Analysis on Chicago Crime (2012-2022)

STAT 447 Final Project

Wenxuan Gu(wg16), Qingyu Huang(qingyuh2), Yuhui Wang(yuhuiw2)

Contents

1 PROJECT DESCRIPTION

1.1 Background

As a city developed by Transport Hub, all kinds of news tell us a fact: Chicago uses prosperity to conceal blood. On November 9, 2021, a Chinese student studying at the University of Chicago died in a shooting, and many similar incidents happened later. Not only did the residents panic, but also the international students questioned the city's safety. So the purpose of this project is to analyze the real crime situation in Chicago.

1.2 Understanding data

There are two tables of original data: Crimes_-*2001_to_Present and ILCOOK1URN.*
1. Crimes-_2001_to_Present: It stores basic crime data and includes 7681524 observations and 30 variables.
2. ILCOOK1URN: It stores unemployment data and includes 394 observations and 2 variables.

1.3 Framework

1. EDA.
2. Analysis the Influential factors with crime.
4. Modeling.
5. Shiny APP.

1.4 Libraries

```
#Loading packages
if (!require("pacman")) install.packages("pacman")
pacman::p_load(tidyverse, httr,jsonlite,shiny,shinydashboard,shinyjs,shinyWidgets,leaflet,data.table,ca
               ggfortify,zoo,plotly,xts,tseries,forecast)
```

2 EDA

2.1 Simply Data Wrangling

First, we load the data from the csv files and format the columns to be the appropriate type for further operations.Then we filter out the cases which happened after 2012. We also merged the similar crime types to reduce the complexity of our data set.

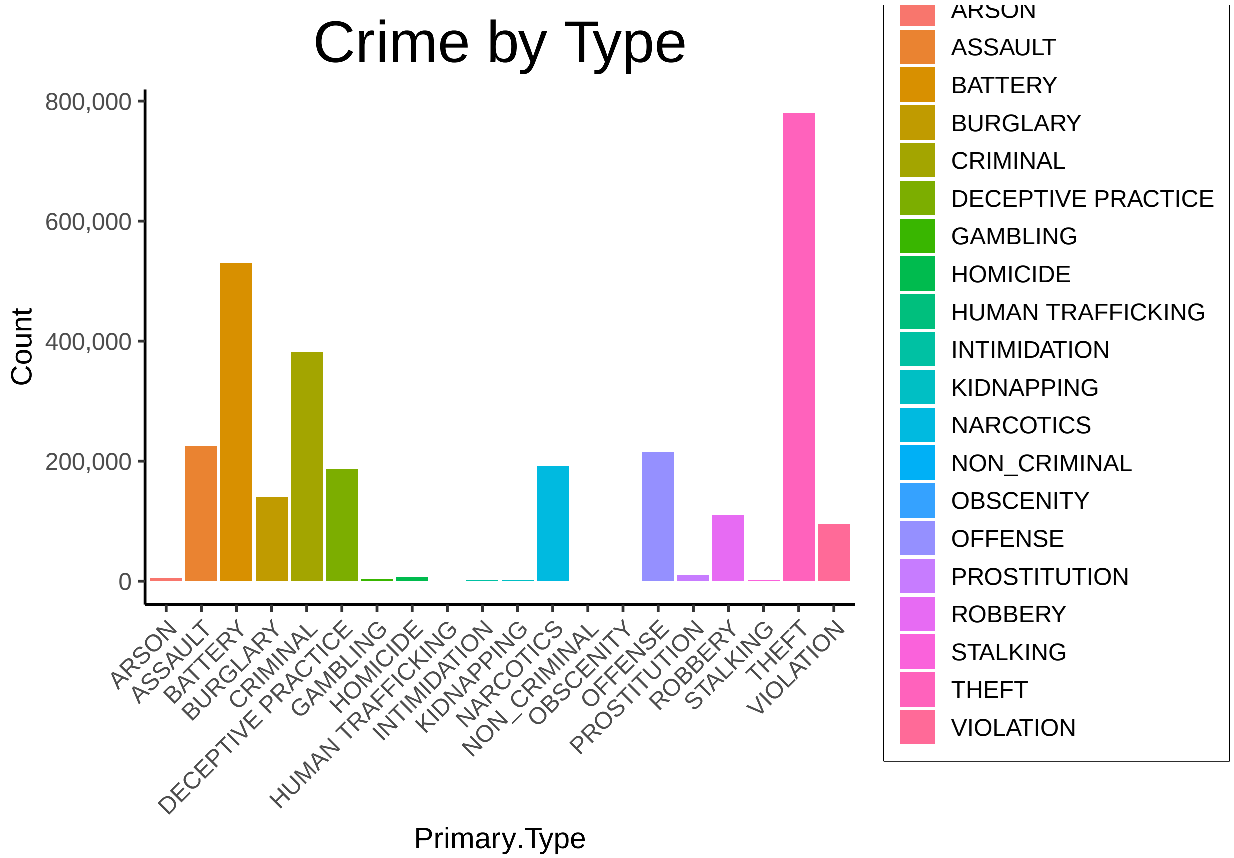
```
crime = read.csv("Crimes_-_2001_to_Present.csv")
unemployment = read.csv("ILCOOK1URN.csv")
crime <- as_tibble(crime)
unemployment <- as_tibble(unemployment)
crime$Year <- as.character(crime$Year)
crime$Latitude <- as.numeric(crime$Latitude)
crime$Longitude <- as.numeric(crime$Longitude)
```

```r
crime$District <- as.character(crime$District)
unemployment$DATE <- format(as.Date(unemployment$DATE, format="%Y-%d-%m"),"%Y")
crime <- crime |>
  filter(Year >= 2012)
crime$Primary.Type[crime$Primary.Type == "ASSAULT" |
                        crime$Primary.Type == "CRIM SEXUAL ASSAULT" |
                        crime$Primary.Type == "CRIMINAL SEXUAL ASSAULT"] <- "ASSAULT"
crime$Primary.Type[crime$Primary.Type == "NON-CRIMINAL" |
                      crime$Primary.Type == "NON - CRIMINAL" |
                      crime$Primary.Type == "NON-CRIMINAL (SUBJECT SPECIFIED)"] <- "NON_CRIMINAL"
crime$Primary.Type[crime$Primary.Type == "CONCEALED CARRY LICENSE VIOLATION" |
                      crime$Primary.Type == "LIQUOR LAW VIOLATION" |
                      crime$Primary.Type == "PUBLIC PEACE VIOLATION"|
                      crime$Primary.Type == "OTHER NARCOTIC VIOLATION"|
                      crime$Primary.Type == "INTERFERENCE WITH PUBLIC OFFICER"|
                      crime$Primary.Type == "PUBLIC INDECENCY"|
                      crime$Primary.Type == "RITUALISM"|
                      crime$Primary.Type == "WEAPONS VIOLATION"] <- "VIOLATION"
crime$Primary.Type[crime$Primary.Type == "CRIMINAL DAMAGE" |
                      crime$Primary.Type == "CRIMINAL TRESPASS"] <- "CRIMINAL"
crime$Primary.Type[crime$Primary.Type == "MOTOR VEHICLE THEFT" |
                      crime$Primary.Type == "THEFT"] <- "THEFT"
crime$Primary.Type[crime$Primary.Type == "SEX OFFENSE" |
                      crime$Primary.Type == "OTHER OFFENSE"|
                      crime$Primary.Type == "OFFENSE INVOLVING CHILDREN" ] <- "OFFENSE"
```

2.2 Visualizations

2.2.1 By Crime Type

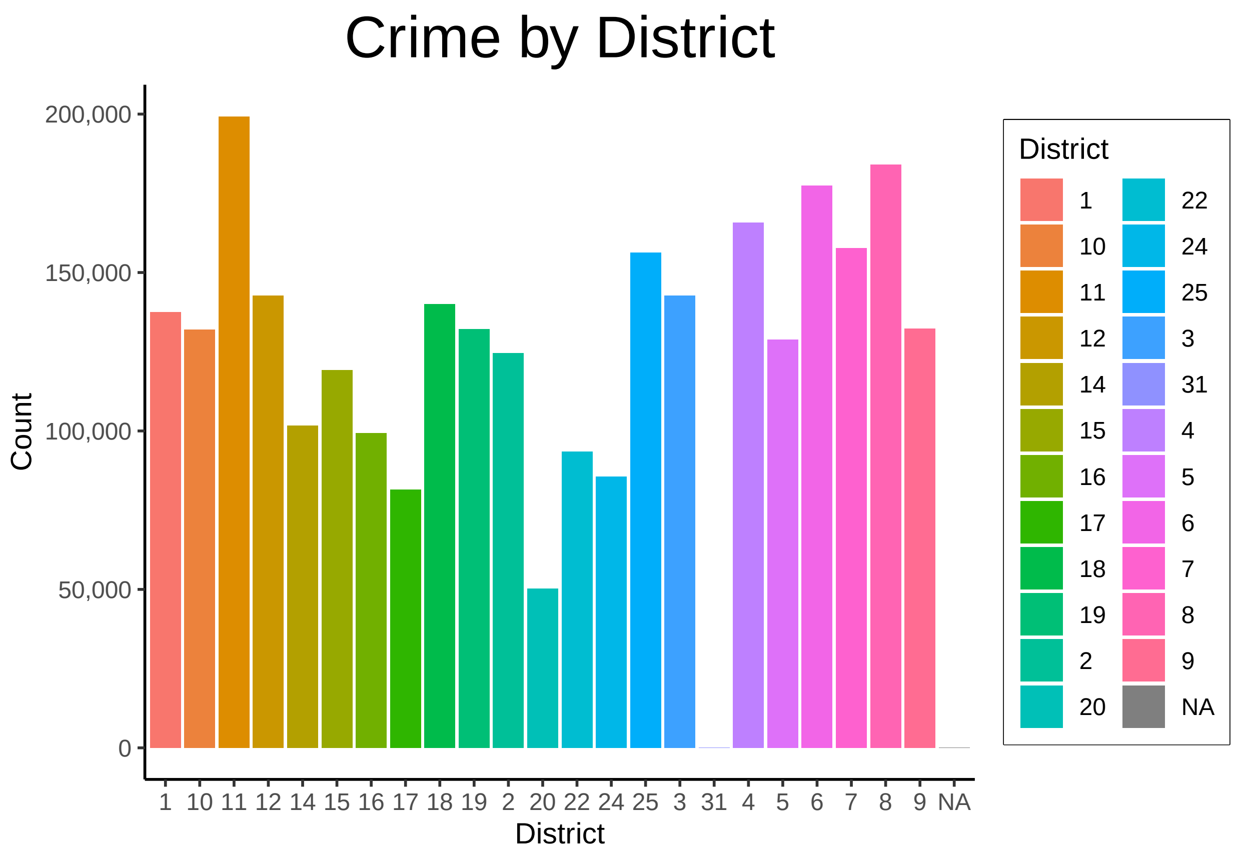
```r
ggplot(crime, aes(Primary.Type, fill = Primary.Type)) +
  geom_bar() +
  theme_classic() +
  theme(legend.key.size = unit(5, 'mm'),
        axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1),
        plot.title = element_text(size=22, hjust = 0.5),
        legend.box.background = element_rect(colour = "black")) +
  ylab("Count") +
  scale_y_continuous(labels = scales::comma) +
  ggtitle("Crime by Type")
```



From the graph above, we can clearly see that theft is the type of crime which happened the most after 2012. Battery and criminal also had really high amounts as compared to the other types of crimes. Many other types of crime such as human trafficking and non criminal all had extremely low quantities.

2.2.2 Plot by District

```
ggplot(crime, aes(District, fill = District)) +
  geom_bar() +
  theme_classic() +
  ggtitle("Crime by District") +
  theme(plot.title = element_text(size=22, hjust = 0.5),
  legend.box.background = element_rect(colour = "black")) +
  scale_y_continuous(labels = scales::comma) +
  ylab("Count")
```




```r
crime |>
  group_by(Block) |>
  summarise(Count = n()) |>
  arrange(desc(Count)) |>
  head(10)
```

```
## # A tibble: 10 x 2
##    Block                              Count
##    <chr>                              <int>
##  1 001XX N STATE ST                    8001
##  2 0000X W TERMINAL ST                 5269
##  3 008XX N MICHIGAN AVE                4423
##  4 0000X N STATE ST                    3615
##  5 076XX S CICERO AVE                  3298
##  6 064XX S DR MARTIN LUTHER KING JR DR 2692
##  7 100XX W OHARE ST                    2577
##  8 033XX W FILLMORE ST                 2506
##  9 011XX S CANAL ST                    2466
## 10 0000X S STATE ST                    2446
```

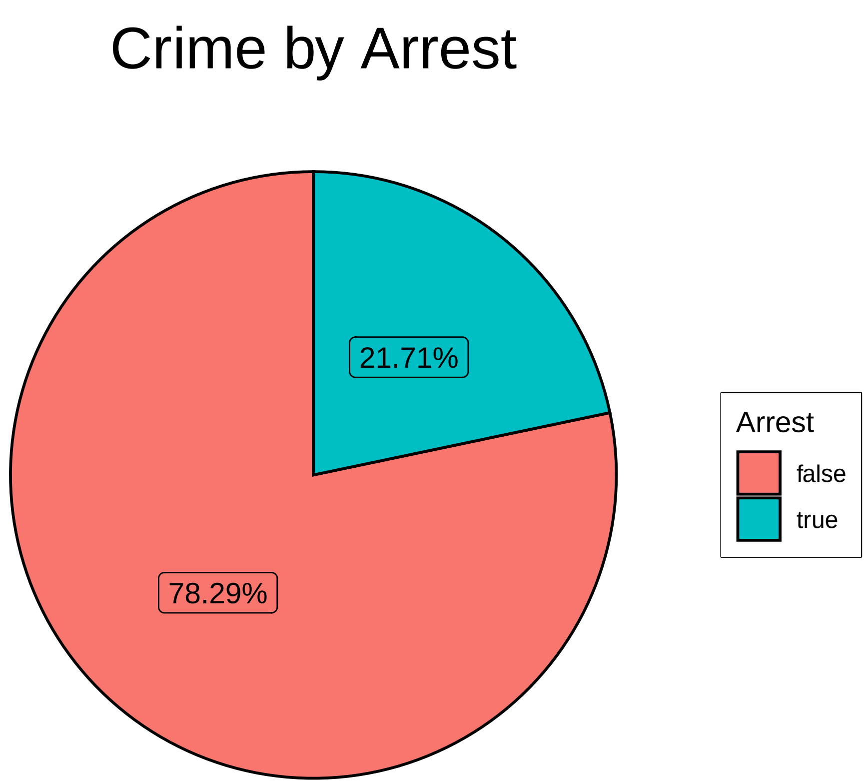
From the plot above, we can observe that 011, 006, and 008 are the three districts with the most crime cases, while 020 is the district with the least cases. Referring to the map of police districts, we can see that southern areas had relatively more crime cases as compared to the northern areas.
More precisely, according to the crime cases after grouping by block, N State Street seems to be the block with the most cases while taking 2 places in the top 4 ranking. Also, the blocks with the most cases are

either located in the urban area or in the transportation facilities area which are both places where large flows of people exist. We can conclude that crimes are more likely to happen in locations where is crowded.

2.2.3 Plot by Arrest

```
dfarrest <- crime |>
  group_by(Arrest) |>
  summarise(count = n()) |>
  mutate(percentage = c("78.29%", "21.71%"))

ggplot(dfarrest, aes(x = "", y = count, fill = Arrest)) +
  geom_col(color = "black") +
  geom_label(aes(label = percentage),
             position = position_stack(vjust = 0.5),
             show.legend = FALSE) +
  coord_polar(theta = "y") +
  theme(panel.grid = element_blank(),
        panel.background = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        plot.title = element_text(size=22, hjust = 0.5),
        legend.box.background = element_rect(colour = "black")) +
  ggtitle("Crime by Arrest")
```

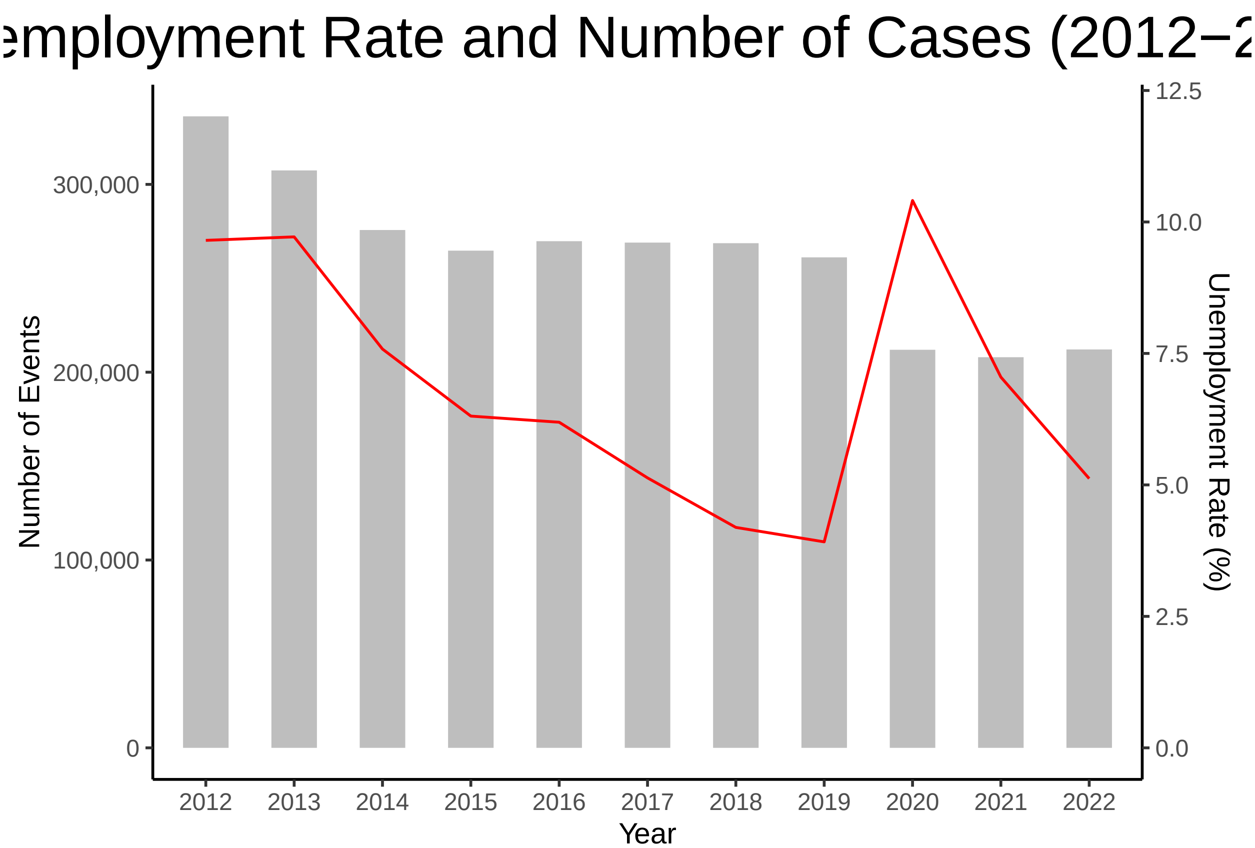


This pie chart is based on if the individual(s) who committed the crime was arrested or not. Over 78 percent of the suspects are not arrested and only about 22 percent of them are arrested.

3 Influential Factors

3.1 Unemployment and Crimes

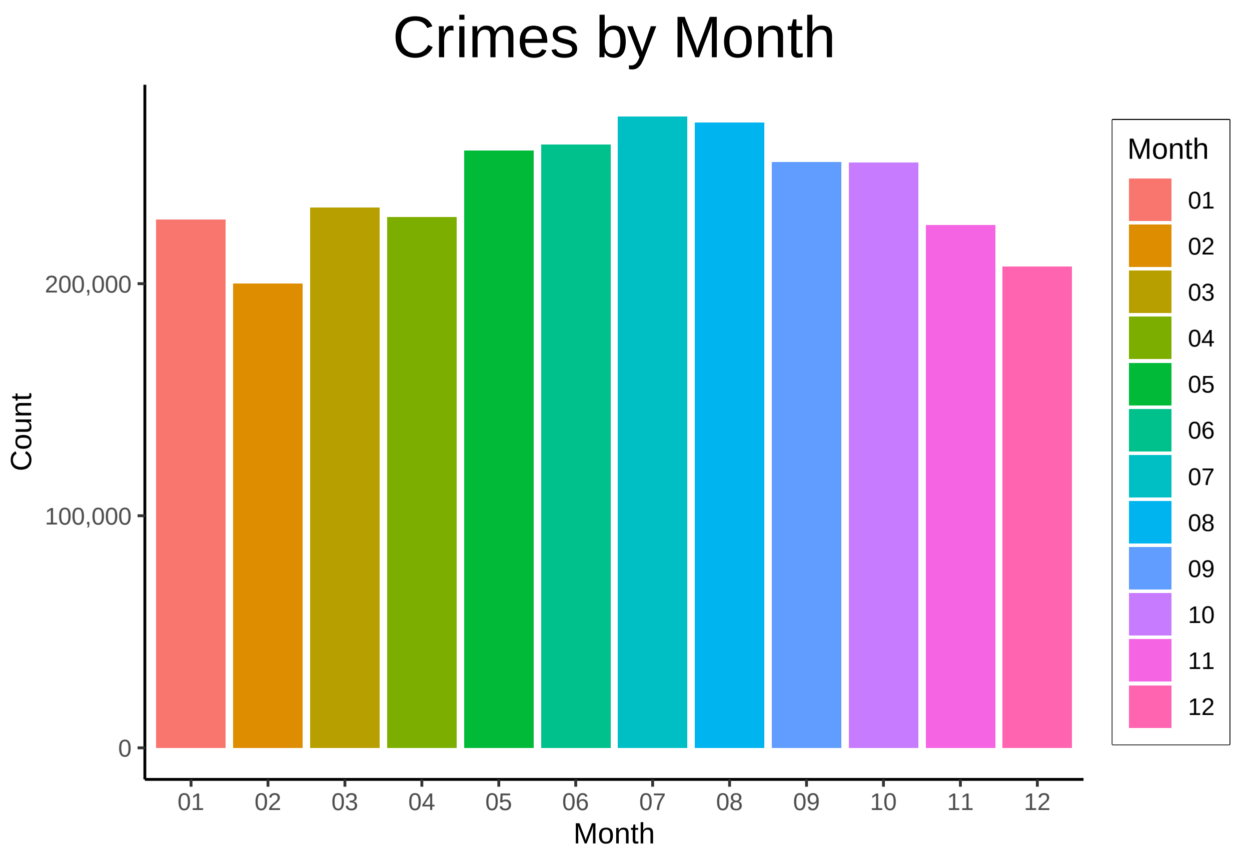
```
newun <- unemployment |>
  filter(DATE >= 2012) |>
  group_by(DATE) |>
  summarise(Mean = mean(ILCOOK1URN))
newcrime <- crime |>
  rename(DATE = Year) |>
  group_by(DATE) |>
  summarise(n = n())
newdf <- as_tibble(data.frame(Year = newcrime$DATE,
                              Cases = newcrime$n,
                              Unemployment = newun$Mean))
ggplot(newdf) +
  geom_segment(aes(x = Year, y = Cases, xend = factor(Year), yend = 0), size = 8, colour = "grey") +
  scale_y_continuous(name = "Number of Events",
                     labels = scales::comma,
                     sec.axis = sec_axis(trans = ~ . / 28000, name = "Unemployment Rate (%)")) +
  geom_line(aes(x = factor(Year), y = Unemployment * 28000, group = 1), colour = "red") +
  ggtitle("Unemployment Rate and Number of Cases (2012-2022)") +
  theme_classic() +
  theme(plot.title = element_text(size = 22, hjust = 0.5),
        legend.box.background = element_rect(colour = "black"))
```



The diagram above shows the relationship between crime cases and unemployment. The y-axis on the left refers to the amount of crime cases and the right one refers to the unemployment rate. We can clearly see that the number of crime cases is positively related with the unemployment rate, except for the year 2020 in which the Covid-19 pademic happened which gave the economic a big strike.

3.2 Season and Crimes

```
Month <- str_extract(crime$Date, "\\d+")
dfmonth1 <- as.data.frame(Month)
ggplot(dfmonth1) +
  geom_bar(aes(Month, fill = Month)) +
  theme_classic() +
  ggtitle("Crimes by Month") +
  ylab("Count") +
  scale_y_continuous(labels = scales::comma) +
  theme(plot.title = element_text(size=22, hjust = 0.5),
        legend.box.background = element_rect(colour = "black"))
```



We extract the months in which the crime cases happened and form a new data frame. Then we plot the number of crime cases by month. The plot clearly shows that summer is the season that had the most number of cases and winter is the season which had the least number of cases.

4 Modeling

4.1 Data Wrangling

We do a similar step like the first time , but this time cleaning more deeply for further modeling.

4.1.1 Loading and tidying up the data

```
##Import Full data
data <- read.csv("Crimes_-_2001_to_Present.csv")
## subset data for 2012 to 2022
data2012 <- data %>% filter(Year>2011)
## Extracting relevant variables
data2012 <- data2012 %>%
  dplyr::select(Date,Primary.Type,Arrest,Domestic,Beat,
                District,Year,Community.Areas,Census.Tracts,
                Police.Beats)
### Checking structure of data
glimpse(data2012)
```

```
## Rows: 2,884,796
## Columns: 10
## $ Date             <chr> "09/05/2015 01:30:00 PM", "09/04/2015 11:30:00 AM", "0~
## $ Primary.Type     <chr> "BATTERY", "THEFT", "THEFT", "NARCOTICS", "ASSAULT", "~
## $ Arrest           <chr> "false", "false", "false", "true", "false", "false", "~
## $ Domestic         <chr> "true", "false", "true", "false", "true", "false", "fa~
## $ Beat             <int> 924, 1511, 631, 1412, 1522, 614, 1434, 1034, 1222, 824~
## $ District         <int> 9, 15, 6, 14, 15, 6, 14, 10, 12, 8, 8, 16, 5, 2, 14, 6~
## $ Year             <int> 2015, 2015, 2018, 2015, 2015, 2015, 2015, 2015, 2015, ~
## $ Community.Areas <int> 59, 26, NA, 22, 26, 70, 25, 32, 28, NA, 63, 11, 45, 5,~
## $ Census.Tracts    <int> 706, 562, NA, 216, 696, 575, 179, 203, 50, NA, 318, 12~
## $ Police.Beats     <int> 108, 67, NA, 168, 81, 237, 192, 151, 77, NA, 209, 58, ~
```

4.1.2 Checking Missing values

```r
## Over all missing values
sum(is.na(data2012))
```

```
## [1] 125563
```

```r
## Checking missing values in each column
sapply(data2012, function (x) sum(is.na(x)))
```

```
##            Date    Primary.Type          Arrest        Domestic            Beat
##               0               0               0               0               0
##        District            Year Community.Areas   Census.Tracts    Police.Beats
##               1               0           42165           41597           41800
```

We do find missing values are there in four columns:Community.Areas, Census.Tracts, Police.Beats, and Distrct. So we replace the missing values with median(because its discrete data).

```r
## Community Areas
data2012$Community.Areas <- ifelse(is.na(data2012$Community.Areas),
                                   median(data2012$Community.Areas,na.rm = T),
                                   data2012$Community.Areas)
## Census Tracts
data2012$Census.Tracts <- ifelse(is.na(data2012$Census.Tracts),
                                   median(data2012$Census.Tracts,na.rm = T),
                                   data2012$Census.Tracts)
## Police Beats
data2012$Police.Beats <- ifelse(is.na(data2012$Police.Beats),
                                   median(data2012$Police.Beats,na.rm = T),
                                   data2012$Police.Beats)
## remove any other missing value
data2012 <- na.omit(data2012)
## Checking missing values again for confirmation
sapply(data2012, function (x) sum(is.na(x)))
```

```
##            Date    Primary.Type          Arrest        Domestic            Beat
##               0               0               0               0               0
##        District            Year Community.Areas   Census.Tracts    Police.Beats
##               0               0               0               0               0
```

There are no more missing value among this data.

4.1.3 Encoding and breaking categorical variables to dummies

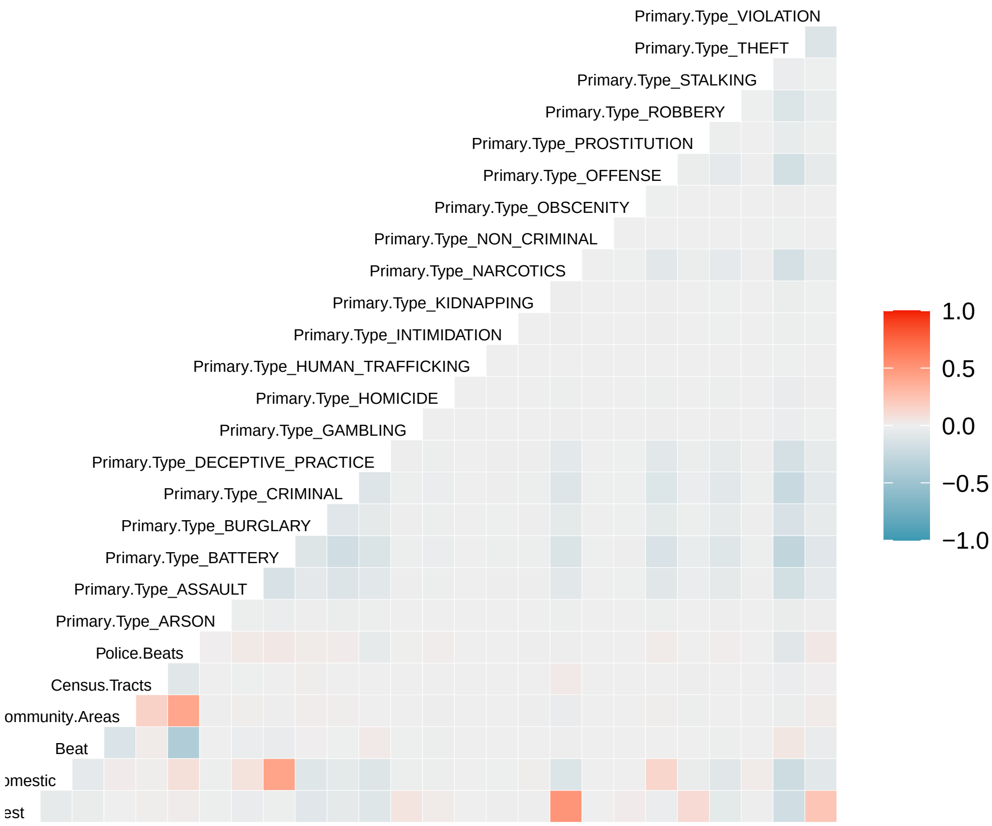
We can use dummies as numeric variable, and each category can be treated as dummy

```
## Convert Arrest to dummy (True = 1, and False = 0)
data2012$Arrest <- ifelse(data2012$Arrest == "false",0,1)
## Convert Domestic to dummy (True = 1, and False = 0)
data2012$Domestic <- ifelse(data2012$Domestic == "false",0,1)
## Converting each "Primary.Type" categories to multiple dummies
## Merging some categories
data2012$Primary.Type[data2012$Primary.Type == "ASSAULT" |
                        data2012$Primary.Type == "CRIM SEXUAL ASSAULT" |
                        data2012$Primary.Type == "CRIMINAL SEXUAL ASSAULT"] <- "ASSAULT"
data2012$Primary.Type[data2012$Primary.Type == "NON-CRIMINAL" |
                      data2012$Primary.Type == "NON - CRIMINAL" |
                      data2012$Primary.Type == "NON-CRIMINAL (SUBJECT SPECIFIED)"] <- "NON_CRIMINAL"
data2012$Primary.Type[data2012$Primary.Type == "CONCEALED CARRY LICENSE VIOLATION" |
                      data2012$Primary.Type == "LIQUOR LAW VIOLATION" |
                      data2012$Primary.Type == "PUBLIC PEACE VIOLATION"|
                      data2012$Primary.Type == "OTHER NARCOTIC VIOLATION"|
                      data2012$Primary.Type == "INTERFERENCE WITH PUBLIC OFFICER"|
                      data2012$Primary.Type == "PUBLIC INDECENCY"|
                      data2012$Primary.Type == "RITUALISM"|
                      data2012$Primary.Type == "WEAPONS VIOLATION"] <- "VIOLATION"
data2012$Primary.Type[data2012$Primary.Type == "CRIMINAL DAMAGE" |
                      data2012$Primary.Type == "CRIMINAL TRESPASS"] <- "CRIMINAL"
data2012$Primary.Type[data2012$Primary.Type == "MOTOR VEHICLE THEFT" |
                      data2012$Primary.Type == "THEFT"] <- "THEFT"
data2012$Primary.Type[data2012$Primary.Type == "SEX OFFENSE" |
                      data2012$Primary.Type == "OTHER OFFENSE"|
                      data2012$Primary.Type == "OFFENSE INVOLVING CHILDREN" ] <- "OFFENSE"
# Creating dummies
data2012 <- data2012 %>% fastDummies::dummy_cols(select_columns = "Primary.Type")
```

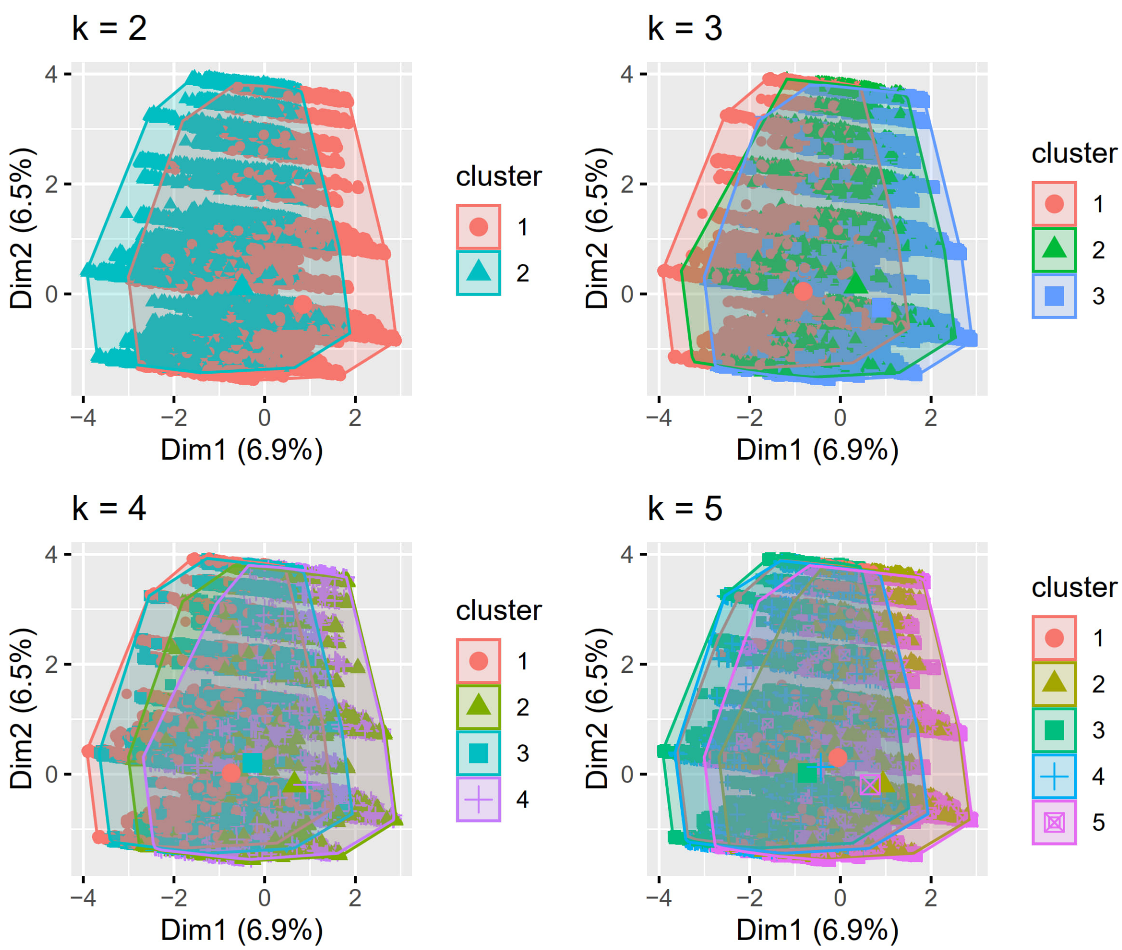
4.2 Clustering and K-means

In order to detect the crime in districts, we need to use K-means.

```
## Clustering
## First there is need to scale data, to equalize magnitude
scaled_data <- data2012[,-c(1,2,6,7)]
## Correlation matrix
ggcorr(scaled_data,size=2,hjust=1,vjust=1)
```




```r
## K-means
# Trying Kmeans with different number of K values
k2 <- kmeans(scaled_data, centers = 2, nstart = 25)
k3 <- kmeans(scaled_data, centers = 3, nstart = 25)
k4 <- kmeans(scaled_data, centers = 4, nstart = 25)
k5 <- kmeans(scaled_data, centers = 5, nstart = 25)
# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = scaled_data) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = scaled_data) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = scaled_data) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = scaled_data) + ggtitle("k = 5")
## Plot all Combine
grid.arrange(p1, p2, p3, p4, nrow = 2)
```



We have tried different values of k from 2 to 5, and applied k-means on that data. We can see that almost all clusters are overlapping, but some of points in K = 2, are found to have many different points in different clusters.

4.2.1 Checking outputs for K=2

```
## Plot CLuster mean for each variable
clust_means <- as.data.frame(k2$centers) %>%
  pivot_longer(1:26,names_to = "Variable",values_to = "Cluster1 Mean") %>%
  mutate(`Cluster1 Mean`=round(`Cluster1 Mean`,3))
cbind.data.frame(clust_means[1:26,],`Cluster2 Mean`=clust_means$`Cluster1 Mean`[27:52])
```

```
##                          Variable Cluster1 Mean Cluster2 Mean
## 1                          Arrest         0.187         0.235
## 2                        Domestic         0.131         0.183
## 3                            Beat      1930.485       687.809
## 4                Community.Areas         33.978        41.630
## 5                  Census.Tracts        391.661       374.337
## 6                   Police.Beats        107.458       174.373
## 7               Primary.Type_ARSON         0.001         0.002
```

```
## 8              Primary.Type_ASSAULT         0.068         0.084
## 9              Primary.Type_BATTERY         0.163         0.196
## 10            Primary.Type_BURGLARY         0.050         0.047
## 11            Primary.Type_CRIMINAL         0.130         0.133
## 12 Primary.Type_DECEPTIVE PRACTICE         0.081         0.055
## 13            Primary.Type_GAMBLING         0.001         0.001
## 14            Primary.Type_HOMICIDE         0.001         0.003
## 15  Primary.Type_HUMAN TRAFFICKING         0.000         0.000
## 16        Primary.Type_INTIMIDATION         0.001         0.001
## 17          Primary.Type_KIDNAPPING         0.001         0.001
## 18           Primary.Type_NARCOTICS         0.050         0.077
## 19        Primary.Type_NON_CRIMINAL         0.000         0.000
## 20           Primary.Type_OBSCENITY         0.000         0.000
## 21             Primary.Type_OFFENSE         0.073         0.076
## 22        Primary.Type_PROSTITUTION         0.002         0.005
## 23             Primary.Type_ROBBERY         0.033         0.041
## 24            Primary.Type_STALKING         0.001         0.001
## 25               Primary.Type_THEFT         0.321         0.241
## 26           Primary.Type_VIOLATION         0.023         0.039
```

4.2.2 Comparing districts with clusters

```
## Adding cluster variable in original data
data2012$cluster <- k2$cluster
## Comparing clusters with District
table(data2012$District,data2012$cluster)
```

```
##
##           1      2
##    1     90 137394
##    2    709 123887
##    3      1 142711
##    4      1 165740
##    5      2 128811
##    6      2 177379
##    7      0 157714
##    8      1 184027
##    9     91 132156
##    10     5 132063
##    11     4 199290
##    12  7362 135419
##    14 101766      4
##    15 119283      1
##    16  99400      4
##    17  81512      1
##    18 140001     45
##    19 132089      1
##    20  50335      1
##    22  93494     16
##    24  85561      1
##    25 156335      2
##    31     56     28
```

It shows that there are 91 times crime happened in cluster 1 of district 1,and 137237 times classified cluster 2, which add together becomes 137328. District 2 will be 124456,District 3 will be 165585, District 4 will be 128739, District 6 will be177211, District 7 will be 157593,District 8 will be 183847,District 9 will be 132126,District 10 will be 131960,District 11 will be 199145,District 12 will be 142589,District 14 will be 101692,District 15 will be119203,District 16 will be 99285,District 17 will be 81432,District 20 will be 50287,District 22 will be 93387,District 24 will be 85464,District 25 will be 156177,District 31 will be 84.So District 11 has the highest crime.

4.3 Time Series

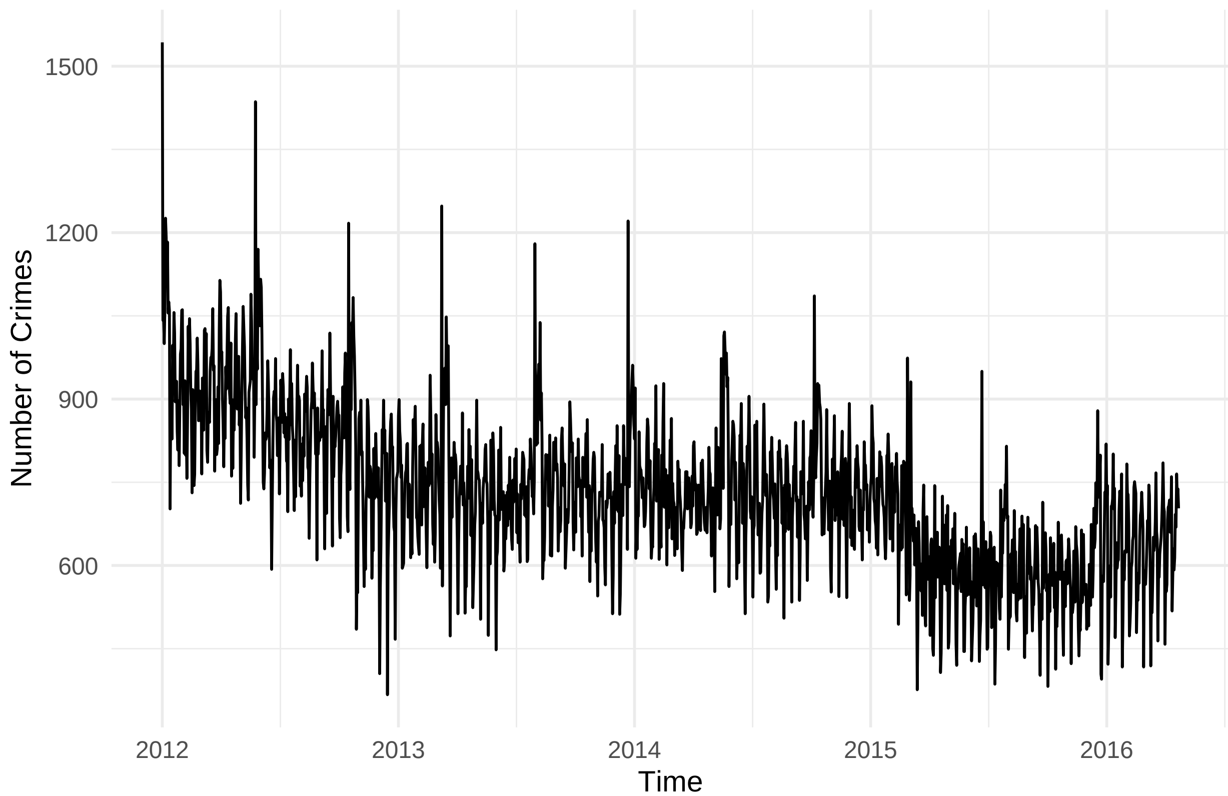
In order to predict number of crimes in the next 100 days, we utilize Time Series Model to achieve it.

```
data <- data %>% filter(Year>2011)
data_time <- data %>%
  dplyr::select(Date) %>% ## Select date variable
  mutate(Date = as.Date(Date, format="%d/%m/%Y")) %>%
  group_by(Date) %>%
  summarise(No.of.Crimes = n())
## Checking structure of data
glimpse(data_time)
```

```
## Rows: 1,573
## Columns: 2
## $ Date         <date> 2012-01-01, 2012-01-02, 2012-01-03, 2012-01-04, 2012-01-~
## $ No.of.Crimes <int> 1543, 1042, 1043, 1000, 1066, 1226, 1191, 1108, 1183, 105~
```

4.3.1 Creating and Plotting Timeseries

```
## Data
Crime_ts <- ts(data_time$No.of.Crimes[-1573], frequency = 365,start=c(2012,01,01))
## Time Plot
autoplot(Crime_ts)+
   theme_minimal()+
   labs(x= "Time",y="Number of Crimes")
```



The series looks stationary, but there is need to confirm stationary through ADF test. Also, there is a clear element seasonality in the series.

4.3.2 ADF Test

```
adf.test(Crime_ts)
```

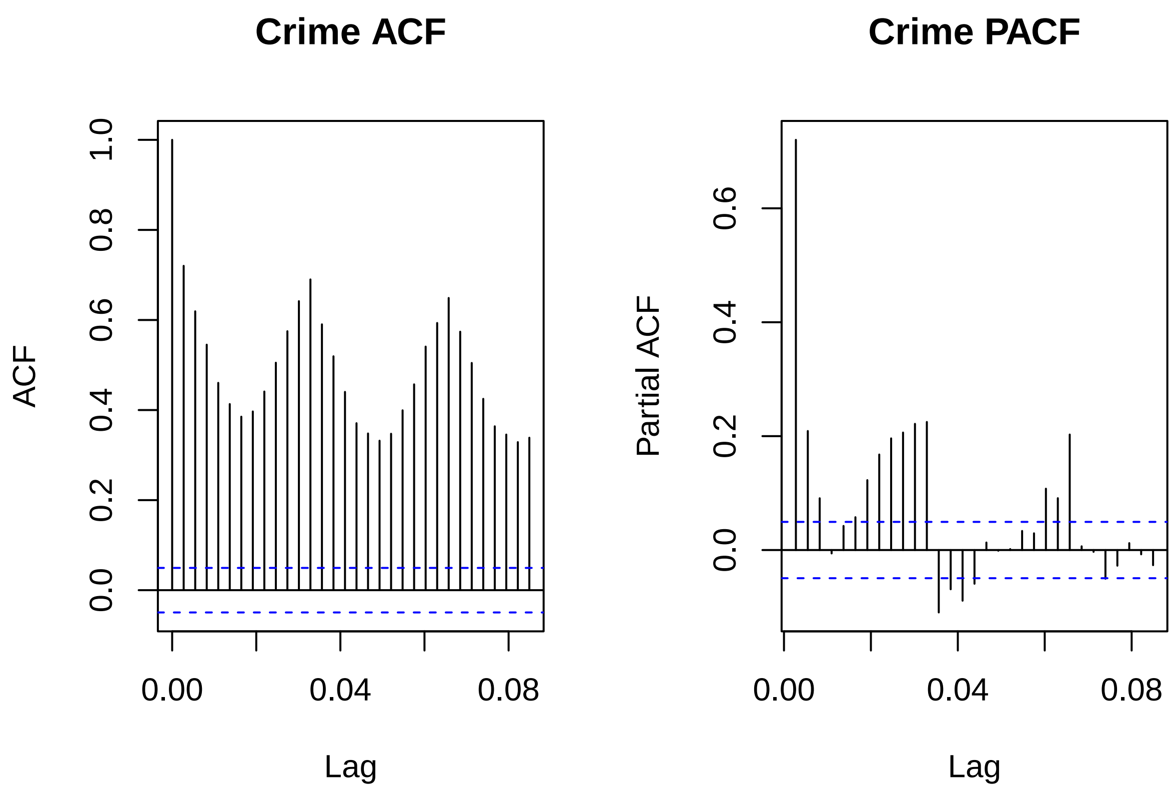
```
## Warning in adf.test(Crime_ts): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Crime_ts
## Dickey-Fuller = -5.0825, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

The Test p-value = 0.01 (less than 0.05), which shows that the null hypothesis of non-stationary is rejected, and the time series is stationary. Now, there is need to check the lag order.

4.3.3 Exploring the Lag order through ACF, and PACF

```r
## For combining Plots
par(mfrow=c(1,2), mar = c(5, 4, 4, 2) + 0.1)
## ACF
acf(Crime_ts,main="Crime ACF")
## PACF
pacf(Crime_ts,main="Crime PACF")
```



It can be observed that the spikes are going outside the limits, at each seasonal lag in proper pattern. Hence, there is a non-zero correlation at seasonal lags, and showing a seasonal patteren. We can use auto.arima function to fit our correct model.

```r
## Fitting model
Model_Arima <- auto.arima(Crime_ts)
## Model summary
summary(Model_Arima)
```

```
## Series: Crime_ts
## ARIMA(5,1,3)
##
## Coefficients:
##          ar1     ar2      ar3      ar4      ar5      ma1      ma2     ma3
##       0.2034  0.7471  -0.3366  -0.2804  -0.3164  -0.7830  -0.7086  0.8843
## s.e.  0.0316  0.0324   0.0308   0.0244   0.0247   0.0219   0.0342  0.0242
##
## sigma^2 = 7176:  log likelihood = -9200.87
```

```
## AIC=18419.73   AICc=18419.85   BIC=18467.97
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE    MAPE      MASE        ACF1
## Training set -1.150817 84.46859 60.16745 -1.29402 8.72615 0.4006958 -0.04046757
```

So, the Auto.arima suggested a better model, but not suggested a seasonal model. So, it suggested a ARIMA model with AR = 2, Diff = 1, and MA= 1, that is ARIMA (2,1,1).

4.3.4 Forecasting

The major purpose of the time-series was to forecast the number of crimes for future. Hence, we have Model fitted with Auto.arima. There is need to forecast, and we are going to for next 100 days.

```
## Forecasting
Forecast_crime <- forecast(Model_Arima,h=100)
## Print forecasting values
Forecast_crime
```

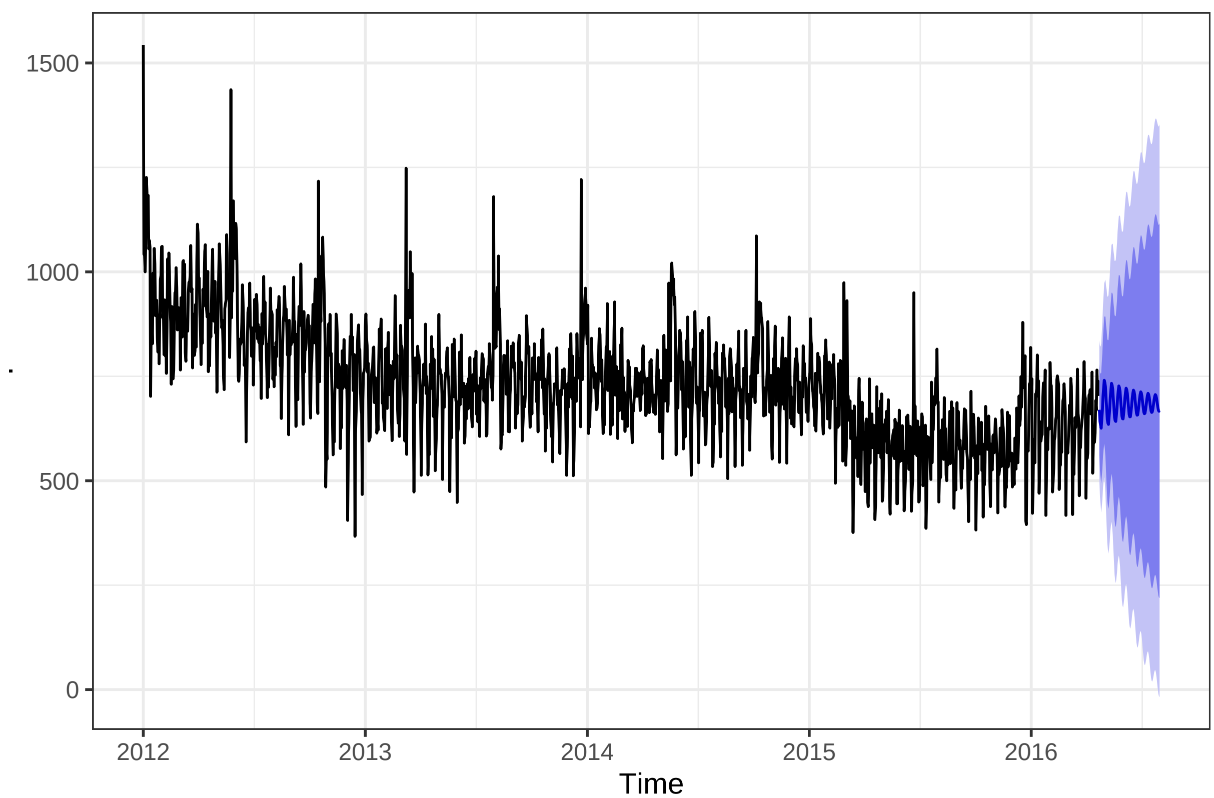
```
##           Point Forecast    Lo 80     Hi 80      Lo 95     Hi 95
## 2016.3068       669.8320 561.2699  778.3941 503.800652  835.8634
## 2016.3096       641.7909 524.0279  759.5539 461.687985  821.8939
## 2016.3123       632.8650 509.4208  756.3092 444.073317  821.6567
## 2016.3151       625.6989 493.3550  758.0428 423.296335  828.1014
## 2016.3178       647.3866 510.6973  784.0759 438.338425  856.4348
## 2016.3205       667.8053 528.7353  806.8754 455.116057  880.4946
## 2016.3233       701.9482 559.1292  844.7672 483.525446  920.3710
## 2016.3260       721.6799 575.3666  867.9931 497.913101  945.4466
## 2016.3288       740.5141 588.8122  892.2160 508.506095  972.5221
## 2016.3315       735.0055 576.6034  893.4076 492.750344  977.2606
## 2016.3342       725.2802 557.9528  892.6077 469.374929  981.1856
## 2016.3370       696.5112 520.3876  872.6348 427.153395  965.8691
## 2016.3397       673.7248 488.4665  858.9831 390.396702  957.0529
## 2016.3425       646.4562 454.1006  838.8118 352.273680  940.6387
## 2016.3452       638.0407 439.7766  836.3049 334.821875  941.2596
## 2016.3479       634.7708 432.6557  836.8859 325.662478  943.8792
## 2016.3507       652.4896 447.2664  857.7128 338.627855  966.3514
## 2016.3534       671.3387 463.8853  878.7920 354.066181  988.6112
## 2016.3562       700.4982 490.5430  910.4534 379.399394 1021.5970
## 2016.3589       718.1256 505.4862  930.7651 392.921654 1043.3296
## 2016.3616       733.2173 516.6590  949.7755 402.020044 1064.4145
## 2016.3644       728.7489 507.4641  950.0337 390.322969 1067.1748
## 2016.3671       719.0413 491.6081  946.4746 371.212185 1066.8705
## 2016.3699       694.4797 460.7636  928.1958 337.041778 1051.9177
## 2016.3726       673.9272 433.7461  914.1082 306.601941 1041.2524
## 2016.3753       651.1428 405.6711  896.6145 275.726186 1026.5594
## 2016.3781       643.5579 393.6173  893.4984 261.306786 1025.8090
## 2016.3808       641.8698 388.7918  894.9478 254.820388 1028.9192
## 2016.3836       657.0637 401.4028  912.7245 266.064117 1048.0632
## 2016.3863       674.3373 416.6668  932.0077 280.264329 1068.4102
## 2016.3890       699.1058 439.2486  958.9630 301.688505 1096.5231
## 2016.3918       714.8069 452.5711  977.0426 313.751874 1115.8619
## 2016.3945       726.9640 461.4825  992.4454 320.945153 1132.9828
```

```
## 2016.3973       723.1783 453.8131  992.5435 311.219824 1135.1368
## 2016.4000       713.7954 439.5770  988.0137 294.414609 1133.1762
## 2016.4027       692.7272 413.4962  971.9581 265.680268 1119.7740
## 2016.4055       674.3301 389.9975  958.6627 239.480917 1109.1792
## 2016.4082       655.2218 366.5743  943.8693 213.773488 1096.6701
## 2016.4110       648.5119 356.1824  940.8414 201.432463 1095.5913
## 2016.4137       647.9404 352.8711  943.0096 196.670905 1099.2098
## 2016.4164       661.0679 363.6995  958.4362 206.282258 1115.8535
## 2016.4192       676.7482 377.4840  976.0123 219.063202 1134.4331
## 2016.4219       697.8645 396.5794  999.1497 237.088624 1158.6404
## 2016.4247       711.7381 408.2456 1015.2306 247.586384 1175.8898
## 2016.4274       721.5574 415.2097 1027.9051 253.039071 1190.0758
## 2016.4301       718.2610 408.5411 1027.9809 244.585280 1191.9367
## 2016.4329       709.3743 395.5858 1023.1629 229.476123 1189.2725
## 2016.4356       691.2274 373.2224 1009.2324 204.880786 1177.5740
## 2016.4384       674.8643 352.6110  997.1176 182.020417 1167.7081
## 2016.4411       658.7877 332.8513  984.7240 160.310983 1157.2643
## 2016.4438       652.9365 323.8238  982.0492 149.602057 1156.2710
## 2016.4466       653.1438 321.5550  984.7327 146.022433 1160.2653
## 2016.4493       664.5562 330.8498  998.2627 154.196251 1174.9162
## 2016.4521       678.6868 343.1576 1014.2161 165.539171 1191.8345
## 2016.4548       696.7444 359.2989 1034.1899 180.666047 1212.8228
## 2016.4575       708.9255 369.3894 1048.4615 189.649874 1228.2010
## 2016.4603       716.8715 374.7436 1058.9993 193.632099 1240.1109
## 2016.4630       713.9364 368.7926 1059.0802 186.084467 1241.7883
## 2016.4658       705.6414 356.9647 1054.3180 172.386437 1238.8963
## 2016.4685       689.9577 337.6209 1042.2946 151.105007 1228.8105
## 2016.4712       675.4767 319.4812 1031.4722 131.028596 1219.9249
## 2016.4740       661.9155 302.6855 1021.1455 112.520613 1211.3103
## 2016.4767       656.8726 294.8217 1018.9235 103.163547 1210.5817
## 2016.4795       657.6121 293.2769 1021.9473 100.409486 1214.8148
## 2016.4822       667.5827 301.2584 1033.9071 107.337946 1227.8276
## 2016.4849       680.2448 312.1519 1048.3378 117.295254 1243.1944
## 2016.4877       695.7250 325.7906 1065.6594 129.959199 1261.4908
## 2016.4904       706.3651 334.4299 1078.3002 137.539312 1275.1908
## 2016.4932       712.8023 338.4696 1087.1349 140.309875 1285.2947
## 2016.4959       710.1446 333.0641 1087.2252 133.449674 1286.8395
## 2016.4986       702.4848 322.2641 1082.7056 120.987451 1283.9822
## 2016.5014       688.8932 305.4261 1072.3603 102.430892 1275.3555
## 2016.5041       676.1295 289.4366 1062.8224  84.733693 1267.5253
## 2016.5068       664.6661 275.0756 1054.2567  68.838781 1260.4935
## 2016.5096       660.3629 268.2159 1052.5098  60.625874 1260.0998
## 2016.5123       661.4545 267.1714 1055.7376  58.450541 1264.4584
## 2016.5151       670.1996 274.0248 1066.3744  64.302496 1276.0967
## 2016.5178       681.4950 283.5962 1079.3938  72.961266 1290.0287
## 2016.5205       694.7919 295.1103 1094.4735  83.531658 1306.0522
## 2016.5233       704.0467 302.4387 1105.6547  89.840251 1318.2532
## 2016.5260       709.2634 305.4099 1113.1168  91.622848 1326.9039
## 2016.5288       706.8284 300.4390 1113.2179  85.309363 1328.3475
## 2016.5315       699.8130 290.5848 1109.0411  73.952558 1325.6734
## 2016.5342       688.0088 275.8552 1100.1624  57.674198 1318.3434
## 2016.5370       676.7956 261.7471 1091.8440  42.033731 1311.5574
## 2016.5397       667.0899 249.4064 1084.7733  28.298103 1305.8816
## 2016.5425       663.4495 243.4148 1083.4843  21.061801 1305.8373
```

```
## 2016.5452       664.7622 242.7105 1086.8139  19.289771 1310.2345
## 2016.5479       672.4556 248.5900 1096.3211  24.209159 1320.7020
## 2016.5507       682.4956 256.9453 1108.0460  31.672498 1333.3187
## 2016.5534       693.9351 266.6511 1121.2190  40.460631 1347.4095
## 2016.5562       701.9565 272.8108 1131.1022  45.634833 1358.2782
## 2016.5589       706.1822 274.9152 1137.4491  46.616333 1365.7480
## 2016.5616       703.9343 270.3044 1137.5641  40.754632 1367.1139
## 2016.5644       697.5496 261.3205 1133.7788  30.394720 1364.7046
## 2016.5671       687.2807 248.3821 1126.1792  16.043219 1358.5181
## 2016.5699       677.4561 235.9236 1118.9885   2.190496 1352.7216
## 2016.5726       669.2285 225.2715 1113.1855  -9.745160 1348.2021
## 2016.5753       666.1734 220.0290 1112.3178 -16.145591 1348.4924
## 2016.5781       667.6118 219.5493 1115.6744 -17.640638 1352.8643
```

4.4 Plotting forecasted values

```
Crime_ts %>% autoplot() +
  autolayer(Forecast_crime) +
  theme_bw()
```



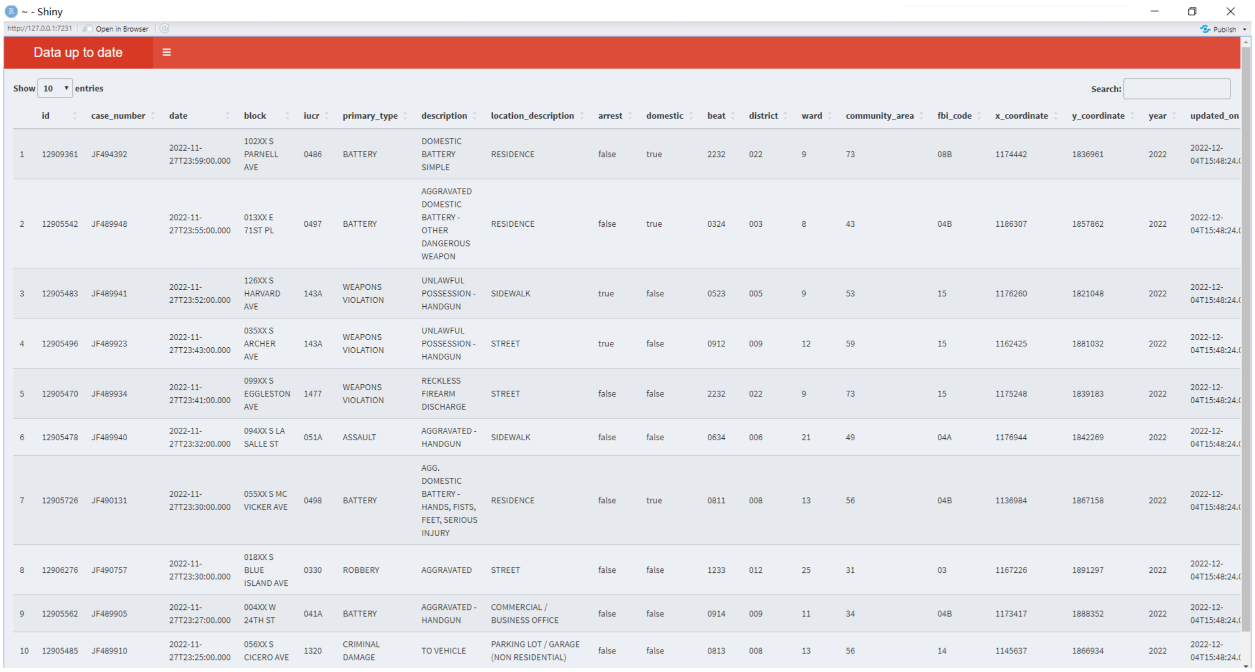
From the dataframe above, 'Point Forecast' shows us the prediction number of crimes in the next 100 days, also the blue line of the plot shows above also means that. We can see, in the next 100 days, the crimes are gradually rise.

5 Shiny App

5.1 Latest data access

```r
library(httr)
library(shinydashboard)
library(shiny)
library(data.table)
library(shinyjs)
library(shinyWidgets)
library(tidyverse)
apiKey <- "XXXXXXXXXXX"
result <- GET("https://data.cityofchicago.org/resource/crimes.json",
              add_headers(Authorization = paste("Key", apiKey)))
ui <- dashboardPage(skin = "red",
  dashboardHeader(title = "Data up to date"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Access data",tabName = "a"))),
  dashboardBody(useShinyjs(),
    tabItems(
      tabItem(tabName = "a",DT::dataTableOutput("table_names"))
      )))
server <- function(input, output) {
  url <- reactive({
  paste("https://data.cityofchicago.org/resource/crimes.json")
  })

  result <- reactive ({
    r_json <- jsonlite::fromJSON(url(), flatten = TRUE)
  })
   output$table_names <- DT::renderDataTable({
    result()
   })
   addClass(selector = "body", class = "sidebar-collapse")}
shinyApp(ui, server)
```



~ - Shiny
http://127.0.0.1:7231 Open in Browser
Publish
Data up to date
Show 10 entries
Search:

	id	case_number	date	block	iucr	primary_type	description	location_description	arrest	domestic	beat	district	ward	community_area	fbi_code	x_coordinate	y_coordinate	year	updated_on
1	12909361	JF494392	2022-11-27T23:59:00.000	102XX S PARNELL AVE	0486	BATTERY	DOMESTIC BATTERY SIMPLE	RESIDENCE	false	true	2232	022	9	73	08B	1174442	1836961	2022	2022-12-04T15:48:24.(
2	12905542	JF489948	2022-11-27T23:55:00.000	013XX E 71ST PL	0497	BATTERY	AGGRAVATED DOMESTIC BATTERY - OTHER DANGEROUS WEAPON	RESIDENCE	false	true	0324	003	8	43	04B	1186307	1857862	2022	2022-12-04T15:48:24.(
3	12905483	JF489941	2022-11-27T23:52:00.000	126XX S HARVARD AVE	143A	WEAPONS VIOLATION	UNLAWFUL POSSESSION - HANDGUN	SIDEWALK	true	false	0523	005	9	53	15	1176260	1821048	2022	2022-12-04T15:48:24.(
4	12905496	JF489923	2022-11-27T23:43:00.000	035XX S ARCHER AVE	143A	WEAPONS VIOLATION	UNLAWFUL POSSESSION - HANDGUN	STREET	true	false	0912	009	12	59	15	1162425	1881032	2022	2022-12-04T15:48:24.(
5	12905470	JF489934	2022-11-27T23:41:00.000	099XX S EGGLESTON AVE	1477	WEAPONS VIOLATION	RECKLESS FIREARM DISCHARGE	STREET	false	false	2232	022	9	73	15	1175248	1839183	2022	2022-12-04T15:48:24.(
6	12905478	JF489940	2022-11-27T23:32:00.000	094XX S LA SALLE ST	051A	ASSAULT	AGGRAVATED - HANDGUN	SIDEWALK	false	false	0634	006	21	49	04A	1176944	1842269	2022	2022-12-04T15:48:24.(
7	12905726	JF490131	2022-11-27T23:30:00.000	055XX S MC VICKER AVE	0498	BATTERY	AGG. DOMESTIC BATTERY - HANDS, FISTS, FEET, SERIOUS INJURY	RESIDENCE	false	true	0811	008	13	56	04B	1136984	1867158	2022	2022-12-04T15:48:24.(
8	12906276	JF490757	2022-11-27T23:30:00.000	018XX S BLUE ISLAND AVE	0330	ROBBERY	AGGRAVATED	STREET	false	false	1233	012	25	31	03	1167226	1891297	2022	2022-12-04T15:48:24.(
9	12905562	JF489905	2022-11-27T23:27:00.000	004XX W 24TH ST	041A	BATTERY	AGGRAVATED - HANDGUN	COMMERCIAL / BUSINESS OFFICE	false	false	0914	009	11	34	04B	1173417	1888352	2022	2022-12-04T15:48:24.(
10	12905485	JF489910	2022-11-27T23:25:00.000	056XX S CICERO AVE	1320	CRIMINAL DAMAGE	TO VEHICLE	PARKING LOT / GARAGE (NON RESIDENTIAL)	false	false	0813	008	13	56	14	1145637	1866934	2022	2022-12-04T15:48:24.(

This Shinyapp allows user to access latest data of criminal information in Chicago from the government website.We use api connection to link the data from Internet into our Shinyapp.
(api connection:“https://data.cityofchicago.org/resource/crimes.json”)

5.2 Filter the data and visualization Shinyapp

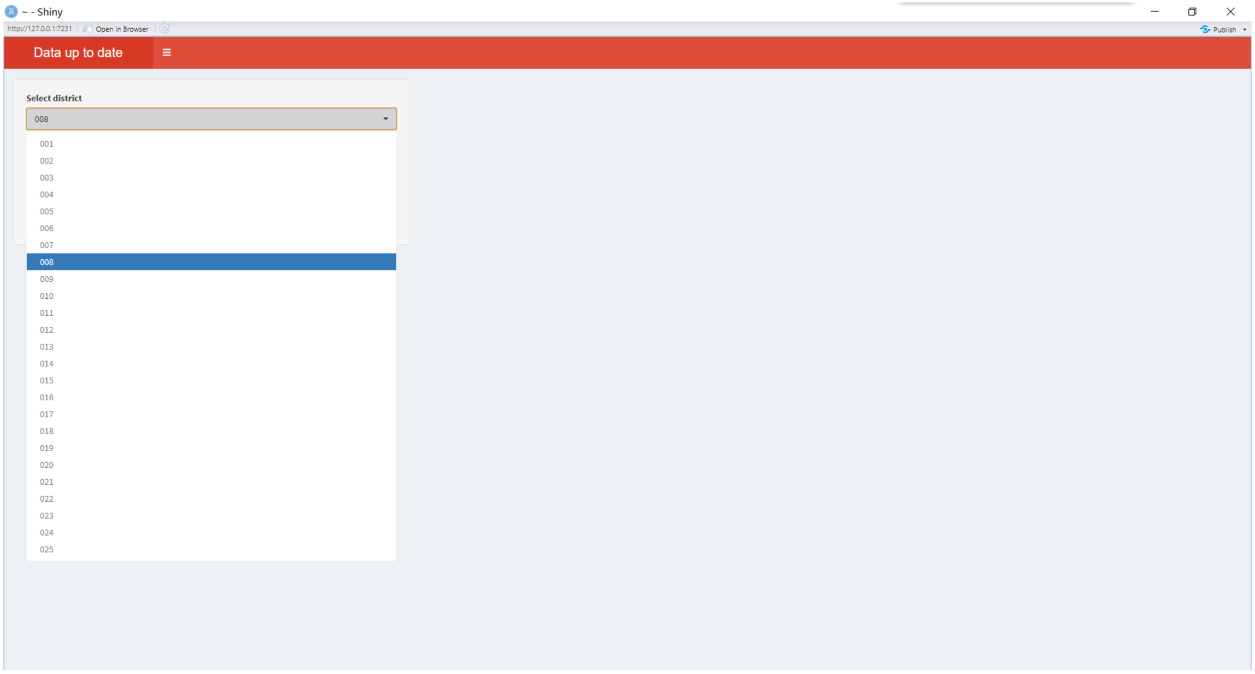
```
apiKey <- "XXXXXXXXXXX"
result <- GET("https://data.cityofchicago.org/resource/crimes.json",
              add_headers(Authorization = paste("Key", apiKey)))
ui <- dashboardPage(skin = "red",
  dashboardHeader(title = "Data up to date"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Plot your file in barchart",tabName = "c"))),
  dashboardBody(useShinyjs(),
    tabItems(
      tabItem(tabName = "c",sidebarLayout(
        sidebarPanel(
          uiOutput("picker1"),
          actionButton("view6", "View Selection"),
          uiOutput("picker"),
          actionButton("view5", "View Selection")
        ),mainPanel(DT::dataTableOutput("table1"),plotOutput("plot1"))
      )
      ))))
server <- function(input, output) {
  url <- reactive({
  paste("https://data.cityofchicago.org/resource/crimes.json")
  })

  result <- reactive ({
```

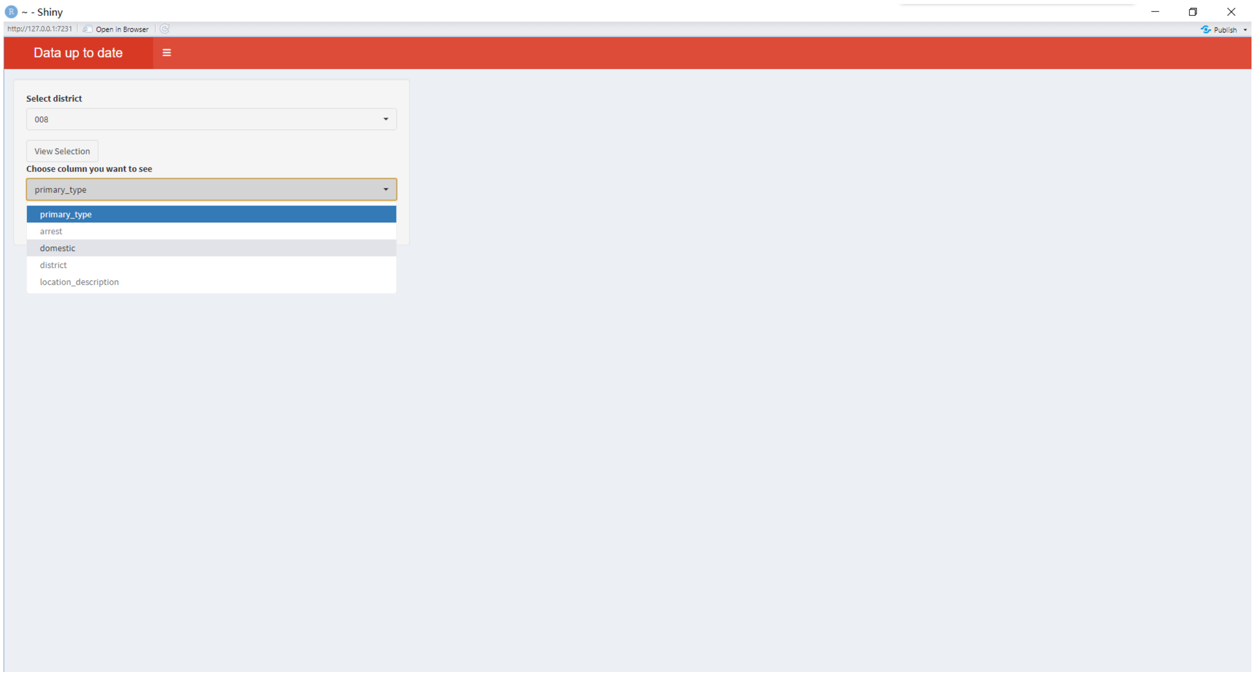
```
  r_json <- jsonlite::fromJSON(url(), flatten = TRUE)
})
 output$table_names <- DT::renderDataTable({
  result()
 })
 addClass(selector = "body", class = "sidebar-collapse")
 datasetInput3 <- eventReactive(input$view5,{
   data2 <- datasetInput4()
  if (input$pick == "primary_type"){
    return(ggplot(data2,aes(primary_type,fill = primary_type)) + geom_bar()+theme_classic()+ theme(leg
  } else if (input$pick == "arrest")
  {
    return(ggplot(data2,aes(arrest,fill = arrest)) + geom_bar()+theme_classic())
  } else if (input$pick == "domestic")
  {
    return(ggplot(data2,aes(domestic,fill = domestic)) + geom_bar()+theme_classic())
  } else if (input$pick == "district")
  {
    return(ggplot(data2,aes(district,fill = district)) + geom_bar()+theme_classic())
  } else if (input$pick == "location_description")
  {
    return(ggplot(data2,aes(location_description,fill = location_description)) + geom_bar()+theme_clas
  }
 })
 output$picker <- renderUI({
  pickerInput(inputId = 'pick',
              label = 'Choose column you want to see',
              choices = c("primary_type","arrest","domestic","district","location_description"),
              options = list(`actions-box` = TRUE),multiple = F)
 })
 datasetInput4 <- eventReactive(input$view6,{
   dat1 <- result()
    if(input$pick2 == "001"){
      data2 <- dat1 %>% filter(district == "001")
    } else if(input$pick2 == "002"){
      data2 <- dat1 %>% filter(district == "002")
    } else if(input$pick2 == "003"){
      data2 <- dat1 %>% filter(district == "003")
    } else if(input$pick2 == "004"){
      data2 <- dat1 %>% filter(district == "004")
    } else if(input$pick2 == "005"){
      data2 <- dat1 %>% filter(district == "005")
    } else if(input$pick2 == "006"){
      data2 <- dat1 %>% filter(district == "006")
    } else if(input$pick2 == "007"){
      data2 <- dat1 %>% filter(district == "007")
    } else if(input$pick2 == "008"){
      data2 <- dat1 %>% filter(district == "008")
    } else if(input$pick2 == "009"){
      data2 <- dat1 %>% filter(district == "009")
    } else if(input$pick2 == "010"){
      data2 <- dat1 %>% filter(district == "010")
    } else if(input$pick2 == "011"){
```

```
        data2 <- dat1 %>% filter(district == "011")
      } else if(input$pick2 == "012"){
        data2 <- dat1 %>% filter(district == "012")
      } else if(input$pick2 == "013"){
        data2 <- dat1 %>% filter(district == "013")
      } else if(input$pick2 == "014"){
        data2 <- dat1 %>% filter(district == "014")
      } else if(input$pick2 == "015"){
        data2 <- dat1 %>% filter(district == "015")
      } else if(input$pick2 == "016"){
        data2 <- dat1 %>% filter(district == "016")
      } else if(input$pick2 == "017"){
        data2 <- dat1 %>% filter(district == "017")
      } else if(input$pick2 == "018"){
        data2 <- dat1 %>% filter(district == "018")
      } else if(input$pick2 == "019"){
        data2 <- dat1 %>% filter(district == "019")
      } else if(input$pick2 == "020"){
        data2 <- dat1 %>% filter(district == "020")
      } else if(input$pick2 == "021"){
        data2 <- dat1 %>% filter(district == "021")
      } else if(input$pick2 == "022"){
        data2 <- dat1 %>% filter(district == "022")
      } else if(input$pick2 == "023"){
        data2 <- dat1 %>% filter(district == "023")
      } else if(input$pick2 == "024"){
        data2 <- dat1 %>% filter(district == "024")
      } else if(input$pick2 == "025"){
        data2 <- dat1 %>% filter(district == "025")
      }})
   output$picker1 <- renderUI({
    pickerInput(inputId = 'pick2',
                label = 'Select district',
                choices = c("001","002","003","004","005","006","007","008","009","010","011","012","01
                options = list(`actions-box` = TRUE),multiple = F)
   })
   value <- reactive({ input$pick2 })
   output$table1 <- DT::renderDataTable({datasetInput4()})
   output$plot1 <- renderPlot({
     datasetInput3()
   })}
shinyApp(ui, server)
```

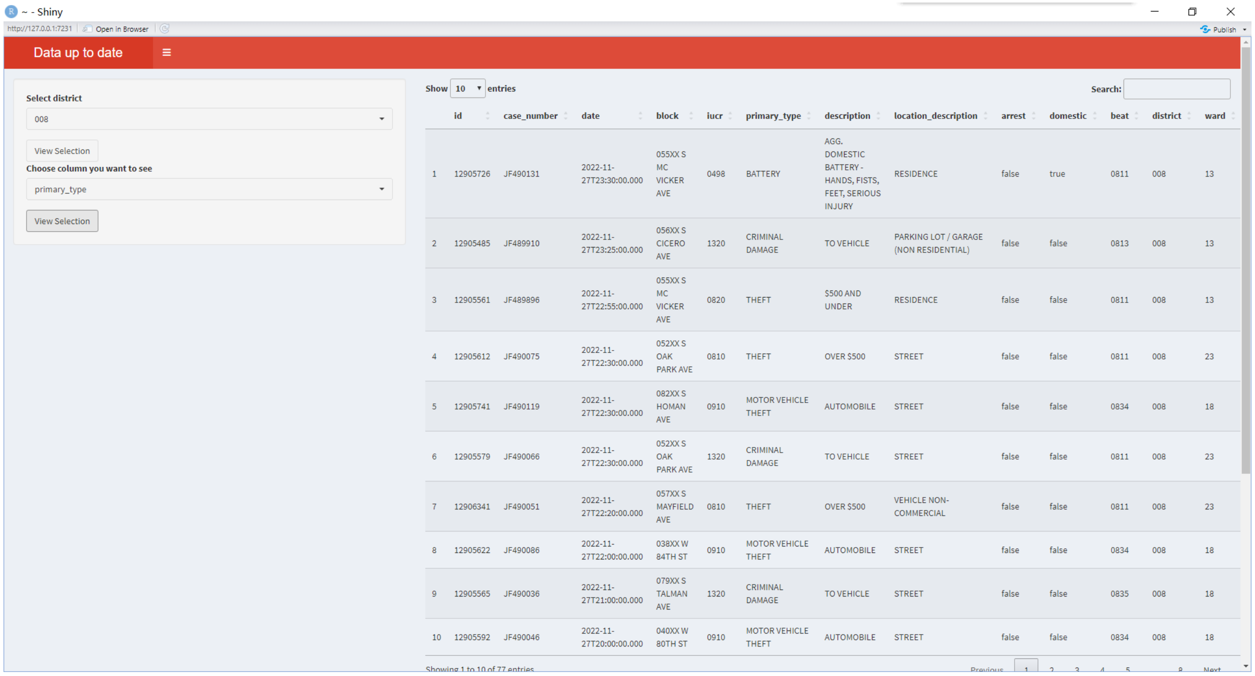
Select which district you want



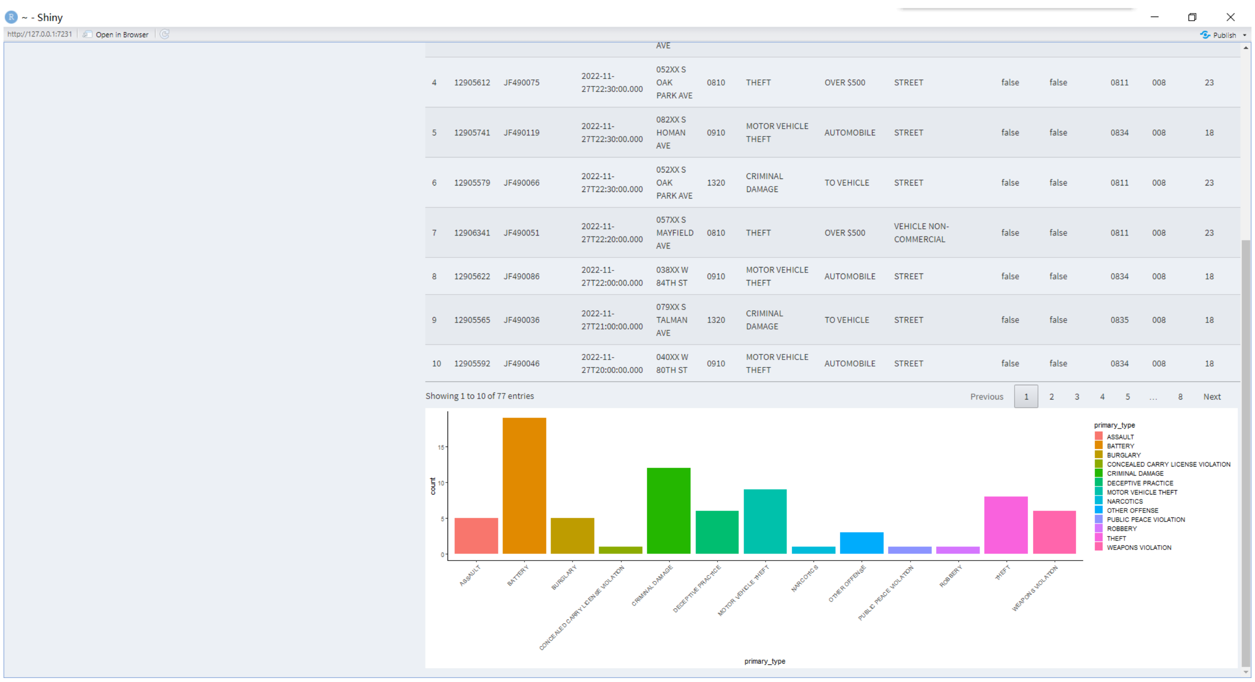
Select which column you want



The Tableoutput of your selection



The plot output of your selection

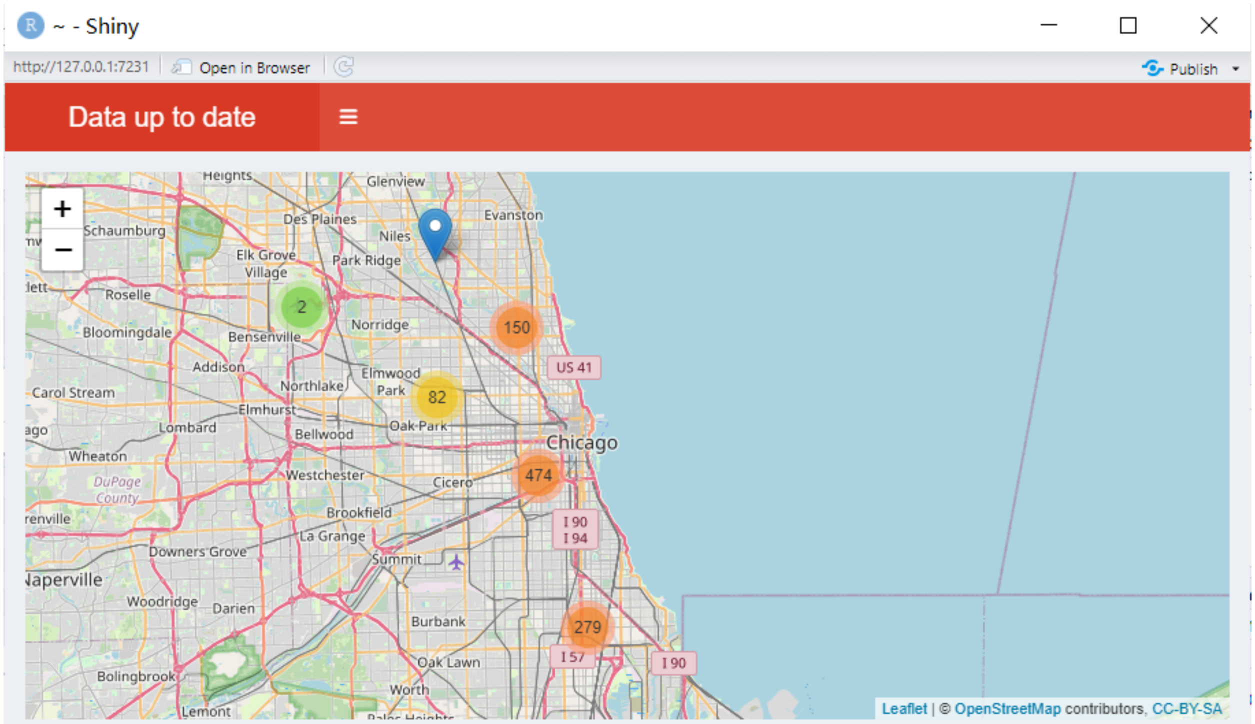


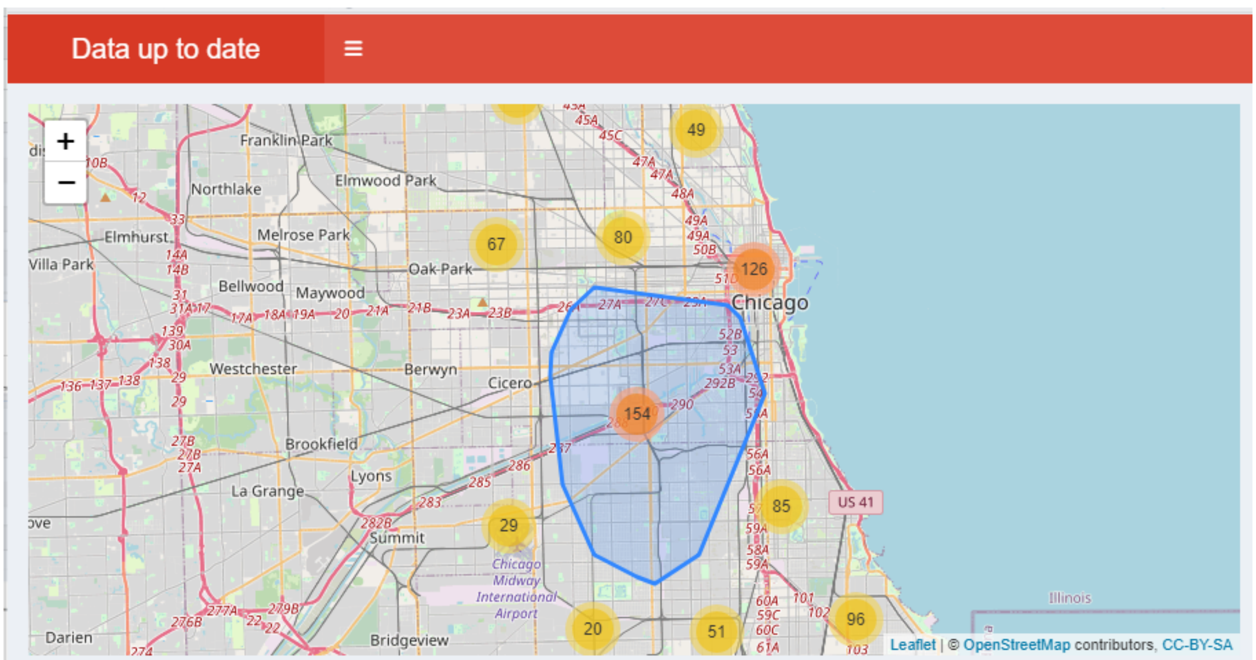
In this Shinyapp, we allow users to select the district from “001” to “025” which is defined by police department and select which column of data they want to see. Then the Shinyapp will produce the filtered table based on users selection. Also, we provide the visualization of the table to let users have a better and clear view of data they want. Based on these information provided by Shinyapp and our analysis above, we hope users can have some ideas about the current situation and avoid some high risk areas. Data in this app also connect to the Internet, so everything inside is up to date.

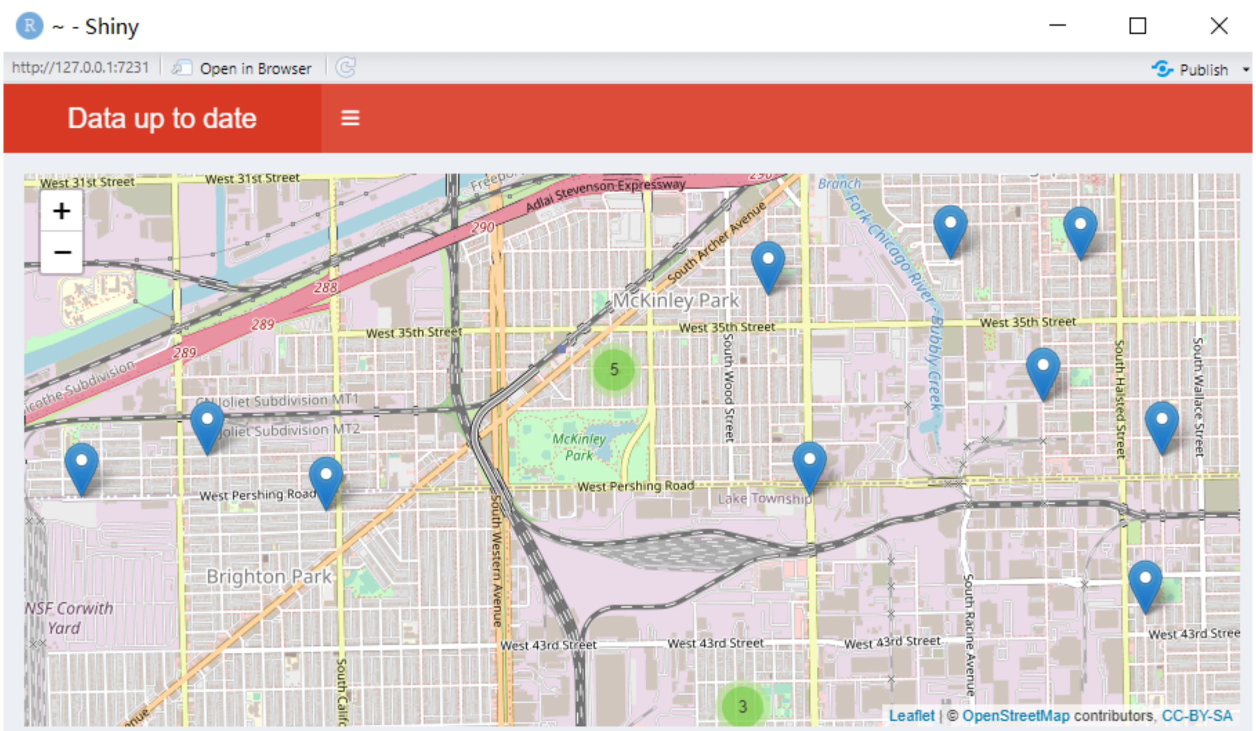
5.3 Latest data access

```
apiKey <- "XXXXXXXXXXX"
result <- GET("https://data.cityofchicago.org/resource/crimes.json",
              add_headers(Authorization = paste("Key", apiKey)))
ui <- dashboardPage(skin = "red",
  dashboardHeader(title = "Data up to date"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Access data",tabName = "a"))),
  dashboardBody(useShinyjs(),
    tabItems(
      tabItem(tabName = "a",leafletOutput("Mapofcrime"))
      )))
server <- function(input, output) {
  url <- reactive({
  paste("https://data.cityofchicago.org/resource/crimes.json")
  })

  result <- reactive ({
    r_json <- jsonlite::fromJSON(url(), flatten = TRUE)
  })
   output$table_names <- DT::renderDataTable({
    result()
   })
   addClass(selector = "body", class = "sidebar-collapse")
   output$Mapofcrime <- renderLeaflet({
     crimemap<- result()
     crimemap$longitude <- as.numeric(crimemap$longitude)
     crimemap$latitude <- as.numeric(crimemap$latitude)
     leaflet(data = crimemap[1:1000,]) %>% addTiles() %>%
  addMarkers(~longitude, ~latitude,clusterOptions = markerClusterOptions())
   })
   }
shinyApp(ui, server)
```


~ - Shiny
http://127.0.0.1:7231
Open in Browser
Publish
Data up to date
Leaflet | © OpenStreetMap contributors, CC-BY-SA


Data up to date
Leaflet | © OpenStreetMap contributors, CC-BY-SA



In this Shinyapp,we create a live map which will update data everyday from the Internet and mark the lastest crime on the map. We filter the first 1000 rows of latest data from Internet, so the map will not be messed up by large number of data. Also we use clustering markers in the map. This can increase users' experience and the information on it will be more directed. Users also can zoom in or zoom out to see the precise point on the map. We hope this map can help users notice which area is high risk area.

6 CONCLUSION

1. For this project, after EDA the crime data and unemployment data, we found that the unemployment rate seems to have some influence,and the month also seems to have some influence, with the winter months of January-April having the lowest crime rates, and the summer months of May-August having the highest crime rates. But overall, the crime rate in Chicago generally around 200,000. What's more, the most frequent "crime" in Chicago are theft while the second most are Battery. Most of the crime happens in 011, 006, and 008.Therefore, to ensure safety when staying in Chicago, people need to be careful of these three districts, as well as thieves on the street.

2. Overall, the number of crimes has shown a downward trend since 2012. In order to analyze which district have more crimes, we use k-means. It shows that District 11 has the highest crimes while District 31 has the lowest crimes. Also, for further predicting crimes, we use time series model, which shows that the number of crimes that keeps going between 600-700 in the next 100 days.It's a reminder to the people in Chicago to stay safe and alert to the dangers.