Crime analysis

Introduction

This project is made for purpose of Data mining course - Mathematics and Computer science department Original data from Kaggle: https://www.kaggle.com/adamschroeder/crimes-new-york-city/version/1 Data : New York crime data Objective : extraction of knowledge related to crimes from dataset General purpose of this project is not classic classification of regression problems, but finding out important features of crime nature in New York.

R Libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(DT)
library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(viridisLite)
```

Data

```
crimeData = read.table("CrimeData.csv",header = TRUE,sep = ',')
crimeData[sample(x = 1:1048575,size = 15),] # a brief look at the data
```

```
##        day month year  time       Borough dayPart Latitude Longitude
## 659907  17     8 2014  3.67     MANHATTAN   22-06 40.75520 -73.96830
## 242415  15     5 2015 16.00      BROOKLYN   12-17 40.66923 -73.92829
## 179169  19     8 2015  8.75      BROOKLYN   06-12 40.66938 -73.89522
## 695406  22     7 2014 17.00 STATEN_ISLAND   12-17 40.60969 -74.15599
## 228790  16     7 2015  9.93      BROOKLYN   06-12 40.66852 -73.99191
## 819695  27     4 2014 17.17      BROOKLYN   17-22 40.69207 -73.93237
## 602482  25     9 2014 21.00     MANHATTAN   17-22 40.74063 -74.00771
## 800238  11     5 2014 22.75      BROOKLYN   22-06 40.70827 -73.95512
## 94006   19    10 2015 18.25     MANHATTAN   17-22 40.72364 -73.99830
## 134602  19     9 2015 19.33         BRONX   17-22 40.86262 -73.90068
## 363834   8     4 2015 22.25         BRONX   22-06 40.84649 -73.88263
## 741206  28     5 2014 12.00      BROOKLYN   06-12 40.61149 -73.96069
## 956195  12     1 2014  3.00      BROOKLYN   22-06 40.67832 -73.94553
## 830345  19     4 2014 19.25     MANHATTAN   17-22 40.78940 -73.97110
## 546308  29    10 2014  7.00        QUEENS   06-12       NA        NA
##                    offenseDescription                          pdDescription
## 659907   ASSAULT_AND_RELATED_OFFENSES                                ASSAULT
## 242415   ASSAULT_AND_RELATED_OFFENSES                                ASSAULT
## 179169       VEHICLE_AND_TRAFFIC_LAWS       TRAFFIC_UNCLASSIFIED_MISDEMEAN
## 695406                  PETIT_LARCENY              LARCENY_PETIT_FROM_AUTO
## 228790                        ROBBERY       ROBBERY_OPEN_AREA_UNCLASSIFIED
## 819695 OFF_AGNST_PUB_ORD_SENSBLTY_AND               AGGRAVATED_HARASSMENT_
## 602482                  GRAND_LARCENY LARCENY_GRAND_FROM_EATERY_UNATTENDED
## 800238                   HARRASSMENT_             HARASSMENT_SUBD_CIVILIAN
## 94006                   PETIT_LARCENY       LARCENY_PETIT_FROM_STORE_SHOPL
## 134602                  PETIT_LARCENY       LARCENY_PETIT_FROM_BUILDING_UN
## 363834              DANGEROUS_WEAPONS                WEAPONS_POSSESSION_
## 741206                    THEFT_FRAUD            FRAUD_UNCLASSIFIED_FELONY
## 956195                  PETIT_LARCENY              LARCENY_PETIT_FROM_AUTO
## 830345                DANGEROUS_DRUGS             MARIJUANA_POSSESSION___5
## 546308                    THEFT_FRAUD            FRAUD_UNCLASSIFIED_FELONY
##        crimeCompleted offenseLevel occurenceLocation       premiseDescription
## 659907      COMPLETED  MISDEMEANOR            INSIDE           BAR_NIGHT_CLUB
## 242415      COMPLETED  MISDEMEANOR            INSIDE                    OTHER
## 179169      COMPLETED  MISDEMEANOR     MISSING_VALUE                   STREET
## 695406      COMPLETED  MISDEMEANOR          FRONT_OF          RESIDENCE_HOUSE
## 228790      COMPLETED       FELONY          FRONT_OF                   STREET
## 819695      COMPLETED  MISDEMEANOR            INSIDE          RESIDENCE_HOUSE
```

```
## 602482      COMPLETED       FELONY     MISSING_VALUE         RESTAURANT_DINER
## 800238      COMPLETED    VIOLATION     MISSING_VALUE        BUS_(NYC_TRANSIT)
## 94006       COMPLETED  MISDEMEANOR            INSIDE              CHAIN_STORE
## 134602      COMPLETED  MISDEMEANOR            INSIDE      COMMERCIAL_BUILDING
## 363834      COMPLETED       FELONY            INSIDE           GROCERY_BODEGA
## 741206      ATTEMPTED       FELONY            INSIDE      RESIDENCE_APT_HOUSE
## 956195      COMPLETED  MISDEMEANOR       OPPOSITE_OF                   STREET
## 830345      COMPLETED  MISDEMEANOR            INSIDE RESIDENCE_PUBLIC_HOUSING
## 546308      COMPLETED       FELONY            INSIDE                BOOK_CARD
```

There is a difference beetwen this data and original from Kaggle. Simple preprocessing is made and some variables(date event was reported,police jurisdiction...) are ejected, some are changed(date variable to month, day and year, hours and minutes to time...) and some are added(dayPart) due to simplicity. Variables "hours" and "minutes" are joined into 1 continuous variable time - for instance: 15h 30min is now 15.5 (15 + 30/60). Variable "time" is divided into categorical variable "dayPart" with 4 classes(parts of the day). All NA's are replaced with "MISSING_VALUE"

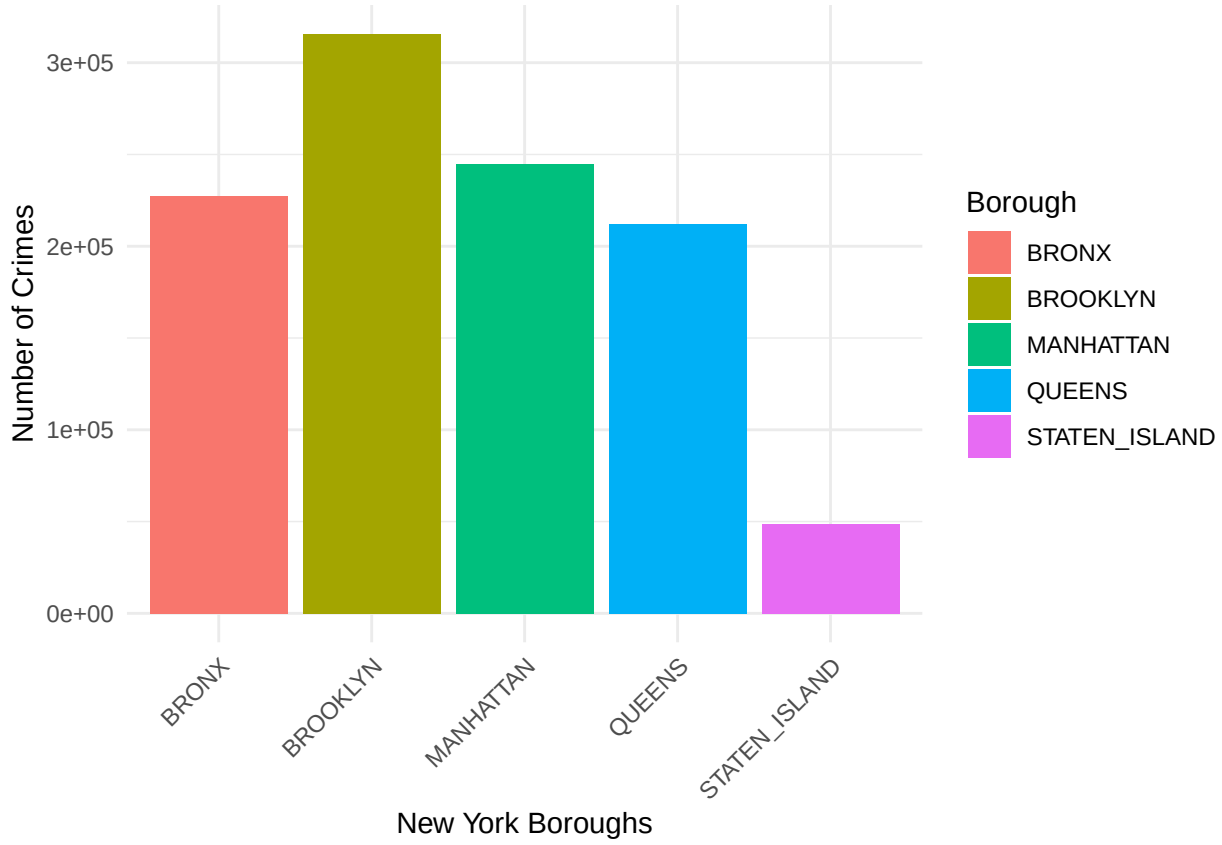
```
summary(crimeData)
```

```
##       day             month              year           time      
##  Min.   : 1.00   Min.   : 1.000   Min.   :1015   Min.   : 0.00  
##  1st Qu.: 8.00   1st Qu.: 4.000   1st Qu.:2014   1st Qu.: 9.00  
##  Median :15.00   Median : 7.000   Median :2014   Median :14.67  
##  Mean   :15.52   Mean   : 6.947   Mean   :2014   Mean   :13.51  
##  3rd Qu.:23.00   3rd Qu.:10.000   3rd Qu.:2015   3rd Qu.:19.00  
##  Max.   :31.00   Max.   :12.000   Max.   :2015   Max.   :23.98  
##  NA's   :65      NA's   :65       NA's   :65                    
##    Borough             dayPart             Latitude       Longitude     
##  Length:1048575     Length:1048575     Min.   :40.50   Min.   :-74.26  
##  Class :character   Class :character   1st Qu.:40.67   1st Qu.:-73.97  
##  Mode  :character   Mode  :character   Median :40.73   Median :-73.93  
##                                        Mean   :40.73   Mean   :-73.93  
##                                        3rd Qu.:40.81   3rd Qu.:-73.88  
##                                        Max.   :40.91   Max.   :-73.70  
##                                        NA's   :32417   NA's   :32417   
##  offenseDescription pdDescription      crimeCompleted     offenseLevel      
##  Length:1048575     Length:1048575     Length:1048575     Length:1048575    
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                            
##                                                                            
##                                                                            
##                                                                            
##  occurenceLocation  premiseDescription
##  Length:1048575     Length:1048575    
##  Class :character   Class :character  
##  Mode  :character   Mode  :character  
##                                       
##                                       
##                                       
##
```

Data visualization

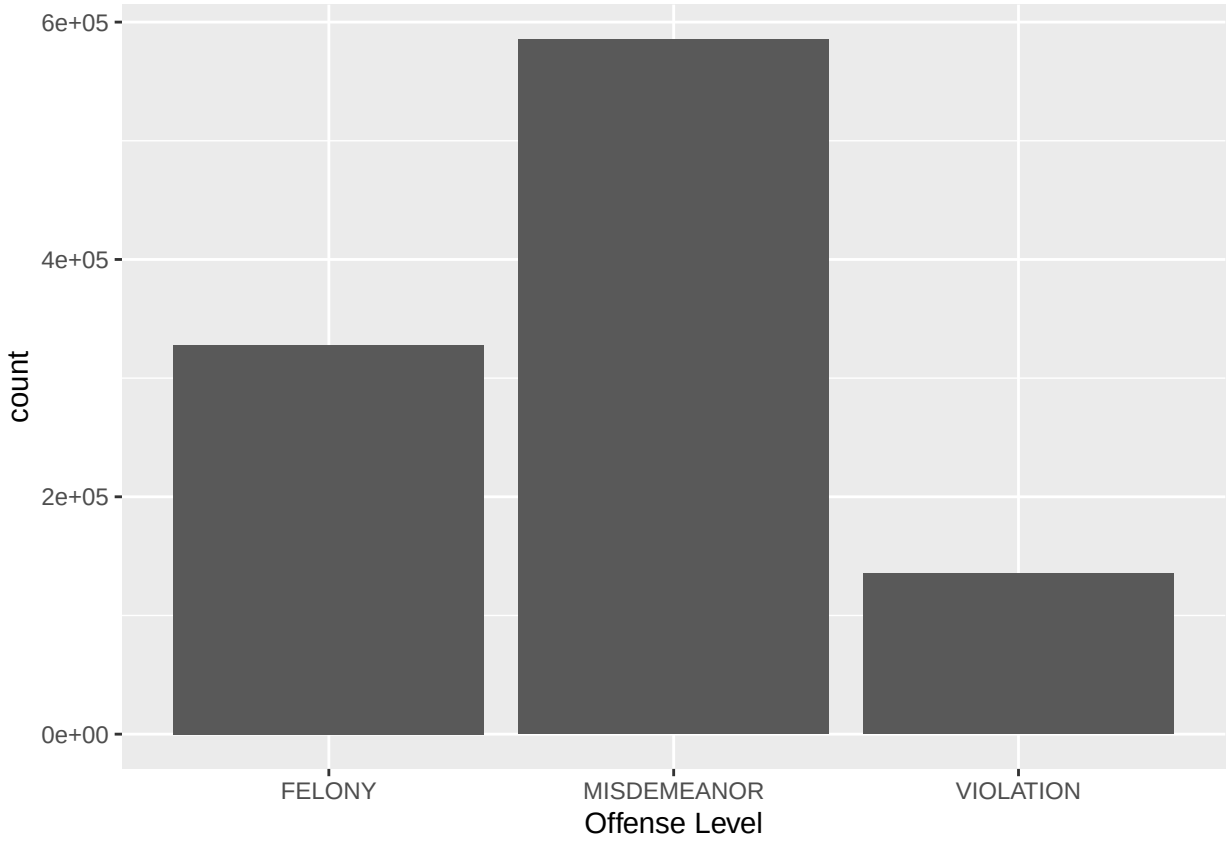
The most crimes generally occur in Brooklyn while least number of crimes occur in Staten Island

```
ggplot(data = crimeData) +
  geom_bar(mapping = aes(x = Borough, fill = Borough)) +
  xlab("New York Boroughs") + ylab("Number of Crimes") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

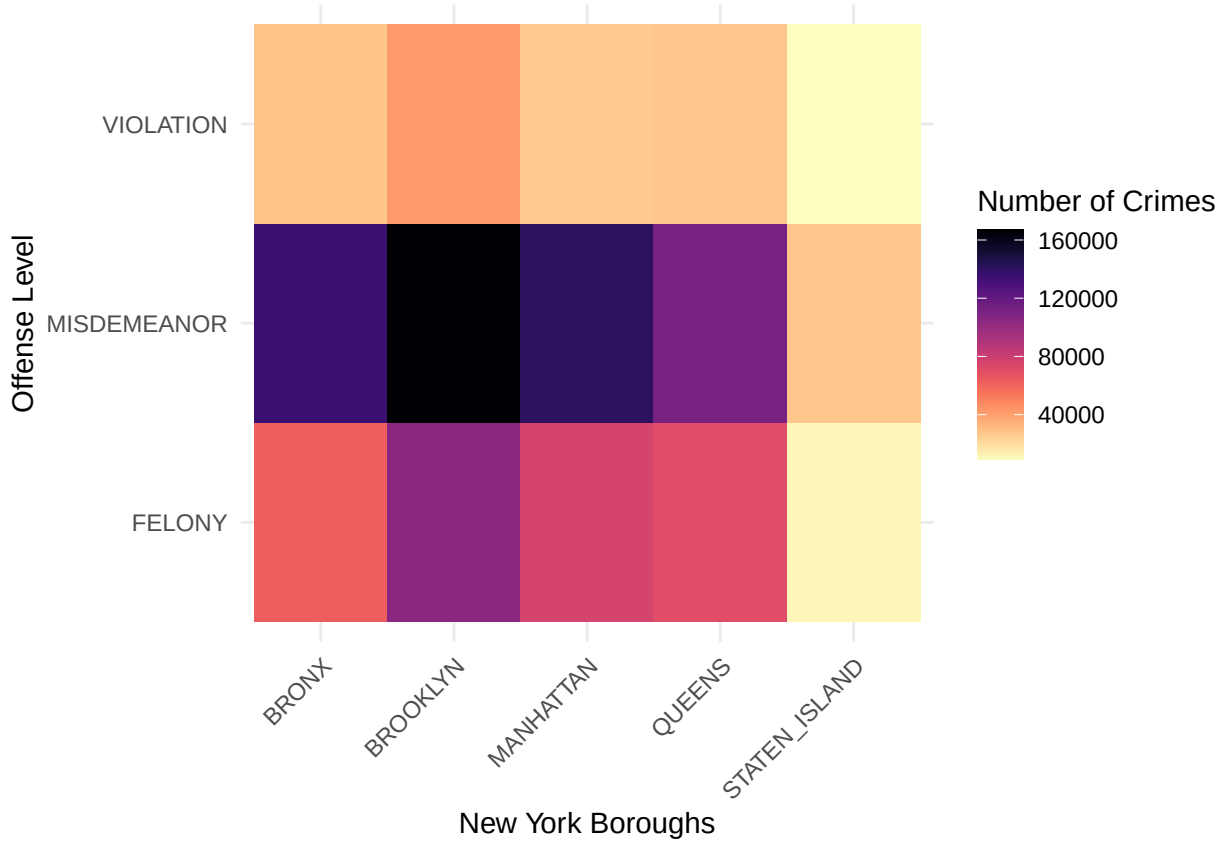


Distribution of "offense level" variable Misdemeanor offense dominate over felony and violation.

```
ggplot(data = crimeData) +
  geom_bar(mapping = aes(x = offenseLevel)) +
  xlab("Offense Level")
```

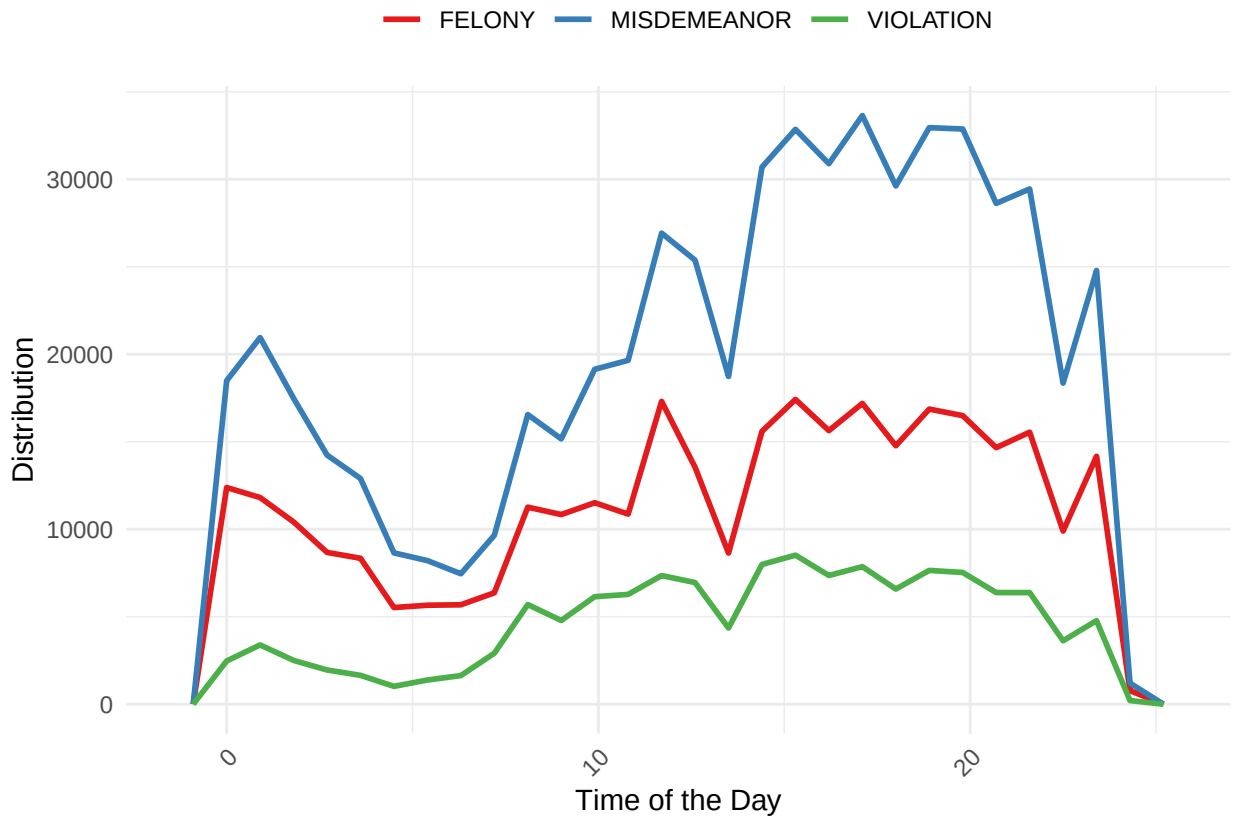



```
subset(crimeData, year >= 2012) %>%
  count(Borough, offenseLevel) %>%
  ggplot(mapping = aes(x = Borough, y = offenseLevel, fill = n)) +
  geom_tile() +
  labs(x = "New York Boroughs", y = "Offense Level", fill = "Number of Crimes") +
  scale_fill_viridis(option = "A", direction = -1) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  theme(legend.position = "right")
```



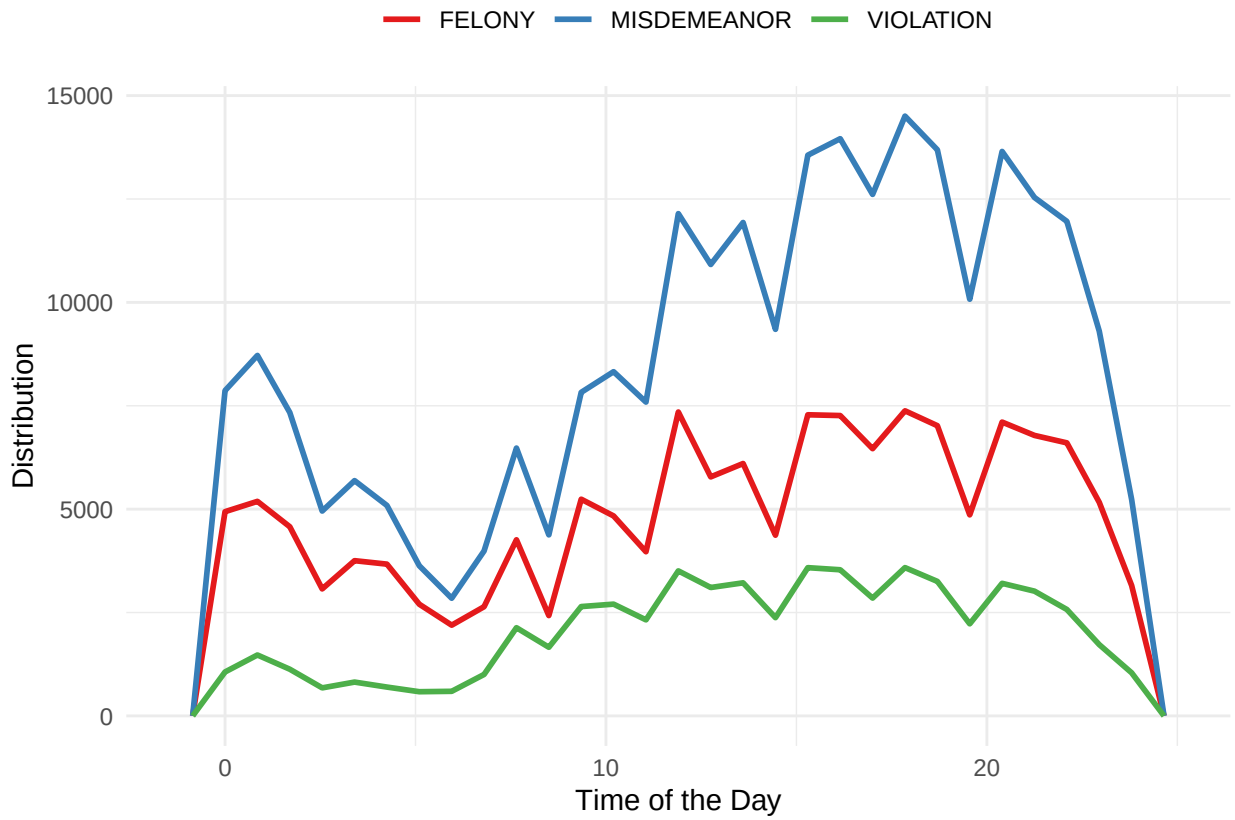
Distribution of each offense level through the day. It is clear that second part of the day (17-21) is the time when most crimes of each level occur and the morning is the part of the day with less crime appearances.

```
options(repr.plot.width = 12, repr.plot.height = 5)
ggplot(crimeData, mapping = aes(x = time, colour = offenseLevel)) +
  geom_freqpoly(binwidth = 0.9, lwd = 1) +
  xlab("Time of the Day") +
  ylab("Distribution") +
  theme_minimal() +
  scale_color_brewer(palette = "Set1") +
  theme(legend.position = "top",
        legend.title = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1))
```



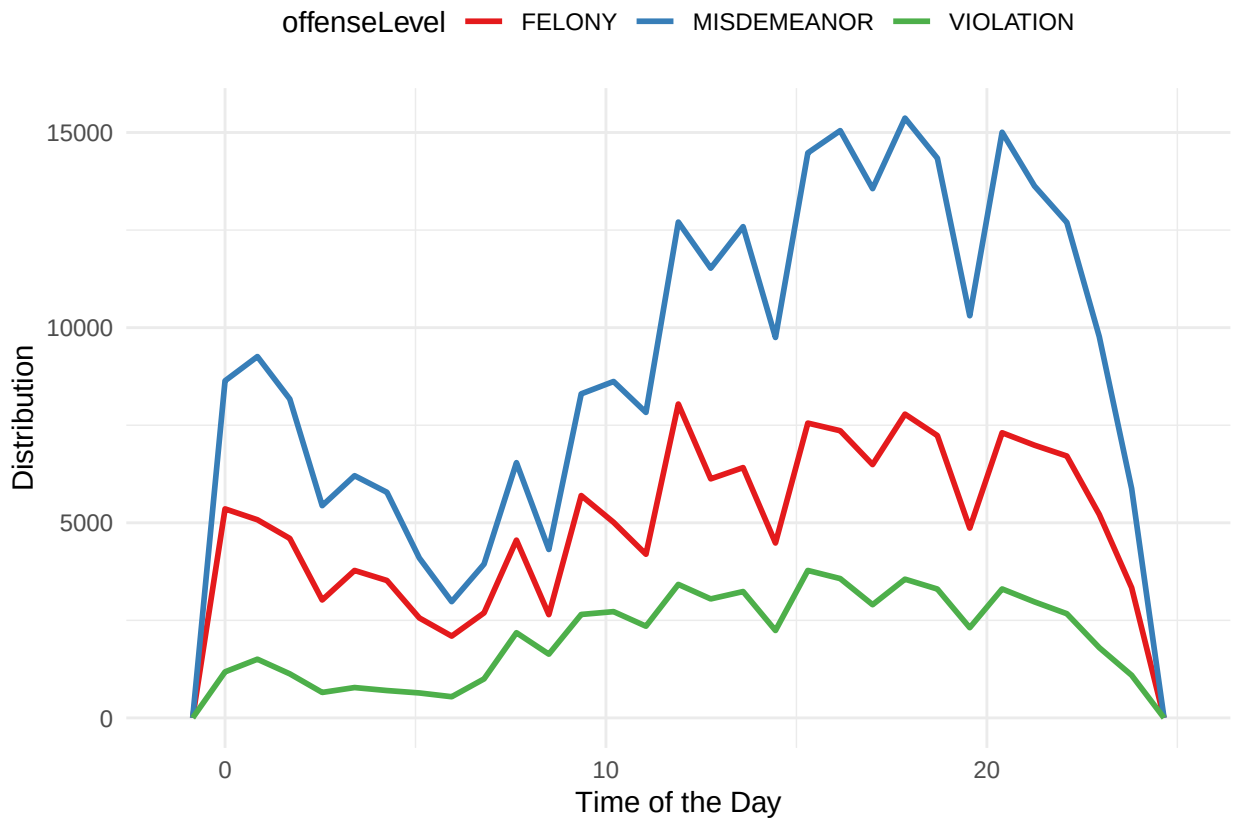
This trend don't change over time. The similar pattern occur if one smaller subset(44% of data) of data is taken (only crimes from last year - 2015)

```
options(repr.plot.width=9, repr.plot.height=5)
subset(crimeData, year == 2015) %>%
  ggplot(mapping = aes(x = time, colour = offenseLevel)) +
  geom_freqpoly(binwidth = 0.85, lwd = 1) +
  xlab("Time of the Day") + ylab("Distribution") +
  theme_minimal() +
  theme(legend.position = "top", legend.title = element_blank()) +
  scale_color_brewer(palette = "Set1")
```



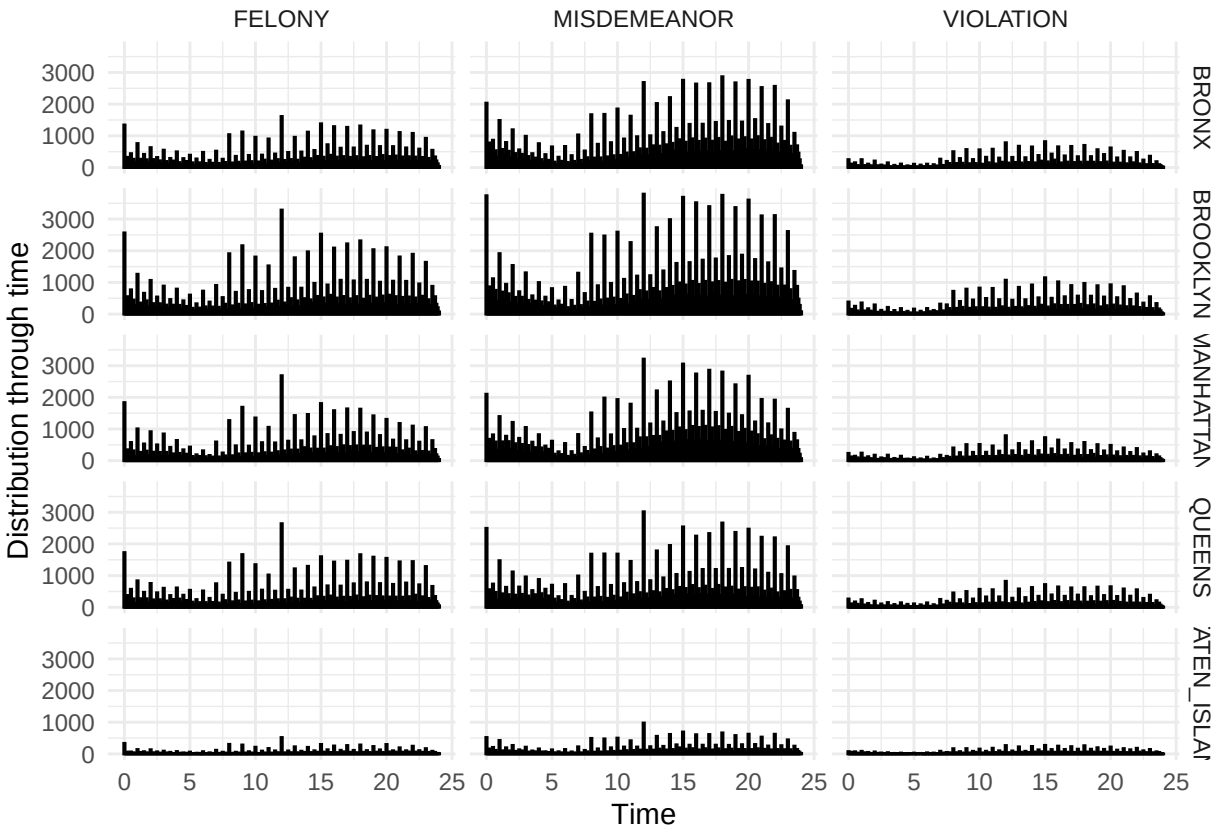
2014 year

```
options(repr.plot.width=9, repr.plot.height=5)
subset(crimeData, year == 2014) %>%
  ggplot(mapping = aes(x = time, colour = offenseLevel)) +
  geom_freqpoly(binwidth = 0.85, lwd = 1) +
  xlab("Time of the Day") + ylab("Distribution") +
  theme_minimal() +
  theme(legend.position = "top") +
  scale_color_brewer(palette = "Set1")
```



Relation beetwen New York boroughs, offense level and time. Crime offense levels mostly don't depend on borough but on time od the day.

```
options(repr.plot.width=10, repr.plot.height=7)
ggplot(data = subset(crimeData, year >= 2012), aes(x = time)) +
  geom_histogram(binwidth = 0.1, color = "black", fill = "blue", alpha = 0.6) +
  facet_grid(Borough ~ offenseLevel) +
  labs(x = "Time", y = "Distribution through time") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



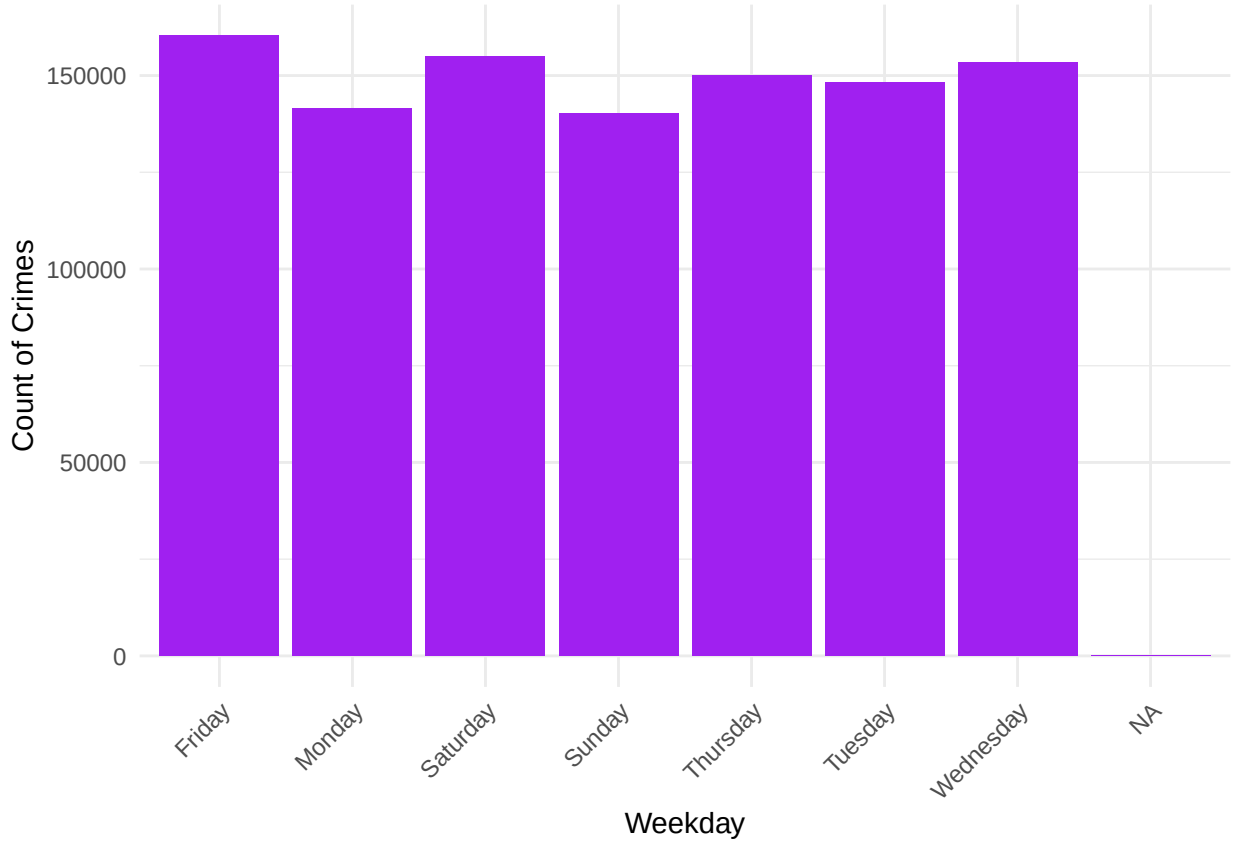
Offense level vs. crime location Inside crimes are dominant independently of crime level.

```
ggplot(data = subset(crimeData, year >= 2013), aes(x = occurenceLocation)) +
  geom_bar(aes(fill = offenseLevel), position = "dodge") +
  labs(x = "Location of Crime", y = "Count of Crimes") +
  facet_wrap(~offenseLevel, ncol = 1, scales = "free_y") +
  coord_flip() +
  theme_minimal() +
  scale_fill_brewer(palette = "Set1")
```

How to get some new information - simple example

One of the main tasks of data mining is extraction new data and information from the old one. Here is a very simple example of getting day of the week from a given date.

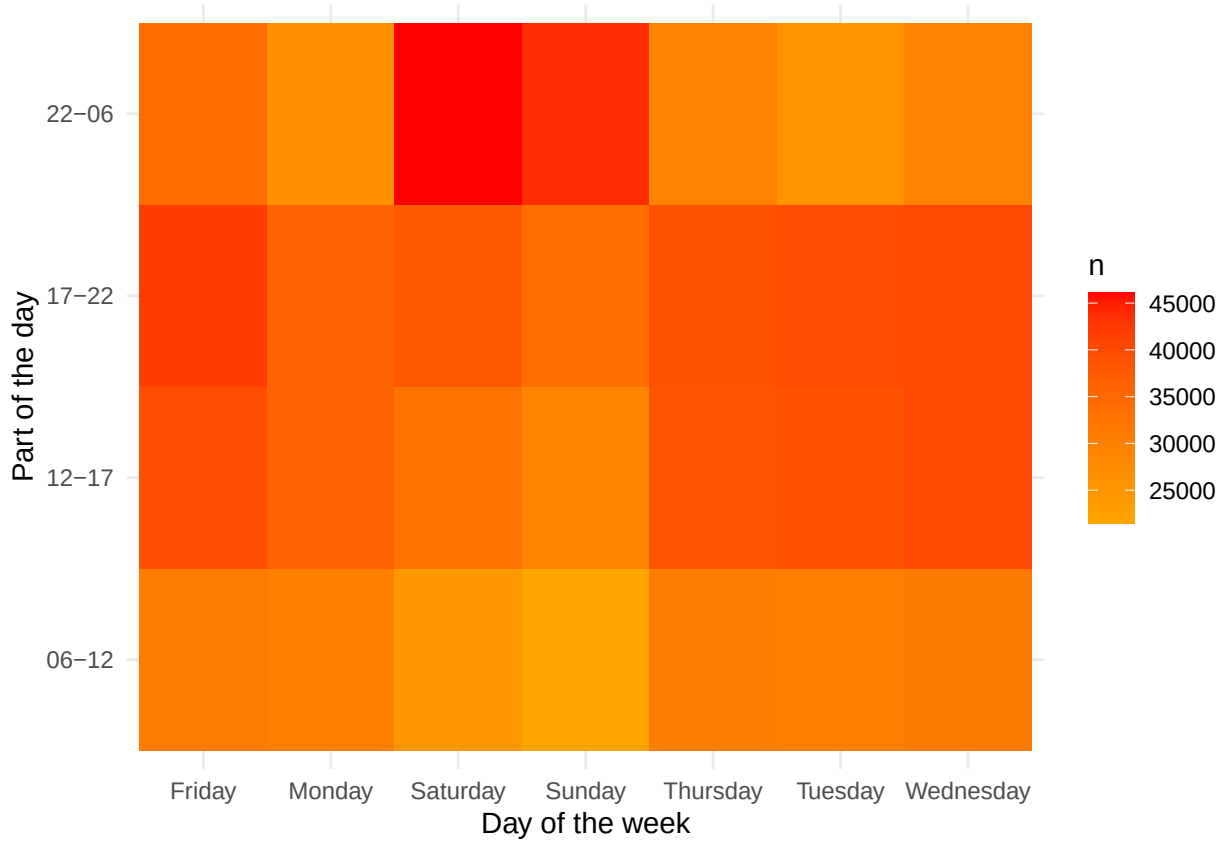
```
crimeData[15] <- crimeData[, c(3, 2, 1)] %>%
  apply(MARGIN = 1, FUN = function(vec) {paste(vec, collapse = '-')}) %>%
  as.Date() %>%
  weekdays()

names(crimeData)[15] <- 'weekDay'
options(repr.plot.width = 7, repr.plot.height = 5)
ggplot(data = crimeData, aes(x = weekDay)) +
  geom_bar(fill = "purple") +
  xlab("Weekday") + ylab("Count of Crimes") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Day of the week vs. time (year 2014 and 2015)

As it can be seen, the most dangerous time of the week is weekend night(saturday and sunday 22-06) and middle of the week through the day, while the less dangerous is the middle of the week at night and weekend mornings. For this and similar analysis, the idea is to track new trends that might happen and for that reason newer data should be taken for analysis(in this case last 2 years).

```
subset(crimeData, !is.na(weekDay) & year >= 2014) %>%
  count(weekDay, dayPart) %>%
  ggplot(mapping = aes(x = weekDay, y = dayPart)) +
  geom_tile(mapping = aes(fill = n)) +
  xlab("Day of the week") + ylab("Part of the day") +
  theme_minimal() +
  scale_fill_gradient(low = "orange", high = "red")
```

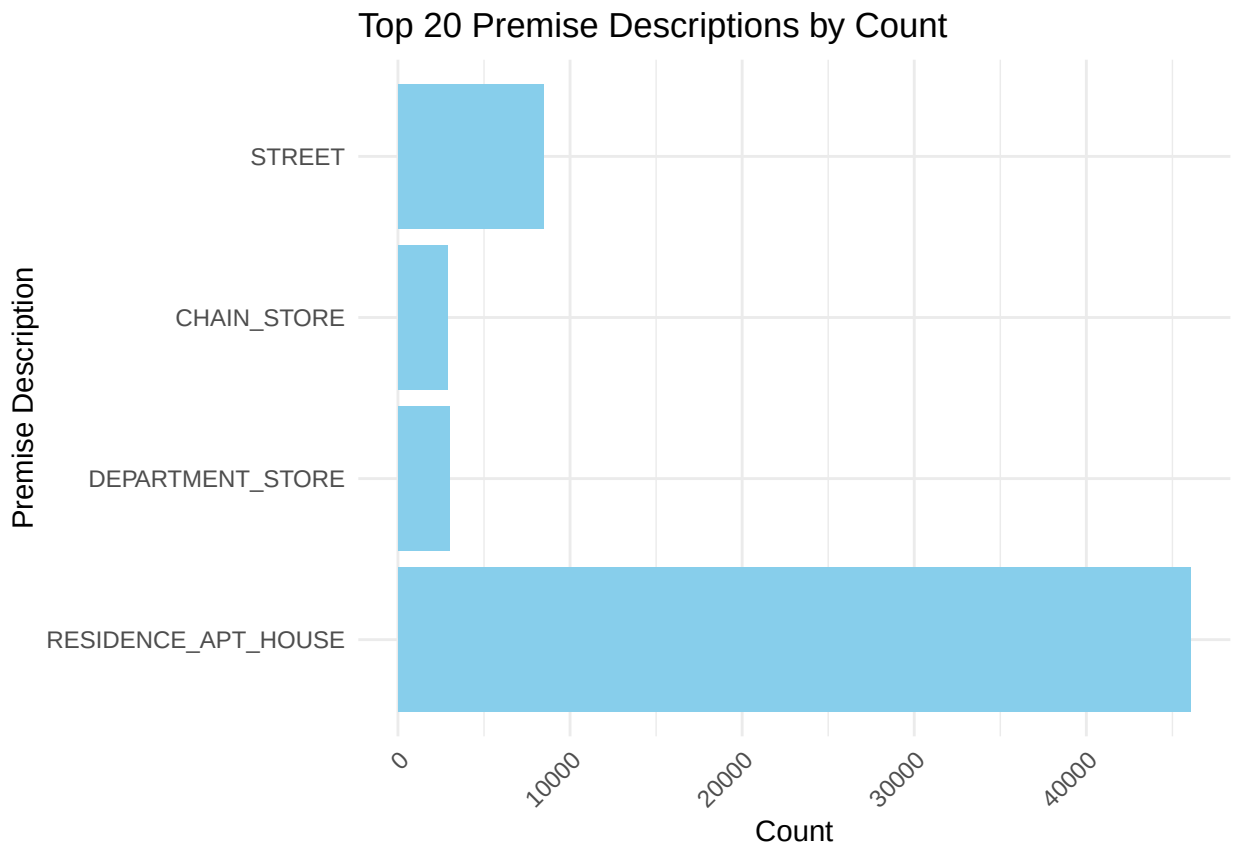


Some things happen more often than the others - interactive data tables

For this kind of analysis DT(data-table) library is used - simple review of particular desired events. From this data-tables it is easy to observe what events are more frequent than the others.

```
subset(crimeData, year >= 2014 & occurenceLocation != "MISSING_VALUE") %>%
  group_by(offenseDescription, Borough, dayPart, premiseDescription) %>%
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
  head(20) %>%
  ggplot(mapping = aes(x = reorder(premiseDescription, -count), y = count)) +
  geom_bar(stat = "identity", fill = "skyblue") +  # Change the fill color
  xlab("Premise Description") + ylab("Count") +
  ggtitle("Top 20 Premise Descriptions by Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  coord_flip()
```

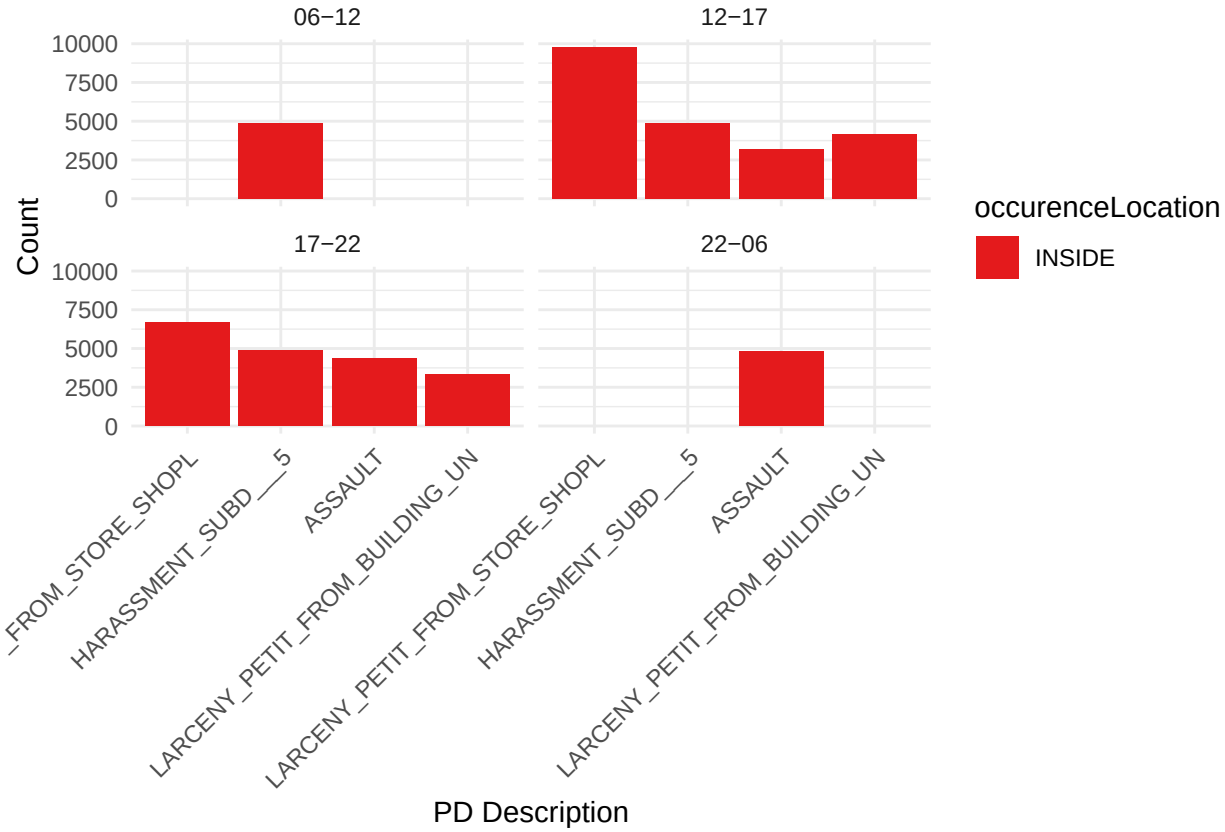
```
## `summarise()` has grouped output by 'offenseDescription', 'Borough', 'dayPart'.
## You can override using the `.groups` argument.
```




```
summary_data <- subset(crimeData, year >= 2014 & occurenceLocation != "MISSING_VALUE") %>%
  group_by(occurenceLocation, Borough, dayPart, pdDescription) %>%
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
  head(20)
```

```
## `summarise()` has grouped output by 'occurenceLocation', 'Borough', 'dayPart'.
## You can override using the `.groups` argument.
```

```
ggplot(summary_data, aes(x = reorder(pdDescription, -count), y = count, fill = occurenceLocation)) +
  geom_bar(stat = "identity", position = "dodge") +
  facet_wrap(~dayPart) +
  labs(x = "PD Description", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(palette = "Set1")
```



Felony

2015 felony related crimes - small pattern emerges. The most dangerous place for this specific category is Brooklyn - 12 h, at beginning of the week. The same pattern come up for 2014 year.

```
subset(crimeData,year == 2015 & offenseLevel == "FELONY") %>% group_by(Borough,time,weekDay) %>% summari
arrange(desc(count)) %>% head(30) %>% datatable(options = list(pageLength = 10,scrollX='400px'))
```

```
## `summarise()` has grouped output by 'Borough', 'time'. You can override using
## the `.groups` argument.
```

Association rules

Association rules are rule-based data mining method for discovering certain relations between variables in data-sets. The main purpose of association rules is to discover strong rules in data-sets using measures of interestingness. Let $I = \{i_1, ..., i_n\}$ be the set of variables in the dataset. Observations of data-set (rows of data frame) are usually called **transactions**. A rule is defined like implication $A \implies B$ where $A, B \subset I$. A is usually called antecedent or left-hand-side (LHS) and Y consequent or right-hand-side (RHS). In some implementations rule is defined like $A \implies i_j$ where $i_j \in I$.

Significant measures

Let X, Y be itemsets, $X \implies Y$ an association rule and T a set of transactions of a given data-set.

Support

Support is an indication of how frequently the itemset appears in the dataset. It is proportion of transactions(rows in data frame) that contain specific itemset, with respect to number of transactions. $supp(X) = \frac{|t \in T; X \subset t|}{|T|}$

Confidence

Confidence is an indication of how often the rule has been found to be true. $conf(X \cup Y) = \frac{supp(X \cup Y)}{supp(X)}$ Confidence can be interpreted as an estimate of the conditional probability $P(Y|X)$, the probability of finding the Y in transactions under the condition that these transactions also contain the X in the left side of the rule.

The lift

The lift of a rule is defined as $lift(X \implies Y) = \frac{supp(X \cup Y)}{supp(X) * supp(Y)}$ It is the ratio of the observed support to that expected if X and Y were independent events. If X and Y are truly independent events, we can expect that about $supp(X) * supp(Y)$ number of transactions will contain both of them. If the rule had a **lift of 1**, it would imply that the probability of occurrence of the antecedent and that of the consequent are independent of each other. When two events are independent of each other, no rule can be drawn involving those two events. If the **lift is $>$ 1**, that lets us know the degree to which those two occurrences are dependent on one another, and makes those rules potentially useful for predicting the consequent in future data sets. If the **lift is $<$ 1**, that lets us know the items are substitute to each other. This means that presence of one item has negative effect on presence of other item and vice versa. Definitons taken from : https://en.wikipedia.org/wiki/Association_rule_learning R implementation: library arules , apriori algorithm.

Example - wrong way of using association rules

One of the obvious wrong ways of using of association rules is to apply it to variables that are obviously correlated in some way. In this dataset for instance we could get rule like: offenseDescription = ASSAULT_AND_RELATED_OFFENSES $\implies$ pdDescription = ASSAULT. It is clear and natural that these 2 variables are in close relationship, so although this rule might have large lift, is not very helpfull. For this reason in examples below algorithm will take only some subset of variables, excluding others that are obviously correlated with them.

Apriori algorithm

Apriori algorithm is classic algorithm for generating association rules from datasets or databases. The key idea of the algorithm is to begin by generating frequent itemsets with just one item (1-itemsets) and to recursively generate frequent itemsets with 2 items, then frequent 3-itemsets and so on till some stopping condition is satisfied. This is where computational complexity comes into the game. Apriori algorithm is based on very simple observation: **subsets of frequent itemsets are also frequent itemsets**. In other words , if some itemset is proven to be non-frequent , then it will not be considered by algorithm any more for forming new frequent itemsets. To identify the k-itemsets that are not frequent algorithm need to examine all subsets of size (k-1) of each candidate k-itemset. It generates candidate itemsets of length k from item sets of length k-1. Then it prunes the candidates which have an infrequent sub-pattern.

```
rules <- apriori(data = subset(crimeData,year >= 2013)[,-c(1,2,3,6,7,8,9,10,11,14)] ,
parameter = list(support = 0.03 , confidence = 0.6,maxlen = 5,target = 'rules'))
```

```
## Warning: Column(s) 1, 2, 3, 4, 5 not logical or factor. Applying default
## discretization (see '? discretizeDF').

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5    0.03      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 31353
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[24 item(s), 1045101 transaction(s)] done [0.23s].
## sorting and recoding items ... [21 item(s)] done [0.04s].
## creating transaction tree ... done [0.84s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [7 rule(s)] done [0.00s].
## creating S4 object  ... done [0.10s].
```

```
inspect(sort(rules,by='lift'))
```

```
##     lhs                                    rhs                            support confidence   coverage
## [1] {offenseLevel=VIOLATION}          => {occurenceLocation=INSIDE} 0.08360436  0.6466426 0.12928990
## [2] {time=[11.3,17.5),
##      Borough=MANHATTAN}               => {occurenceLocation=INSIDE} 0.05160458  0.6279414 0.08218057
## [3] {Borough=MANHATTAN,
##      offenseLevel=FELONY}             => {occurenceLocation=INSIDE} 0.04407995  0.6062536 0.07270876
## [4] {Borough=BRONX,
##      occurenceLocation=MISSING_VALUE} => {offenseLevel=MISDEMEANOR} 0.03050040  0.6278635 0.04857808
## [5] {time=[17.5,24],
##      Borough=BRONX}                   => {offenseLevel=MISDEMEANOR} 0.04869864  0.6273884 0.07762121
## [6] {time=[0,11.3),
##      occurenceLocation=MISSING_VALUE} => {offenseLevel=MISDEMEANOR} 0.04490858  0.6237491 0.07199783
## [7] {occurenceLocation=MISSING_VALUE} => {offenseLevel=MISDEMEANOR} 0.12501663  0.6005801 0.20815979
```

In the example above, the first couple of rules have the lift that is slightly greater than 1 which means there might be light correlation between these itemsets. On the other hand, this might be because value "INSIDE" (1st rule) for occurenceLocation is dominating over the other values of occurenceLocation. Intuitive way of interpreting this rule is something like "when crime belongs to the level VIOLATION, it is slightly more likely that it happened INSIDE than then somewhere else". However, confidence of this rule could be somewhat better so we can't accept that this is strong connection between these 2 variables although lift implies some dependence.

Trying to detect what is the cause of rare events

From summary table it is clear that most crimes have value COMPLETED for category crimeCompleted, much less number of crimes are registered as just ATTEMPTED. Association rules could allow us to find

some specific moments that imply this rare events. Although lift is really high for these events, their count is small(2-3) and these are not indicators of any kind of correlation with ATTEMPTED value.

```
rules <- apriori(data = subset(crimeData,year >= 2011)[,c(4,5,9,11,14)] ,
parameter = list(support = 0.000001 , confidence = 0.85,maxlen = 5),
appearance = list(rhs = c('crimeCompleted=ATTEMPTED')))
```

```
## Warning: Column(s) 1, 2, 3, 4, 5 not logical or factor. Applying default
## discretization (see '? discretizeDF').
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.85    0.1    1 none FALSE            TRUE       5   1e-06      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 1
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[150 item(s), 1046908 transaction(s)] done [0.23s].
## sorting and recoding items ... [146 item(s)] done [0.03s].
## creating transaction tree ... done [1.04s].
## checking subsets of size 1 2 3 4 5 done [0.01s].
## writing ... [10 rule(s)] done [0.00s].
## creating S4 object  ... done [0.13s].
```

```
inspect(head(sort(rules,by='lift'),10))
```

```
##      lhs                                                  rhs                              suppor
## [1]  {offenseDescription=FRAUDULENT_ACCOSTING,
##       premiseDescription=PARKING_LOT_GARAGE_(PUBLIC)}     => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [2]  {offenseDescription=RAPE,
##       premiseDescription=BEAUTY_&_NAIL_SALON}             => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [3]  {time=[11.3,17.5),
##       offenseDescription=RAPE,
##       premiseDescription=TRANSIT_NYC_SUBWAY}              => {crimeCompleted=ATTEMPTED} 2.865581e-0
## [4]  {Borough=STATEN_ISLAND,
##       offenseDescription=BURGLARY,
##       premiseDescription=BANK}                            => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [5]  {time=[11.3,17.5),
##       Borough=MANHATTAN,
##       offenseDescription=KIDNAPPING_AND_RELATED_OFFENSES,
##       premiseDescription=STREET}                          => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [6]  {time=[0,11.3),
##       Borough=MANHATTAN,
##       offenseDescription=ROBBERY,
```

```
##       premiseDescription=BUS_STOP}                          => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [7]  {time=[17.5,24],
##       Borough=MANHATTAN,
##       offenseDescription=ROBBERY,
##       premiseDescription=BUS_STOP}                          => {crimeCompleted=ATTEMPTED} 2.865581e-0
## [8]  {time=[11.3,17.5),
##       Borough=MANHATTAN,
##       offenseDescription=RAPE,
##       premiseDescription=TRANSIT_NYC_SUBWAY}                => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [9]  {time=[0,11.3),
##       Borough=STATEN_ISLAND,
##       offenseDescription=BURGLARY,
##       premiseDescription=BANK}                              => {crimeCompleted=ATTEMPTED} 1.910388e-0
## [10] {time=[11.3,17.5),
##       Borough=STATEN_ISLAND,
##       offenseDescription=BURGLARY,
##       premiseDescription=SMALL_MERCHANT}                    => {crimeCompleted=ATTEMPTED} 1.910388e-0
```

Greater count implies that we need to sacrifice confidence. Left-hand side of these rules with great lift value, contains some specific events like explicit part of the day when "KIDNAPPING_AND_RELATED_OFFENSES" crimes happend on the street.

```
rules <- apriori(data = subset(crimeData,year >= 2013)[,c(4,5,9,11,14)] ,
parameter = list(support = 0.00001 , confidence = 0.55,maxlen = 5),
appearance = list(rhs = c('crimeCompleted=ATTEMPTED')))
```

```
## Warning: Column(s) 1, 2, 3, 4, 5 not logical or factor. Applying default
## discretization (see '? discretizeDF').
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.55    0.1    1 none FALSE            TRUE       5   1e-05      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 10
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[150 item(s), 1045101 transaction(s)] done [0.23s].
## sorting and recoding items ... [135 item(s)] done [0.03s].
## creating transaction tree ... done [1.12s].
## checking subsets of size 1 2 3 4 5 done [0.01s].
## writing ... [3 rule(s)] done [0.00s].
## creating S4 object  ... done [0.15s].
```

```r
inspect(head(sort(rules,by='lift'),10))
```

```
##     lhs                                                  rhs                              support
## [1] {time=[11.3,17.5),
##      offenseDescription=KIDNAPPING_AND_RELATED_OFFENSES,
##      premiseDescription=STREET}                       => {crimeCompleted=ATTEMPTED} 1.435268e-05
## [2] {Borough=BRONX,
##      offenseDescription=ROBBERY,
##      premiseDescription=CHECK_CASHING_BUSINESS}       => {crimeCompleted=ATTEMPTED} 1.339583e-05
## [3] {time=[0,11.3),
##      offenseDescription=ROBBERY,
##      premiseDescription=CHECK_CASHING_BUSINESS}       => {crimeCompleted=ATTEMPTED} 1.913691e-05
```

Association rules allow us to discover nature of serious crimes, like burglary and larceny (for 2015 year). Rules with higher lift and confidence are good candidates for better research because they imply that there might be some connection between certain variables in this subset of data.

```r
rules <- apriori(data = subset(crimeData,year == 2015)[,c(4,5,9,11,14,15)] ,
parameter = list(support = 0.00001 , confidence = 0.7,maxlen = 5,target='rules'),
appearance = list(rhs = c('offenseDescription=BURGLARY')))
```

```
## Warning: Column(s) 1, 2, 3, 4, 5, 6 not logical or factor. Applying default
## discretization (see '? discretizeDF').
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    0.1    1 none FALSE            TRUE       5   1e-05      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 4
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[154 item(s), 468576 transaction(s)] done [0.12s].
## sorting and recoding items ... [143 item(s)] done [0.02s].
## creating transaction tree ... done [0.35s].
## checking subsets of size 1 2 3 4 5
```

```
## Warning in apriori(data = subset(crimeData, year == 2015)[, c(4, 5, 9, 11, :
## Mining stopped (maxlen reached). Only patterns up to a length of 5 returned!
```

```
##  done [0.04s].
## writing ... [12 rule(s)] done [0.00s].
## creating S4 object  ... done [0.06s].
```

```
inspect(head(sort(rules,by='count'),10))
```

```
##      lhs                                          rhs                                support confidenc
## [1]  {time=[11.3,17.5),
##       Borough=BRONX,
##       premiseDescription=CONSTRUCTION_SITE,
##       weekDay=Friday}                          => {offenseDescription=BURGLARY} 2.987776e-05  0.823529
## [2]  {time=[0,11.3),
##       Borough=QUEENS,
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=RESTAURANT_DINER}     => {offenseDescription=BURGLARY} 2.560951e-05  0.750000
## [3]  {time=[17.5,24],
##       Borough=BROOKLYN,
##       premiseDescription=CONSTRUCTION_SITE,
##       weekDay=Thursday}                        => {offenseDescription=BURGLARY} 1.920713e-05  0.750000
## [4]  {time=[0,11.3),
##       Borough=BROOKLYN,
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=RESTAURANT_DINER}     => {offenseDescription=BURGLARY} 1.920713e-05  0.750000
## [5]  {time=[11.3,17.5),
##       Borough=BRONX,
##       premiseDescription=CONSTRUCTION_SITE,
##       weekDay=Saturday}                        => {offenseDescription=BURGLARY} 1.707300e-05  0.800000
## [6]  {time=[11.3,17.5),
##       Borough=BROOKLYN,
##       premiseDescription=CONSTRUCTION_SITE,
##       weekDay=Sunday}                          => {offenseDescription=BURGLARY} 1.280475e-05  0.750000
## [7]  {time=[0,11.3),
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=RESTAURANT_DINER,
##       weekDay=Monday}                          => {offenseDescription=BURGLARY} 1.280475e-05  0.857142
## [8]  {time=[0,11.3),
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=RESTAURANT_DINER,
##       weekDay=Saturday}                        => {offenseDescription=BURGLARY} 1.280475e-05  0.750000
## [9]  {crimeCompleted=ATTEMPTED,
##       premiseDescription=CHURCH,
##       weekDay=Saturday}                        => {offenseDescription=BURGLARY} 1.067063e-05  0.833333
## [10] {Borough=QUEENS,
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=CHURCH}               => {offenseDescription=BURGLARY} 1.067063e-05  1.000000
```

```
rules <- apriori(data = subset(crimeData,year == 2015)[,c(4,5,9,11,14,15)] ,
parameter = list(support = 0.00001 , confidence = 0.75,maxlen = 5,target='rules'),
appearance = list(rhs = c('offenseDescription=GRAND_LARCENY')))
```

```
## Warning: Column(s) 1, 2, 3, 4, 5, 6 not logical or factor. Applying default
## discretization (see '? discretizeDF').
```

```
## Apriori
##
## Parameter specification:
```

```
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.75    0.1    1 none FALSE            TRUE       5   1e-05      1
##  maxlen target  ext
##       5  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 4
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[154 item(s), 468576 transaction(s)] done [0.14s].
## sorting and recoding items ... [143 item(s)] done [0.02s].
## creating transaction tree ... done [0.35s].
## checking subsets of size 1 2 3 4 5

## Warning in apriori(data = subset(crimeData, year == 2015)[, c(4, 5, 9, 11, :
## Mining stopped (maxlen reached). Only patterns up to a length of 5 returned!

##  done [0.04s].
## writing ... [26 rule(s)] done [0.00s].
## creating S4 object  ... done [0.06s].
```

```
inspect(head(sort(rules,by='count'),10))
```

```
##      lhs                             rhs                                    support confidence     co
## [1]  {time=[0,11.3),
##       Borough=QUEENS,
##       premiseDescription=ATM}    => {offenseDescription=GRAND_LARCENY} 4.695076e-05  0.8461538 5.548
## [2]  {time=[0,11.3),
##       Borough=QUEENS,
##       crimeCompleted=COMPLETED,
##       premiseDescription=ATM}    => {offenseDescription=GRAND_LARCENY} 4.268251e-05  0.8333333 5.121
## [3]  {time=[0,11.3),
##       premiseDescription=ATM,
##       weekDay=Thursday}          => {offenseDescription=GRAND_LARCENY} 3.628013e-05  0.7727273 4.695
## [4]  {time=[0,11.3),
##       crimeCompleted=COMPLETED,
##       premiseDescription=ATM,
##       weekDay=Thursday}          => {offenseDescription=GRAND_LARCENY} 3.414601e-05  0.7619048 4.481
## [5]  {time=[0,11.3),
##       premiseDescription=ATM,
##       weekDay=Wednesday}         => {offenseDescription=GRAND_LARCENY} 3.201188e-05  0.7500000 4.268
## [6]  {crimeCompleted=ATTEMPTED,
##       premiseDescription=ATM}    => {offenseDescription=GRAND_LARCENY} 2.774363e-05  0.7647059 3.628
## [7]  {time=[0,11.3),
##       crimeCompleted=ATTEMPTED,
##       premiseDescription=ATM}    => {offenseDescription=GRAND_LARCENY} 1.920713e-05  1.0000000 1.920
## [8]  {Borough=QUEENS,
##       premiseDescription=ATM,
##       weekDay=Friday}            => {offenseDescription=GRAND_LARCENY} 1.920713e-05  0.7500000 2.560
## [9]  {Borough=QUEENS,
```

```
##       crimeCompleted=COMPLETED,
##       premiseDescription=ATM,
##       weekDay=Friday}            => {offenseDescription=GRAND_LARCENY} 1.920713e-05  0.7500000 2.560
## [10] {Borough=QUEENS,
##       premiseDescription=ATM,
##       weekDay=Thursday}          => {offenseDescription=GRAND_LARCENY} 1.707300e-05  0.8888889 1.920
```

Hotspots detection

Crime hotspots are areas within the city that experience a high concentration of criminal activity.

The primary motivation behind analyzing crime hotspots is to enable law enforcement to allocate resources effectively, focusing on potential hubs of criminal activity.

The analysis of crime locations and their associated data is a fundamental aspect of crime analysis.

This kind of analysis holds significant importance because it highlights that the risk of becoming a victim of a particular type of crime is not uniformly distributed geographically.

According to crime pattern theory, crimes do not occur randomly. The initial definition implies that identifying crime clusters based on density is a crucial approach."

This revised text maintains the original content while improving the flow and readability of the information.

```
library(dbscan)
```

```
##
## Attaching package: 'dbscan'
```

```
## The following object is masked from 'package:stats':
##
##     as.dendrogram
```

```
library(ggmap)
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

```
library(leaflet)
# Citation:
citation("ggmap")
```

```
## To cite package 'ggmap' in publications use:
##
##   D. Kahle and H. Wickham. ggmap: Spatial Visualization with ggplot2.
##   The R Journal, 5(1), 144-161. URL
##   http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf
##
## A BibTeX entry for LaTeX users is
##
##   @Article{,
##     author = {David Kahle and Hadley Wickham},
##     title = {ggmap: Spatial Visualization with ggplot2},
##     journal = {The R Journal},
##     year = {2013},
##     volume = {5},
##     number = {1},
##     pages = {144--161},
##     url = {https://journal.r-project.org/archive/2013-1/kahle-wickham.pdf},
##   }
```

DBSCAN algorithm

For purpose of detecting crime hotspots it is appropriate to use DBSCAN(density-based spatial clustering of applications with noise) clustering algorithm. For a given a set of points in space, it groups together points that are closely packed together(nearby neigbors). Source: https://en.Wikipedia.org/wiki/DBSCAN

Simple example

```
data <- subset(crimeData, year >= 2014 & Borough == "BRONX" &
             offenseDescription == "MURDER_AND_NON_NEGL_MANSLAUGHTER" &
             !is.na(Longitude) & !is.na(Latitude))[, c(7, 8)]
options(repr.plot.width = 9, repr.plot.height = 7)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(
    lng = data$Longitude,
    lat = data$Latitude,
    radius = 7,
    color = "orange",
    fill = TRUE,
    fillOpacity = 0.6,
    fillColor = "brown"
  )
```

Murders in Bronx in period 2014 and 2015

```
data <- subset(crimeData, year >= 2014 & Borough == "BRONX" &
              offenseDescription == "MURDER_AND_NON_NEGL_MANSLAUGHTER" &
              !is.na(Longitude) & !is.na(Latitude))[, c(7, 8)]
options(repr.plot.width = 9, repr.plot.height = 7)
leaflet() %>%
  addTiles() %>%
```

```
  addCircleMarkers(
    data = data,
    lng = ~Longitude,
    lat = ~Latitude,
    radius = 5,
    color = "blue",
    fill = TRUE,
    fillOpacity = 0.7,
    stroke = TRUE,
    weight = 1,
    popup = ~paste("Latitude: ", Latitude, "<br>Longitude: ", Longitude)
  ) %>%
  addProviderTiles("CartoDB.PositronNoLabels")
```

```
options(repr.plot.width = 8, repr.plot.height = 7)
clust = dbscan(x = data, eps = 0.01, minPts = 20, borderPoints = FALSE)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(
    lng = data$Longitude[which(clust$cluster == 1)],
    lat = data$Latitude[which(clust$cluster == 1)],
    radius = 5,
    fillColor = "purple",
    color = "black",
    fillOpacity = 0.5,
    stroke = TRUE,
    weight = 2
  )
```

Robberies in Queens (2015)

Although crime hotspots can be found relatively easy with DBSCAN, they might be very natural because of greater density of population in that places(not visible from this data). Great density of population might imply greater density of some specific crime level.

```
options(repr.plot.width = 8, repr.plot.height = 8)
data <- subset(crimeData, year >= 2015 & month >= 10 & Borough == "QUEENS" &
               offenseDescription == "ROBBERY" & !is.na(Longitude) & !is.na(Latitude))[, c(7, 8)]
marker_color <- "coral2"
leaflet() %>%
  addProviderTiles("CartoDB.Positron") %>%
  addCircleMarkers(lng = data$Longitude, lat = data$Latitude,
                   color = marker_color,
                   radius = 5)
```

```
clust = dbscan(x = data, eps = 0.0095, minPts = 35, borderPoints = FALSE)
leaflet() %>%
  addTiles() %>%
  addCircleMarkers(
    lng = data$Longitude[which(clust$cluster >= 1)],
    lat = data$Latitude[which(clust$cluster >= 1)],
    color = "deeppink4",            # Change marker color
```

```
    radius = 7,                 # Change marker size
    fillOpacity = 0.7,          # Adjust fill opacity
    stroke = FALSE              # Remove marker border
  )
```

In example above DBSCAN algorithm found few clusters that could represent possible hotspots for certain level of crime. However, there are other methods for searching hotspots, like test for clustering. Testing for clustering is the first step in revealing whether data has crime hotspots.

In the example above, the DBSCAN algorithm identified several clusters that could potentially represent hotspots for specific levels of crime.

However, there are alternative methods for identifying hotspots, such as testing for clustering. Testing for clustering is the initial step in determining the presence of crime hotspots.

##Nearest Neighbor Index (NNI) NNI is a simple and quick method for assessing clustering. This test compares the actual distribution of crime data to a dataset of the same size but with a random distribution.

The NNI test involves the following steps:

Calculate the observed average nearest neighbor distance. For each point, find its closest neighbor, calculate the distance, and then average these distances. Repeat the same process for a random distribution of the same size to compute the average random nearest neighbor distance. Calculate the NNI as the ratio of the observed average nearest neighbor distance to the average random nearest neighbor distance. If the NNI result is 1, it suggests that the crime data are randomly distributed. If the NNI is less than 1, it indicates evidence of clustering. An NNI greater than 1 suggests a uniform pattern in the crime data.

##Z-Score Test Statistics To gain confidence in the NNI result, you can apply a z-score test statistic. This statistical test measures how different the actual average nearest neighbor distance is from the average random nearest neighbor distance.

The general principle is that a more negative z-score provides greater confidence in the NNI result.

##Example In the example above, the NNI is approximately 0.62, indicating that there is evidence of clustering, and it is unlikely to be a random occurrence."

```
library(spatstat)
```

```
## Loading required package: spatstat.data
```

```
## Loading required package: spatstat.geom
```

```
## spatstat.geom 3.2-5
```

```
##
## Attaching package: 'spatstat.geom'
```

```
## The following object is masked from 'package:arules':
##
##     compatible
```

```
## Loading required package: spatstat.random
```

```
## spatstat.random 3.1-6
```

```
## Loading required package: spatstat.explore

## Loading required package: nlme

##
## Attaching package: 'nlme'

## The following object is masked from 'package:dplyr':
##
##     collapse

## spatstat.explore 3.2-3

## Loading required package: spatstat.model

## Loading required package: rpart

## spatstat.model 3.2-6

## Loading required package: spatstat.linnet

## spatstat.linnet 3.1-1

##
## spatstat 3.0-6
## For an introduction to spatstat, type 'beginner'
```

```
library(sp)
```

Crime oportunity - vehicle crimes

Some places have great oportunity for vehicle crimes

```
options(repr.plot.width = 8, repr.plot.height = 5)
data <- subset(crimeData, year >= 2014 & Borough == "STATEN_ISLAND" & offenseDescription == "VEHICLE_AN
leaflet() %>%
  addProviderTiles("CartoDB.Positron") %>%
  addCircleMarkers(data = data, radius = 5, fillOpacity = 0.7, color = "grey", stroke = FALSE)
```

Conclusion

Crime remains an integral facet of modern society, particularly in urban and metropolitan areas. Over the past few decades, advancements in technology and extensive statistical research have provided scientists and researchers with sophisticated tools for crime analysis and prevention.

These analytical approaches have consistently demonstrated that crime is not a random occurrence; rather, it is often driven by identifiable factors that can be understood and, to some extent, predicted. Even basic statistical analyses and tests can uncover hidden correlations within the data that may not be immediately apparent.

However, a fundamental question arises in the wake of crime analysis and prevention efforts: Do these initiatives effectively reduce crime in specific locations, or do they merely displace criminal activities to other areas? For instance, hotspot analysis may identify and allow authorities to combat crimes associated with drugs in certain areas. Yet, over time, these efforts may inadvertently lead to the emergence of new hotspots for drug-related crimes in previously unaffected regions. This dynamic underscores the complex and evolving nature of crime patterns and the need for a continuous and adaptive approach to crime prevention.