PROJECT II: San Francisco Crime (2016)

FH

December 28, 2020

Inroduction

The analysis performed within this project is based on a data set entitled San Francisco Crime is available on Kaggle and can be found here:

- https://www.kaggle.com/roshansharma/sanfranciso-crime-dataset

The original dataset is from https://datasf.org/opendata, the central clearinghouse for data published by the City and County of San Francisco. The San Francisco Police Department does not guarantee the accuracy, completeness, timeliness or correct sequencing of the information as the data is subject to change as modifications and updates are completed. The complete data set can be found here:

- https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-Historical-2003/tmnf-yvry

Due to computation power issues only a subset of the data was used. The algorithms were developed using the **sf_crime** data, which is a subset of the of the SF Opendata and includes only 2016. For the evaluation of the recommendation algorithm a **validation** data set was generated. The **validation** data set was only used in the final step to test the final algorithm and contained only 20% of **sf_crime** data.

The following libraries were used:

```r
library(tinytex)
library(factoextra)
library(ggthemes)
library(Hmisc)
library(tidyr)
library(dslabs)
library(tidyverse)
library(caret)
library(data.table)
library(corrplot)
library(dplyr)
library(ggplot2)
library(lubridate)
library(stringr)
library(readr)
library(gridExtra)
```

Data Exploration and Data Processing

The **sf_cime** data set contains 150,500 observations and 13 variables represented in 13 columns. Each row represents one incident with an ID given by the Police Department (*PdId*).

```
#  Number of variables
print(ncol(sf_crime))
```

```
## [1] 13
```

```
#  Number of observations
print(nrow(sf_crime))
```

```
## [1] 150500
```

This data set contains incidents derived from SFPD Crime Incident Reporting system. The data ranges from January to December 2016. The downloaded data set **sf_crime** consists of following 13 variables:

- *PdId* - ID given by Police Department District
- *IncidntNum* - ID of the incident
- *Date* - date of the incident in mm/dd/yy format
- *Time* - time of the incident
- *Category* - category of the crime incident
- *Descript* - detailed description of the crime incident
- *DayOfWeek* - the day of the week
- *PdDistrict* - name of the Police Department District
- *Resolution* - how the crime incident was resolved
- *Address* - the approximate street address of the crime incident
- *X* - Longitude
- *Y* - Latitude
- *Location* - Longitude and Latitude combined in one variable

The data set **sf_crime** contains no missing data.

```
# Check missing values
sum(is.na(sf_crime))
```

```
## [1] 0
```

In order to visualize and analyze the data set several new variables were created through transformation of existing variables.

The variable *Date* was transformed and split into *Date_Month*, *Date_Day*, *Date_Year*. Variable *Time* was converted into a numerical variable *Time_Hour* that shows time of the day when crime took place. *Resolution_dummy* was created to indicate if the incident was resolved, it was set to 1 if resolved otherwise 0. In order to simplify variable *Category* a dummy variable *Category_Violent_dummy* was assigned value 1 when "ASSAULT", "ROBBERY", "SEX OFFENSES, FORCIBLE", "KIDNAPPING" and value 0 otherwise. New variable *Count_Address_Occur* indicates how many times incidents took place at the same address.

```
# Summary of the data set
summary(sf_crime)
```

```
##       PdId                 IncidntNum                Category
##  Min.   : 1135121075000   Min.   : 11351210   LARCENY/THEFT :40409
##  1st Qu.:16032832064300   1st Qu.:160328320   OTHER OFFENSES:19599
##  Median :16065407015100   Median :160654070   NON-CRIMINAL  :17866
##  Mean   :16164403714000   Mean   :161644037   ASSAULT       :13577
##  3rd Qu.:16097640758300   3rd Qu.:160976408   VANDALISM     : 8589
##  Max.   :99100899765000   Max.   :991008997   VEHICLE THEFT : 6419
##                                               (Other)       :44041
##  Count_Category_Occur Category_Violent_dummy
##  Min.   :    3        Min.   :0.0000
##  1st Qu.: 5802        1st Qu.:0.0000
##  Median :17866        Median :0.0000
##  Mean   :18642        Mean   :0.1201
##  3rd Qu.:40409        3rd Qu.:0.0000
##  Max.   :40409        Max.   :1.0000
##
##                           Descript      Resolution       Resolution_dummy
##  GRAND THEFT FROM LOCKED AUTO : 17741   Length:150500      Min.   :0.0000
##  LOST PROPERTY                :  4596   Class :character   1st Qu.:0.0000
##  AIDED CASE, MENTAL DISTURBED :  4566   Mode  :character   Median :0.0000
##  PETTY THEFT OF PROPERTY      :  4416                      Mean   :0.2839
##  MALICIOUS MISCHIEF, VANDALISM:  4262                      3rd Qu.:1.0000
##  BATTERY                      :  4211                      Max.   :1.0000
##  (Other)                      :110708
##      DayOfWeek        Date_Day       Date_Year      Date_Month
##  Monday   :20783   Min.   : 1.00   Min.   :2016   Min.   : 1.000
##  Tuesday  :21242   1st Qu.: 8.00   1st Qu.:2016   1st Qu.: 4.000
##  Wednesday:21332   Median :16.00   Median :2016   Median : 7.000
##  Thursday :21395   Mean   :15.76   Mean   :2016   Mean   : 6.539
##  Friday   :23371   3rd Qu.:23.00   3rd Qu.:2016   3rd Qu.:10.000
##  Saturday :22172   Max.   :31.00   Max.   :2016   Max.   :12.000
##  Sunday   :20205
##        Month         Time_Hour       Date_time            PdDistrict
##  October  :13331   Min.   : 0.00   Min.   :2016-01-01   Length:150500
##  January  :12946   1st Qu.: 9.00   1st Qu.:2016-04-01   Class :character
##  December :12926   Median :14.00   Median :2016-07-02   Mode  :character
##  May      :12713   Mean   :13.33   Mean   :2016-07-02
##  November :12670   3rd Qu.:19.00   3rd Qu.:2016-10-04
##  September:12473   Max.   :23.00   Max.   :2016-12-31
##  (Other)  :73441
##    Address          Count_Address_Occur       Y               X
##  Length:150500      Min.   :   1.0      Min.   :37.71   Min.   :-122.5
##  Class :character   1st Qu.:   9.0      1st Qu.:37.76   1st Qu.:-122.4
##  Mode  :character   Median :  23.0      Median :37.78   Median :-122.4
##                     Mean   : 154.8      Mean   :37.77   Mean   :-122.4
##                     3rd Qu.:  71.0      3rd Qu.:37.79   3rd Qu.:-122.4
##                     Max.   :3561.0      Max.   :37.82   Max.   :-122.4
##
##    Location
##  Length:150500
##  Class :character
##  Mode  :character
##
##
```

```
##
##
```

PdId is a unique identifier for each crime incident, whereas *IncidntNum* can be assigned to several crimes that happened during the same incident.

```r
# Total number of duplicates among "IncidntNum"(incident number)
print(sum(duplicated(sf_crime$IncidntNum)))
```

```
## [1] 33801
```

33801 *IncidntNum* incident numbers have more than one crime category assigned to them. That means that 23% of all incidents happened in year 2016 involved more than one crime.

```r
# Find index of the duplicates among "IncidntNum" (incident ID)
dup <- sf_crime$IncidntNum[duplicated(sf_crime$IncidntNum)]

# Print 10 incident ID's with more than one crime
head(dup, 10) %>% knitr::kable("pipe")
```

x
16059760
16059760
100475254
100475254
110330052
120058272
120058272
120058272
120058272
120058272

The new processed data set contains 21 variables instead of original 13

```r
#   Description of variables in processed sf_crime
summary(sf_crime)
```

```
##       PdId                  IncidntNum                Category
##  Min.   : 1135121075000   Min.   : 11351210   LARCENY/THEFT :40409
##  1st Qu.:16032832064300   1st Qu.:160328320   OTHER OFFENSES:19599
##  Median :16065407015100   Median :160654070   NON-CRIMINAL  :17866
##  Mean   :16164403714000   Mean   :161644037   ASSAULT       :13577
##  3rd Qu.:16097640758300   3rd Qu.:160976408   VANDALISM     : 8589
##  Max.   :99100899765000   Max.   :991008997   VEHICLE THEFT : 6419
##                                               (Other)       :44041
##  Count_Category_Occur Category_Violent_dummy
##  Min.   :    3        Min.   :0.0000
##  1st Qu.: 5802        1st Qu.:0.0000
##  Median :17866        Median :0.0000
##  Mean   :18642        Mean   :0.1201
```

```
##  3rd Qu.:40409        3rd Qu.:0.0000
##  Max.   :40409        Max.   :1.0000
##
##                          Descript      Resolution       Resolution_dummy
##  GRAND THEFT FROM LOCKED AUTO : 17741   Length:150500      Min.   :0.0000
##  LOST PROPERTY                :  4596   Class :character   1st Qu.:0.0000
##  AIDED CASE, MENTAL DISTURBED :  4566   Mode  :character   Median :0.0000
##  PETTY THEFT OF PROPERTY      :  4416                      Mean   :0.2839
##  MALICIOUS MISCHIEF, VANDALISM:  4262                      3rd Qu.:1.0000
##  BATTERY                      :  4211                      Max.   :1.0000
##  (Other)                      :110708
##      DayOfWeek        Date_Day       Date_Year      Date_Month
##  Monday   :20783   Min.   : 1.00   Min.   :2016   Min.   : 1.000
##  Tuesday  :21242   1st Qu.: 8.00   1st Qu.:2016   1st Qu.: 4.000
##  Wednesday:21332   Median :16.00   Median :2016   Median : 7.000
##  Thursday :21395   Mean   :15.76   Mean   :2016   Mean   : 6.539
##  Friday   :23371   3rd Qu.:23.00   3rd Qu.:2016   3rd Qu.:10.000
##  Saturday :22172   Max.   :31.00   Max.   :2016   Max.   :12.000
##  Sunday   :20205
##        Month         Time_Hour       Date_time          PdDistrict
##  October  :13331   Min.   : 0.00   Min.   :2016-01-01   Length:150500
##  January  :12946   1st Qu.: 9.00   1st Qu.:2016-04-01   Class :character
##  December :12926   Median :14.00   Median :2016-07-02   Mode  :character
##  May      :12713   Mean   :13.33   Mean   :2016-07-02
##  November :12670   3rd Qu.:19.00   3rd Qu.:2016-10-04
##  September:12473   Max.   :23.00   Max.   :2016-12-31
##  (Other)  :73441
##    Address          Count_Address_Occur       Y              X
##  Length:150500      Min.   :   1.0      Min.   :37.71   Min.   :-122.5
##  Class :character   1st Qu.:   9.0      1st Qu.:37.76   1st Qu.:-122.4
##  Mode  :character   Median :  23.0      Median :37.78   Median :-122.4
##                     Mean   : 154.8      Mean   :37.77   Mean   :-122.4
##                     3rd Qu.:  71.0      3rd Qu.:37.79   3rd Qu.:-122.4
##                     Max.   :3561.0      Max.   :37.82   Max.   :-122.4
##
##    Location
##  Length:150500
##  Class :character
##  Mode  :character
##
##
##
##
```

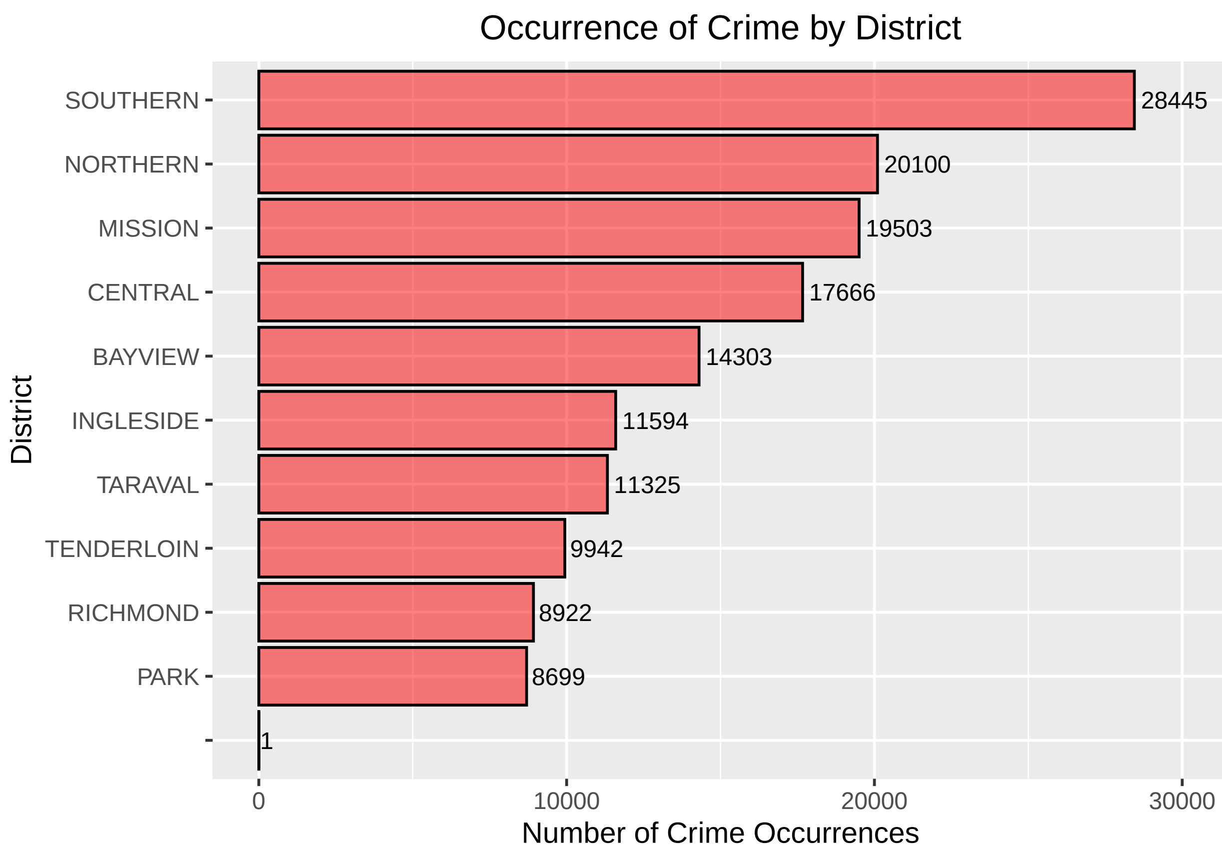
The number of unique crime incidents is 150,500 and there are 39 listed crime categories. The *Description* of crimes includes 726 different descriptions. This variable was not strictly defined and was up to the police officers to choose. 16130 different addresses are present in the data set. There are 10 districts (plus one incident without a district) of Police Department and 14 resolution of crime categories.

n_ids	n_crime_category	n_description	n_district	n_resolution	n_address
150500	39	726	11	14	16130

Visualization

District

The occurrence of crime in 10 different districts shows that by far the most criminal district is Southern district followed by Northern and Mission. Park district shows the least crime occurrences, with 8699 incidents for 2016.



Category

The variable *Category* is a factor variable with 39 different levels.

```
# Crime Category
class(sf_crime$Category)
```

```
## [1] "factor"
```

```
nlevels(sf_crime$Category)
```

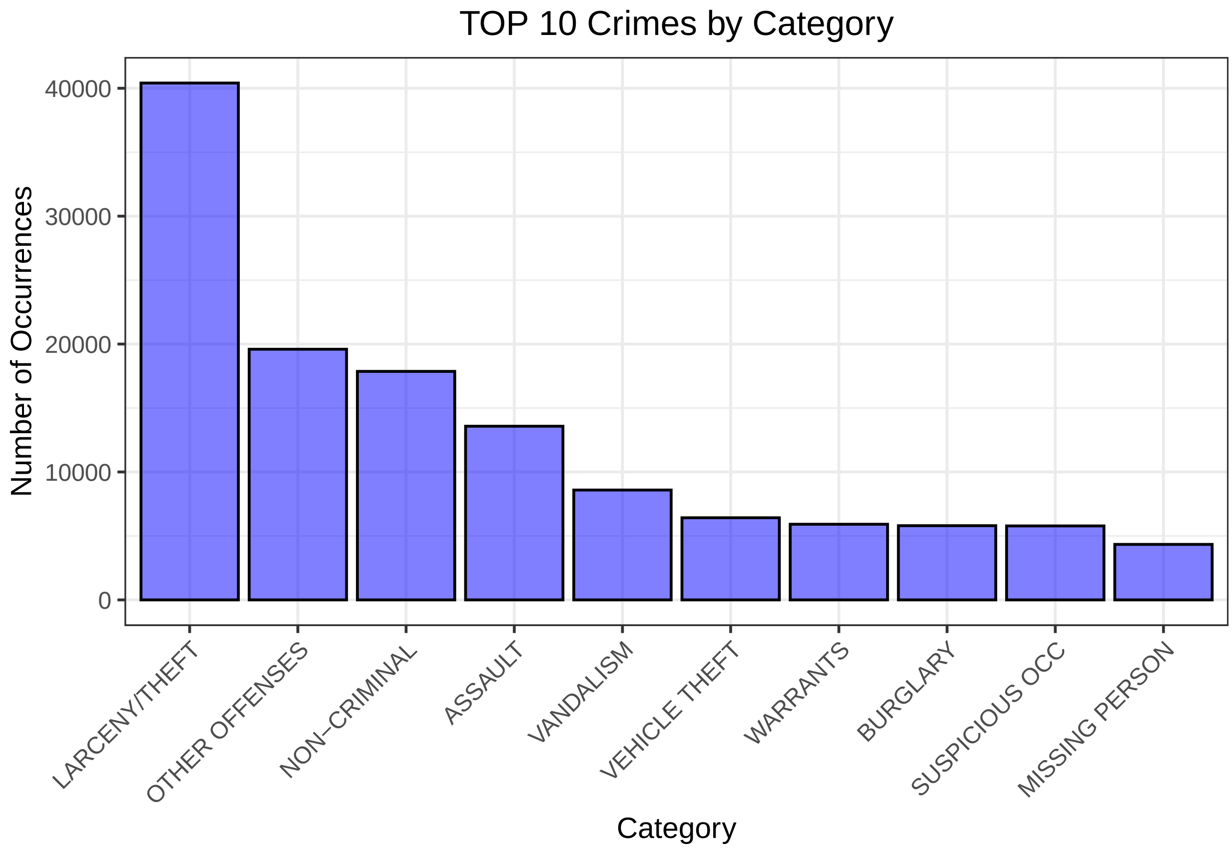
```
## [1] 39
```

```
print(unique(sf_crime$Category))
```

```
##  [1] MISSING PERSON              LARCENY/THEFT
##  [3] OTHER OFFENSES              BURGLARY
##  [5] SUSPICIOUS OCC              WARRANTS
##  [7] ASSAULT                     NON-CRIMINAL
##  [9] STOLEN PROPERTY             WEAPON LAWS
## [11] DRUG/NARCOTIC               EMBEZZLEMENT
## [13] RUNAWAY                     DRUNKENNESS
## [15] FORGERY/COUNTERFEITING      ROBBERY
## [17] VEHICLE THEFT               FRAUD
## [19] SEX OFFENSES, FORCIBLE      SECONDARY CODES
## [21] KIDNAPPING                  VANDALISM
## [23] PROSTITUTION                DRIVING UNDER THE INFLUENCE
## [25] RECOVERED VEHICLE           SEX OFFENSES, NON FORCIBLE
## [27] TRESPASS                    ARSON
## [29] DISORDERLY CONDUCT          LIQUOR LAWS
## [31] FAMILY OFFENSES             EXTORTION
## [33] BAD CHECKS                  LOITERING
## [35] SUICIDE                     BRIBERY
## [37] GAMBLING                    PORNOGRAPHY/OBSCENE MAT
## [39] TREA
## 39 Levels: MISSING PERSON LARCENY/THEFT OTHER OFFENSES ... TREA
```

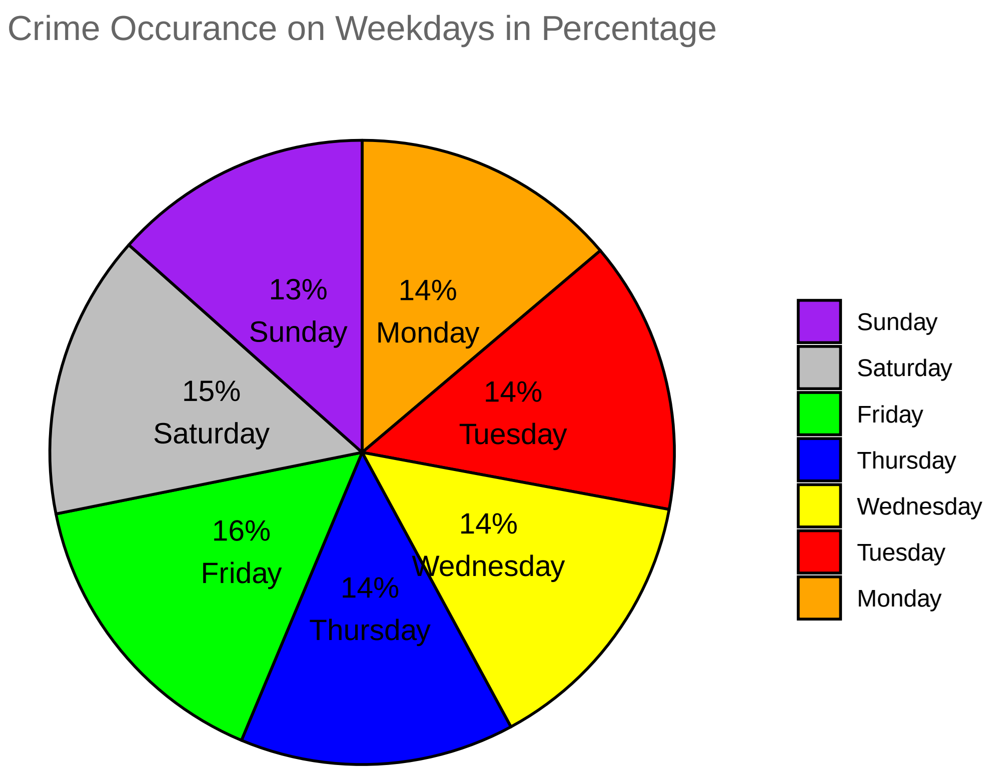
The following categories are the 10 most common. "LARCENY/THEFT" accounts for the most cases and "MISSING PERSON" is the smallest category in the top ten.

Category	count
LARCENY/THEFT	40409
OTHER OFFENSES	19599
NON-CRIMINAL	17866
ASSAULT	13577
VANDALISM	8589
VEHICLE THEFT	6419
WARRANTS	5914
BURGLARY	5802
SUSPICIOUS OCC	5782
MISSING PERSON	4338



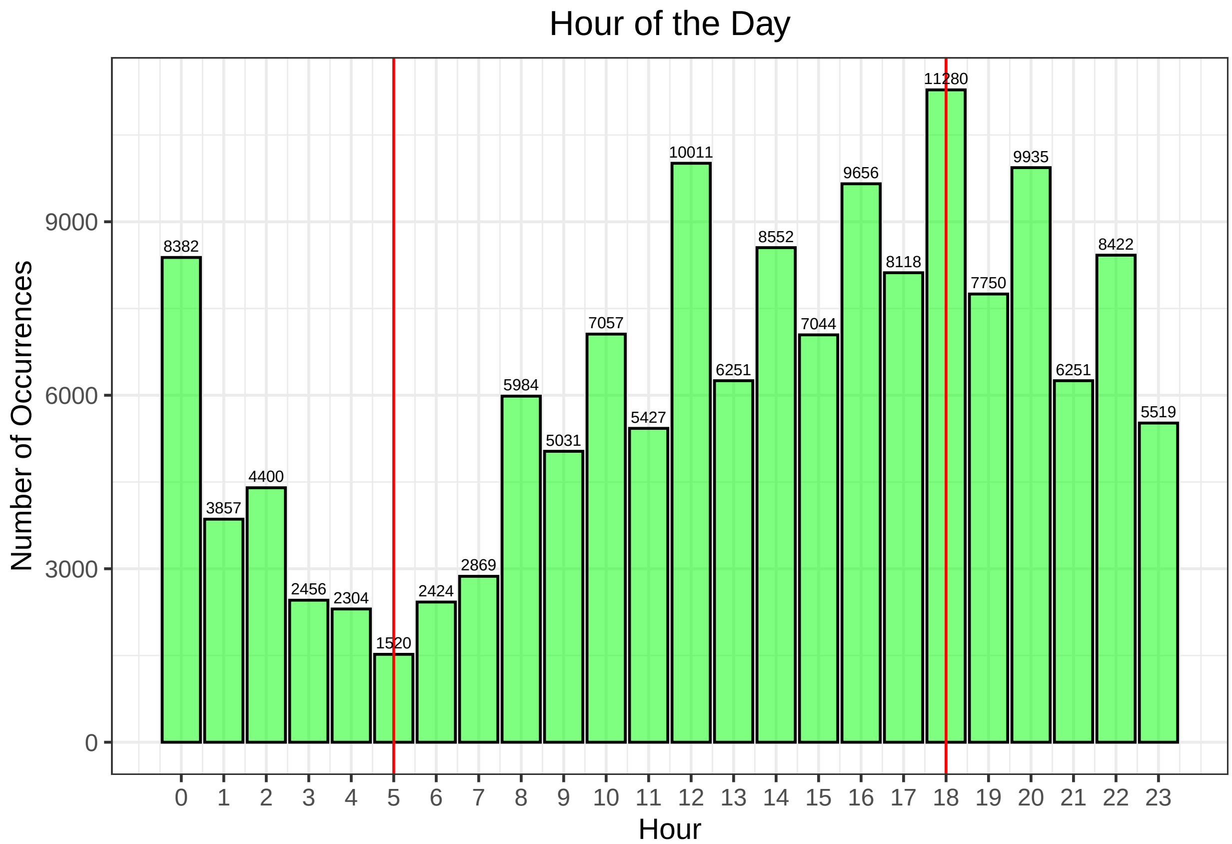
Weekdays

The following pie-chart shows that crime occurrence over the days of the week was almost evenly distributed.

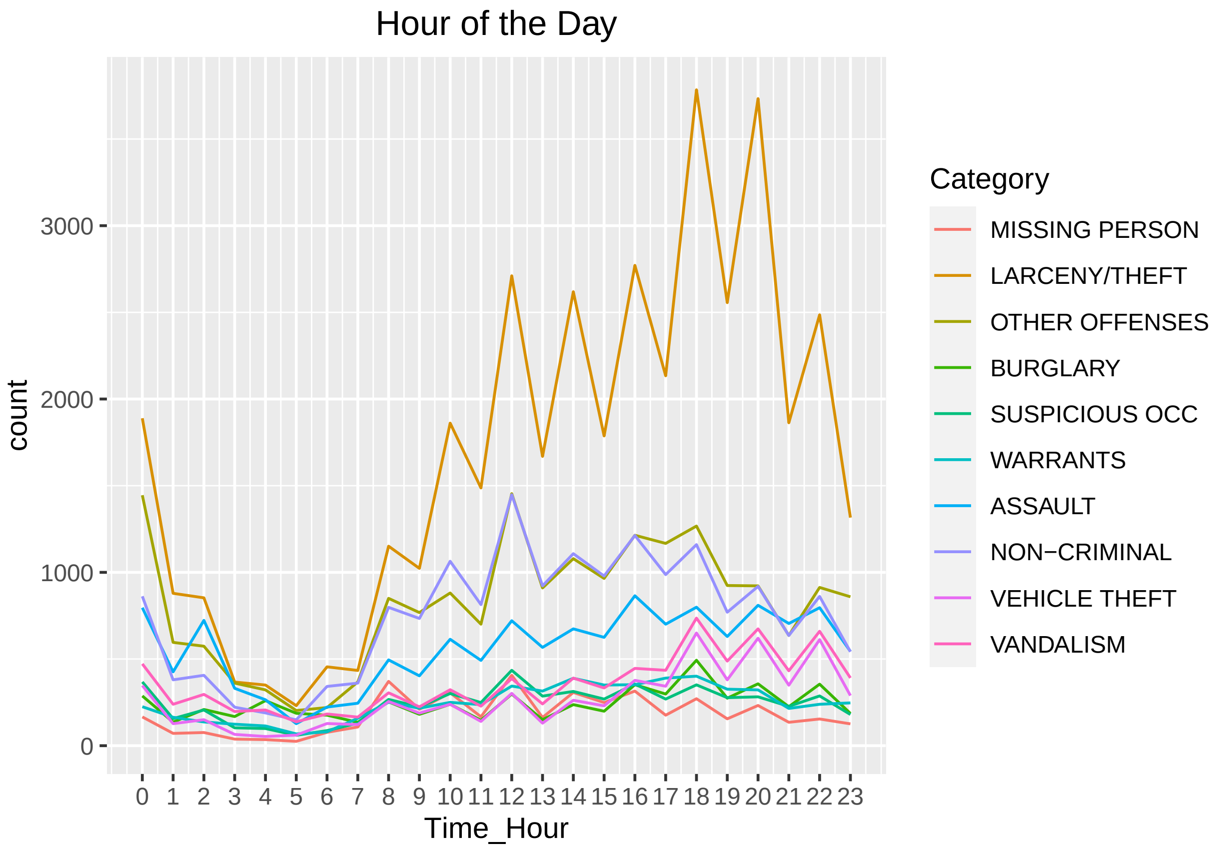


Daytime

The day time plot shows that most crime occurred around 18:00 (6pm) with N = 11280 cases. According to the data for 2016 the safest hour was 5am in the morning with N=1520 incidents.

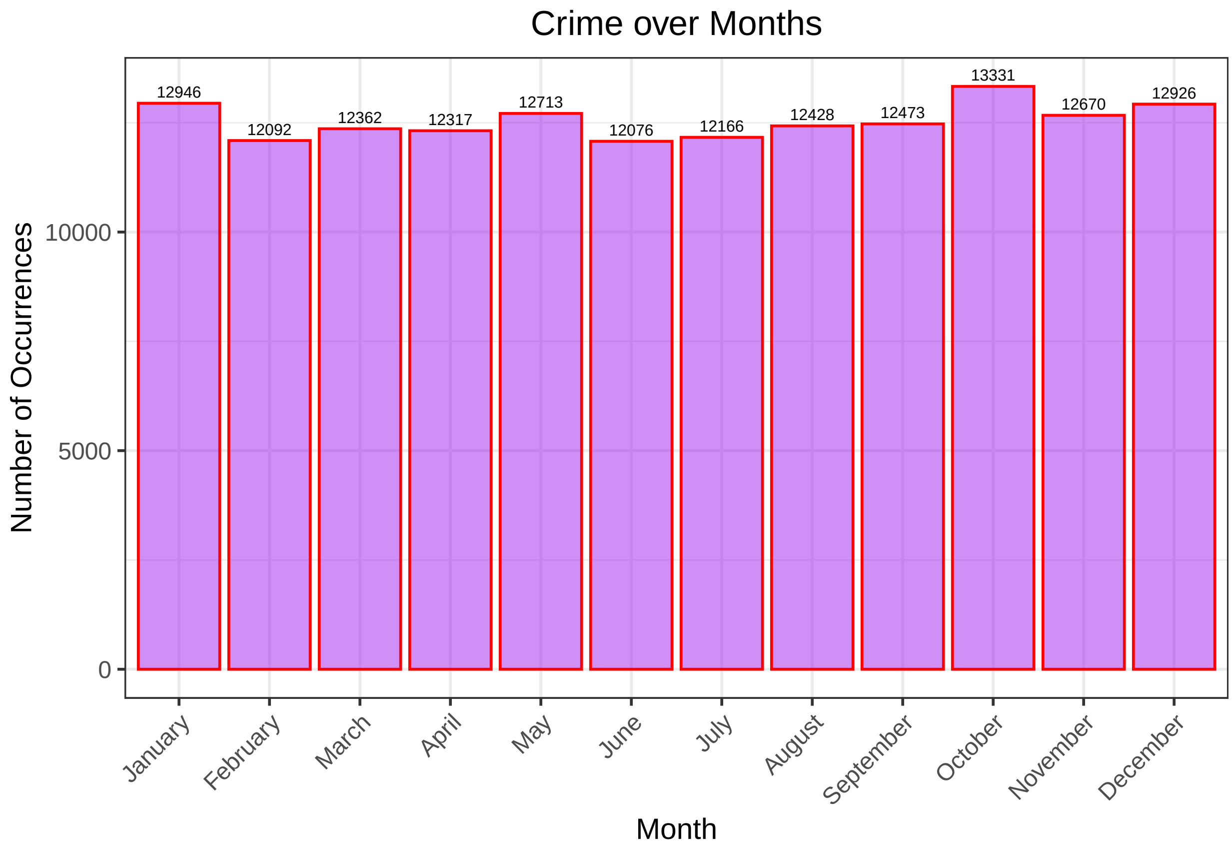


Crime Category "LARCENY/THEFT" remains the most committed crime throughout the day. The number of "LARCENY/THEFT" increases in the evening hours.



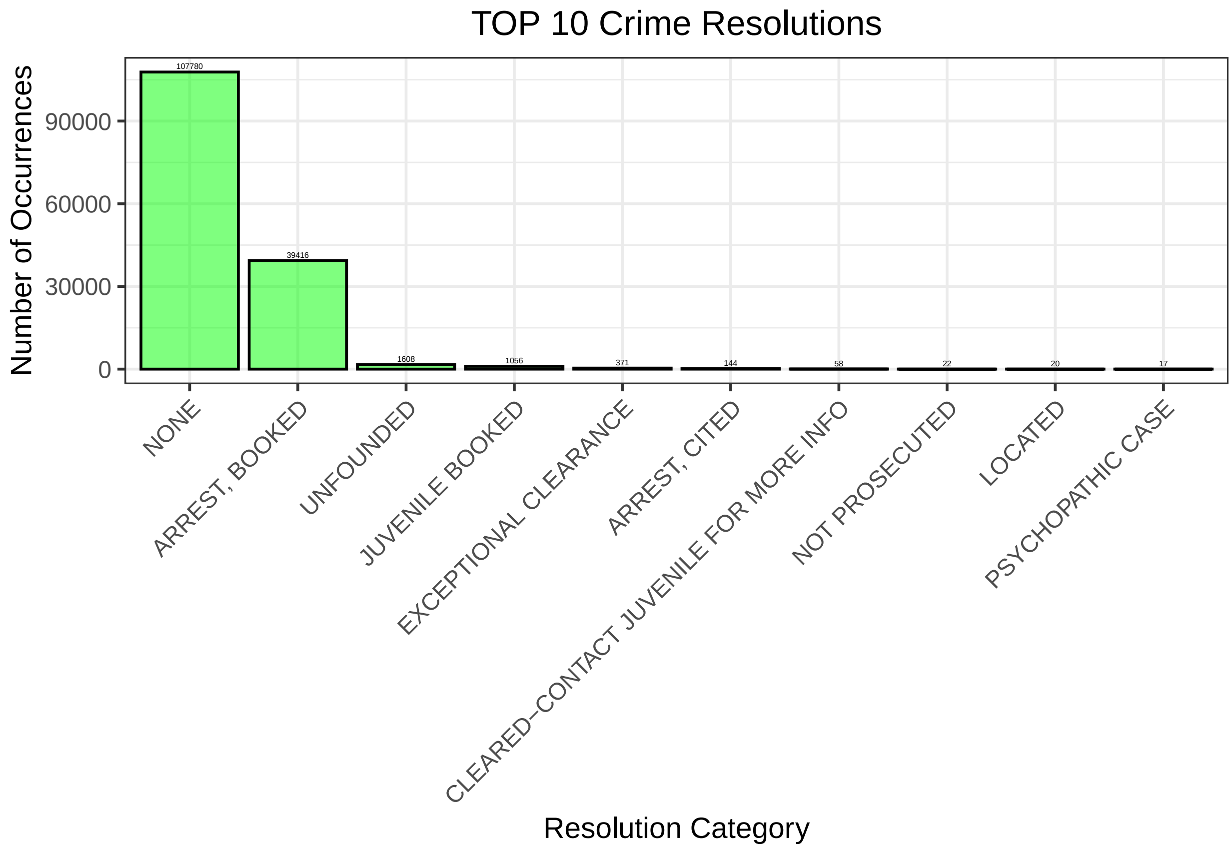
Months

The crime distribution among the months of 2016 also shows a fairly even distribution, with October with the largest number of crime incidents.

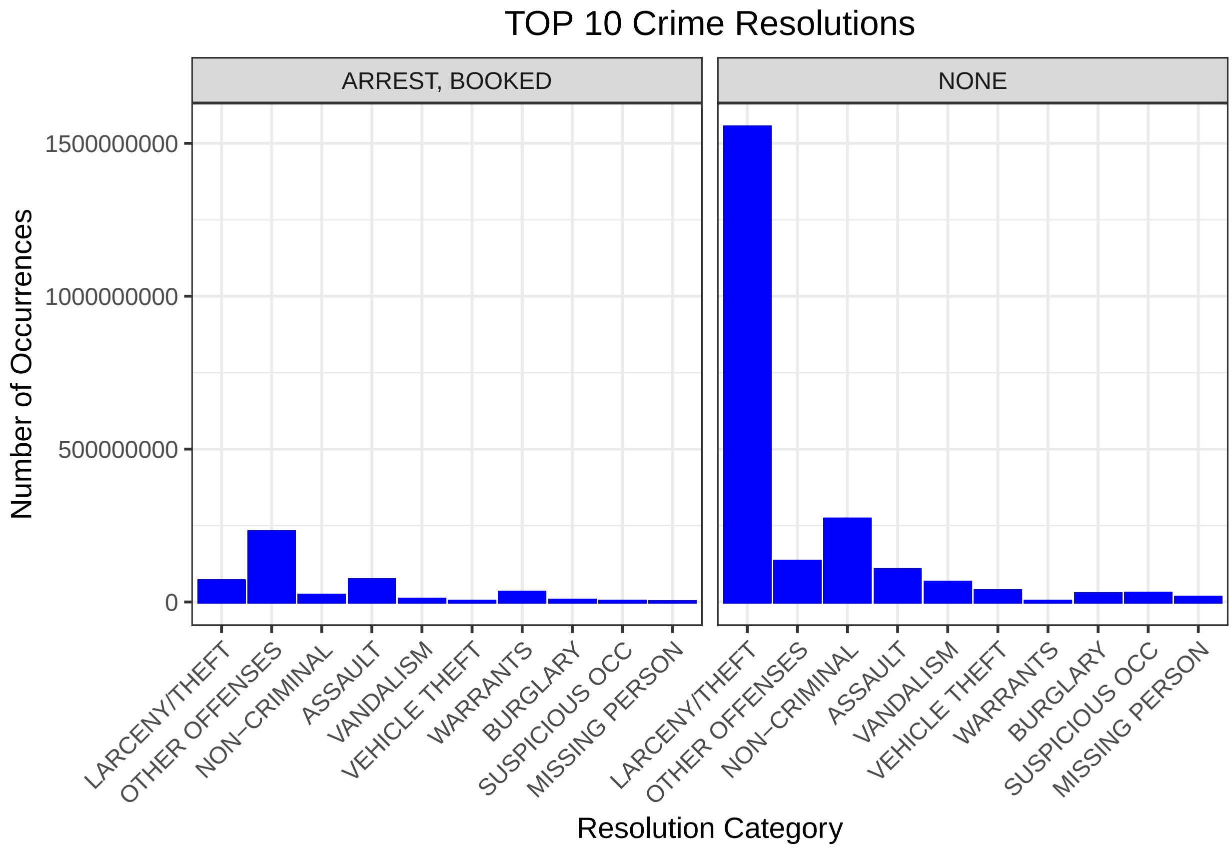


Resolution of Crime

An important variable in the crime statistics is a crime resolution variable. As we can see most of the cases were not resolved. The second biggest category within the variable **Resolution** is "ARREST, BOOKED".



Among the top ten crimes category "ARREST, BOOKED" accounts most for "OTHER OFFENSES" and "ASSAULT", while category "NONE" resolutions is highest for "LARCENY/THEFT".



Description

The variable *Description* contains 726 different non predefined descriptions of crime and therefor can not be taken as a reliable source into the analysis. Following table shows to 10 of used descriptions to describe crime incidents.

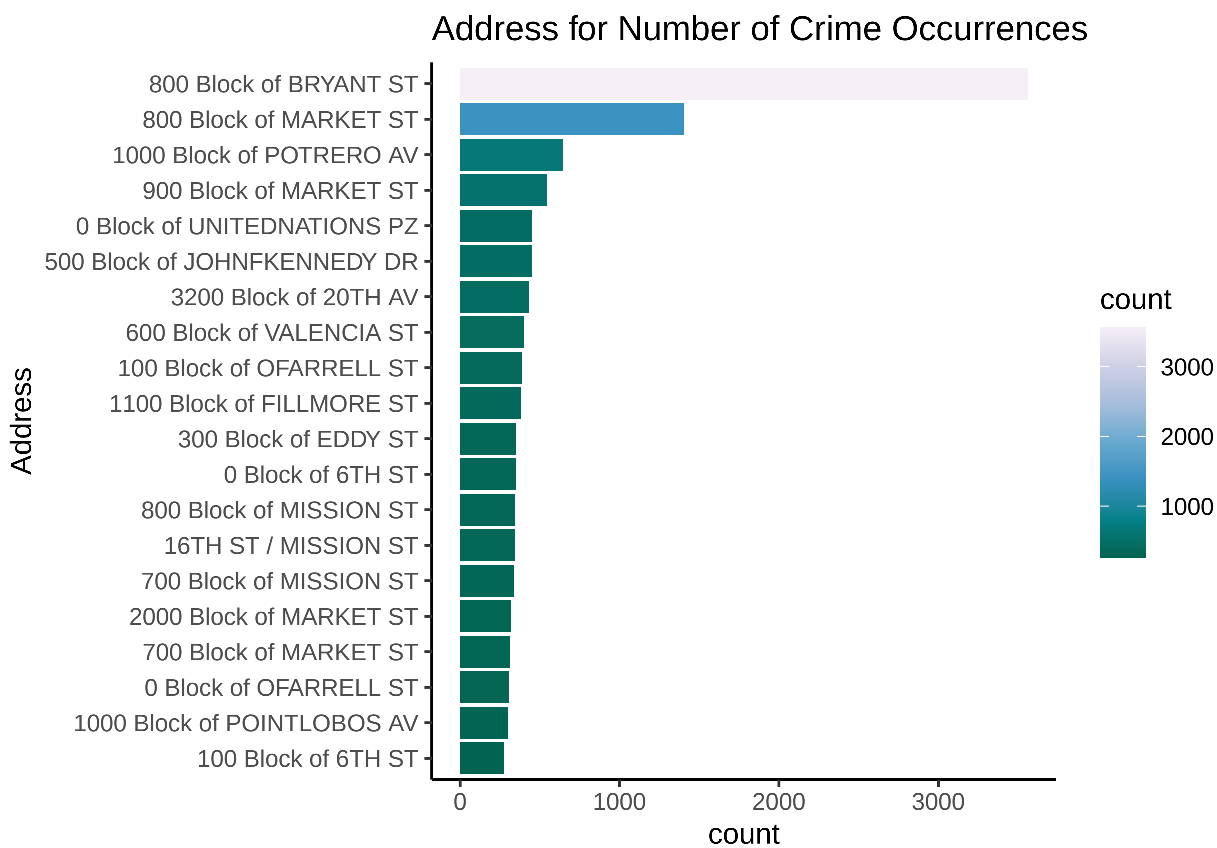
Descript	count
GRAND THEFT FROM LOCKED AUTO	17741
LOST PROPERTY	4596
AIDED CASE, MENTAL DISTURBED	4566
PETTY THEFT OF PROPERTY	4416
MALICIOUS MISCHIEF, VANDALISM	4262
BATTERY	4211
PETTY THEFT FROM LOCKED AUTO	3994
STOLEN AUTOMOBILE	3603
DRIVERS LICENSE, SUSPENDED OR REVOKED	3376
WARRANT ARREST	3089

Address

Some addresses appeared more often in the data than others. There are 16130 different addresses in the data set. The following visualization shows the top 20 most frequent addresses in the crime data for 2016.

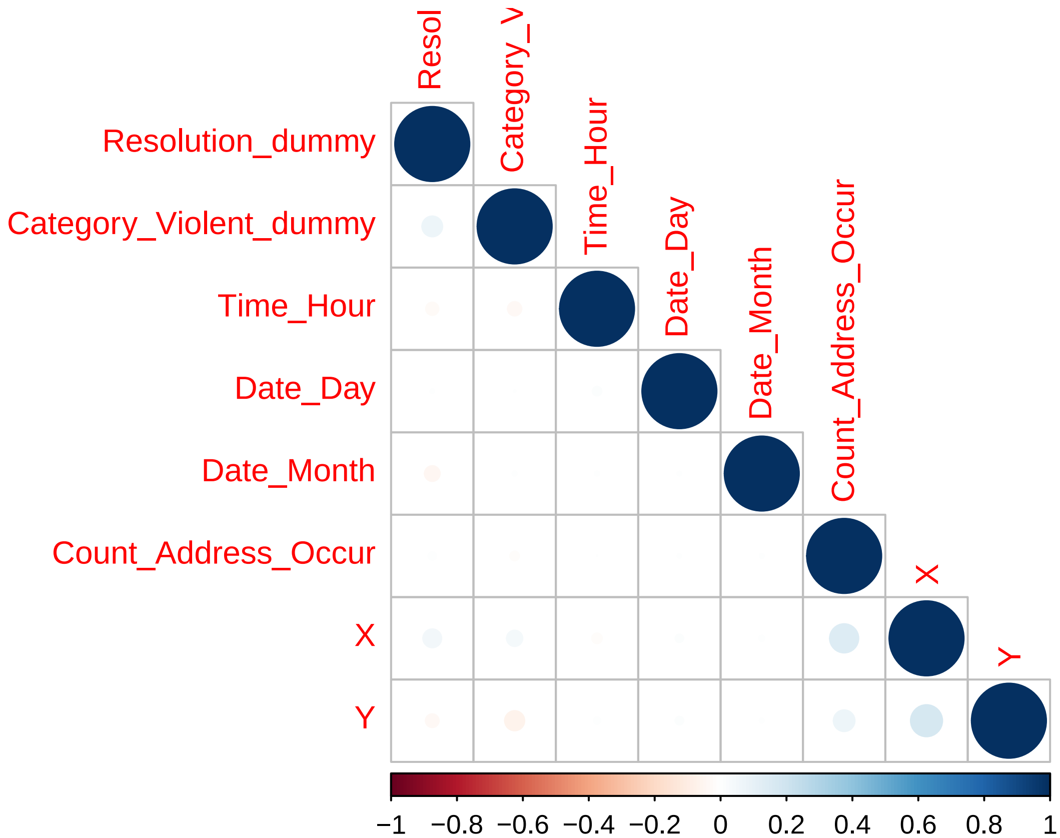
Address	count
800 Block of BRYANT ST	3561
800 Block of MARKET ST	1405
1000 Block of POTRERO AV	644
900 Block of MARKET ST	547
0 Block of UNITEDNATIONS PZ	452
500 Block of JOHNFKENNEDY DR	448
3200 Block of 20TH AV	431
600 Block of VALENCIA ST	399
100 Block of OFARRELL ST	389
1100 Block of FILLMORE ST	382
0 Block of 6TH ST	347
300 Block of EDDY ST	347
800 Block of MISSION ST	345
16TH ST / MISSION ST	343
700 Block of MISSION ST	336
2000 Block of MARKET ST	320
700 Block of MARKET ST	310
0 Block of OFARRELL ST	306
1000 Block of POINTLOBOS AV	298
100 Block of 6TH ST	272

The top address that occurred 3561 times in 2016 is "800 Block of BRYANT ST", which is the address of a county jail.



Method

The goal of the analysis is to predict the category of crime by using predictors considered in the Visualization part. First we take a look at the correlation between the variables.



In order to predict a categorical variable, *Logistic Regression*[1] will be applied. As an outcome of the prediction *Category_Violent_dummy* with assigned values of 0 (non-violent) and 1 (violent) will be used. *Logistic Regression* is a subset of the *Generalized linear Models* and converts probability to log odds. (More details are given in 31.1 Chapter of https://rafalab.github.io/dsbook).

Accuracy was used to judge the performance of the prediction of a categorical variable. For the evaluation of the algorithm a **validation** data set was generated. The **sf_crime** data set was partitioned into **validation** and **sf_crime_p**. The **validation** data set was only used in the final step to test the final algorithm and contained only 20% of **sf_crime** data. The final model with the highest accuracy was chosen to be applied to the **sf_crime_p** data to calculate the parameters of the model. For the final step, this model was evaluated by calculating the accuracy of the **validation** set.

In order to train and test the algorithm, **sf_crime_p** was divided into a train and test set, where test_set contained only 20% of the **sf_crime_p** data.

Logistic regression with one predictor

Only numerical variables were used as predictors in the *Logistic Regression* model. First only one predictor *Resolution_dummy*, a dummy variable with values of 1 for resolved crime and value of 0 for non-resolved crime. The *glm*() function with specified $family = "binomial"$ is used to fit the *Logistic Regression* model.

[1] https://rafalab.github.io/dsbook/

```r
# Fit Logistic regression with one predictor
fit_glm <- glm(Category_Violent_dummy ~ Resolution_dummy,
               data=train_set, family = "binomial")

# Predict "Category_Violent_dummy"
p_hat_glm <- predict(fit_glm, test_set, type = "response")
```

The decision rule is to predict a category as violent if p_hat_glm > 0.15.

```r
# Set values to 1 (violent) if > 0.15 and otherwise 0 (non-violent)
y_hat_glm <- ifelse(p_hat_glm > 0.15, 1, 0)
```

```r
#   Set factors to the same factor level
y_hat_glm<-y_hat_glm %>%factor()
test_set$Category_Violent_dummy<-test_set$Category_Violent_dummy %>%factor()
```

Overall accuracy is low and is below the guessing rate. The proportion of violent crime is much lower than a non-violent crime.

```r
#  Show byClass to check sensitivity, specificity
confusionMatrix(y_hat_glm, test_set$Category_Violent_dummy)$byClass %>% knitr::kable()
```

	x
Sensitivity	0.4066178
Specificity	0.5255302
Pos Pred Value	0.7978581
Neg Pred Value	0.1612825
Precision	0.7978581
Recall	0.4066178
F1	0.5386962
Prevalence	0.8216087
Detection Rate	0.3340807
Detection Prevalence	0.4187220
Balanced Accuracy	0.4660740

```r
# Print results
accuracy_results %>% knitr::kable()
```

Model	Accuracy
Logistic regression with one predictor	0.4278307

The result shows the Accuracy is below the guessing rate. The next model will include the following predictors:

- *Resolution_dummy* shows if crime was resolved
- *Time_Hour* time of the day
- *Date_Day* day of the month
- *Date_Month* which month of the year

- *Count_Address_Occur* frequency of how often crime took place
- *X* and *Y* location of the crime

Logistic regression with more than one predictor

```r
fit_glm_mp <- glm(Category_Violent_dummy ~ Resolution_dummy + Time_Hour + Date_Day +
                  Date_Month + Count_Address_Occur + X + Y,
                  family = "binomial", data = train_set)

#   Predict "Category_Violent_dummy"
p_hat_glm_mp <- predict(fit_glm_mp, test_set, type = "response")

#  Set values to 1 (violent) if > 0.15 and otherwise 0 (non-violent)
y_hat_glm_mp <- ifelse(p_hat_glm_mp > 0.15, 1, 0)

#   Set factors to the same factor level
y_hat_glm_mp <- y_hat_glm_mp %>% factor()
test_set$Category_Violent_dummy <- test_set$Category_Violent_dummy %>% factor()
```

```r
#  Show byClass to check sensitivity, specificity
confusionMatrix(y_hat_glm_mp, test_set$Category_Violent_dummy)$byClass %>% knitr::kable()
```

	x
Sensitivity	0.6936722
Specificity	0.3479969
Pos Pred Value	0.8305085
Neg Pred Value	0.1978562
Precision	0.8305085
Recall	0.6936722
F1	0.7559480
Prevalence	0.8216087
Detection Rate	0.5699271
Detection Prevalence	0.6862388
Balanced Accuracy	0.5208345

```r
# Print results
accuracy_results %>% knitr::kable()
```

Model	Accuracy
Logistic regression with one predictor	0.4278307
Logistic regression with more than one predictor	0.6320067

Since the result has improved and is above the guessing rate, this model will be applied on the **validation** and **sf_crime_p** data.

Final Logistic regression with more than one predictor on validation and sf__crime__p

```
fit_glm_mp_v <- glm(Category_Violent_dummy ~ Resolution_dummy + Time_Hour + Date_Day +
                    Date_Month + Count_Address_Occur + X + Y,
                    family = "binomial", data=sf_crime_p)

# Predict "Category_Violent_dummy"
p_hat_glm_mp_v <- predict(fit_glm_mp_v, validation, type = "response")

# Set values to 1 (violent) if > 0.15 and otherwise 0 (non-violent)
y_hat_glm_mp_v <- ifelse(p_hat_glm_mp_v > 0.15, 1, 0)

# Set factors to the same factor level
y_hat_glm_mp_v <- y_hat_glm_mp_v %>% factor()
validation$Category_Violent_dummy <- validation$Category_Violent_dummy %>% factor()
```

```
# Show byClass to check the sensitivity, specificity
confusionMatrix(y_hat_glm_mp_v, validation$Category_Violent_dummy)$byClass %>% knitr::kable()
```

	x
Sensitivity	0.6908085
Specificity	0.3307735
Pos Pred Value	0.8247630
Neg Pred Value	0.1900410
Precision	0.8247630
Recall	0.6908085
F1	0.7518659
Prevalence	0.8201283
Detection Rate	0.5665516
Detection Prevalence	0.6869265
Balanced Accuracy	0.5107910

```
#   Print results
accuracy_results %>% knitr::kable()
```

Model	Accuracy
Logistic regression with one predictor	0.4278307
Logistic regression with more than one predictor	0.6320067
Final Logistic regression with more than one predictor	0.6260483

Principal component Analyses (PCA)

Another method will be applied to improve the results of the prediction. PCA is is one of ML methods to reduce high-dimensionality of the data set in case of many predictors. PCA works best with numerical data, all non-numerical variables are excluded from the data with predictors.

```r
train_set_pca <- train_set[ , c("Resolution_dummy",
                                "Count_Address_Occur", "Time_Hour",
                                "Date_Day", "Date_Month", "X", "Y") ]

test_set_pca <- test_set[ , c("Resolution_dummy",
                              "Count_Address_Occur", "Time_Hour",
                              "Date_Day", "Date_Month", "X", "Y") ]
```

This data is passed to the *prcomp*() function assigning the output to **sf_crime.pca**. *prcomp*() applies a linear orthogonal transformation of the passed data. Argument *center* = $TRUE$ means the columns are centered. Argument *scale* = $TRUE$ only makes only sense if variables are measured on the same scale, which is not the case. The center component corresponds to the means of the variables.

```r
# PCA method
sf_crime.pca <- prcomp(train_set_pca, center = TRUE)

# Mean value
sf_crime.pca$center
```

```
##    Resolution_dummy Count_Address_Occur           Time_Hour            Date_Day
##           0.2623695         156.2538649          13.3649325          15.7733587
##          Date_Month                   X                   Y
##           6.5438143        -122.4236047          37.7690801
```

PCA returns 3 components: $\$x$ the principal components, $\$rotation$ to transform the matrix and $\$sdev$ standard deviation.

```r
# PC's components
head(sf_crime.pca$x)
```

```
##               PC1        PC2         PC3        PC4        PC5          PC6
## [1,] -3404.74526  12.782785   8.9030226 -4.6618525  0.7055661  0.005376495
## [2,]   150.25337  -2.005883 -13.3883058  5.4981272 -0.2592466 -0.009617967
## [3,]   133.25441  10.774061  -0.1819448 -1.5232879 -0.2682553 -0.004706588
## [4,]   106.25319  -8.996115 -13.5267653  0.4827855  0.7146573 -0.008754000
## [5,]   106.25319  -8.996115 -13.5267653  0.4827855  0.7146573 -0.008754000
## [6,]    17.25342 -12.231881   0.4102094 -3.5596393 -0.2804770  0.062174028
##                PC7
## [1,]  0.0004978731
## [2,] -0.0319581409
## [3,] -0.0157827503
## [4,] -0.0037113332
## [5,] -0.0037113332
## [6,] -0.0059572162
```

```r
# Standard Deviation of the components
sf_crime.pca$sdev
```

```
## [1] 559.87476439   8.86995981   6.55556115   3.47250793   0.43940011
## [6]   0.02702924   0.02214819
```

```r
# Rotation parameter
head(sf_crime.pca$rotation)
```

```
##                                PC1            PC2            PC3
## Resolution_dummy    -0.000007874970 -0.00009573964 -0.00187540524
## Count_Address_Occur -0.999999998241  0.00004610396  0.00001746913
## Time_Hour            0.000016732187 -0.01755944098  0.99983984256
## Date_Day            -0.000046361886 -0.99984490358 -0.01756306133
## Date_Month          -0.000031145401 -0.00134984600  0.00288186443
## X                   -0.000006857514 -0.00002837212 -0.00006669869
##                               PC4             PC5             PC6
## Resolution_dummy    -0.00497221511  0.999978236373  0.001860221743
## Count_Address_Occur -0.00003109455 -0.000008008742  0.000007536878
## Time_Hour           -0.00291440712  0.001859233550 -0.000037143424
## Date_Day            -0.00129837520 -0.000135167636  0.000037724568
## Date_Month           0.99998254747  0.004977363347  0.000045564834
## X                    0.00003224978  0.003337217386 -0.864218259130
##                                 PC7
## Resolution_dummy    -0.0034375949208
## Count_Address_Occur -0.0000006237334
## Time_Hour            0.0000556129813
## Date_Day             0.0000070675901
## Date_Month          -0.0000185410167
## X                    0.5031060099274
```

7 principal components were obtained. Each PC explains a percentage of the total variation in the data. PC1, PC2 and PC3 explain almost 100% of the total variance.

```r
# Summary of the output
summary(sf_crime.pca)
```

```
## Importance of components:
##                             PC1     PC2     PC3     PC4    PC5     PC6     PC7
## Standard deviation     559.8748 8.86996 6.55556 3.47251 0.4394 0.02703 0.02215
## Proportion of Variance   0.9996 0.00025 0.00014 0.00004 0.0000 0.00000 0.00000
## Cumulative Proportion    0.9996 0.99982 0.99996 1.00000 1.0000 1.00000 1.00000
```

```r
summary(sf_crime.pca)$importance
```

```
##                              PC1     PC2      PC3      PC4       PC5        PC6
## Standard deviation     559.87476 8.86996 6.555561 3.472508 0.4394001 0.02702924
## Proportion of Variance   0.99957 0.00025 0.000140 0.000040 0.0000000 0.00000000
## Cumulative Proportion    0.99957 0.99982 0.999960 1.000000 1.0000000 1.00000000
##                               PC7
## Standard deviation     0.02214819
## Proportion of Variance 0.00000000
## Cumulative Proportion  1.00000000
```

```r
# Calculate the Variance
sf_crime.pca.var <- sf_crime.pca$sdev^2
sf_crime.pca$sdev^2
```

```
## [1] 313459.7518048031      78.6761869609      42.9753820024      12.0583113410
## [5]      0.1930724541       0.0007305796       0.0004905423
```

The percentage of variation of each PC is calculated as:

```r
# Percentage of Variation
sf_crime.pca.var.per <- sf_crime.pca.var/sum(sf_crime.pca.var) * 100
sf_crime.pca.var.per
```

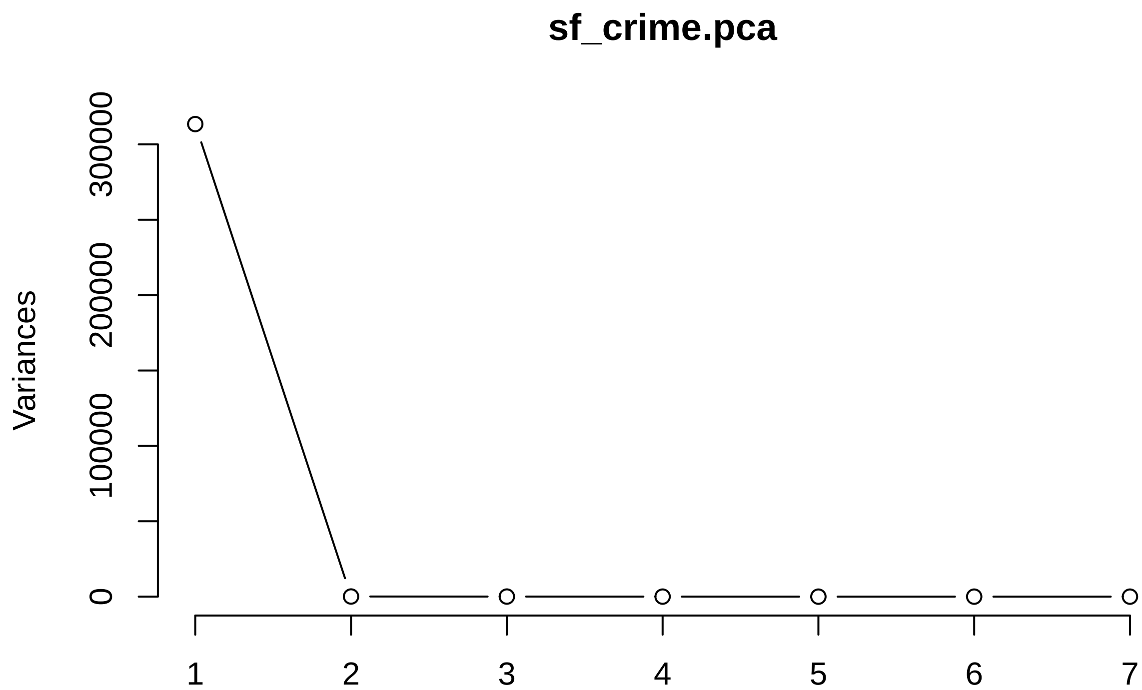
```
## [1] 99.9573001011572  0.0250885773551  0.0137041617976  0.0038452025770
## [5]  0.0000615677168  0.0000002329702  0.0000001564261
```

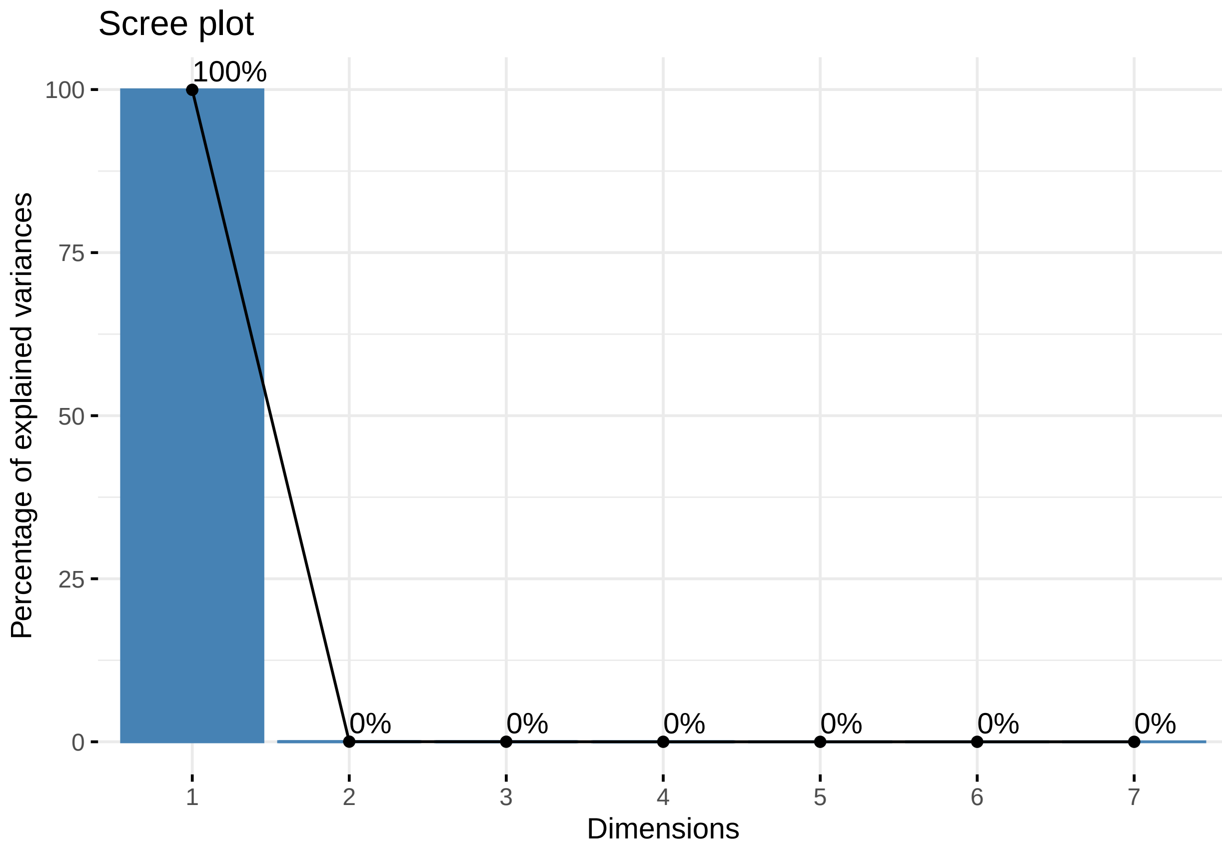
Here we can see what variables have the largest effect:

```r
sf_crime.pca.rotation <- sf_crime.pca$rotation[ , 1]
sort(abs(sf_crime.pca.rotation) , decreasing = TRUE) %>% knitr::kable()
```

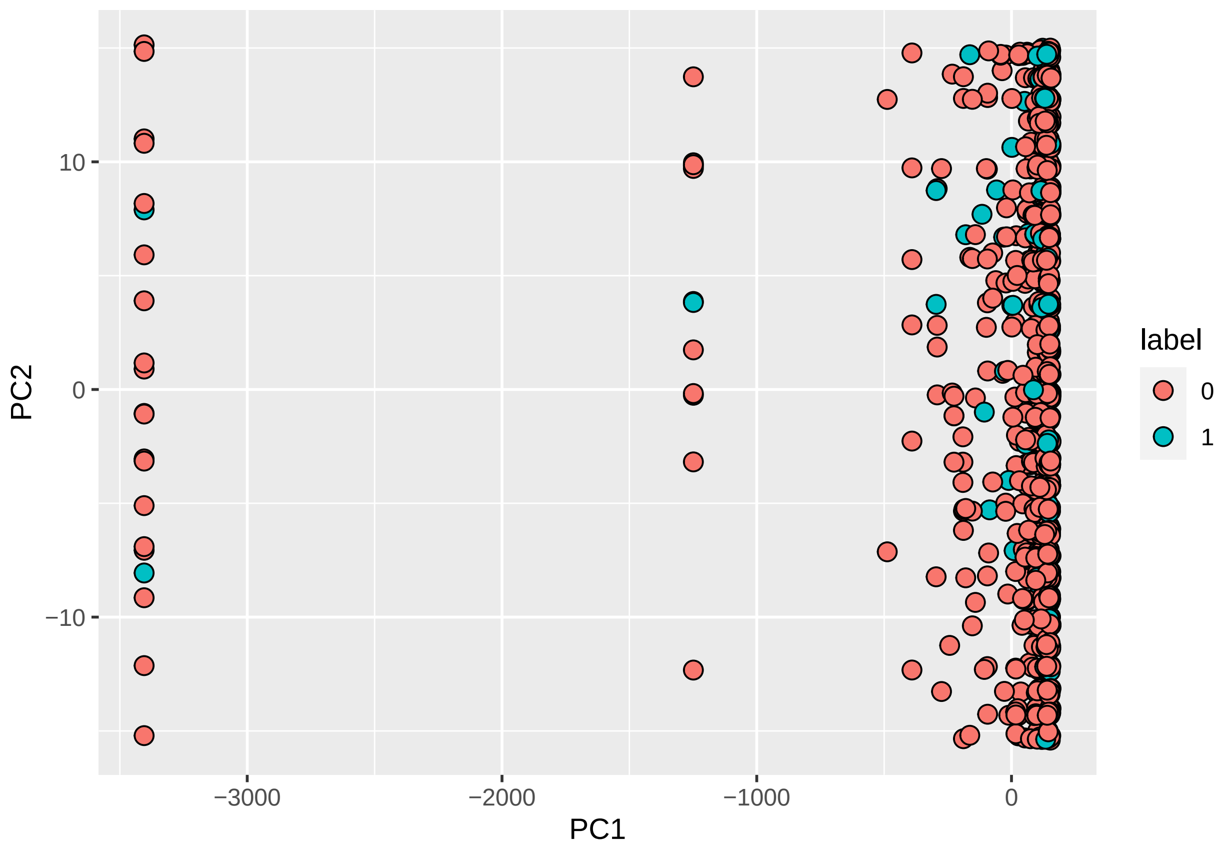
	x
Count_Address_Occur	1.0000000
Date_Day	0.0000464
Date_Month	0.0000311
Time_Hour	0.0000167
Resolution_dummy	0.0000079
X	0.0000069
Y	0.0000032

Visualization of the Variation





The first 2 components account for the largest amount of variability. This plot shows how close are the observations to each other if they build clusters:



In the next step a transformation will be applied to train and test data. The Knn method will be fit on the reduced data set with relevant predictors.

```
# Preparation for model fitting: calculate means of Columns
col_means<- colMeans(test_set_pca)
#  Set x_train equal to PC's
x_train <- sf_crime.pca$x[ ,1:3]
y <- factor(train_set$Category_Violent_dummy)
#  Fit knn model
fit <- knn3(x_train, y)
#  Transform the test_set
x_test <- as.matrix(sweep(test_set_pca, 2, col_means)) %*% sf_crime.pca$rotation
x_test <- x_test[ ,1:3]
#  Predict Category
y_hat <- predict(fit, x_test, type = "class")
# Print Accuracy of the model
confusionMatrix(y_hat,
                factor(test_set$Category_Violent_dummy))$overall["Accuracy"] %>%
                 knitr::kable("pipe")
```

	x
Accuracy	0.7987668

Final PCA on validation and sf_crime_p

Since the Accuracy improved by using the PCA, the same procedure will be applied to the **validation** and **sf_crime_p**.

```
#  Keep only numerical variables in sf_crime_p_pca data set
sf_crime_p_pca <- sf_crime_p[ , c( "Resolution_dummy",
                                   "Count_Address_Occur", "Time_Hour","Date_Day",
                                   "Date_Month", "X", "Y") ]

#  Keep only numerical variables in validation_pca data set
validation_pca <- validation[ , c( "Resolution_dummy",
                                   "Count_Address_Occur", "Time_Hour","Date_Day",
                                   "Date_Month", "X", "Y") ]

#  Perform PCA
sf_crime.pca <- prcomp(sf_crime_p_pca, center = TRUE)

# Preparation for model fitting: calculate means of columns
col_means<- colMeans(validation_pca)

#  Set x_train equal to PC's
x_train <- sf_crime.pca$x[ ,1:3]

y <- factor(sf_crime_p$Category_Violent_dummy)

#  Fit knn with k = 5 model
fit <- knn3(x_train, y)

#  Transform the test_set
x_test <- as.matrix(sweep(validation_pca, 2, col_means)) %*% sf_crime.pca$rotation
x_test <- x_test[ ,1:3]

#  Predict Category
y_hat <- predict(fit, x_test, type = "class")

# Print Accuracy of the model on Validation data
confusionMatrix(y_hat,
                factor(validation$Category_Violent_dummy))$overall["Accuracy"] %>%
                knitr::kable("pipe")
```

	x
Accuracy	0.7943759

Results and Disscussion

Due to the computational complexity a small data set was chosen, which led to limitations in the choice of models. The goal of the analysis was to predict the crime category. There are 39 different crime categories listed. Analysis applied to predict variable *Category* yielded a very low Accuracy. Some of the 39 Categories don't include enough observations or information to build a reliable prediction. Therefore the variable *Category* was transformed into a dummy variable. *Category_Violent_dummy* was assigned value 1 when

“ASSAULT”, “ROBBERY”, “SEX OFFENSES, FORCIBLE”, “KIDNAPPING” and value 0 otherwise. All performed methods first were applied on train and test and additionally on the **validation** and **sf_crime_p** data. First *Logistic Regression* with one predictor was applied with $Accuracy = 0.427$ which is below the guessing rate. By extending the number of predictors by 7 the the Accuracy increased to 63%. This model was fit to the **validation** and **sf_crime_p** data and showed $Accuracy = 0.626$.

In order to improve the results and to reduce the number of predictors *Principal Component Analysis* (PCA) was used. First the method was applied to the train data with 7 predictors such as: *Resolution_dummy*, *Count_Address_Occur*, *Time_Hour*, *Date_Day*, *Date_Month*, *X* and *Y* for location. PCA showed that *Count_Address_Occur*, *Date_Day* and *Date_Month* have the largest effect and that first 3 PC's explain almost 100% of the variation in the data. Therefore only first 3 PC's were taken into account to predict *Category_Violent_dummy*. The obtained result was $Accuracy = 0.794$. Results obtained from PCA yielded higher *Accuracy* with less predictors than results obtained from *Logistic Regression*.

The difficulty of having memory capacity limitations led to the reduction of the analysis techniques.