Crime Classification Report

Yaxin Yu, Daniel Minsu Kim

December 12, 2015

Introduction and Problems of Interest

Back in early and mid 20th century, San Francisco was known for numerous crime occurrences and notorious criminals. Nowadays, although its name is often associated with prosperity in techonology, abundance of crime instances remains as an issue due to rising wealth disparity, housing shortages, and so on.

Intending to better understand this issue, our team would like to investigate the patterns in criminal records from the past 12 years. More specifically, we place our focus on the lawful categories of past crime instances and aim to find correlations between crime category and other factors such as time and location.

Here are the specific questions we are interested in aswering: 1. What are the indicative predictors of crime category? 3. How characteristic are they? How much should each of the predictors weigh? 4. How is crime category correlated with various predictors? 5. How to utilize visualization to explore such correlations?

Data Description

Primary data set:

Past criminal records from 1/1/2003 to 5/13/2015 provided by SF OpenData

We downloaded this dataset from SF OpenData website with R. It provides information about over 870,000 crime instances. It includes eight columns – dates, category, detailed description of the crime (Descript), day of the week, police department district (PdDistrict), resolution, street address, longitude (X), and latitude (Y).

```r
head(crime, 5)
```

```
##   IncidntNum                   Category
## 1   50436712                    ASSAULT
## 2   80049078              LARCENY/THEFT
## 3  130366639                    ASSAULT
## 4   30810835 DRIVING UNDER THE INFLUENCE
## 5  130839567             OTHER OFFENSES
##                                   Descript DayOfWeek       Date
## 1                                  BATTERY Wednesday 2005-04-20
## 2                GRAND THEFT FROM A BUILDING    Sunday 2008-01-13
## 3              AGGRAVATED ASSAULT WITH A KNIFE    Sunday 2013-05-05
## 4 DRIVING WHILE UNDER THE INFLUENCE OF ALCOHOL   Tuesday 2003-07-08
## 5                   TRAFFIC VIOLATION ARREST    Friday 2013-10-04
##         Time PdDistrict     Resolution                     Address
## 1   4H 0M 0S    MISSION           NONE         18TH ST / CASTRO ST
## 2  18H 0M 0S       PARK           NONE    1100 Block of CLAYTON ST
## 3  4H 10M 0S  INGLESIDE ARREST, BOOKED 0 Block of SGTJOHNVYOUNG LN
## 4   1H 0M 0S   SOUTHERN ARREST, BOOKED          MASON ST / TURK ST
## 5 20H 53M 0S TENDERLOIN ARREST, BOOKED    TURK ST / LEAVENWORTH ST
##           X        Y                              Location distance hour
## 1 -122.4350 37.76089 (37.7608878061245, -122.435002864271) 647.4388    4
## 2 -122.4468 37.76226 (37.7622550270122, -122.446837820235) 647.4388   18
## 3 -122.4447 37.72493 (37.7249307267936, -122.444707063455) 451.0388    4
## 4 -122.4090 37.78329 (37.7832878735491, -122.408953598286) 476.8488    1
## 5 -122.4141 37.78279 (37.7827931071006, -122.414056291891) 805.6124   20
##   year month                   street
## 1 2005     4      18TH ST / CASTRO ST
## 2 2008     1               CLAYTON ST
## 3 2013     5          SGTJOHNVYOUNG LN
## 4 2003     7        MASON ST / TURK ST
## 5 2013    10 TURK ST / LEAVENWORTH ST
```

```
summary(crime)
```

```
##    IncidntNum                  Category
##  Min.   :     3979   LARCENY/THEFT :173713
##  1st Qu.: 51390717   OTHER OFFENSES:125852
##  Median : 90053912   NON-CRIMINAL  : 92089
##  Mean   : 86809964   ASSAULT       : 76927
##  3rd Qu.:120243536   DRUG/NARCOTIC : 54630
##  Max.   :991582377   VEHICLE THEFT : 54072
##                      (Other)       :300766
##                                    Descript          DayOfWeek
##  GRAND THEFT FROM LOCKED AUTO          : 58936   Friday   :133852
##  LOST PROPERTY                         : 32099   Monday   :121297
##  BATTERY                               : 27611   Saturday :126110
##  STOLEN AUTOMOBILE                     : 26850   Sunday   :116178
##  DRIVERS LICENSE, SUSPENDED OR REVOKED: 26608   Thursday :125609
##  WARRANT ARREST                        : 23843   Tuesday  :125611
##  (Other)                               :682102   Wednesday:129392
##       Date                              Time
##  Min.   :2003-01-01 00:00:00   Min.   :1M 0S
##  1st Qu.:2005-12-04 00:00:00   1st Qu.:9H 30M 0S
##  Median :2009-01-06 00:00:00   Median :14H 44M 0S
##  Mean   :2009-01-15 01:15:55   Mean   :13H 44M 7.00122658303007S
##  3rd Qu.:2012-03-09 00:00:00   3rd Qu.:19H 0M 0S
##  Max.   :2015-11-20 00:00:00   Max.   :23H 59M 0S
##
##       PdDistrict                 Resolution       Address
##  SOUTHERN  :156653   NONE             :524733   Length:878049
##  MISSION   :120567   ARREST, BOOKED   :205776   Class :character
##  NORTHERN  :105351   ARREST, CITED    : 78554   Mode  :character
##  BAYVIEW   : 89049   LOCATED          : 17556
##  CENTRAL   : 84997   PSYCHOPATHIC CASE: 14829
##  TENDERLOIN: 82083   UNFOUNDED        :  9589
##  (Other)   :239349   (Other)          : 27012
##        X                 Y            Location            distance
##  Min.   :-122.5   Min.   :37.71   Length:878049      Min.   :   4.088
##  1st Qu.:-122.4   1st Qu.:37.75   Class :character   1st Qu.: 272.651
##  Median :-122.4   Median :37.78   Mode  :character   Median : 529.917
##  Mean   :-122.4   Mean   :37.77                      Mean   : 595.597
##  3rd Qu.:-122.4   3rd Qu.:37.78                      3rd Qu.:1083.770
##  Max.   :-120.5   Max.   :90.00                      Max.   :1119.328
##
```

```
##       hour             year            month           street
##  18     : 54908   2013   : 78179   1      : 80757   Length:878049
##  17     : 53710   2014   : 76667   10     : 76862   Class :character
##  12     : 52237   2004   : 75899   8      : 75605   Mode  :character
##  16     : 49793   2003   : 75854   3      : 75027
##  19     : 49346   2005   : 72694   9      : 74502
##  15     : 47866   2008   : 72622   5      : 73534
##  (Other):570189   (Other):426134   (Other):421762
```

Secondary data set:

Location of Starbucks shops in San Francisco

We retrieved San Francisco Starbucks addresses from City-Data.com. Using R, we parsed the originla html source code of all seven pages and looked for nodes containing the addresses of Starbucks shops. Then we obtained their values, put the addresses in a data frame, and saved the addresses as a csv file.

```r
rawStarAdd <- read.csv("rawdata/StarbucksAdd.csv")
head(rawStarAdd, 5)
```

```
##                      address
## 1           4094 18th Street
## 2           3995 24th Street
## 3         2018 Market Street
## 4            2020 Market St.
## 5 5290 Diamond Heights Blvd
```

Data Cleaning and Preprocessing

—Crime Data— Many of the columns such as resolution and category in our crime data frame are qualitative, so we factorized them for model building purposes later. In addition, the original date column was too specific, thus it is impossible to generate pattern from it. Therefore, we separated the column into year, month, and hour and added these as additional columns to the crime data frame. For similar reasons, we also extracted street name from the address column. For detailed examples, see below:

Here are the newly extracted columns:

—Starbucks Address Data—

Using built-in functions such as geocode in R ggmap package and setting google as the source, we searched for the longitude and latitude of each Starbucks shop based on its street address. Since the raw Starbucks address dataset contains only street addresses, it occasionally leads google to finding longitudes and latitudes with the same street address but are very far away from San Francisco. In fact, 20 out of the 79 pairs of longitude and latitude data were outside of the bay area.

Therefore, after retrieving longitudes and latitudes for the first time, we added "San Francisco" to each of addresses of those 20 Starbucks shops and retrieved longitudes and latitudes again. This tactic turned out to be effective and guaranteed accurate geographic information. Now the Starbucks addresses, longitudes, and latitudes are ready for analysis.

Again, here are the resulting columns:

Analysis Approach

— Feature Engineering — As mentioned earlier, we extracted more factors out of existing columns for our crime and Starbucks address data frames.

— Crime Type Labeling — The original dataset has 39 categories, so it would be computationaly infeasible. Also, the original dataset is oversized, thus we decided to sample 0.1% of the original dataset and placed the crime instances in three broader groups – blue-collar crime, white-collar crime, and other crime – and labeled each crime with “blue,” “white,” and “other” accordingly.

— Incorporating Starbucks Location Data — We hypothesized that the closer it is from Starbucks, the lower the crime rate would be. Since Starbucks are oftentimes located in crowded areas, we anticipated more white-collar crime instances than blue-collar crime around Starbucks shops. To quantify this hypothesis, we calculated the distance from the closest Starbucks for each crime spot.

— Visualization — Since the subject of our analysis was in a particular geographical region and location was one of the predictors we intended to consider, we explored the dataset visually. We downlaoded a map of San Francisco and graphed the crime instances as dots along with the map in order to visualize the distribution.

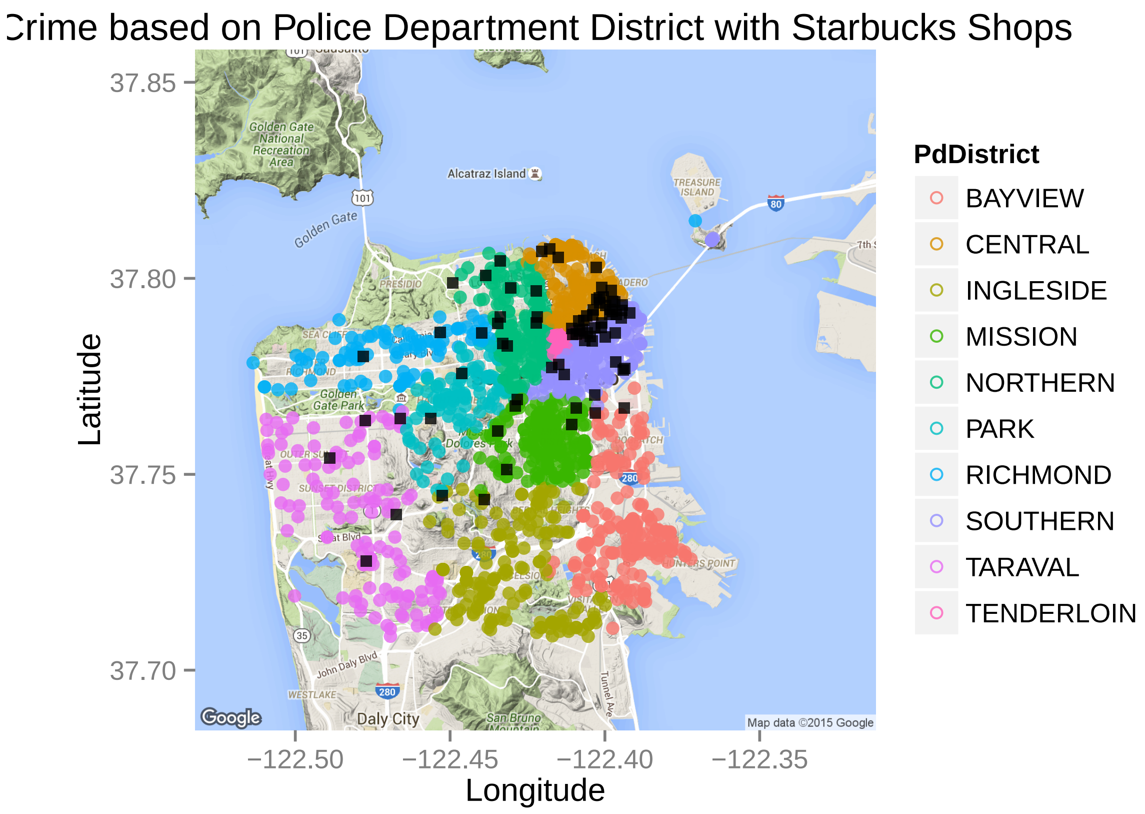
Furthermore, we graphed the density of each of the three groups of crime on the same map to find out the relationship between type of crime and their geographical distribution.

— Modeling — As this project concentrates on a classification problem, we incorporated classification tree models. For more intuitive results, we applied ctree and rpart to see the split points. For more accurate prediciton, we applied bagging and ensemble methods such as Gradient Boosting Machine and Random Forest algorithms.

Descriptions of Tables, Images, and Results

As a simple example, here is a map with first 2000 crime instances as well as all Starbucks shops plotted. The colored dots are crime by police department districts, and the black dots are Starbucks shops.

```
crimeStarbucksMapNoCirclesFromTo(1, 2000)
```



— Distribution of Three Groups of Crime — After dividing the crime instances into blue-collar, white-collar, and other groups, we graphed the crime density on our map for each group.

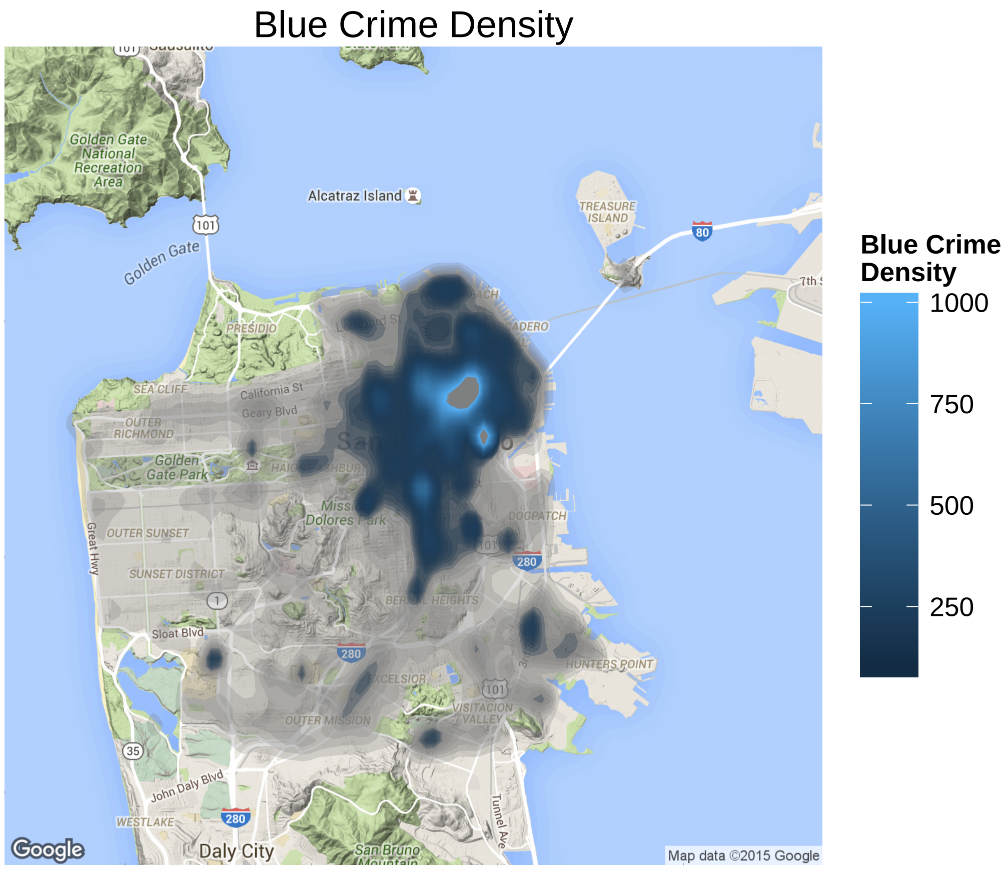
```
## There are 399572 blue-collar crime.
```

```
## There are 399572 blue-collar crime.
```

```
## There are 399572 blue-collar crime.
```

```r
graphBlueCrimeMap()
```

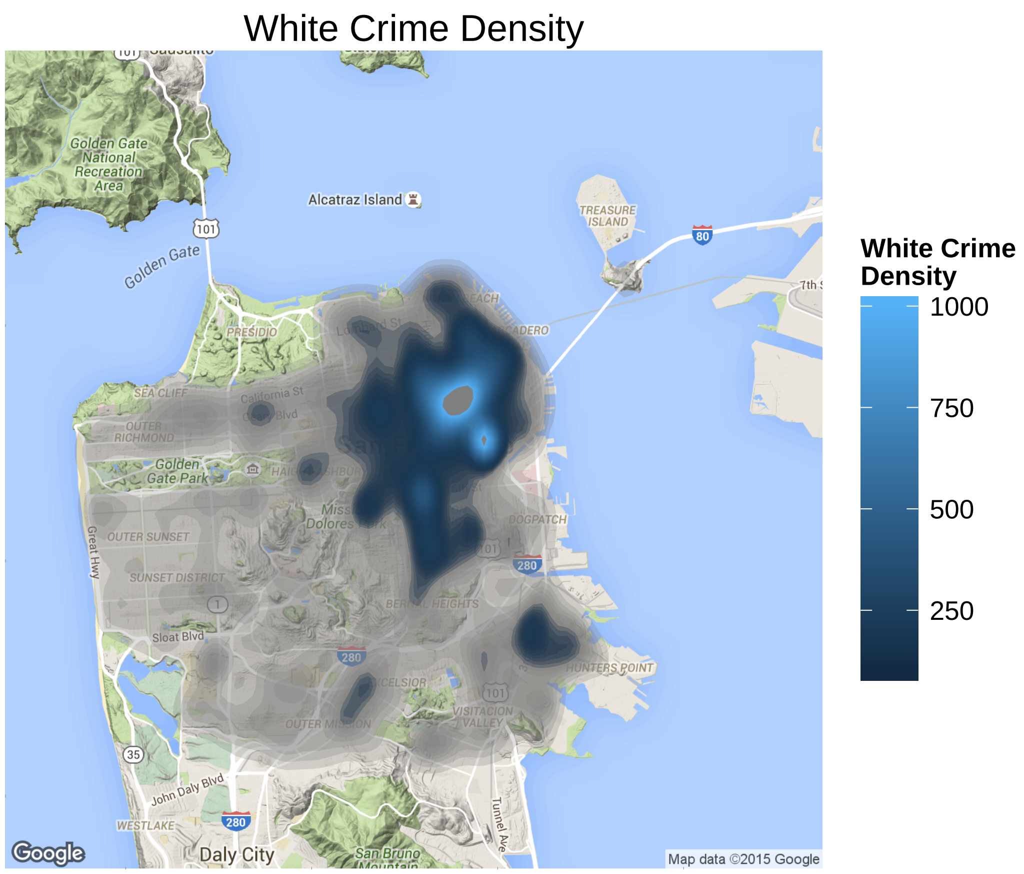
```
## Warning: Removed 34 rows containing non-finite values (stat_density2d).
```



As shown by the map, blue-collar crime concentrated extensively in the northeast corner of San Francisco, with the highest crime density in the center of that region. Meanwhile, small clusters of crime occured far way from the northeast.

```r
graphWhiteCrimeMap()
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density2d).
```

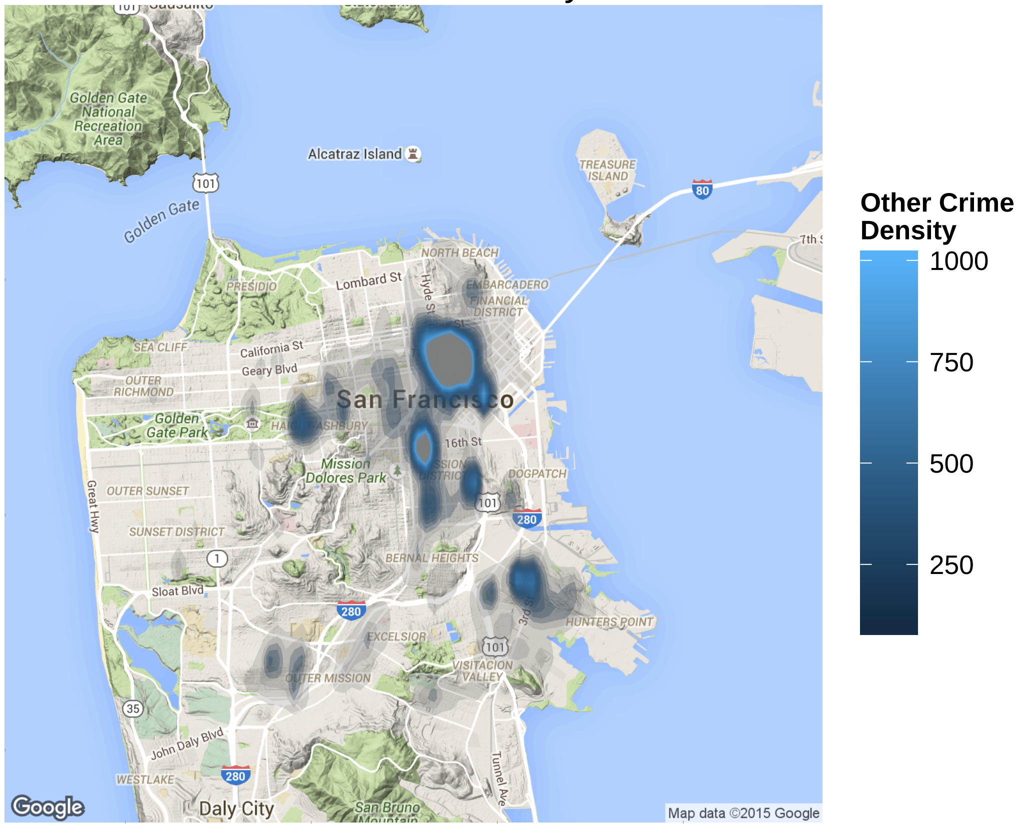


Contrasting to our intuition, white-collar crime occured in almost the same area as blue-collar crime. Moreover, they were even more concentrated in the northeast corner of the city, as the white-collar crime density map shows fewer small clusters on the edge than the blue-collar crime density map. Blue-collar crime are over 5 times more than white-collar crime, yet the density of white-collar crime around the northeast corner is almost the same as that of blue-collar crime. Even though they have extremely simlar distributions, blue-collar crime occur much more often than white-collar ones.

```r
graphOtherCrimeMap()
```

```
## Warning: Removed 5 rows containing non-finite values (stat_density2d).
```

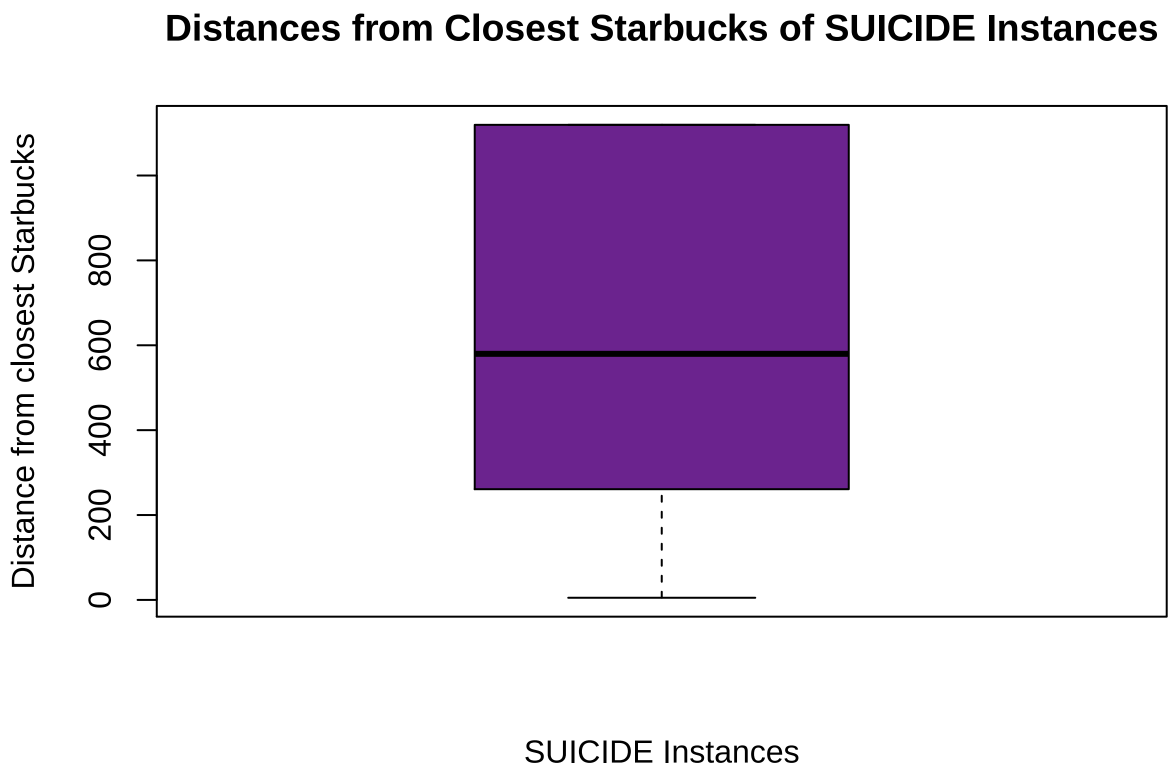
Other Crime Density



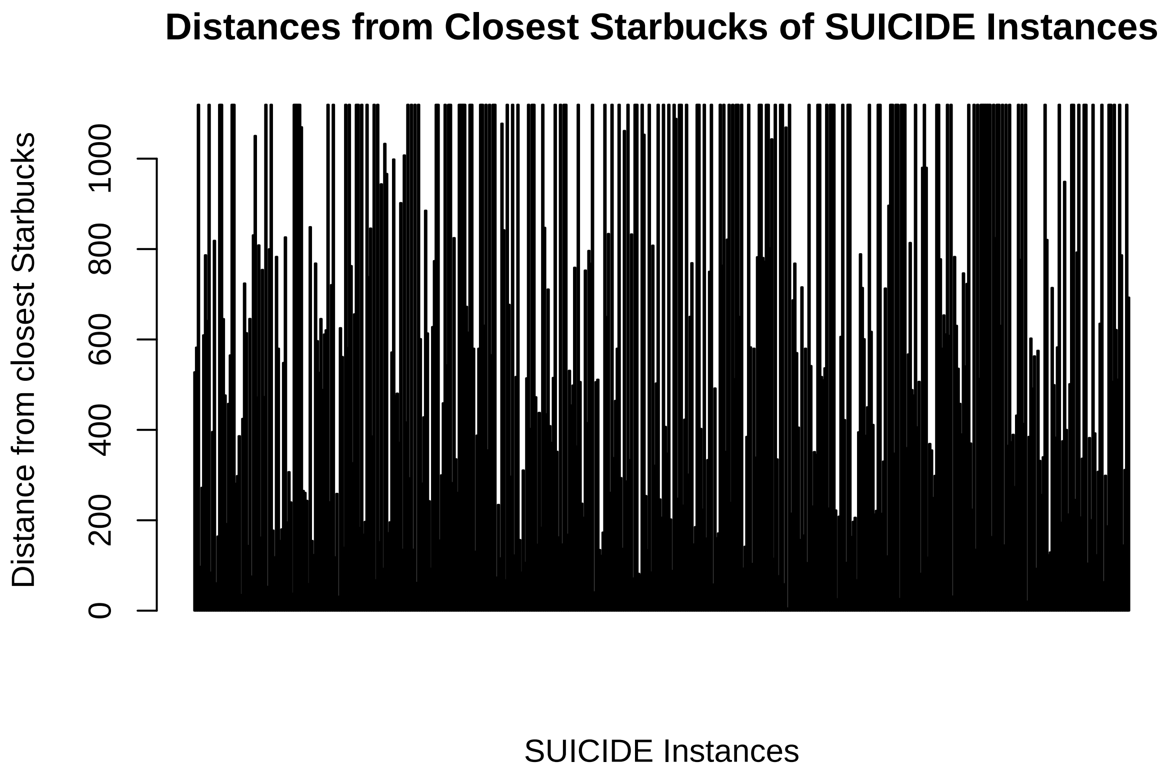
Instead of concentrating in a particular area, the rest of crime instances located around several different places in the city. However, similar to blue-collar and white-collar crime, very few occured in the west or south regions of San Francisco. One potential reason for this pattern is that criminals mostly commit crime around populated areas where most major companies, malls, and restaurants are located.

— Minimal Distances from Starbucks — After quantifying the relationship between crime and Starbucks locations by calculating the distances in between, we explored such relationship visually first by crime category and eventually of all crime instances.

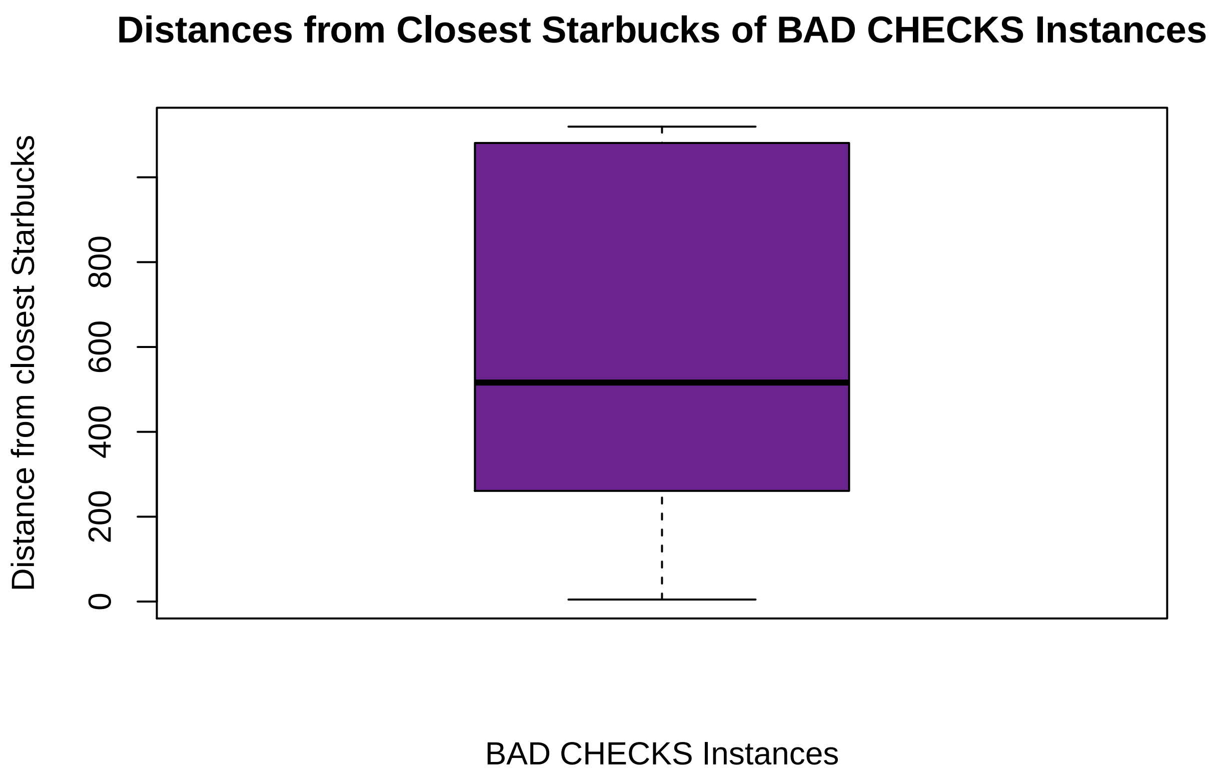
```
boxplotMinStarDistOf("SUICIDE")
```



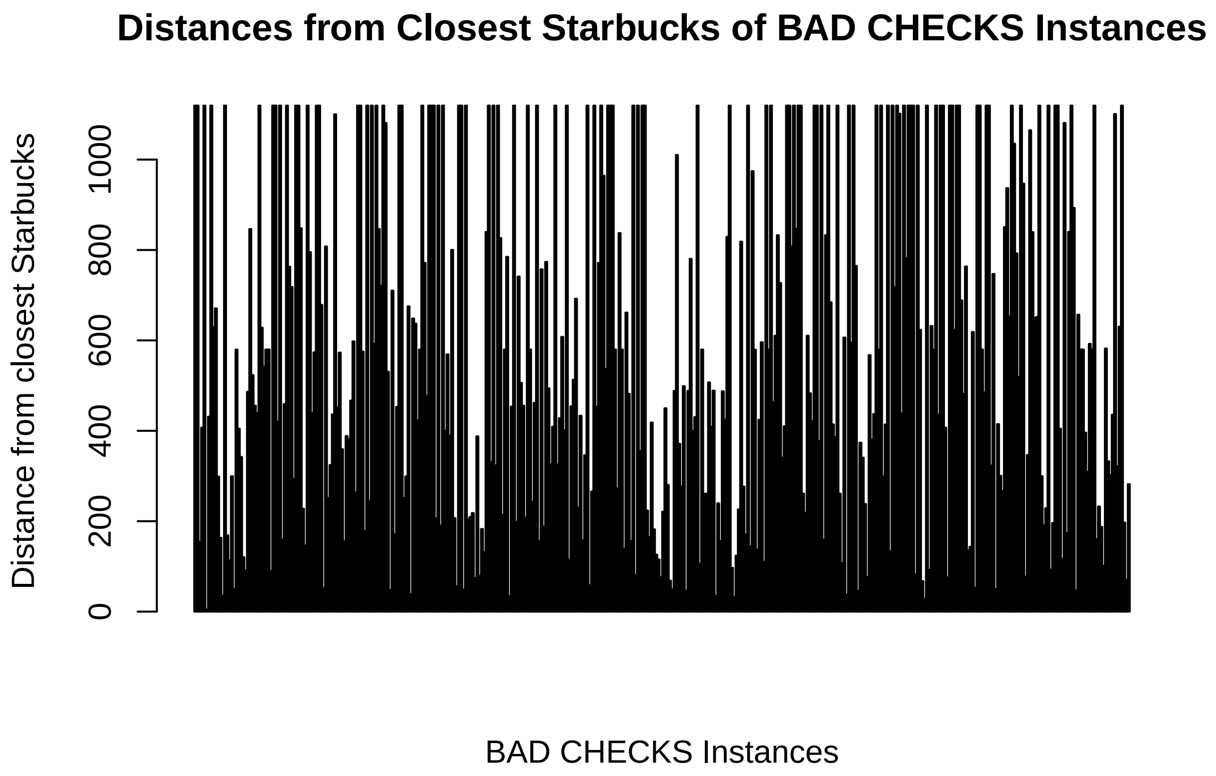
```
barplotMinStarDistOf("SUICIDE")
```



```r
boxplotMinStarDistOf("BAD CHECKS")
```



```r
barplotMinStarDistOf("BAD CHECKS")
```

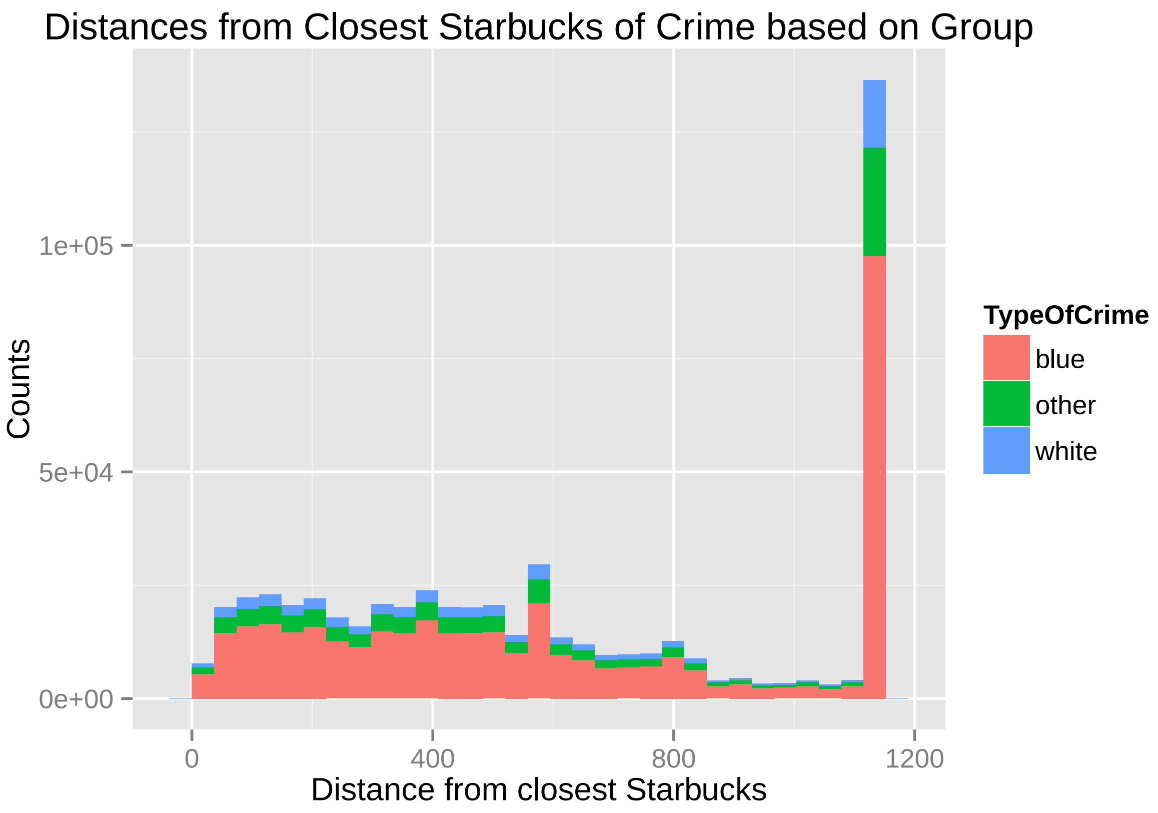


Above are plots of only several interesting categories. Due to limited length of this report, we will not demonstrate the correlation between crime and Starbucks locations of every category. However, one should be able to realize the differences of distribution amongst different crime categories. We also encourage readers to utilize our ploting functions in plotCrimeMinStarBucksDist.R to explore further.

Lastly, we used bar plot to visualize distances from closest Starbucks for all crime instances and colored the bars by crime group.

```
barplotMinStarDistByGroup()
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



As shown by the above graph, the crime rate increases significantly when the mininal distances from Starbucks exceeds roughly 1100. And the crime rate decreases as the distances increase in range 600-1000. This observation matches with the crime density graphs obtained previously: Starbucks shops are concentrated in the northeast corner of San Francisco, so were the crime instances. In other words, areas farther away from the northeast area of San Francisco have fewer Starbucks, and respectively lower crime rate.

Prediction Results

We used ctree and rpart to get the split points to understand general pattern of data. Because PdDistrict column contains more factors than other columns, both algorithms weighted more on PdDistrict column to start from. And then, we used other bagging and ensemble methods to calculate the accuracy of data. All algorithms incorporate a bootstrapping method internally to measure the prediction accuracy.

Followings are fitted algorithms. ctree, rpart, randomforset, supoort vector machine, gradient boosting machine respectively.

```
fitCtree
```

```
##
##   Conditional inference tree with 3 terminal nodes
##
## Response:  TypeOfCrime
## Inputs:  DayOfWeek, PdDistrict, distance, year, month, hour
## Number of observations:  449
##
## 1) PdDistrict == {BAYVIEW, MISSION, TENDERLOIN}; criterion = 0.998, statistic = 46.18
##   2)*  weights = 145
## 1) PdDistrict == {CENTRAL, INGLESIDE, NORTHERN, PARK, RICHMOND, SOUTHERN, TARAVAL}
##   3) DayOfWeek == {Friday, Monday, Saturday, Thursday}; criterion = 0.986, statistic = 52.347
##     4)*  weights = 186
##   3) DayOfWeek == {Sunday, Tuesday, Wednesday}
##     5)*  weights = 118
```

```
fitRpart
```

```
## n= 449
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 449 129 blue (0.71269488 0.17594655 0.11135857)
##     2) PdDistrict=CENTRAL,INGLESIDE,NORTHERN,PARK,RICHMOND,SOUTHERN,TARAVAL 304  60 blue (0.80263158
##       4) DayOfWeek=Friday,Monday,Saturday,Thursday 186  22 blue (0.88172043 0.08064516 0.03763441) *
##       5) DayOfWeek=Sunday,Tuesday,Wednesday 118  38 blue (0.67796610 0.15254237 0.16949153)
##        10) hour=6,7,11,12,13,16,17,21,22 42   4 blue (0.90476190 0.04761905 0.04761905) *
##        11) hour=0,1,2,3,8,9,10,14,15,18,19,20,23 76  34 blue (0.55263158 0.21052632 0.23684211)
##          22) month=3,4,5,7,9,10,11,12 56  21 blue (0.62500000 0.25000000 0.12500000)
##            44) PdDistrict=INGLESIDE,NORTHERN,PARK,RICHMOND 22   3 blue (0.86363636 0.09090909 0.0454
##            45) PdDistrict=CENTRAL,SOUTHERN,TARAVAL 34  18 blue (0.47058824 0.35294118 0.17647059)
##              90) year=2003,2004,2005,2009,2010,2011,2012,2014 27  11 blue (0.59259259 0.18518519 0.2
##               180) hour=0,1,10,14,18,19,20,23 20   6 blue (0.70000000 0.20000000 0.10000000) *
##               181) hour=8,9,15 7   3 white (0.28571429 0.14285714 0.57142857) *
##              91) year=2006,2007,2008,2013 7   0 other (0.00000000 1.00000000 0.00000000) *
##          23) month=1,2,6,8 20   9 white (0.35000000 0.10000000 0.55000000)
##            46) hour=1,2,8,9,19,20 9   3 blue (0.66666667 0.00000000 0.33333333) *
##            47) hour=0,10,14,15 11   3 white (0.09090909 0.18181818 0.72727273) *
##     3) PdDistrict=BAYVIEW,MISSION,TENDERLOIN 145  69 blue (0.52413793 0.31724138 0.15862069)
##       6) hour=0,1,2,3,4,6,8,9,12,13,16,18,19,21,22,23 94  35 blue (0.62765957 0.18085106 0.19148936)
##        12) year=2004,2005,2006,2011,2013,2014 53  11 blue (0.79245283 0.09433962 0.11320755) *
##        13) year=2003,2007,2008,2009,2010,2012 41  24 blue (0.41463415 0.29268293 0.29268293)
##          26) hour=0,1,2,3,4,6,8,13,16,19,21,22 30  14 blue (0.53333333 0.33333333 0.13333333)
##            52) month=1,3,5,6,10,12 15   6 other (0.40000000 0.60000000 0.00000000) *
##            53) month=4,7,8,9,11 15   5 blue (0.66666667 0.06666667 0.26666667) *
##          27) hour=9,12,18,23 11   3 white (0.09090909 0.18181818 0.72727273) *
##       7) hour=5,7,10,11,14,15,17,20 51  22 other (0.33333333 0.56862745 0.09803922)
##        14) month=2,3,5,8 15   6 blue (0.60000000 0.13333333 0.26666667) *
##        15) month=1,4,6,7,9,10,11,12 36   9 other (0.22222222 0.75000000 0.02777778) *
```

```
fitRandomForest
```

```
## Random Forest
##
## 449 samples
##  17 predictor
##   3 classes: 'blue', 'other', 'white'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 449, 449, 449, 449, 449, 449, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa       Accuracy SD  Kappa SD
##    2    0.7019067  0.00000000  0.03465027   0.00000000
##   32    0.6772965  0.06355075  0.02649796   0.04684593
##   62    0.6648652  0.05780726  0.03042530   0.04671468
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
fitSVM
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 449 samples
##  17 predictor
##   3 classes: 'blue', 'other', 'white'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 449, 449, 449, 449, 449, 449, ...
##
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa          Accuracy SD  Kappa SD
##   0.25  0.7104395  -0.0004069224  0.02682581   0.002034612
##   0.50  0.7065725   0.0057057269  0.02939690   0.021224653
##   1.00  0.6962353   0.0330718502  0.03230672   0.039672375
##
## Tuning parameter 'sigma' was held constant at a value of 0.008846004
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.008846004 and C = 0.25.
```

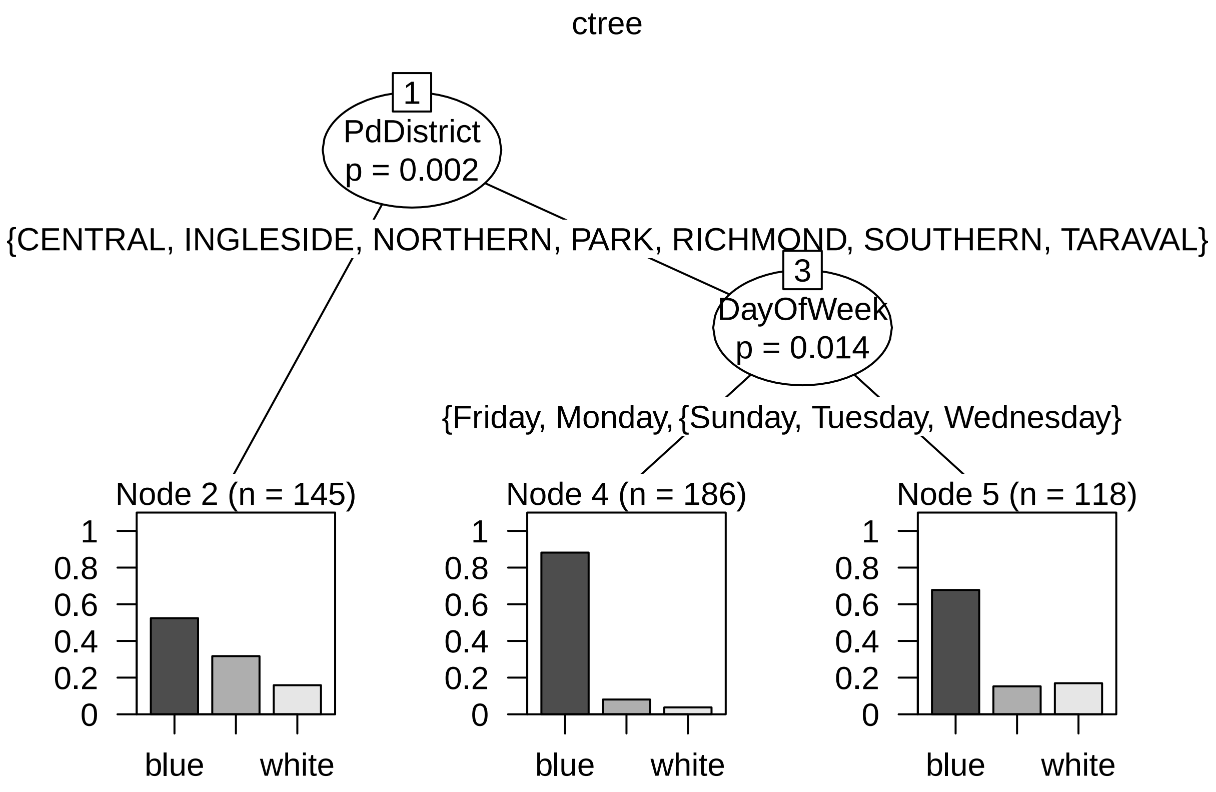
```
fitGbm
```

```
## Stochastic Gradient Boosting
##
## 449 samples
##  17 predictor
##   3 classes: 'blue', 'other', 'white'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 449, 449, 449, 449, 449, 449, ...
##
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa       Accuracy SD
##   1                   50      0.7001798  0.05959706  0.03374835
##   1                  100      0.6816146  0.08353381  0.03604250
##   1                  150      0.6751325  0.09786226  0.03663254
##   2                   50      0.6890728  0.07036164  0.03081483
##   2                  100      0.6686084  0.08638362  0.03712655
##   2                  150      0.6618606  0.10032299  0.03750860
##   3                   50      0.6767206  0.07651151  0.03014703
##   3                  100      0.6608077  0.09325344  0.04155839
##   3                  150      0.6501307  0.09025318  0.04276356
##   Kappa SD
##   0.04665435
##   0.04494299
##   0.05190831
##   0.04294838
##   0.05738592
##   0.05654775
##   0.05316715
##   0.05730304
##   0.06565241
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 50, interaction.depth
##  = 1, shrinkage = 0.1 and n.minobsinnode = 10.
```

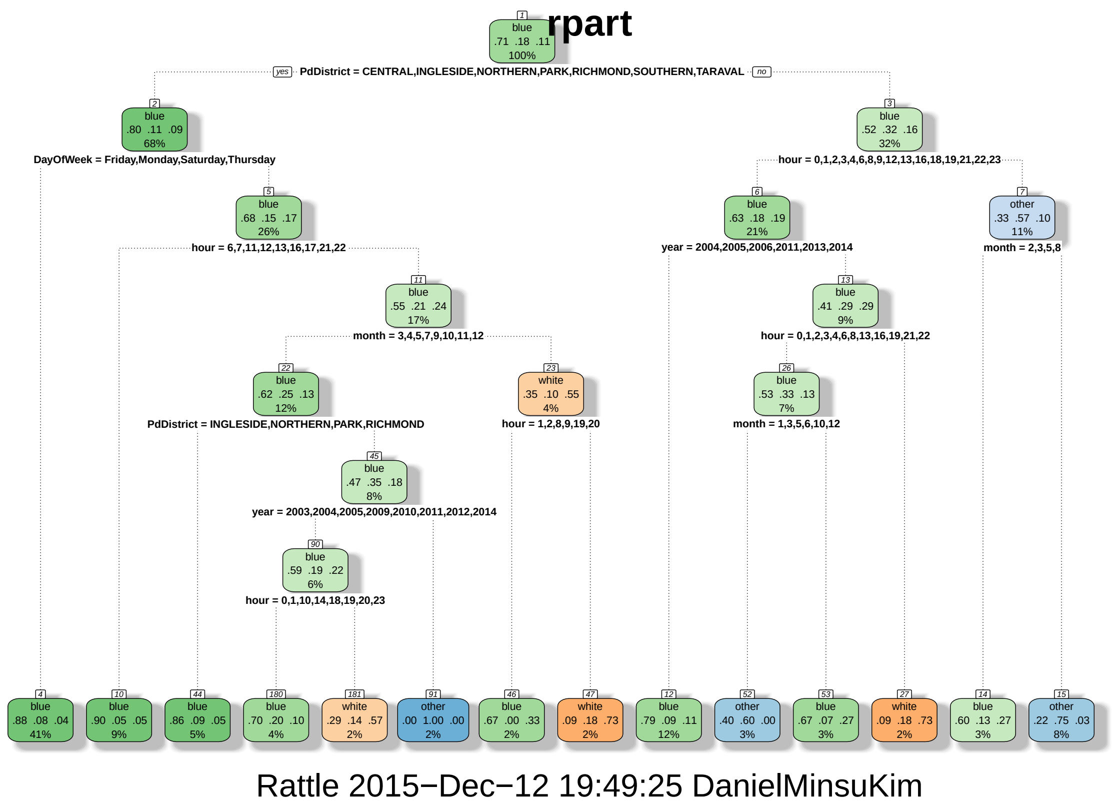
We plotted trained ctree and rpart algorithms to see how each algorithm generated decision branches, and plotted the results of SVM, Random Forest, and GBM models to find out the optimal tuning parameters.

Followings are the results of each predictive model. ctree, rpart, randomforest, svm, gbm. As the plot generated by ctree shows that PdDistrict is a considerably important factor to classify the types of crime. The plot generated by rpart also classified data first with the PdDstrict column. As we move down from the root to the leaves, we can how each entry is divided into each category based on thresholds denoted by nodes. Plots generated by Random Forest, SVM, GBM show the optimal tuning parameters estimated by boostrapping method.

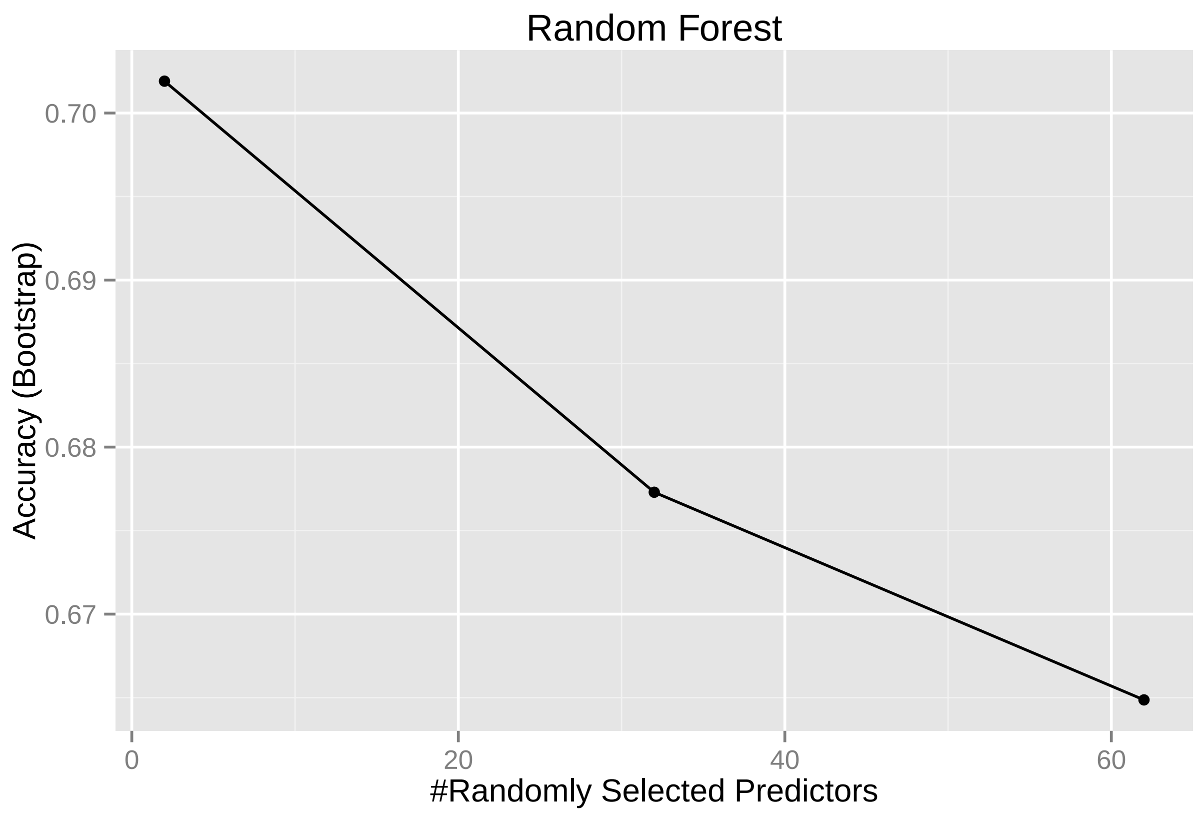
```r
plot(fitCtree, main = "ctree")
```



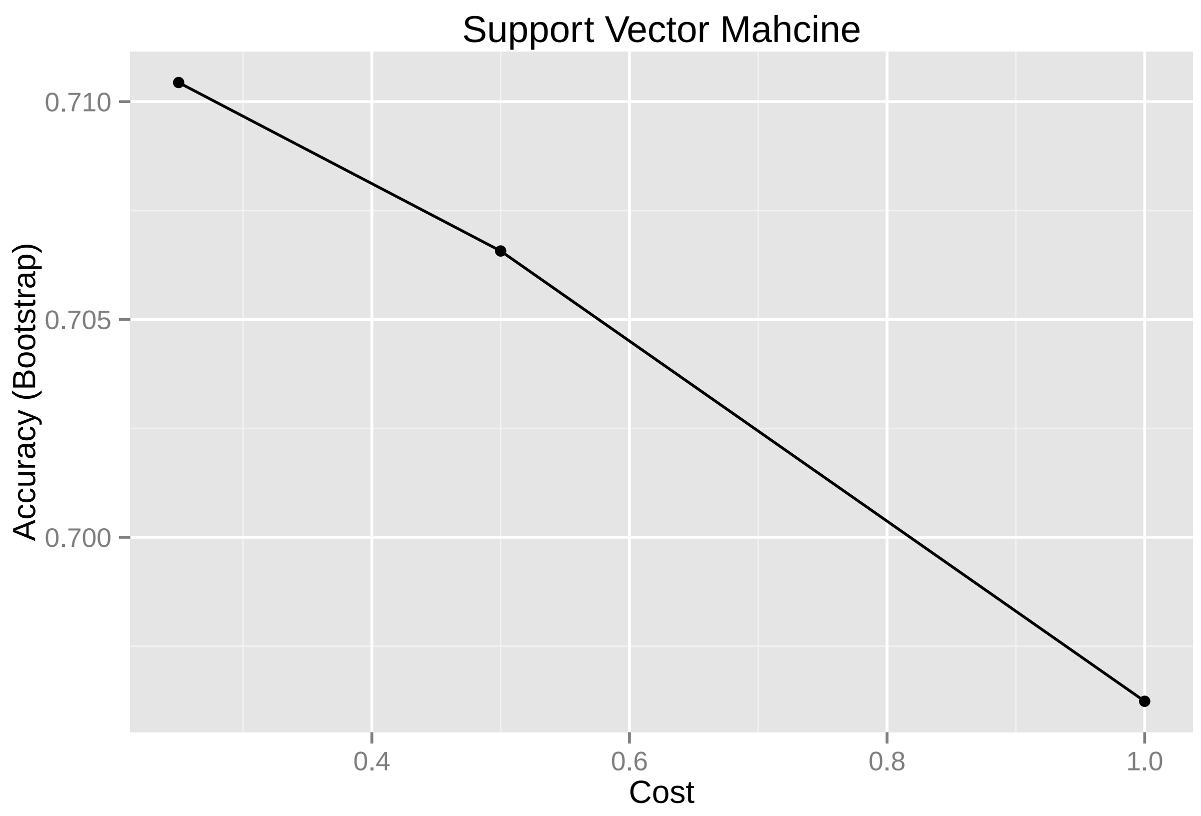
```
fancyRpartPlot(fitRpart, main = "rpart")
```


rpart
PdDistrict = CENTRAL,INGLESIDE,NORTHERN,PARK,RICHMOND,SOUTHERN,TARAVAL
yes
no
DayOfWeek = Friday,Monday,Saturday,Thursday
hour = 0,1,2,3,4,6,8,9,12,13,16,18,19,21,22,23
hour = 6,7,11,12,13,16,17,21,22
year = 2004,2005,2006,2011,2013,2014
month = 2,3,5,8
month = 3,4,5,7,9,10,11,12
hour = 0,1,2,3,4,6,8,13,16,19,21,22
PdDistrict = INGLESIDE,NORTHERN,PARK,RICHMOND
hour = 1,2,8,9,19,20
month = 1,3,5,6,10,12
year = 2003,2004,2005,2009,2010,2011,2012,2014
hour = 0,1,10,14,18,19,20,23
Rattle 2015−Dec−12 19:49:25 DanielMinsuKim

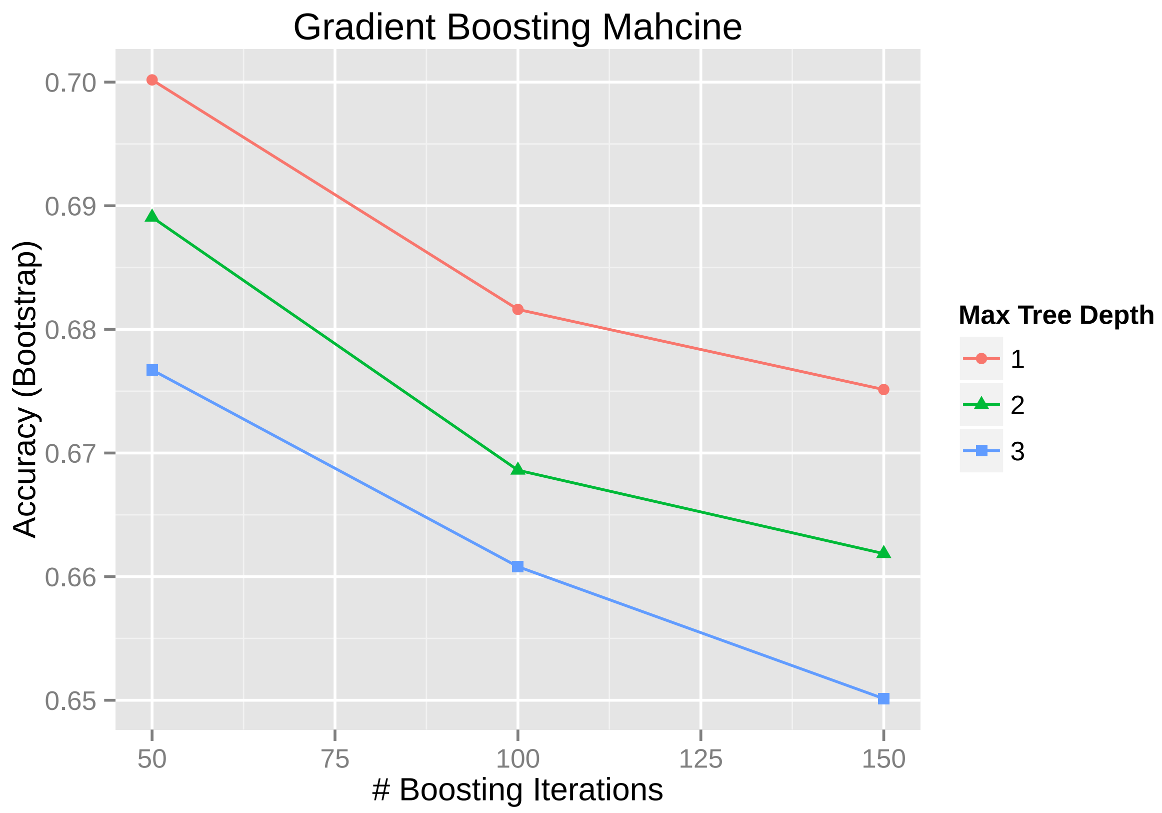
```r
ggplot(fitRandomForest) + ggtitle("Random Forest")
```



```r
ggplot(fitSVM) + ggtitle("Support Vector Mahcine")
```

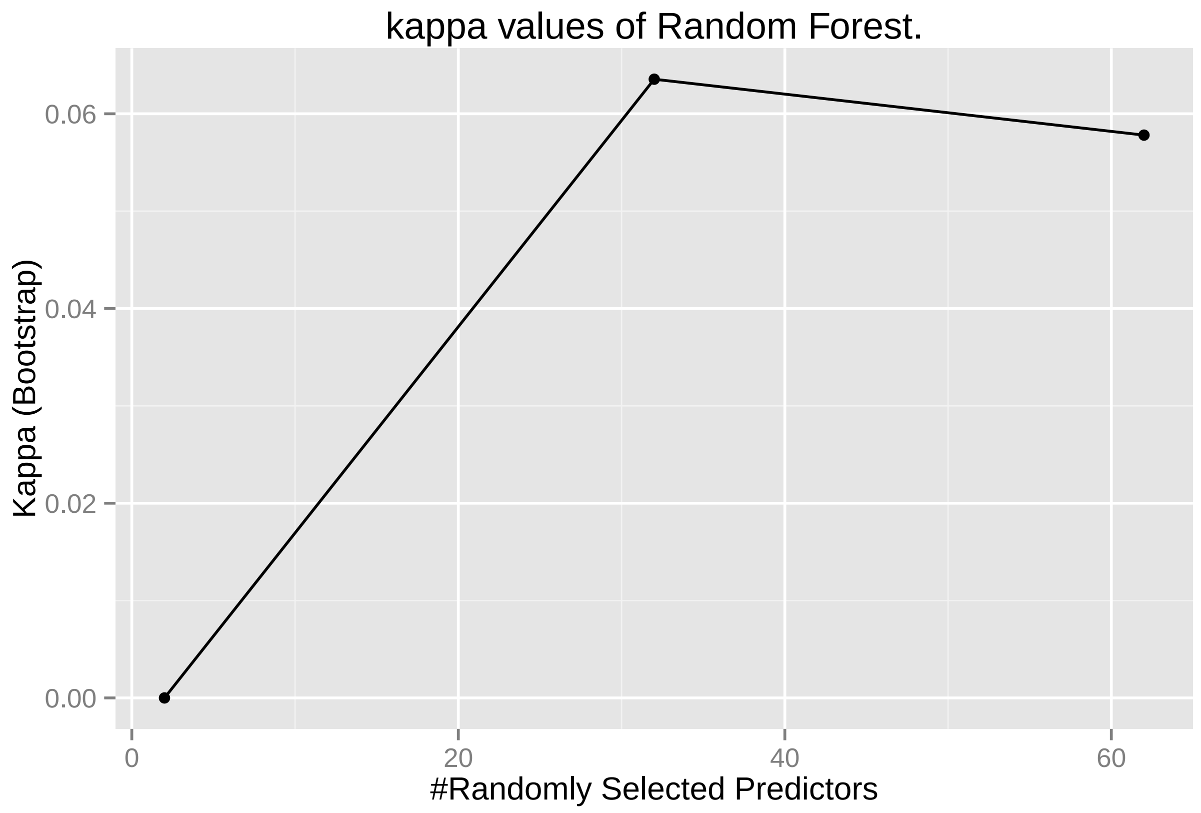


```r
ggplot(fitGbm) + ggtitle("Gradient Boosting Mahcine")
```

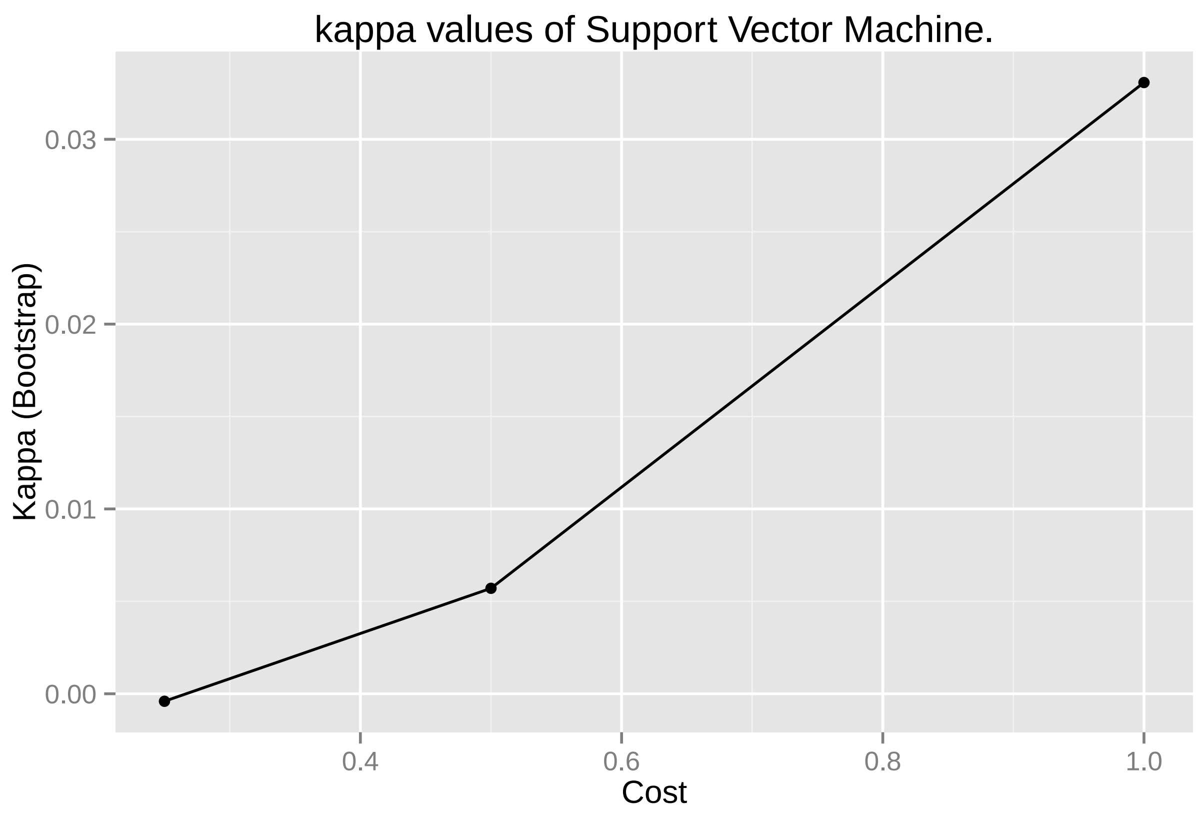


Followings are plots of kappa values of each model based on tuning parameters. We observed kappa values to measure the interrater reliability. Unfortuantely, all of the kappa values generated by models are low, which shows a poor level of agreement.

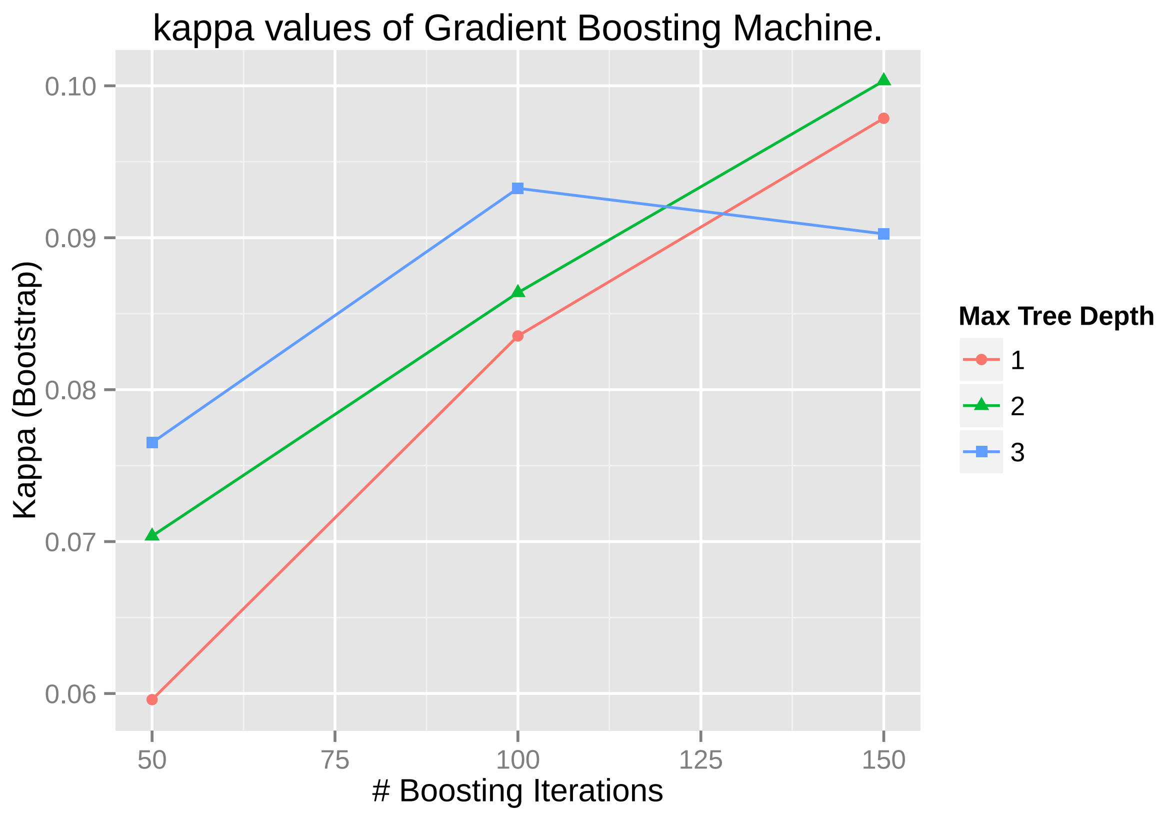
```r
ggplot(fitRandomForest, metric = "Kappa") + ggtitle("kappa values of Random Forest.")
```



```r
ggplot(fitSVM, metric = "Kappa") + ggtitle("kappa values of Support Vector Machine.")
```



```
ggplot(fitGbm, metric = "Kappa") + ggtitle("kappa values of Gradient Boosting Machine.")
```



Followings are confusion matrices of each model. These show the results of the comparisons between reference and prediction internally. As these models show, blue-collar crimes are easier to predict than white-collar and other crime. Also, Random Forest and SVM predict every crime instacne as a blue-crime crime. We assumed that it is because there are more blue-collar crime instances in general and there is no vivid pattern among crime instances, or the grouping is too general to catch the distinct pattern of each type of crimes.

```r
trellis.par.set(caretTheme())
confusionMatrix(fitRandomForest)
```

```
## Bootstrapped (25 reps) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction blue other white
##      blue  70.2  18.6  11.2
##      other  0.0   0.0   0.0
##      white  0.0   0.0   0.0
```

```r
confusionMatrix(fitSVM)
```

```
## Bootstrapped (25 reps) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction blue other white
##      blue  71.0  17.7  11.2
##      other  0.0   0.0   0.0
##      white  0.0   0.0   0.0
```

```r
confusionMatrix(fitGbm)
```

```
## Bootstrapped (25 reps) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction blue other white
##      blue  68.3  16.3   9.7
##      other  2.2   1.7   0.7
##      white  0.7   0.3   0.1
```

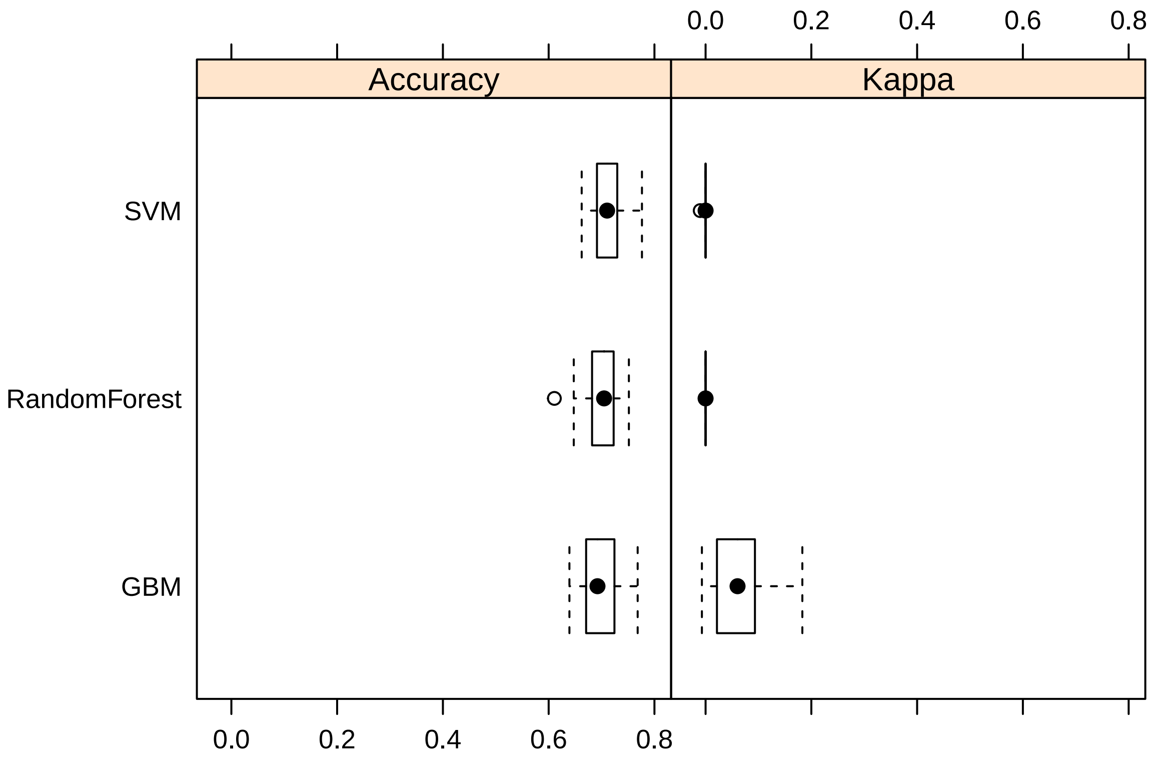
These are plot summaries of model comparisons. When it comes to the overall average accuracy, SVM is superior to other models. However, GBM has a higher kappa value among all. Model comparisons

```r
results <- resamples(list(RandomForest=fitRandomForest, SVM = fitSVM, GBM = fitGbm))
summary(results)
```

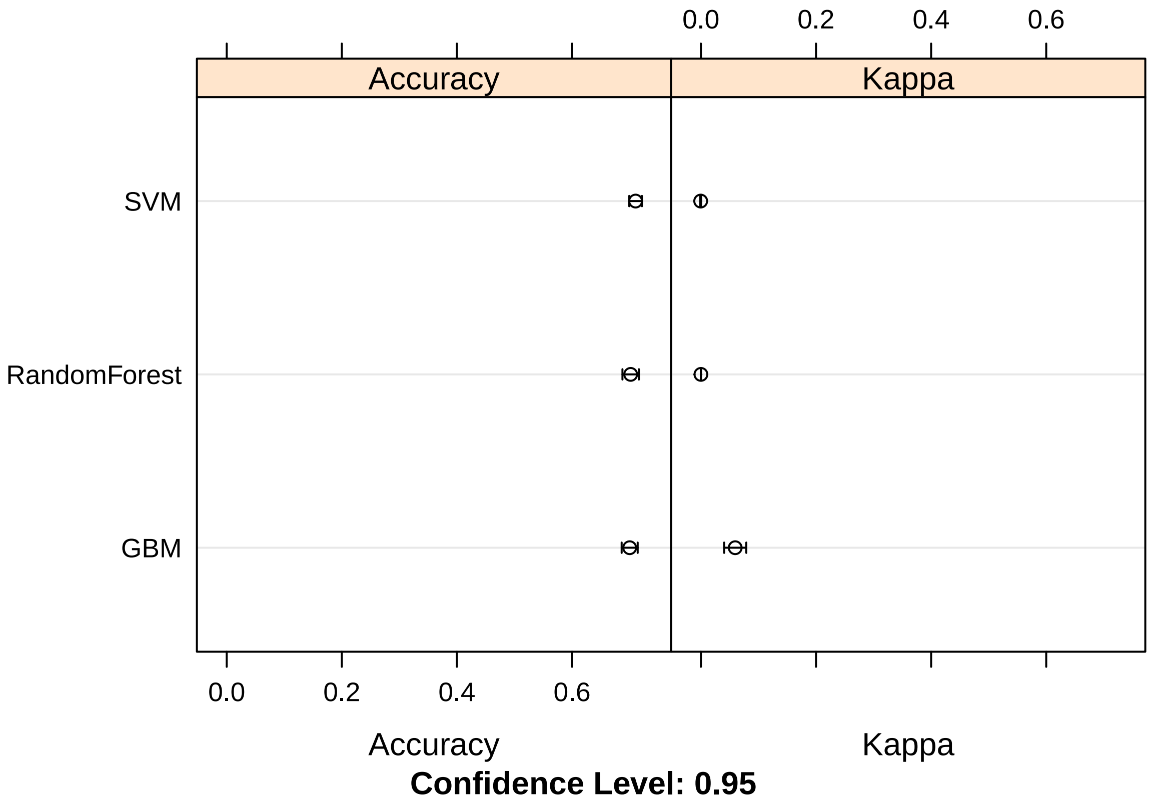
```
##
## Call:
```

```
## summary.resamples(object = results)
##
## Models: RandomForest, SVM, GBM
## Number of resamples: 25
##
## Accuracy
##                Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## RandomForest 0.6108  0.6821 0.7048 0.7019  0.7229 0.7517    0
## SVM          0.6624  0.6914 0.7105 0.7104  0.7297 0.7764    0
## GBM          0.6392  0.6708 0.6923 0.7002  0.7244 0.7683    0
##
## Kappa
##                   Min. 1st Qu.  Median       Mean 3rd Qu.  Max. NA's
## RandomForest  0.000000 0.00000 0.00000  0.0000000 0.00000 0.000    0
## SVM          -0.010170 0.00000 0.00000 -0.0004069 0.00000 0.000    0
## GBM          -0.006936 0.02153 0.06045  0.0596000 0.09337 0.183    0
```

```
bwplot(results)
```

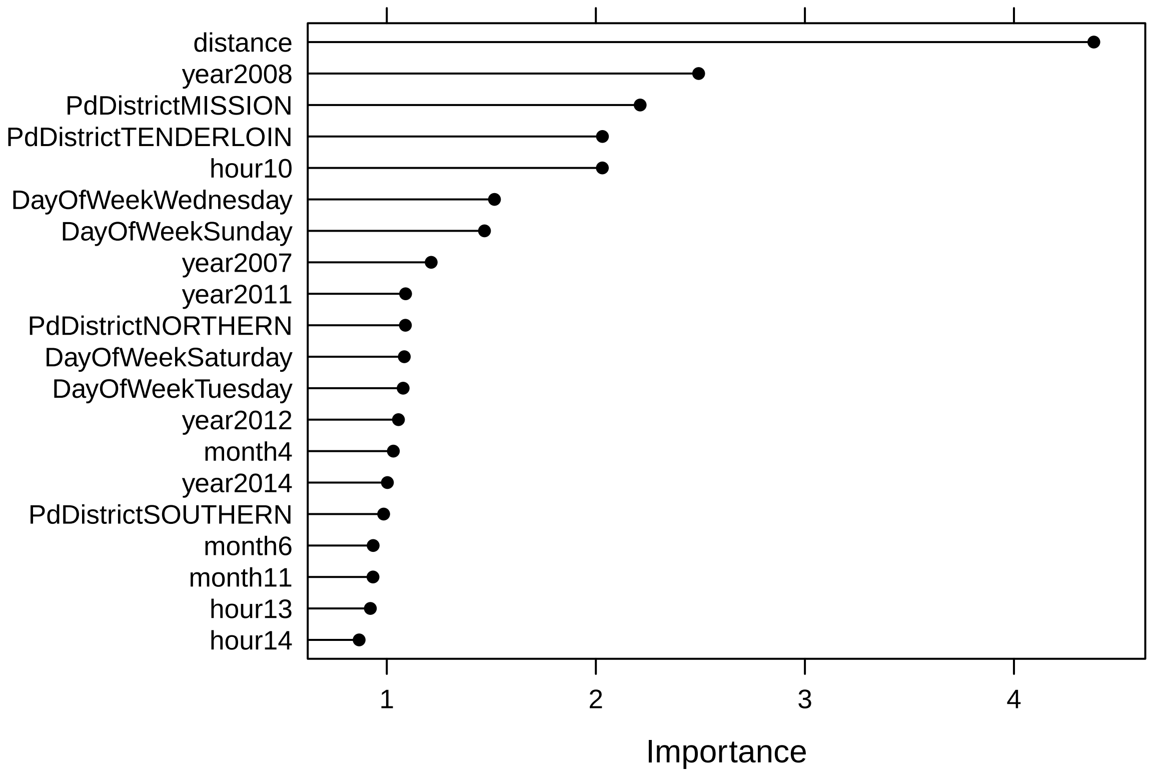


```
dotplot(results)
```

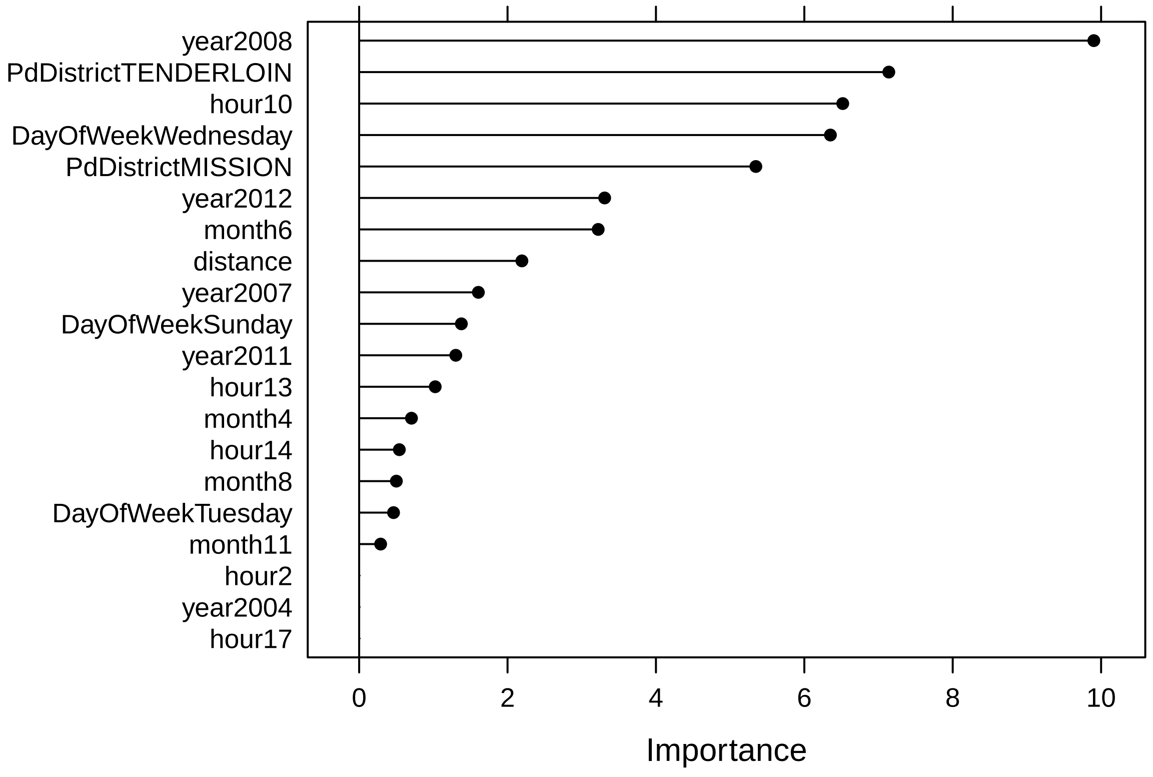


There are plots of variable importance calculated by each model. Random Forest model shows that distance, year 2008, PdDistrict, Wednesday and hour 10. GBM model also agrees this result. We can conclude that distacne from nearby Starbucks to the criem spot matters a lot, and there must have been distinct trend of crime during 2008 that makes it different from other years. Also, PdDistricts Mission, Tenderloin have distinct patterns from other areas.

```r
gbmImp <- varImp(fitRandomForest, scale = FALSE)
plot(gbmImp, top = 20)
```



```r
gbmImp <- varImp(fitGbm, scale = FALSE)
plot(gbmImp, top = 20)
```


year2008
PdDistrictTENDERLOIN
hour10
DayOfWeekWednesday
PdDistrictMISSION
year2012
month6
distance
year2007
DayOfWeekSunday
year2011
hour13
month4
hour14
month8
DayOfWeekTuesday
month11
hour2
year2004
hour17
Importance