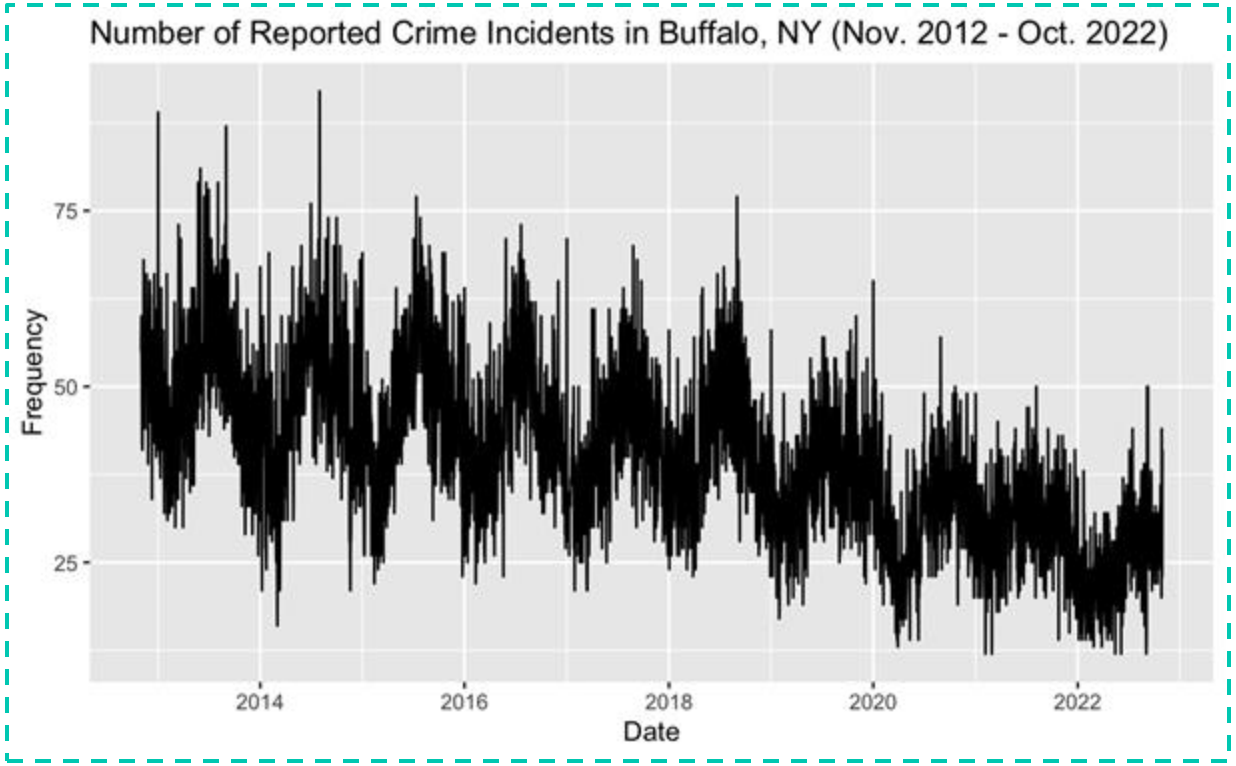
1. Create new dataframe displaying the number of incidents on each day

1. Convert to tsibble for time series analysis

	frequency	Date
1	55	2012-11-01
2	60	2012-11-02
3	54	2012-11-03
4	51	2012-11-04
5	42	2012-11-05
6	41	2012-11-06

Time Series Plot

- Decreasing trend over the 10 year period
- What can the increase in 2022 be attributed to?
    - Return to normalcy after COVID-19 restrictions?
    - Or is it just another increase like in past years?



Collapse into Monthly Data

Motivation: In determining sustainable policy, government officials may be more interested in crime rates on a monthly basis than a daily basis

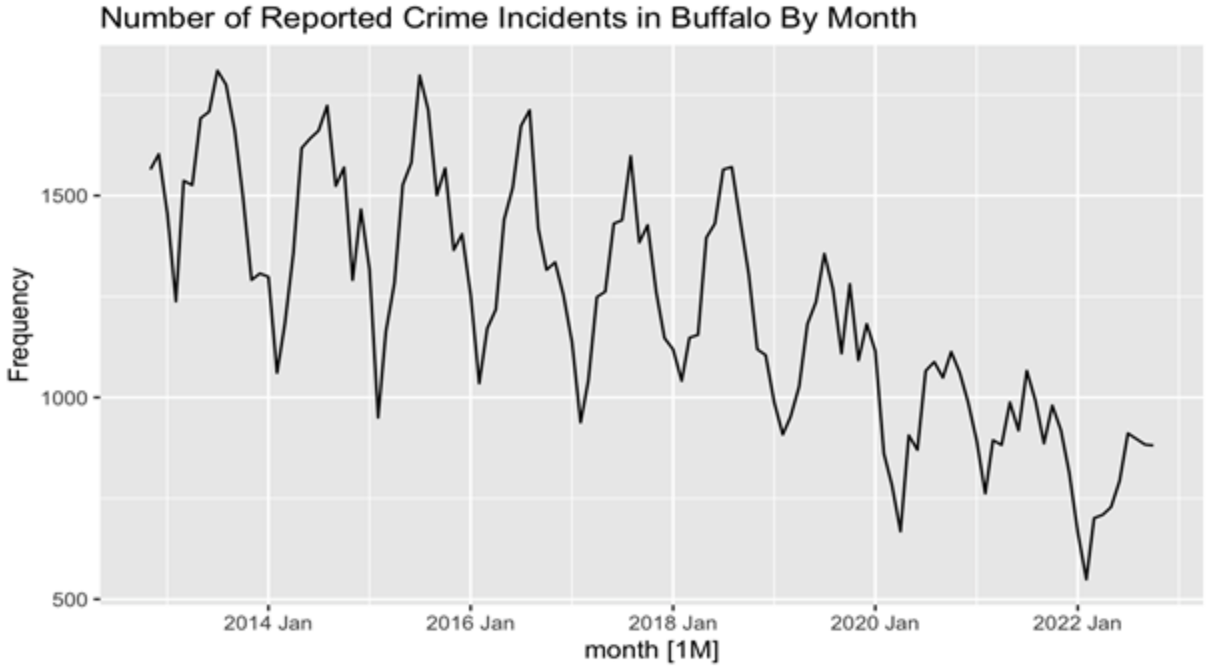
1. Sum the incidents for each month of data

1. Convert to tsibble for time series analysis

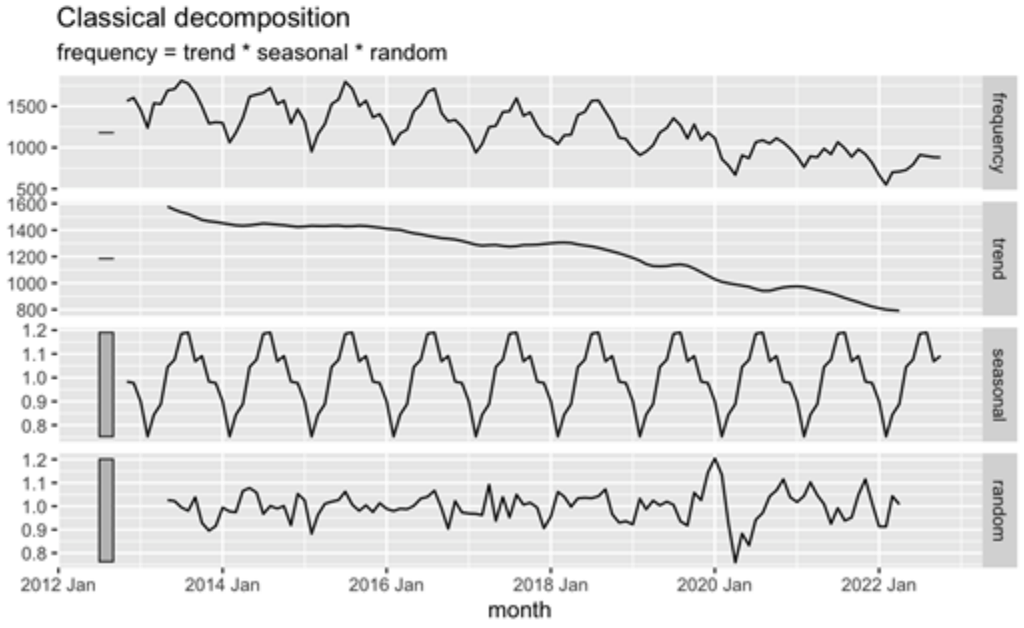
	month	frequency
1	2012 Nov	1565
2	2012 Dec	1603
3	2013 Jan	1456
4	2013 Feb	1238
5	2013 Mar	1537
6	2013 Apr	1526

Time Series Plot

- Decreasing trend over the 10 year period
- Classical Decomposition displays a degree of seasonality in the data

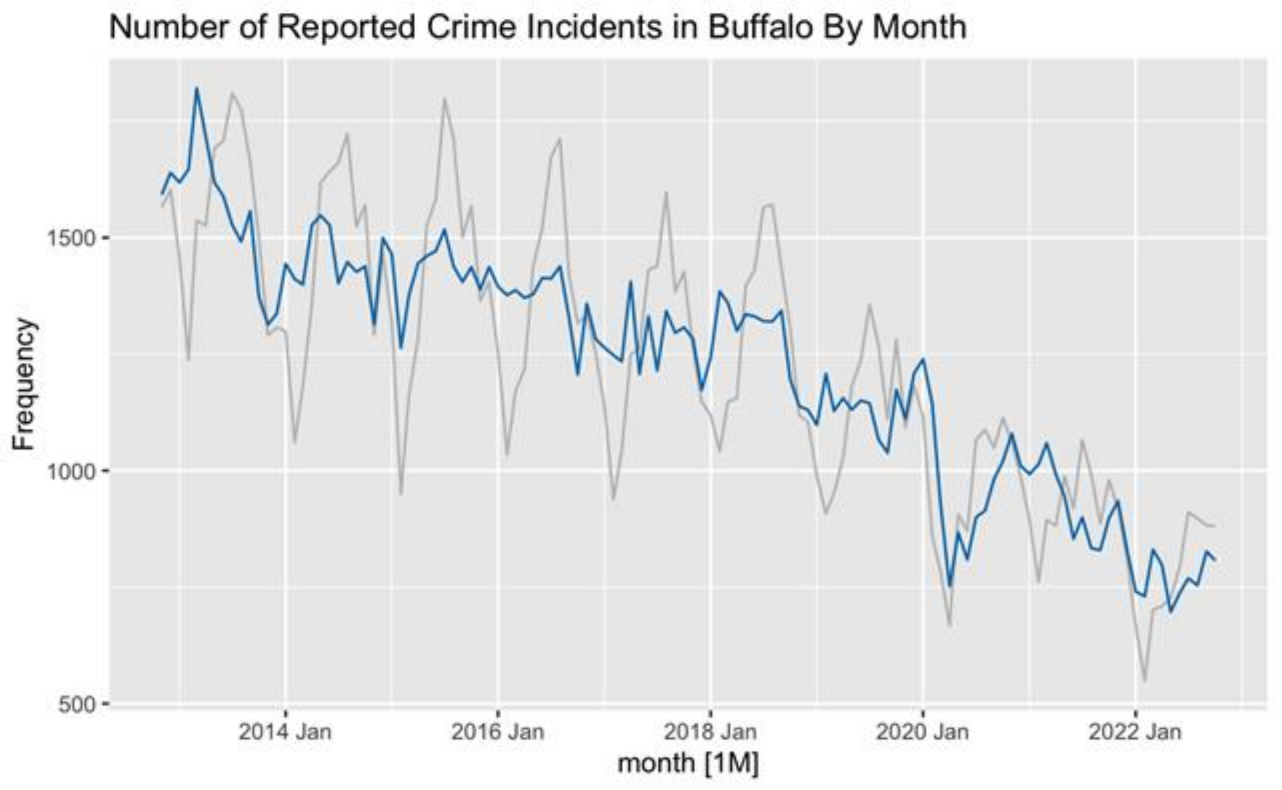


Classical Decomposition: Multiplicative



Seasonally Adjusted Data

- Aids government in examining a particular issue and may facilitate policy and action
  - May provide city officials with valuable information that can be used to fight crime in Buffalo



Seasonally Adjusted Data Shown in Blue

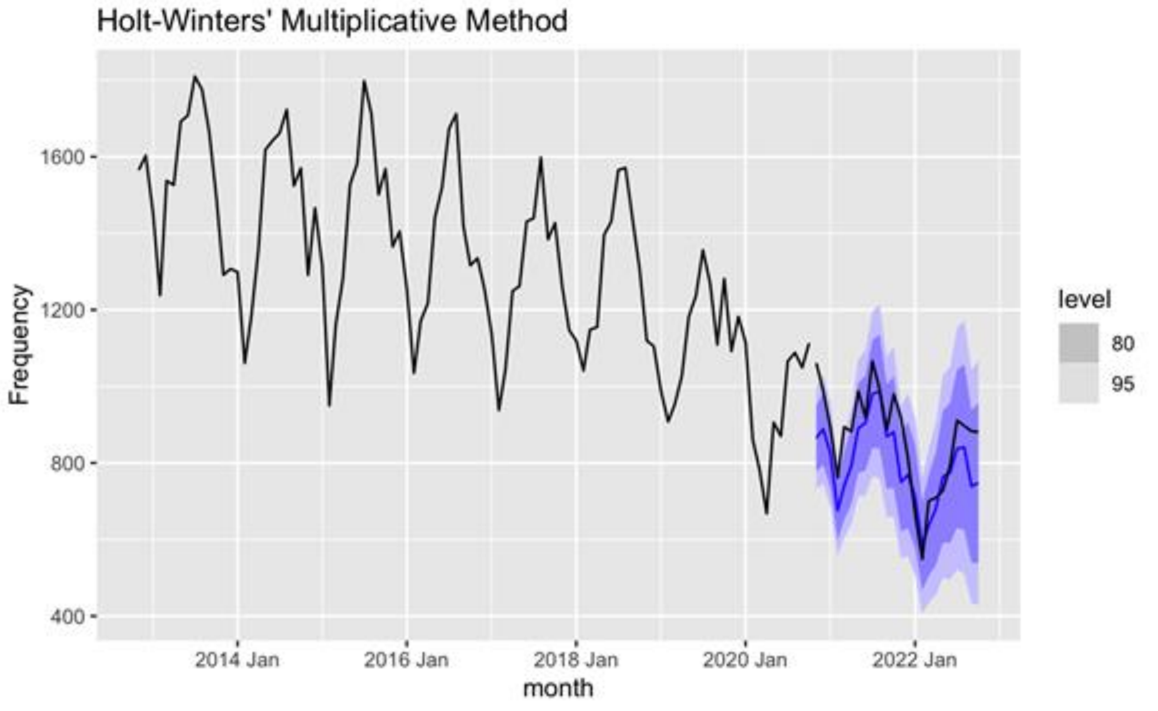
Accuracy Evaluation

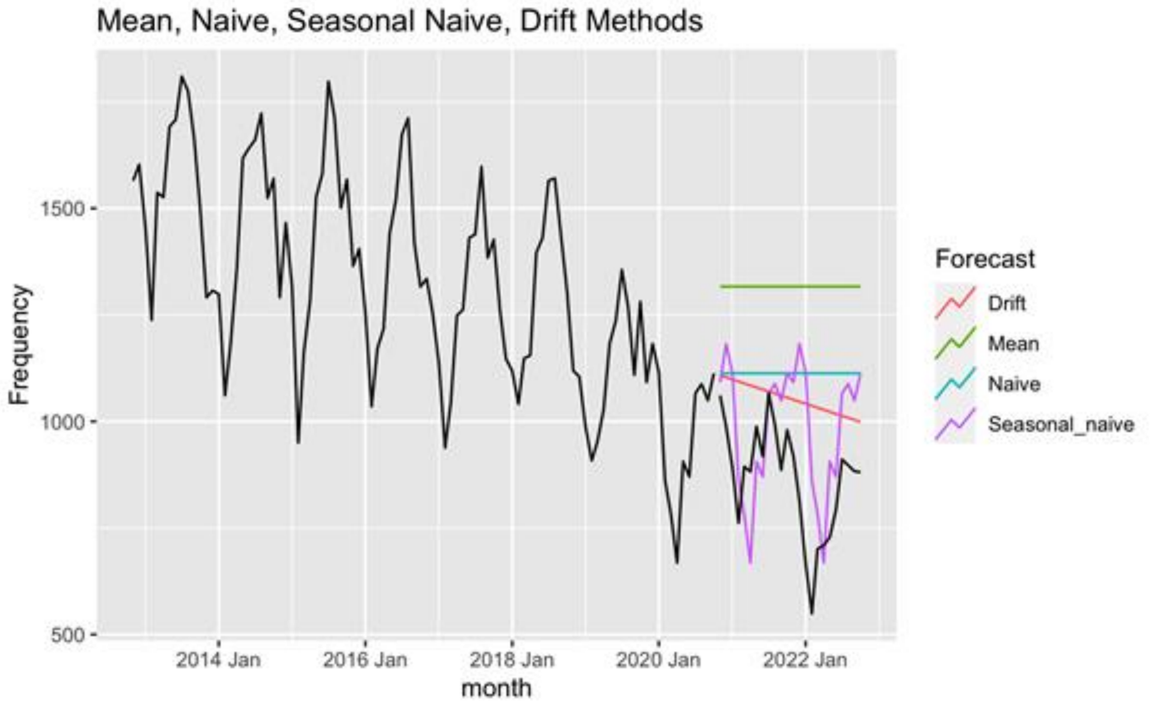
- Train-test split the data set to see which type of model yields the highest accuracy and lowest test error

The following methods will be used to create a fit on the training set

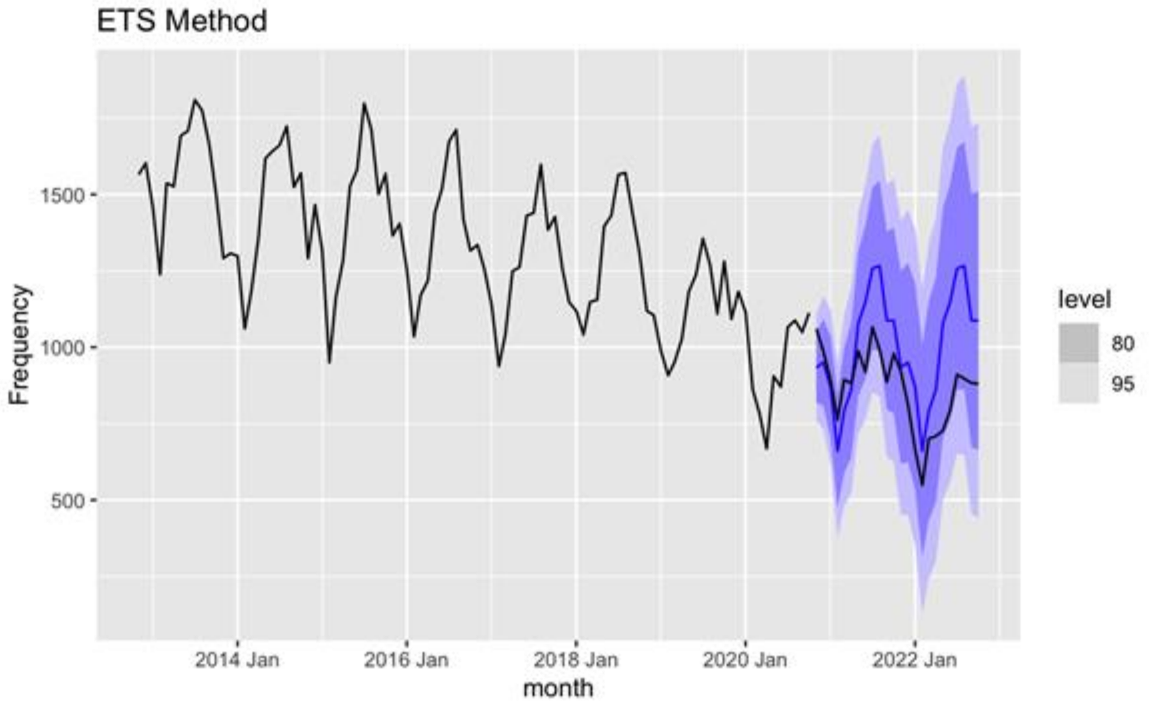
1. Holt-Winters' Multiplicative Method
2. Mean
3. Naive
4. Seasonal Naive
5. Drift
6. ETS
7. ARIMA

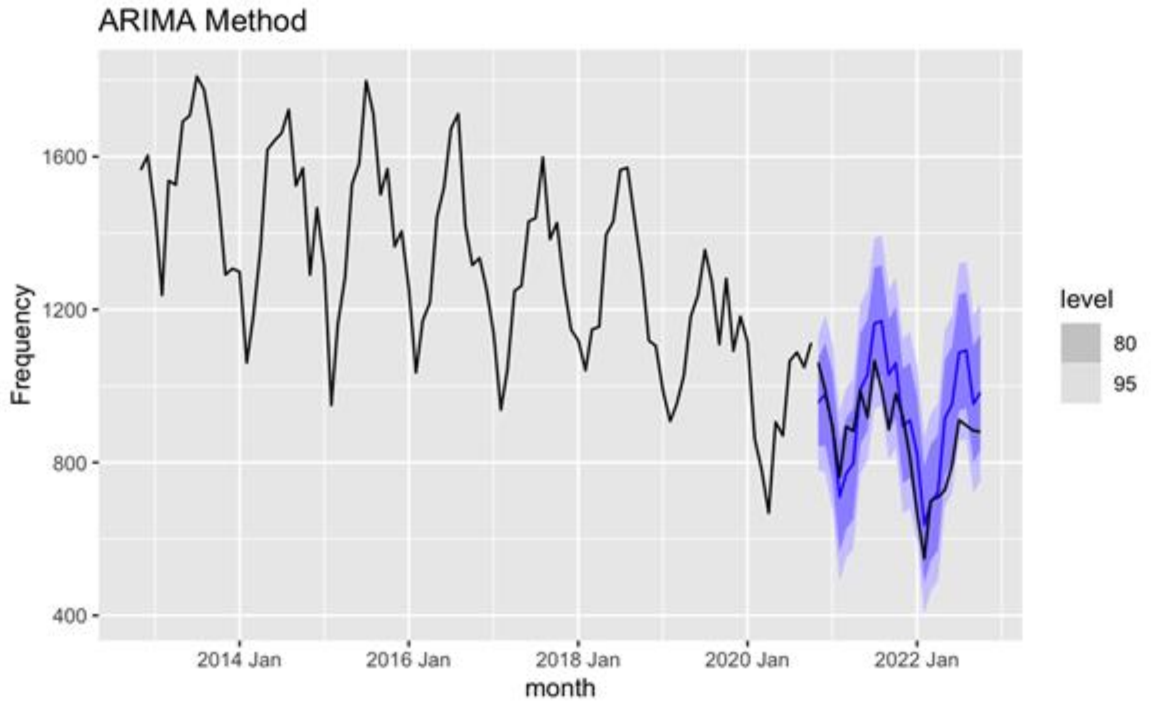
Forecast Plots





Forecast Plots

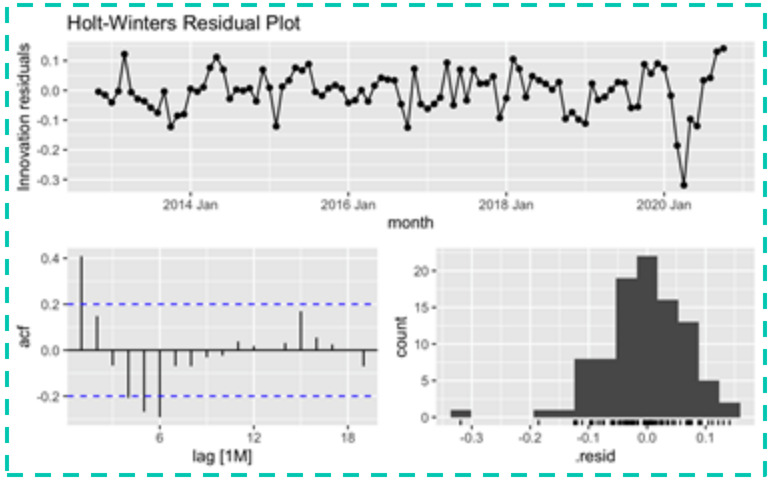


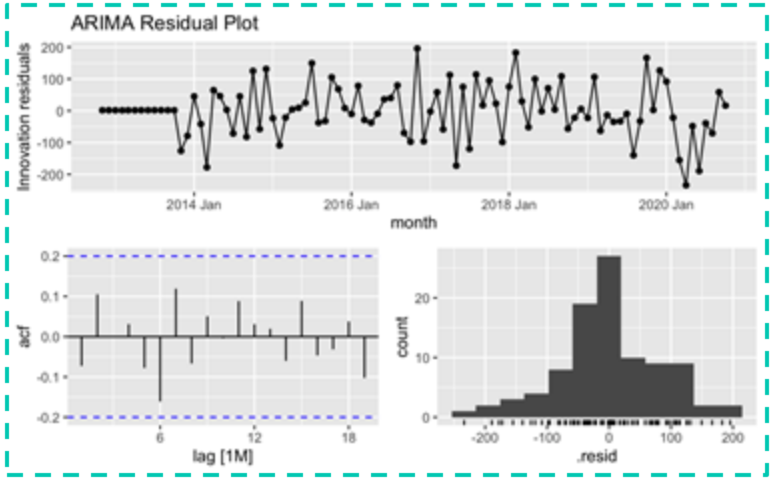


Test Errors

Method	Test Error (RMSE)
Holt-Winters’	92.87663
Mean	468.6411
Naive	277.9149
Seasonal Naive	190.4494
Drift	220.9877
ETS	168.768
ARIMA	112.8066

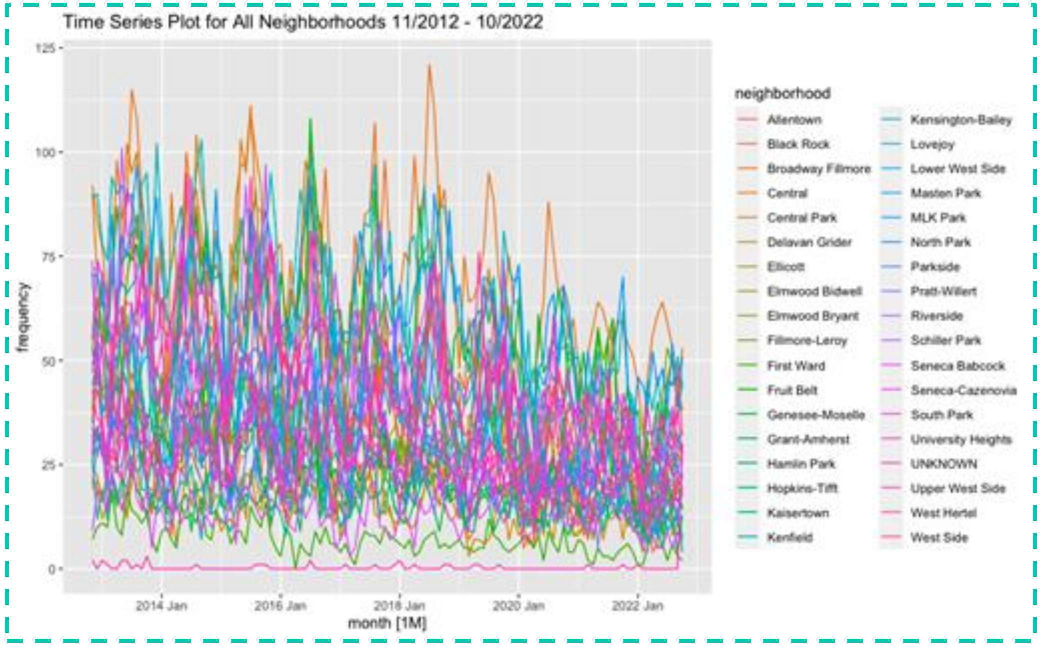
- Based on the test errors, the Holt-Winters’ and ARIMA methods forecasted the last two years most accurately
  - Looking at the residual plots, the ARIMA model appears to be the overall best choice

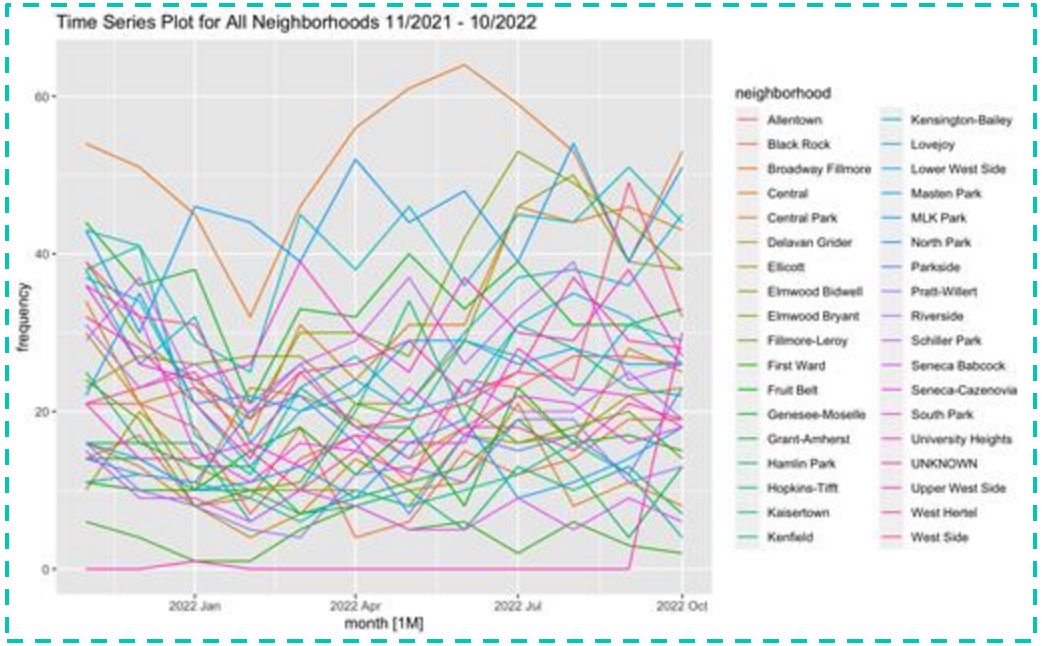




Time Series Plots By Neighborhood

- The City of Buffalo consists of 35 neighborhoods

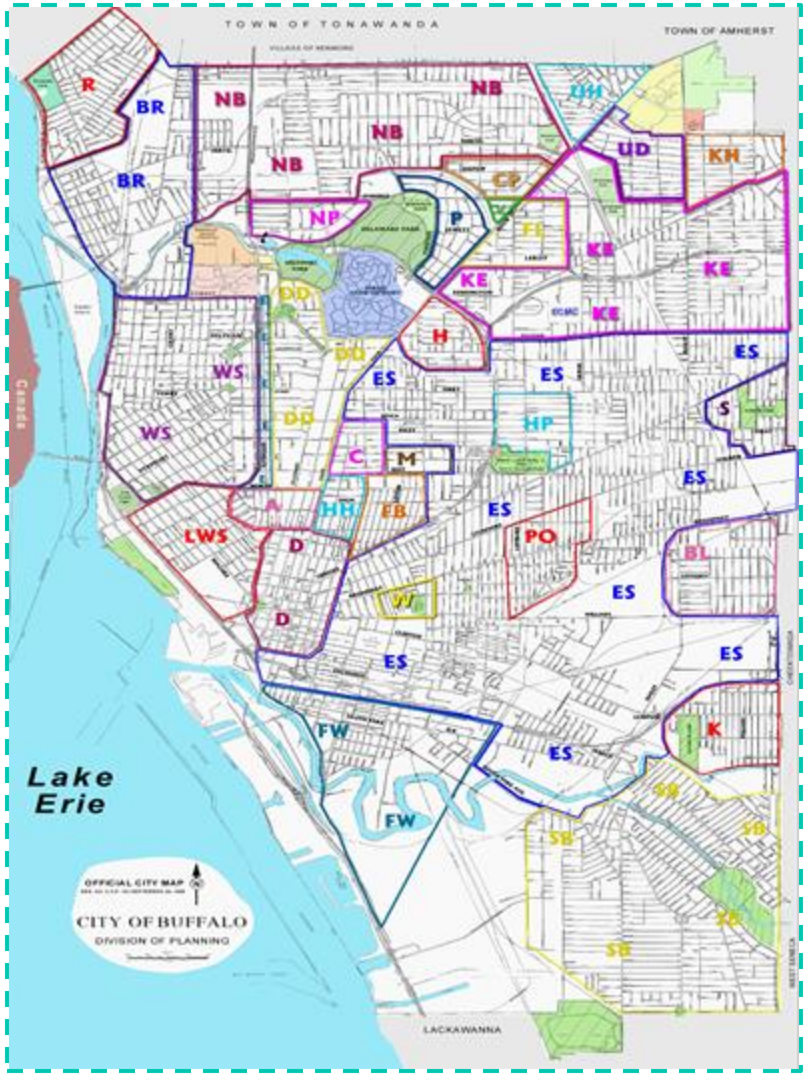




The graph on the left is extremely difficult to analyze. The graph on the right displays the last 12 months of data on a more readable scale.

Motivation for Analysis by Neighborhood

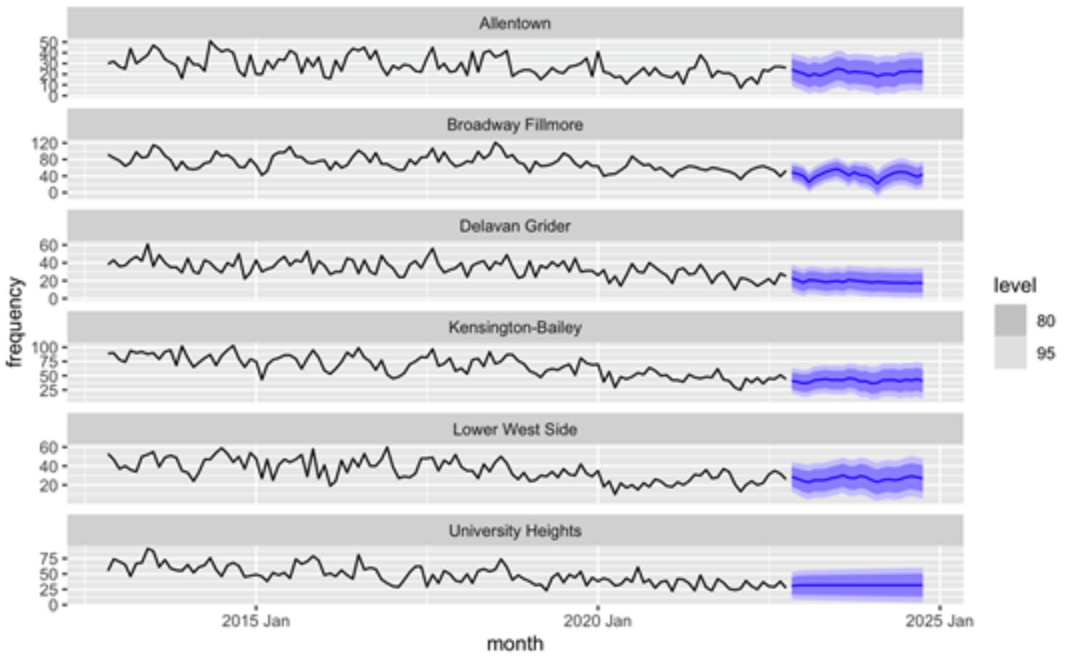
- Near- and long-term implications
- May aid city officials in deciding where to allocate their resources in an effort to fight crime and poverty
    - Where police officers are assigned on duty
    - Where to consider implementing welfare initiatives

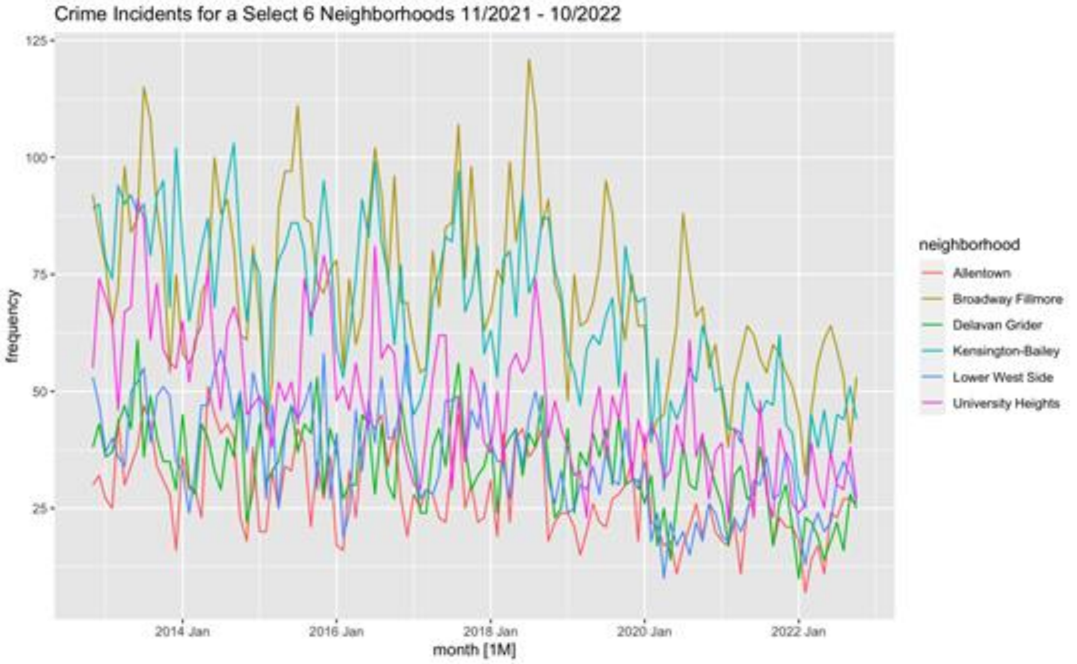


Forecasting Crime in a Subset of Neighborhoods

- A select 6 neighborhoods are fit to ARIMA models
  - Difficult to format and visualize all 35 neighborhoods in R

ARIMA Forecasts:





Thank You!

Please reply with any comments and/or questions you may have.

University at Buffalo
School of Engineering and Applied Sciences

EAS 509 Final Project

2023-05-08

Below we import the necessary libraries for our analysis.

```
library(tibble)
library(dplyr)
library(tidyr)
library(readr)
library(lubridate)
library(ggplot2)
library(data.table)
library(fpp)
library(ggpubr)
library(tsibble)
library(fable)
library(fabletools)
library(feasts)
library(tsibbledata)
library(zoo)
```

Next, we read the file from the Open Buffalo Data repository that contains crime data.

```
crime <- fread('/Users/ryanglasser/Desktop/University at Buffalo/Graduate/Year 1/Spring 2023/EAS 509 - S
```

In the next code block, we subset the data frame so that it only includes relevant columns and convert the column containing incident dates to a format of interest. From there, because the data set is extremely large and we are only interested in relatively recent crime data, we choose to analyze crime incidents that occurred between the time period of November 2012 and October 2022. Beyond this point in time, there are some jumps in crime records, which would make it difficult in our time series forecasting.

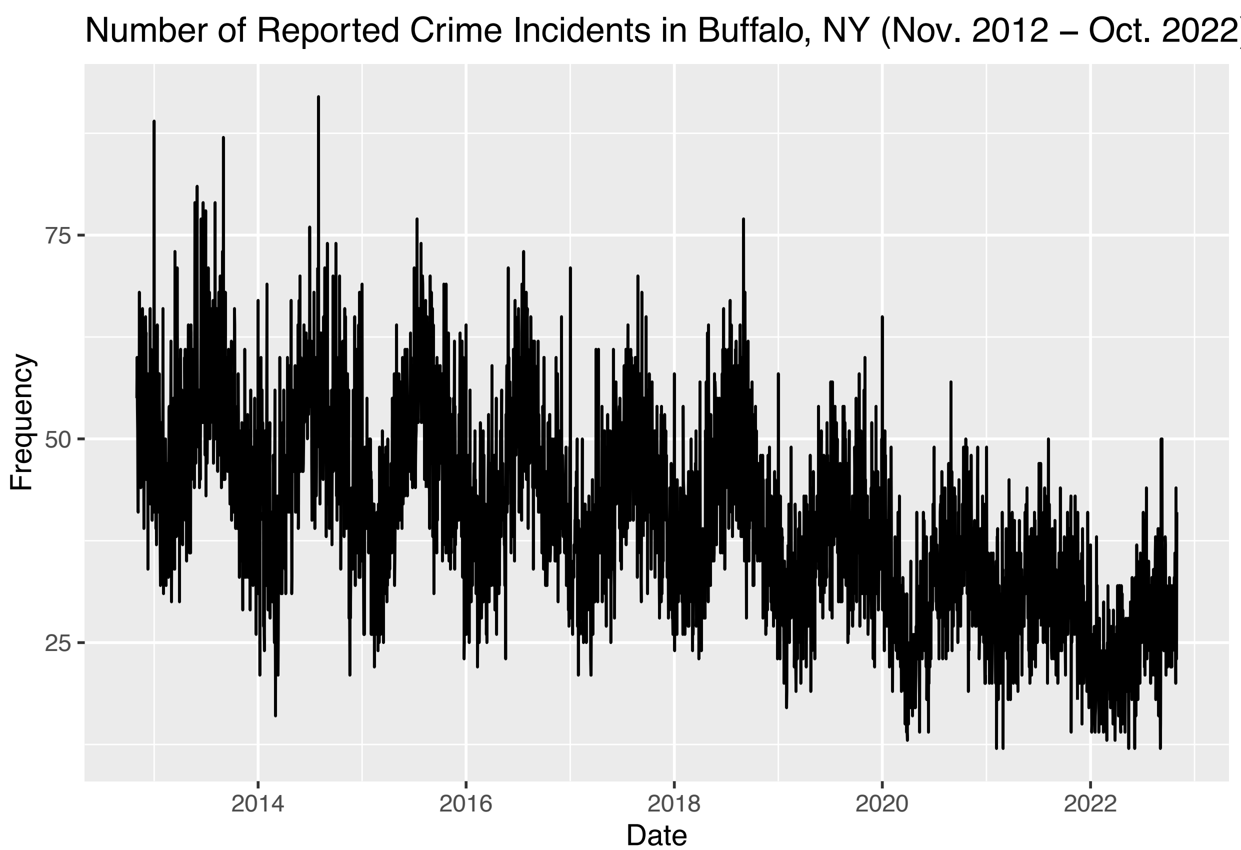
```
#Select the columns that are necessary for our analyses
crime <- crime[, c('Incident Datetime', 'Parent Incident Type', 'Hour of Day',
                   'Day of Week', 'Neighborhood', 'Police District',
                   'Council District', 'Neighborhoods')] %>% drop_na()

#Convert dates to a format of interest
dates <- as.character(crime$`Incident Datetime`,
                      format = "%Y/%m/%d %I:%M:%S %p")
crime$`Incident Datetime` <- as.Date(dates, format = '%m/%d/%Y')

#Select a subset of 10 years of data from November 2012 - October 2022
crime <- crime[crime$`Incident Datetime` >= "2012-11-01" & crime$`Incident Datetime` <= "2022-10-31", ]

#Order the data
crime <- crime[order(crime$`Incident Datetime`), ]
```

Now that the data is organized in a way of interest, we will create a separate data frame that will store a tally for the number of incidents recorded on each day in our time period of interest. Then, the data frame is converted to a time series object for further analysis.

```
#Create a separate data frame for number of incidents on each day
crime_count <- data.frame(table(crime$`Incident Date`))
colnames(crime_count) <- c('date', 'frequency')

#Convert to tsibble
crime_count <- crime_count %>%
mutate(Date = ymd(date)) %>%
  select(-date) %>%
  as_tsibble(index = Date)
```

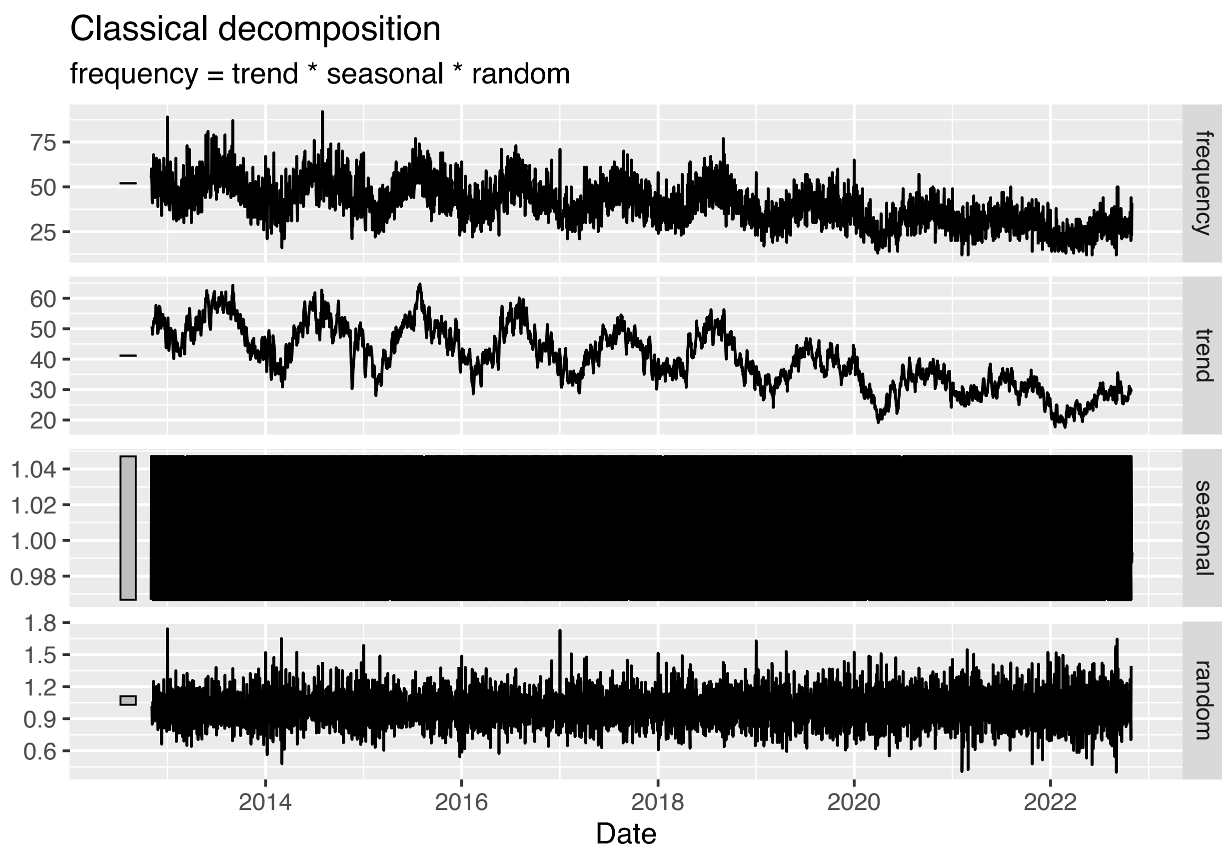
Next, we plot the time series data which displays the number of crime incidents daily between Nov. 2012 and Oct. 2022.

```
#Plot the time series data
crime_count %>% autoplot(frequency) +
  labs(x = 'Date', y = 'Frequency',
       title = 'Number of Reported Crime Incidents in Buffalo, NY (Nov. 2012 - Oct. 2022)')
```



At first glance, it appears that there is a decreasing trend in crime over time until about the start of 2022. Beyond that, the time series appears to be on the incline. We can perform classical decomposition on the data set to further analyze this.

```
#Classical decomposition
dcmp <- crime_count %>%
  model(classical_decomposition(frequency, type = "multiplicative"))

components(dcmp) %>% autoplot()
```



This again shows that overall there is a decreasing trend over the course of our time period of interest, yet the slight increase in crime incidents in 2022 is noteworthy. Perhaps this increase can be attributed to societal operations returning back to normal after a few years of intense COVID-19 restrictions?

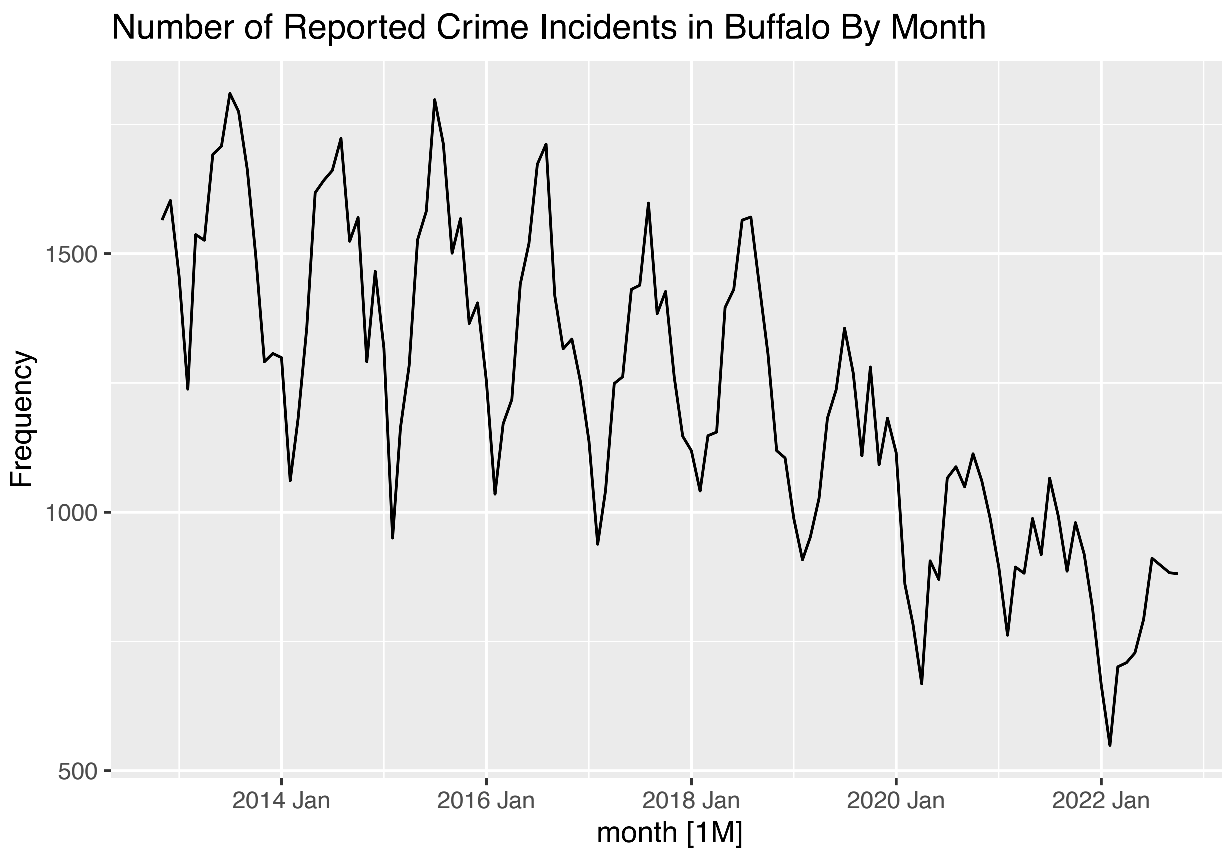
Since the time series plot of thousands of days of crime incident data can be appear to be scrunched, we will now sum the number incidents for each month between November 2012 and October 2022. This is done by using the `lubridate` package and creating another data frame titled `monthly` that is also converted to a time series object.

```
#The following 2 lines will allow us to sum the incidents for each month
crime_count <- crime_count %>%
  group_by(month = lubridate::floor_date(Date, 'month'))

#Data frame called `monthly` displays the number of incidents for each month
monthly <- data.frame(aggregate(frequency~month, crime_count, sum))
monthly$month <- as.yearmon(monthly$month, '%b %Y')

#Convert to tsibble
monthly <- monthly %>%
mutate(month = yearmonth(month)) %>%
```

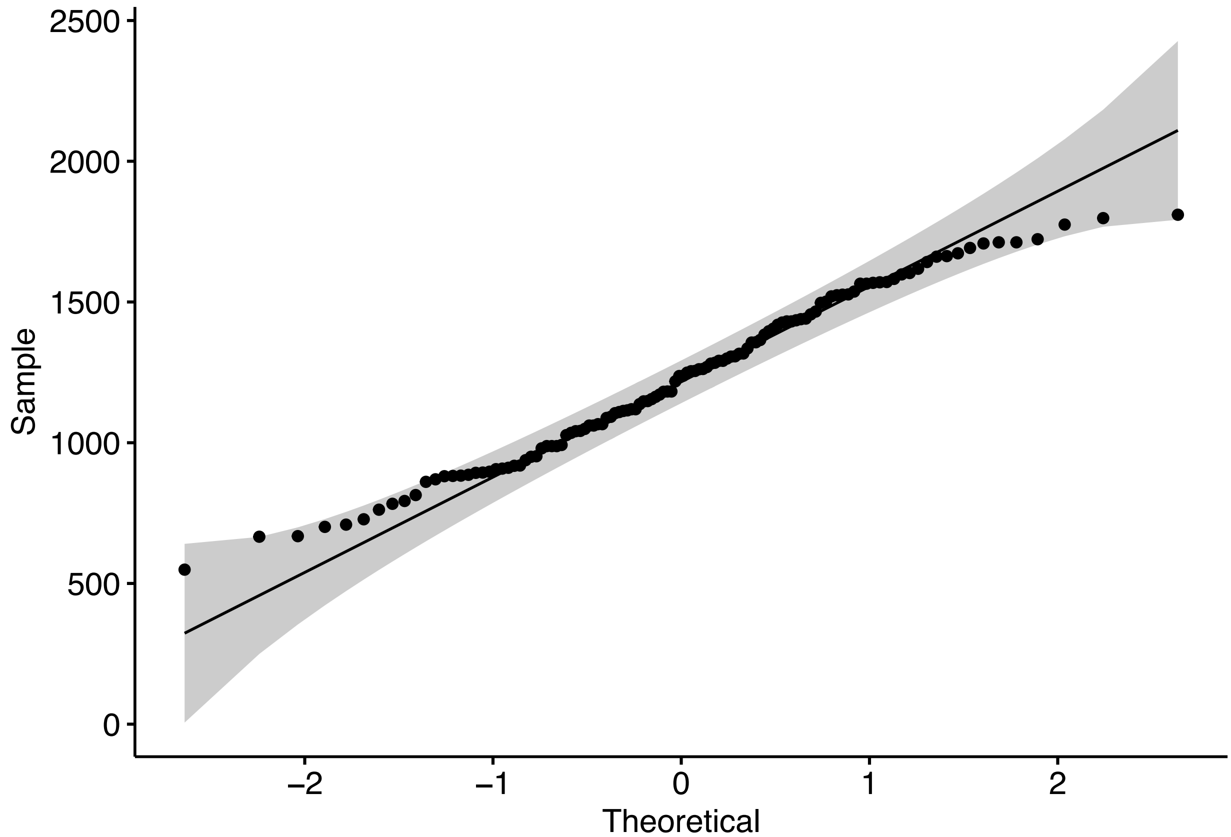
```
  as_tsibble(index = month)
```

Next, we plot the time series data for each month. In addition, we take a look at a Q-Q plot to determine if the data needs any transformation.

```
#Plot the time series data
monthly %>% autoplot(frequency) +
  labs(y = "Frequency",
       title = "Number of Reported Crime Incidents in Buffalo By Month")
```

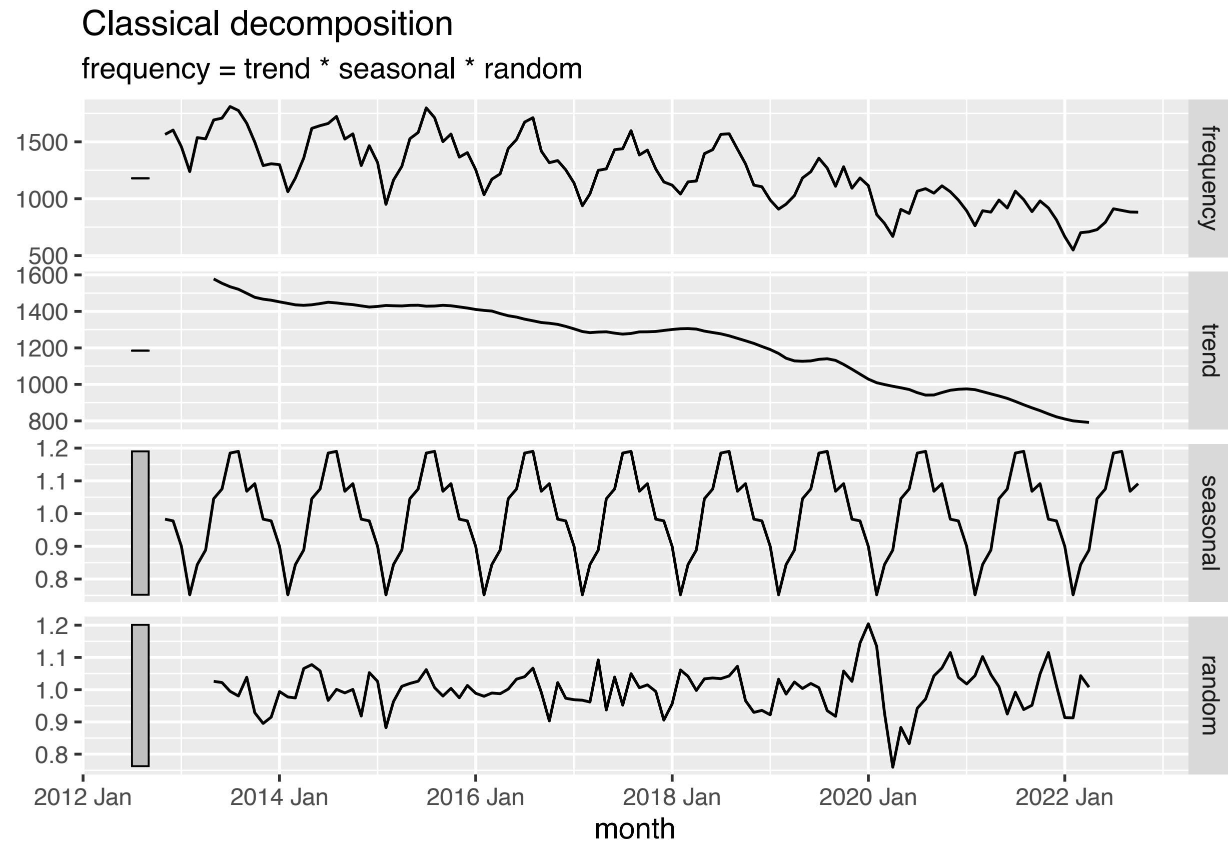



```
#Q-Q plot
ggqqplot(monthly$frequency)
```



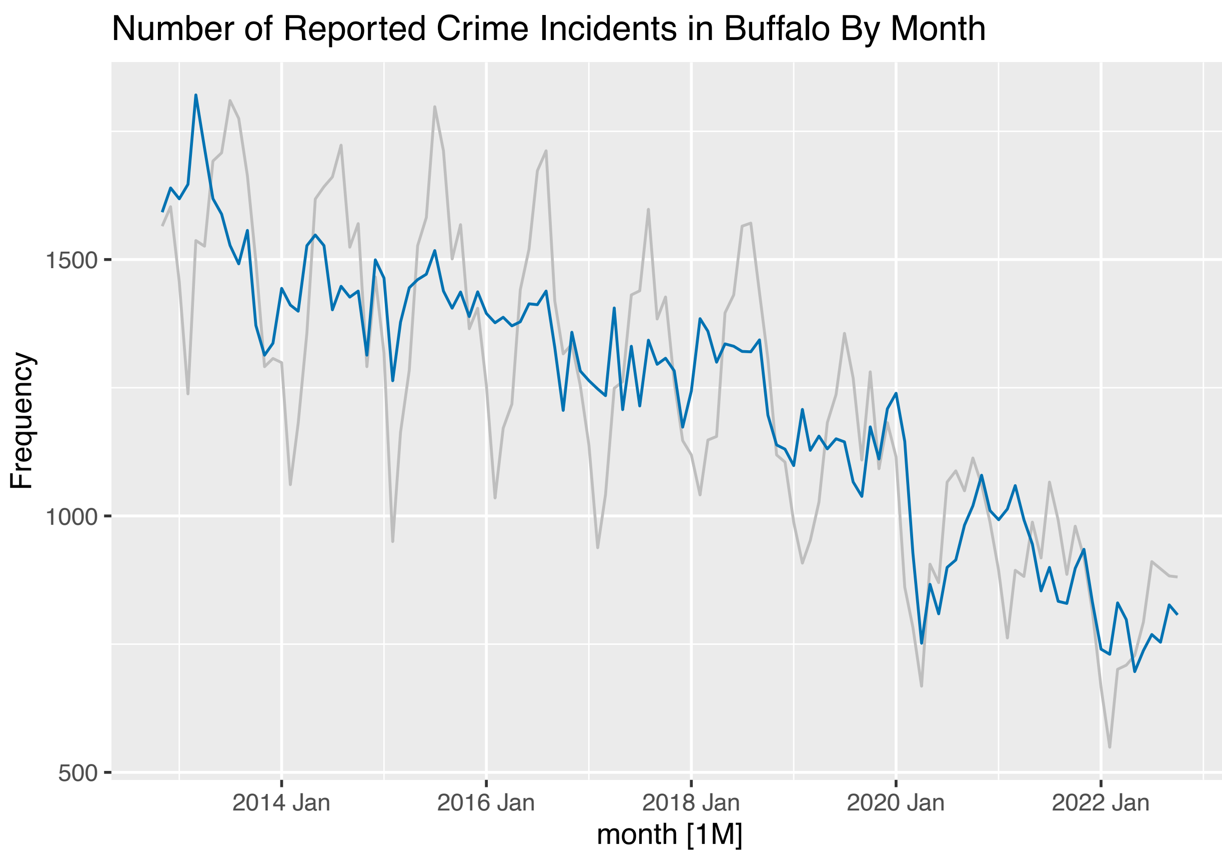
When we take a look at the number of crime incidents for each month, it appears that there may be some sort of seasonality in addition to a decreasing trend over time. With regards to the Q-Q plot, just about every point falls along or within the grey region, indicating that a transformation is not absolutely necessary.Next, we perform classical decomposition on the data to further look into the possibility in data seasonality.

```
#Classical decomposition
dcmp <- monthly %>%
  model(classical_decomposition(frequency, type = "multiplicative"))

components(dcmp) %>% autoplot()
```



After performing classical decomposition, it is clear that there is a decreasing trend over time, and the seasonal plot does indicated a degree of seasonality. We now plot the seasonally adjusted data. Typically, seasonally adjusted data is useful in aiding government to examine a particular issue and may facilitate policy and action. In the case of crime data, perhaps the seasonally adjusted data provides city officials with valuable information that they use in fighting crime and poverty in the region?

```
#Plot the seasonally adjusted data
monthly %>%
  autoplot(frequency, color = 'gray') +
  autolayer(components(dcmp), season_adjust, color = '#0072B2') +
  labs(y = 'Frequency',
       title = 'Number of Reported Crime Incidents in Buffalo By Month')
```



We will now perform accuracy evaluation on the time series data and evaluate several models in attempt to find one that yields the highest accuracy and lowest test error. The last two years of our time period of interest serve as the testing set, while the prior eight years serve as the training set.

```
train_cc <- monthly[1:96,]
test_cc <- monthly[97:120,]
```

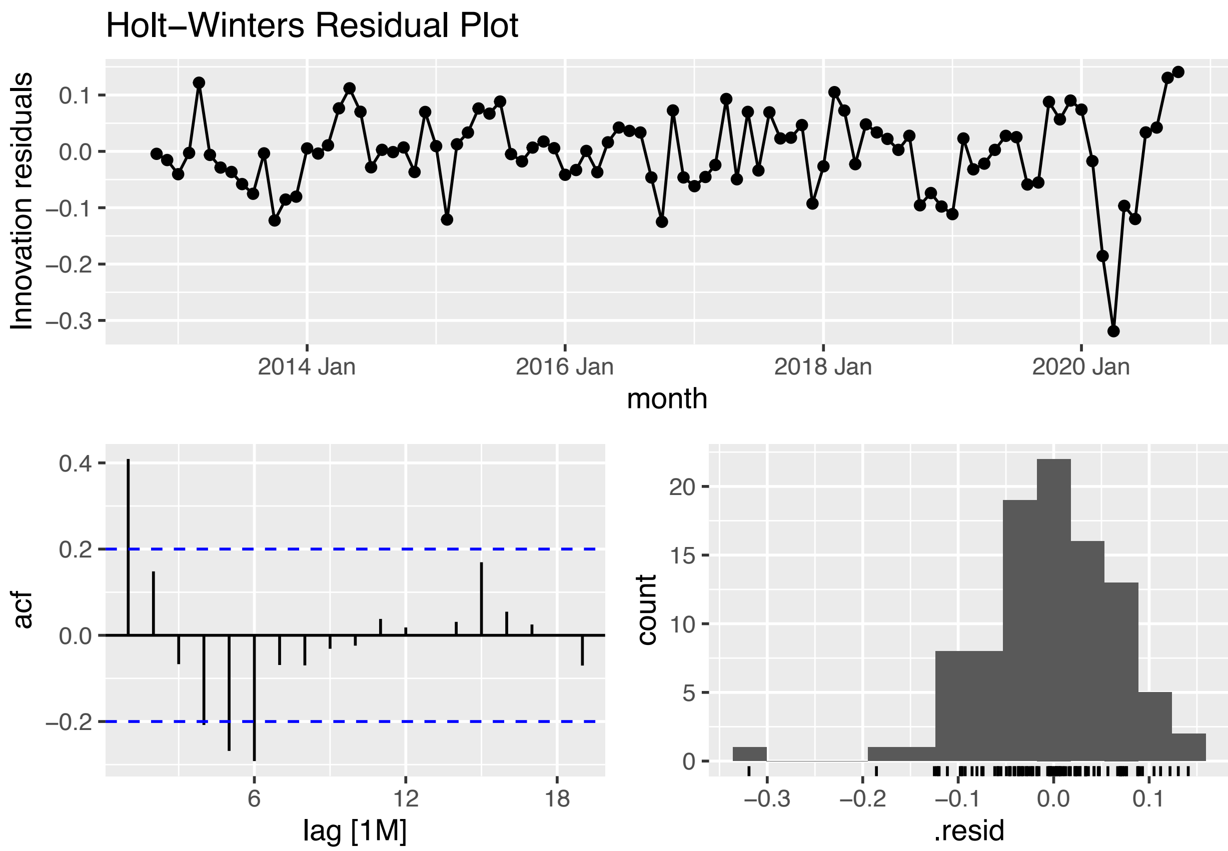
First, we use the training set to create a fit for Holt-Winters' multiplicative method and forecast two years of crime incident data.

```
crime_hw_fit <- train_cc %>%
  model(multiplicative = ETS(frequency ~ error("M") + trend("A") + season("M")))

crime_hw_fc <- crime_hw_fit %>%
  forecast(h = 24)

residuals(crime_hw_fit, type = 'response')
```

```
## # A tsibble: 96 x 3 [1M]
## # Key:       .model [1]
##    .model           month  .resid
##    <chr>            <mth>   <dbl>
##  1 multiplicative 2012 Nov   -6.89
##  2 multiplicative 2012 Dec  -25.1
##  3 multiplicative 2013 Jan  -61.5
```

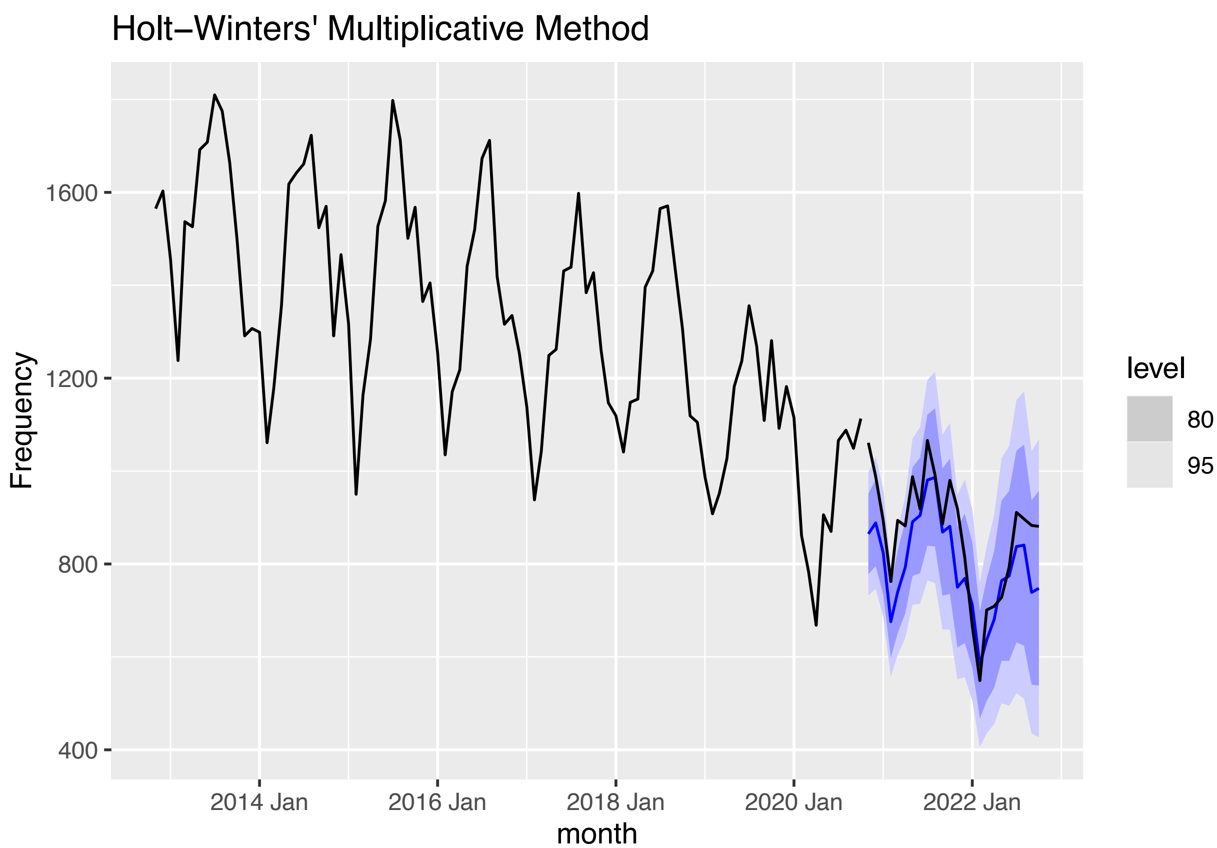
```
##   4 multiplicative 2013 Feb   -3.49
##   5 multiplicative 2013 Mar  167.
##   6 multiplicative 2013 Apr   -9.64
##   7 multiplicative 2013 May  -50.1
##   8 multiplicative 2013 Jun  -64.8
##   9 multiplicative 2013 Jul -112.
##  10 multiplicative 2013 Aug -144.
## # i 86 more rows
```

```
crime_hw_fit %>%
  gg_tsresiduals() + labs(title = 'Holt-Winters Residual Plot')
```




```
crime_hw_fc %>%
  autoplot(train_cc) +
  labs(
  title = "Holt-Winters' Multiplicative Method",
  y = "Frequency"
  ) +
  guides(colour = guide_legend(title = "Forecast")) +
  autolayer(test_cc) #actual data
```

```
## Plot variable not specified, automatically selected `.vars = frequency`
```




```r
accuracy(crime_hw_fit)
```

```
## # A tibble: 1 x 10
##   .model         .type        ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
##   <chr>          <chr>     <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 multiplicative Training -5.31  86.7  67.2 -0.965  5.55 0.558 0.577 0.372
```

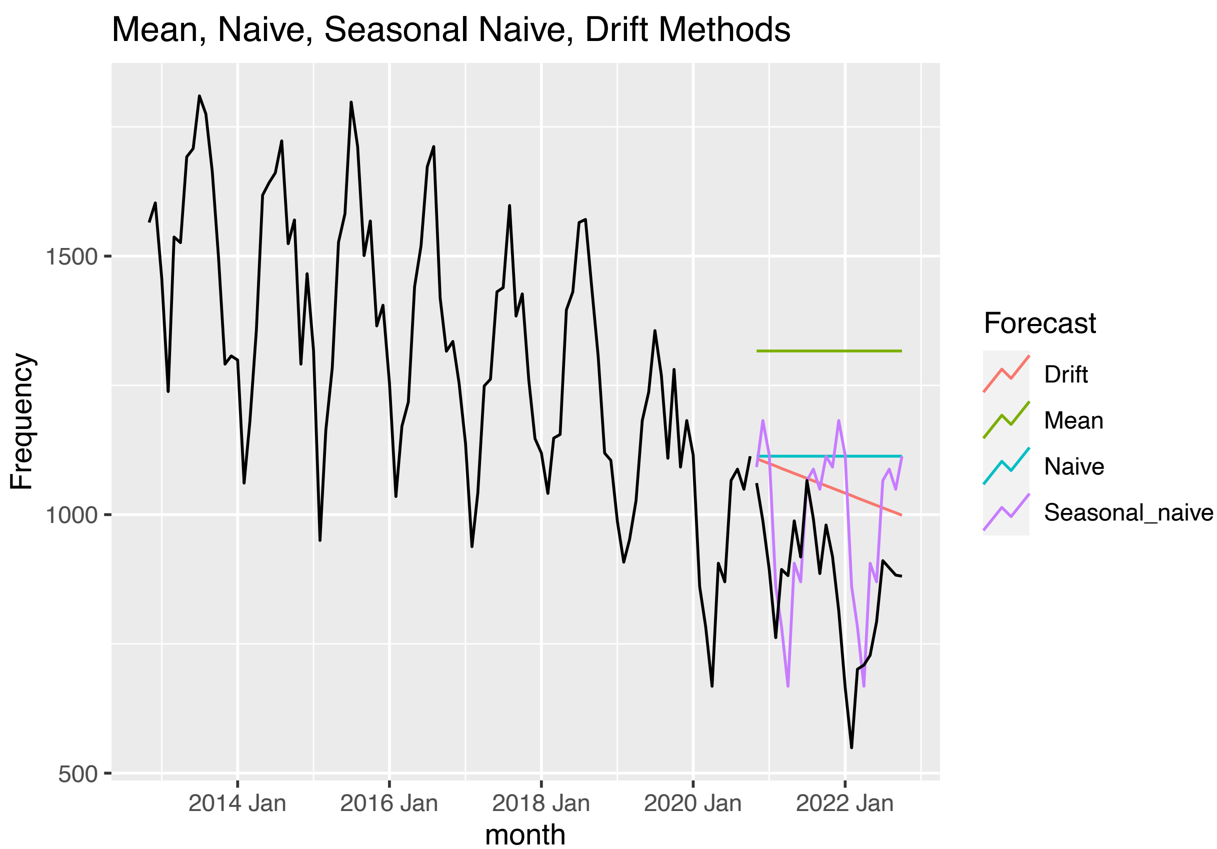
```r
accuracy(crime_hw_fc, monthly)
```

```
## # A tibble: 1 x 10
##   .model         .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>          <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 multiplicative Test   67.9  92.9  77.5  7.26  8.76 0.644 0.618 0.360
```

The RMSE for this method is 92.87663. Next, we use the training set to create a fit for mean, naive, seasonal naive, and drift methods. Then, we forecast the two years following that.

```r
cc_fit <- train_cc %>%
  model(
    Mean = MEAN(frequency),
    Naive = NAIVE(frequency),
    Seasonal_naive = SNAIVE(frequency),
    Drift = RW(frequency ~ drift())
  )
```

```r
cc_fc <- cc_fit %>%
  forecast(h = 24)

cc_fc %>%
  autoplot(train_cc, level = NULL) +
  labs(
  title = "Mean, Naive, Seasonal Naive, Drift Methods",
  y = "Frequency"
  ) +
  guides(colour = guide_legend(title = "Forecast")) +
  autolayer(test_cc, level = NULL) #actual data
```




```r
accuracy(cc_fit)
```

```
## # A tibble: 4 x 10
##   .model         .type           ME  RMSE   MAE    MPE  MAPE  MASE RMSSE   ACF1
##   <chr>          <chr>        <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1 Mean           Training  0          252.  209. -4.14  17.2   1.74 1.68  0.820
## 2 Naive          Training -4.76e+ 0  148.  123. -1.08   9.89  1.02 0.988 0.0289
## 3 Seasonal_naive Training -8.66e+ 1  150.  120. -7.83  10.4   1    1     0.525
## 4 Drift          Training  9.93e-14  148.  124. -0.706  9.90  1.03 0.987 0.0289
```

```r
accuracy(cc_fc, monthly)
```

```
## # A tibble: 4 x 10
##   .model         .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>          <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Drift          Test  -188.  221.  188. -24.5  24.5  1.56  1.47 0.642
## 2 Mean           Test  -452.  469.  452. -55.9  55.9  3.75  3.12 0.678
## 3 Naive          Test  -248.  278.  248. -31.8  31.8  2.06  1.85 0.678
## 4 Seasonal_naive Test  -118.  190.  159. -15.2  19.8  1.32  1.27 0.668
```

The RMSEs for these four methods are as follows:

Drift: 220.9877 Mean: 468.6411 Naive: 277.9149 Seasonal Naive: 190.4494

Next, we repeat the process by fitting an ETS model.

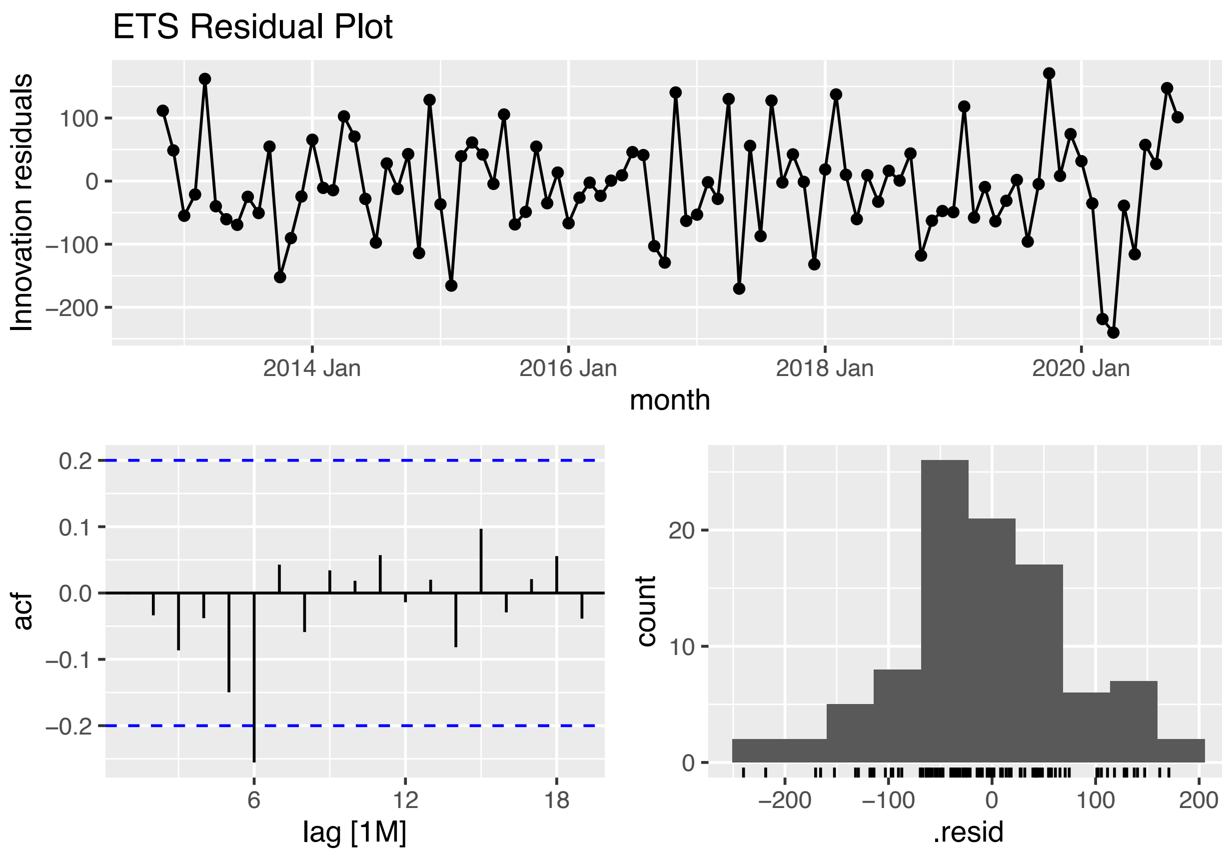
```r
ets_fit <- train_cc %>%
  model(auto = ETS(frequency)) %>%
  report()
```

```
## Series: frequency
## Model: ETS(A,N,A)
##   Smoothing parameters:
##     alpha = 0.7471786
##     gamma = 0.0001022917
##
##   Initial states:
##     l[0]     s[0]    s[-1]    s[-2]    s[-3]    s[-4]    s[-5]     s[-6]
##  1519.81 87.80346 87.75308 267.2918 257.9597 148.6545 78.59584 -137.7297
##      s[-7]     s[-8]     s[-9]    s[-10]    s[-11]
##  -207.5089 -339.1982 -128.5235 -48.78057 -66.31754
##
##   sigma^2:  7856.185
##
##      AIC     AICc      BIC
## 1314.074 1320.074 1352.540
```

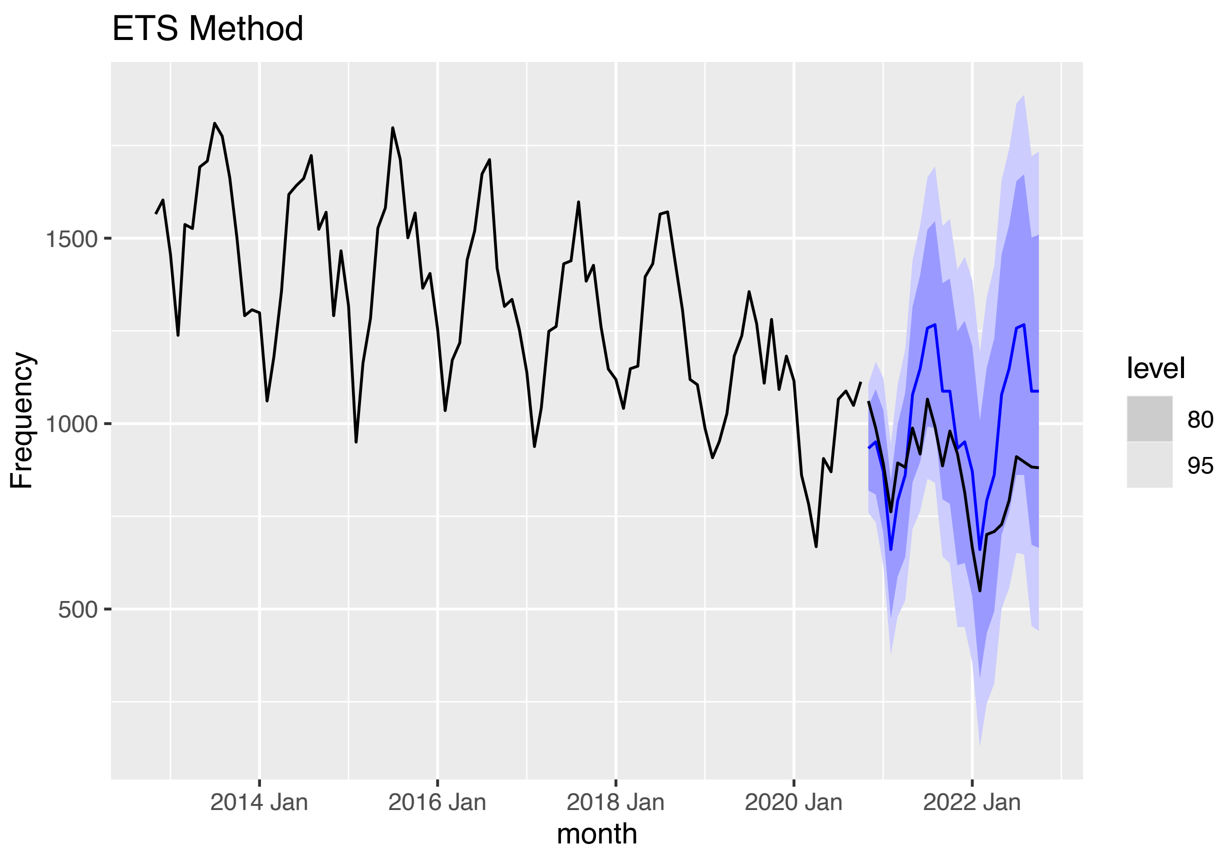
```r
residuals(ets_fit, type = 'response')
```

```
## # A tsibble: 96 x 3 [1M]
## # Key:       .model [1]
##    .model    month .resid
##    <chr>     <mth>  <dbl>
##  1 auto   2012 Nov  112.
##  2 auto   2012 Dec   48.7
##  3 auto   2013 Jan  -55.0
##  4 auto   2013 Feb  -21.2
##  5 auto   2013 Mar  162.
##  6 auto   2013 Apr  -39.8
##  7 auto   2013 May  -60.4
##  8 auto   2013 Jun  -69.3
##  9 auto   2013 Jul  -24.8
## 10 auto   2013 Aug  -50.6
## # i 86 more rows
```

```r
ets_fit %>%
  gg_tsresiduals() + labs(title = 'ETS Residual Plot')
```




```r
ets_fc <- ets_fit %>%
  forecast(h = 24)

ets_fc %>%
  autoplot(train_cc) + autolayer(test_cc) +
  labs(
  title = "ETS Method",
  y = "Frequency"
  )
```




```r
accuracy(ets_fc, monthly)
```

```
## # A tibble: 1 x 10
##   .model .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 auto   Test  -135.  200.  169. -16.3  20.1  1.40  1.33 0.754
```

The RMSE for the ETS model is 200.0021. Finally, we repeat the process with an ARIMA model below.

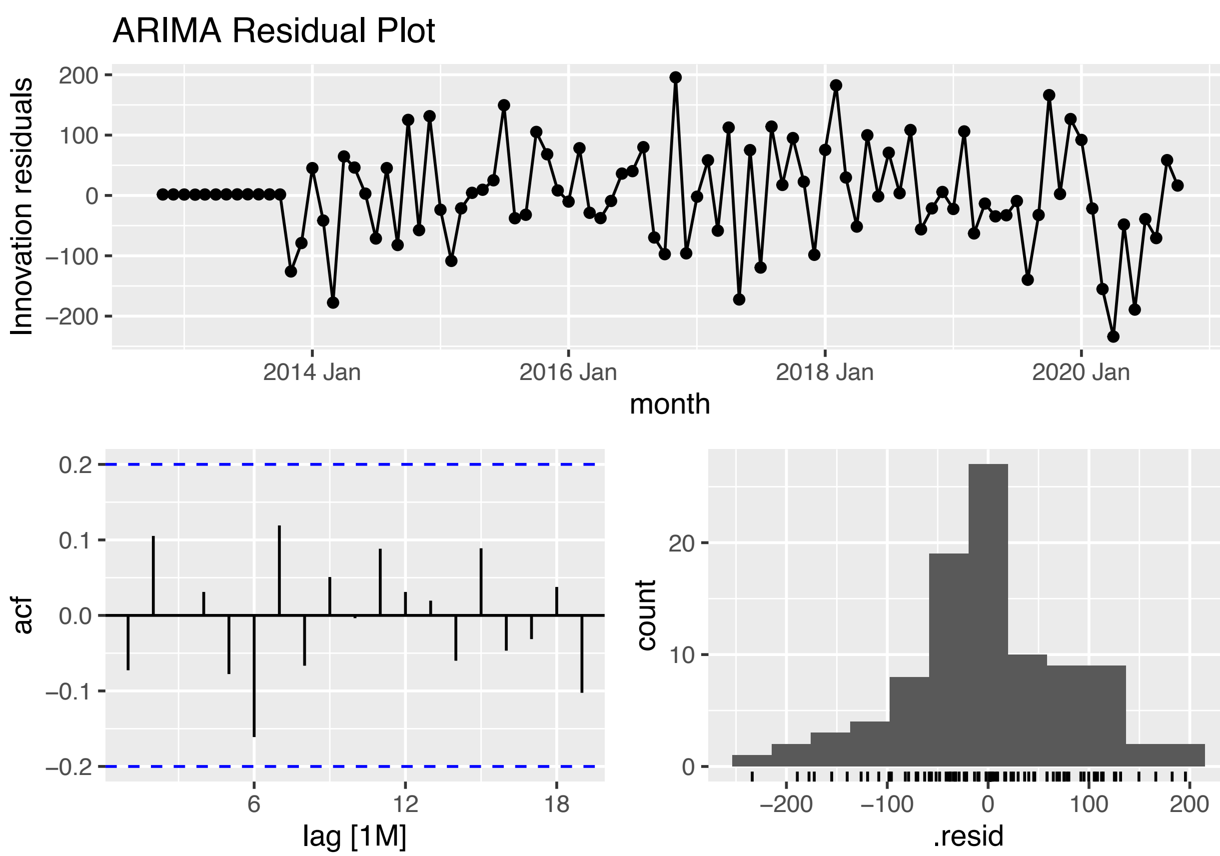
```r
arima_fit <- train_cc %>%
  model(ARIMA(frequency)) %>%
  report()
```

```
## Series: frequency
## Model: ARIMA(1,0,0)(0,1,1)[12] w/ drift
##
## Coefficients:
##          ar1     sma1  constant
##       0.6221  -0.7132  -29.0044
## s.e.  0.0846   0.1506    4.2373
##
## sigma^2 estimated as 7913:  log likelihood=-499.15
## AIC=1006.29   AICc=1006.8   BIC=1016.01
```

```r
residuals(arima_fit, type = 'response')
```

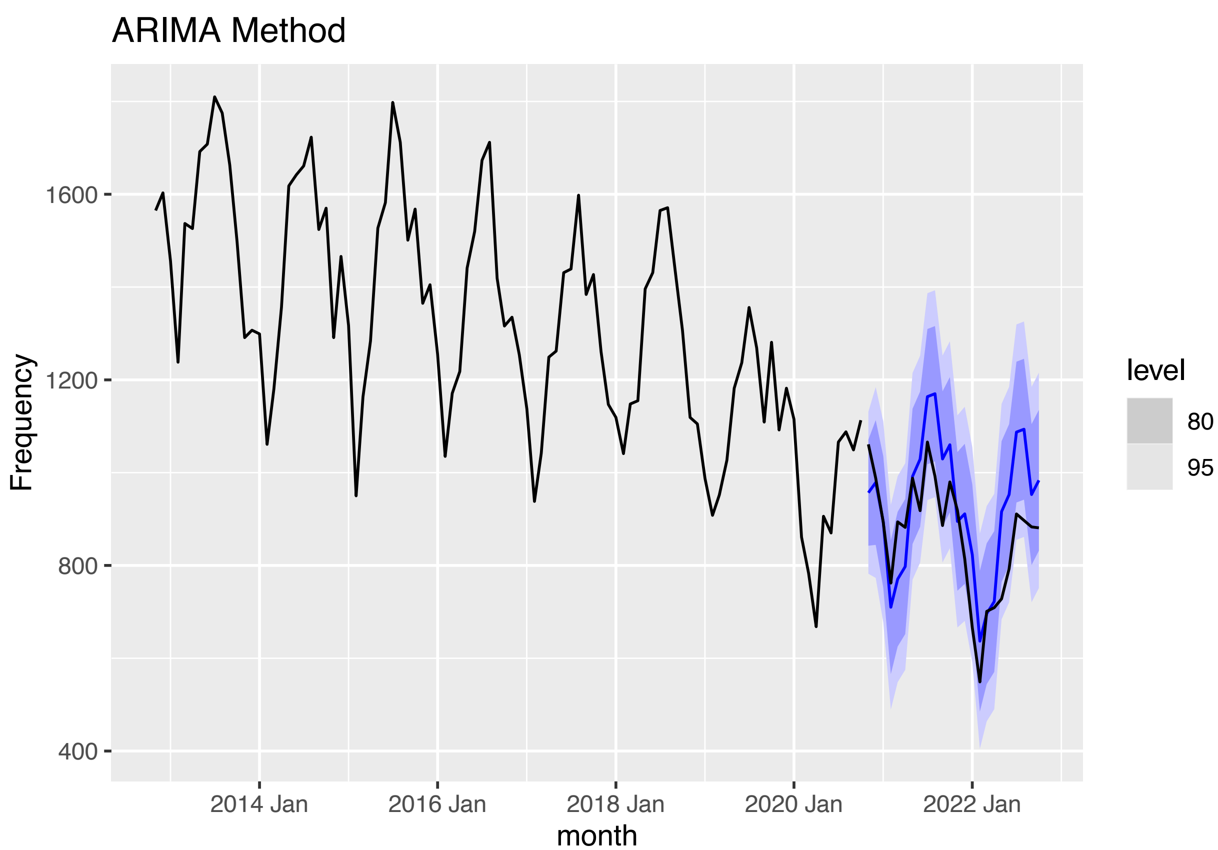
```
## # A tsibble: 96 x 3 [1M]
## # Key:       .model [1]
##    .model               month .resid
##    <chr>                <mth>  <dbl>
##  1 ARIMA(frequency) 2012 Nov   1.64
##  2 ARIMA(frequency) 2012 Dec   1.68
##  3 ARIMA(frequency) 2013 Jan   1.53
##  4 ARIMA(frequency) 2013 Feb   1.31
##  5 ARIMA(frequency) 2013 Mar   1.61
##  6 ARIMA(frequency) 2013 Apr   1.60
##  7 ARIMA(frequency) 2013 May   1.77
##  8 ARIMA(frequency) 2013 Jun   1.78
##  9 ARIMA(frequency) 2013 Jul   1.89
## 10 ARIMA(frequency) 2013 Aug   1.85
## # i 86 more rows
```

```r
arima_fit %>%
  gg_tsresiduals() + labs(title = 'ARIMA Residual Plot')
```




```r
arima_fc <- arima_fit %>%
  forecast(h = 24)
```

```
arima_fc %>%
  autoplot(train_cc) + autolayer(test_cc) +
  labs(
  title = "ARIMA Method",
  y = "Frequency"
  )
```




```
accuracy(arima_fc, monthly)
```

```
## # A tibble: 1 x 10
##   .model           .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>            <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 ARIMA(frequency) Test  -60.9  113.  94.5 -7.50  11.2 0.784 0.751 0.605
```

The RMSE after fitting an ARIMA model is 112.8066.

Upon fitting several different models, we see that Holts Winter's multiplicative method yields the lowest RMSE, and the ARIMA model fit yields the second lowest RMSE. Typically, we would want to choose the model that yields the lowest RMSE, but looking at the residual tests for the aforementioned methods, when choosing a model for future forecasting, the ARIMA model may be the best bet.

Now, we create a data frame that includes the frequency of crime for each day in each of the 35 neighborhoods of Buffalo.

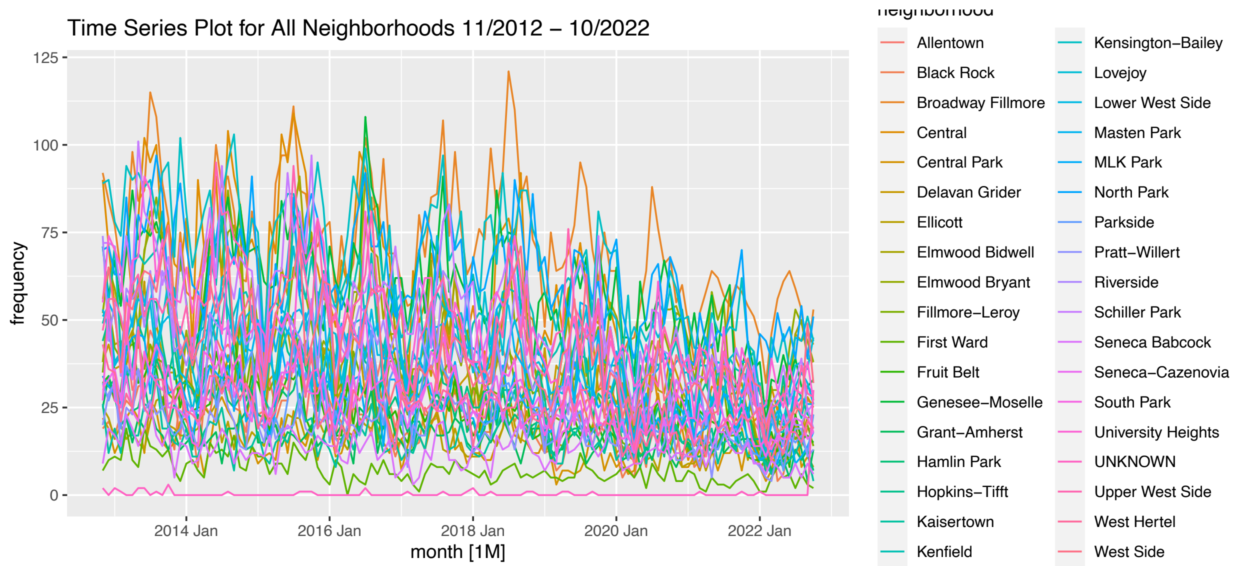
```r
#Data frame with crime incident frequency by neighborhood for each day
neighborhood_count <- data.frame(table(crime$`Incident Date`, crime$Neighborhood))
colnames(neighborhood_count) <- c('date', 'neighborhood', 'frequency')

#Convert to tsibble
neighborhood_count <- neighborhood_count %>%
mutate(Date = ymd(date)) %>%
  select(-date) %>%
  as_tsibble(key = neighborhood, index = Date)
```

Next, we create a data frame that sums up the number of crime incidents for each month in each of the 35 neighborhoods of Buffalo. We also subset that data frame to include only the last two years of data for plotting purposes later.

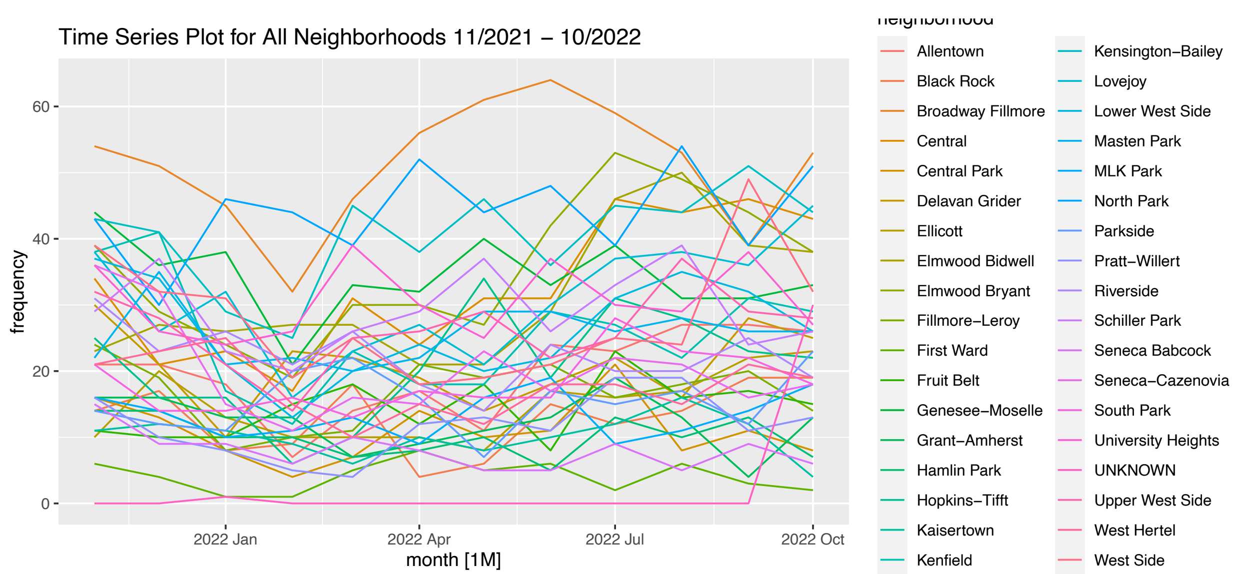
```r
#Allows us to sum up the number of crimes in each month for every neighborhood
neighborhood_count <- neighborhood_count %>%
  group_by(month = lubridate::floor_date(Date, 'month'))

#Sums up the frequency of crime incidents for each neighborhood per month
neighborhood_monthly <- data.frame(aggregate(frequency~month + neighborhood,
                                             neighborhood_count, sum))

#Data frame that contains the last 2 years of data from `neighborhood monthly`
subset_of_data <- subset(neighborhood_monthly, month >= '2021-11-01')

#Convert to date format
neighborhood_monthly$month <- as.yearmon(neighborhood_monthly$month, '%b %Y')
subset_of_data$month <- as.yearmon(subset_of_data$month, '%b %Y')

#Convert to tsibble
neighborhood_monthly <- neighborhood_monthly %>%
mutate(month = yearmonth(month)) %>%
  as_tsibble(key = neighborhood, index = month)

#Convert to tsibble
subset_of_data <- subset_of_data %>%
mutate(month = yearmonth(month)) %>%
  as_tsibble(key = neighborhood, index = month)
```

We now plot the `neighborhood_monthly` and `subset_of_data` data frames. We see that the plot of all 35 neighborhoods over the course of 10 years of data is extremely hard to pick apart and analyze. Thus, by plotting only the last two years of data, we can see how the time series data fluctuates for each of the 35 neighborhoods on a more readable scale.

```r
neighborhood_monthly %>%
  autoplot(frequency) + labs(title = 'Time Series Plot for All Neighborhoods 11/2012 - 10/2022')
```




```
subset_of_data %>%
  autoplot(frequency) + labs(title = 'Time Series Plot for All Neighborhoods 11/2021 - 10/2022')
```



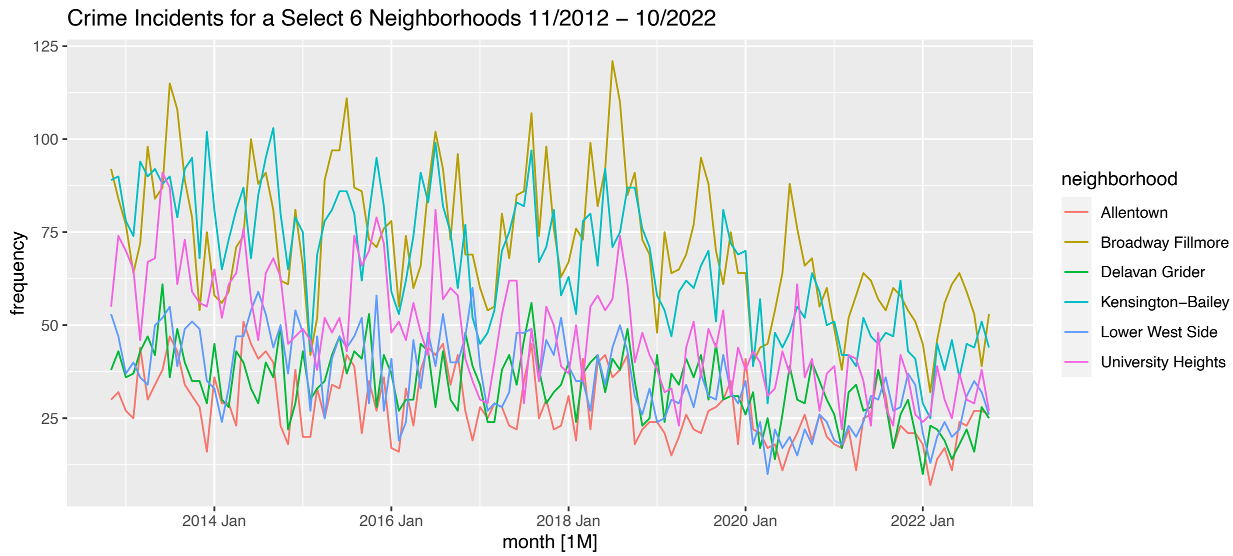
Since fitting ARIMA models and forecasting two years of data for all 35 neighborhoods in Buffalo would be difficult to format and visualize in R, we select six neighborhoods to fit. From there, we plot the two-year forecasts for each.

Relating this data analysis to the real-world context of crime incidents, it may be useful to perform model forecasting for each of the 35 neighborhoods to see how crime rates can change in the near- or long-term. Such valuable analysis may aid city officials in deciding where to allocate their resources (police officer assignments, where to implement welfare initiatives, etc.) in an effort to fight crime and poverty.

First, we define the data frame.

```
#Select six neighborhoods
six_n <- subset(neighborhood_monthly,
                neighborhood %in% c('Allentown', 'Broadway Fillmore',
                                    'Delavan Grider', 'Kensington-Bailey',
```

```r
                                    'Lower West Side', 'University Heights'))

#Convert to tsibble
six_n <- six_n %>%
mutate(month = yearmonth(month)) %>%
  as_tsibble(key = neighborhood, index = month)

#Plot the time series data
six_n %>%
  autoplot(frequency) + labs(title = 'Crime Incidents for a Select 6 Neighborhoods 11/2012 - 10/2022')
```

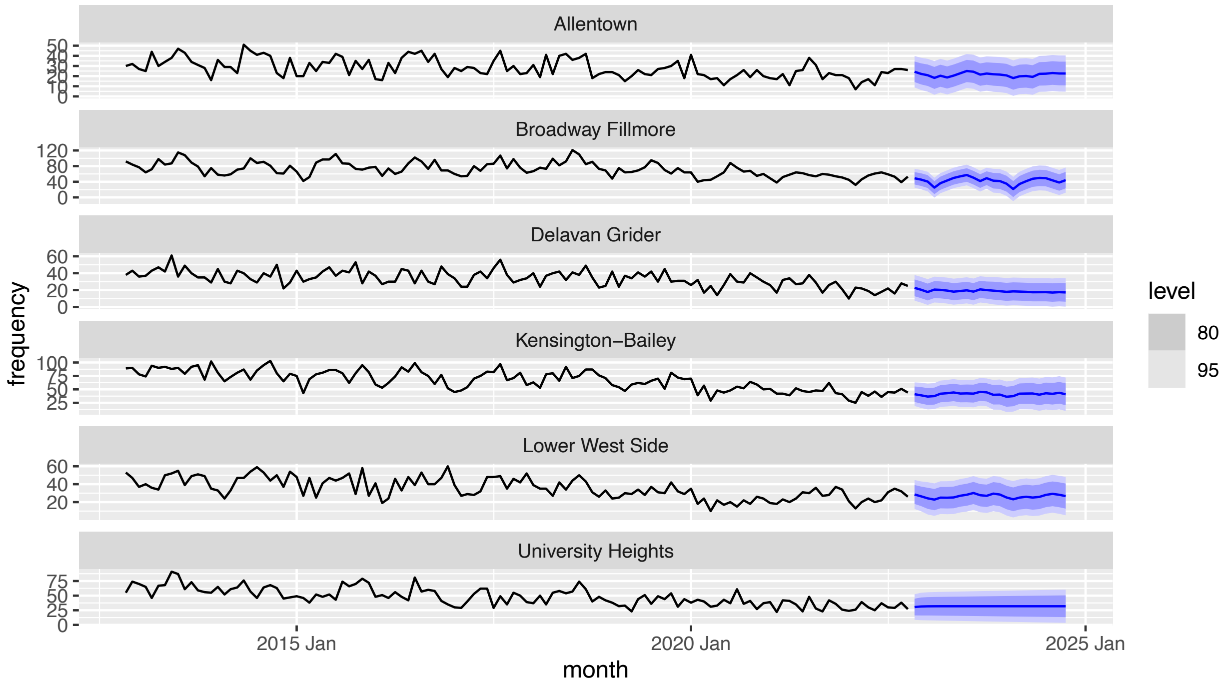


Next, we fit the ARIMA models.

```r
arima_nf <- six_n %>%
  model(ARIMA(frequency))
```

Finally, we plot the two-year forecasts for each of the six neighborhoods.

```r
arima_nf_fc <- arima_nf %>%
  forecast(h = 24)

arima_nf_fc %>%
  autoplot(six_n)
```


Allentown
Broadway Fillmore
Delavan Grider
Kensington–Bailey
Lower West Side
University Heights
frequency
month
2015 Jan
2020 Jan
2025 Jan
level
80
95