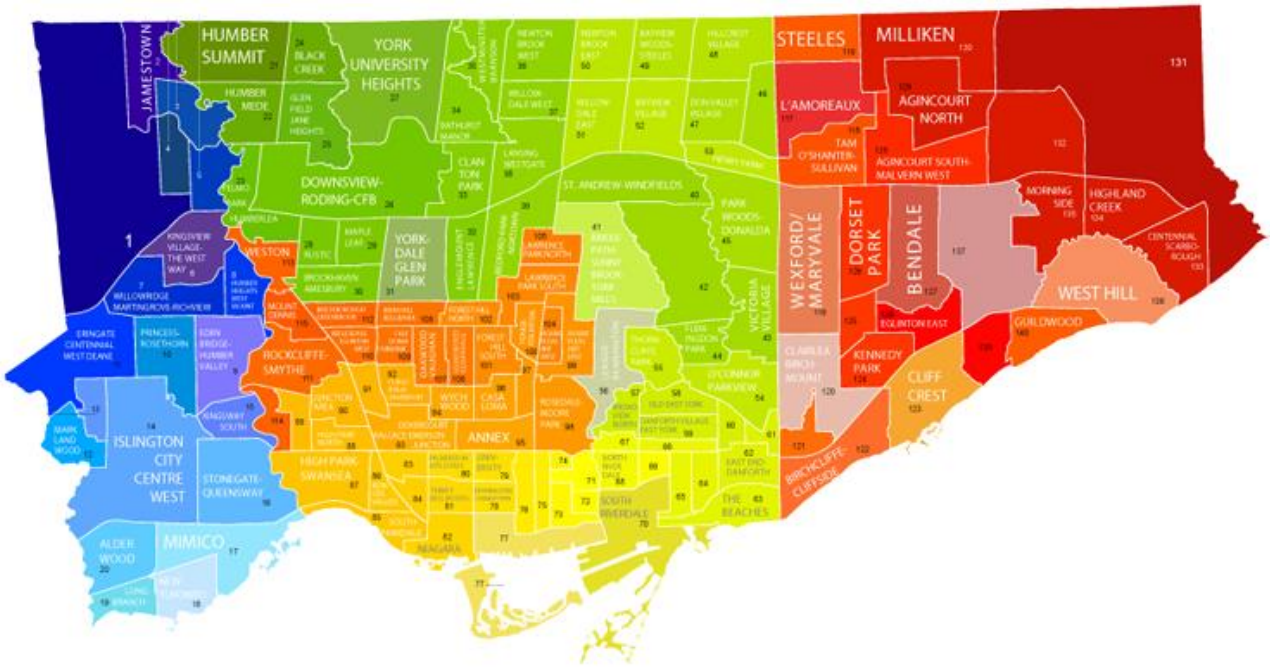
Toronto Crime Analysis & Prediction

July 2020

Abstract



An effort to predict crime with supervised machine learning algorithms

CKME136 Capstone Project

Certificate in Data Analytics, Big Data, and Predictive Analytics

Submitted by: **Vidyasankar Sundar**

Supervised by: **Tamer Abdou, PhD**

Ryerson University

Contents

LIST OF TABLES AND FIGURES

Toronto Crime Analysis & Prediction

Abstract

Can we predict crime before it happens? The capstone project proposes to tackle this key question, analyze crime data and build predictive models to predict crime in Toronto. Supervised machine learning algorithms such as classification will be utilized for this prediction task. The techniques will also cover to include pattern identification, prediction, and visualization.

The following questions are also addressed using exploratory/descriptive statistical data analysis methods:

1. Top crimes committed by type - which crimes are most frequently committed and what are the trends in the crimes being committed?
2. When do these crimes occur and is there a pattern?
3. Crime hotspots in Toronto - which locations are these frequent crimes being committed to?
4. What are the safest neighborhoods in Toronto?

Introduction

Crime prediction is a law-enforcement technique that uses data, and machine learning for the identification of crimes most likely to occur. Toronto is one of the most populous, ethnically diverse, and multicultural urban cities in Canada. A supervised prediction technique, namely, classification can be utilized to create a predictive model that can predict the category of crimes in Toronto. More specifically, four algorithms namely Decision tree, KNN Classifier, Naïve Bayes, and Random Forest will be tested, compared and evaluated in order to identify the best performing model for crime prediction. This work does not focus on the victim and the offender, but on the prediction of occurrence of a certain crime type per location and time using past data.

Literature Review

Crime prediction has always been subject to continued research in many parts of the world and various researchers have proposed different crime-prediction algorithms.

Identifying crime hotspots (defined as the areas in which the crime rates are above the average level) has particularly garnered interest among most researchers. One of the papers reviewed utilized a data-driven machine-learning algorithm, spatial analysis, and visualization techniques [1] to analyze drug-related crime data in Taiwan to predict crime hotspots. Another research utilized human behavioral data [2] from mobile network activity combined with demographic information using real crime data to predict crime hotspots in London, UK.

On reviewing several crime-prediction methods [3], an effective tool for predicting crime involved the combined use of techniques such as statistical modeling, machine learning, database storage, and AI technologies. Other reviewed papers have proposed different crime-prediction algorithms and the accuracy of prediction depended a lot on the attributes selected within the dataset. As an example, a study provided a comparison [4], between two classification algorithms, Decision Tree and Naive Bayesian, to predict crime. A comparative study to measure the accuracy and effectiveness of linear regression, additive regression, and decision stump algorithms [5] was

conducted to predict the crime in the state of Mississippi. Other cities such as the City of Chicago, the city of San Francisco have also used numerous models to predict crimes in their respective cities. [6&7].

Of note, the most important research that will be leveraged for this capstone project is the study that leveraged the open Vancouver crime data set from 2003-2018 and utilized the classification algorithms KNN and boosted decision tree [8] to predict crime in the city of Vancouver. These techniques resulted in a crime prediction accuracy that ranged from 39% to 44% when predicting crime in Vancouver.

A similar approach shall be used to predict crime in Toronto using the open data provided by the city and an effort to compare various algorithms such as decision tree, KNN, Naïve Bayes, and Random Forest methods will be explored in this project. Of note, the criteria used for the selection of the proposed classifiers are based entirely on prior work done by other researchers i.e. the papers reviewed in this section mostly used these classifiers on the crime dataset for their respective cities with varying levels of accuracies.

Dataset

Toronto crime data from 2014-2019 is made available/ open by the Toronto Police Service public safety data portal. Link to this dataset can be found here. The GitHub repository for this project where all the files including R code is stored can be found here.

The dataset is structured and contains all Major Crime Indicators (MCI) and offences between 2014 and 2019 including location, neighborhood, day, month, and time. The MCI categories are Assault, Break and Enter, Auto Theft, Robbery and Theft Over (act of stealing property in excess of $5,000). Note that the dataset excludes Homicides and Sexual Assaults. The location of crime occurrences have been deliberately offset to the nearest road intersection node to protect the privacy of parties involved in the occurrence. All location data must be considered as an approximate location of the occurrence henceforth in the analysis.

The original dataset contained 27 features (attributes or variables) and 206435 observations prior to data cleaning. The 26 features of this dataset include (object id is a feature that is not shown in the table below):

Table 1 Dataset Variables Description

Fields	Field_Description	
Index	Record identifier	**1**
event_unique_id	Event identifier	**2**
Occurrence date	Date of crime occurrence in UNIX format	**3**
Reported date	Date when crime was reported in UNIX format	**4**
Premise type	Where crime occurred	**5**
ucr_code	UCR code – unique code	**6**
ucr_ext	UCR code extension	**7**
offence	Offence related to the event	**8**
Reported year	Year crime was reported	**9**
reportedmonth	Month crime was reported	**10**
Reportedday	Day crime was reported	**11**
Reported day of year	Day of week crime was reported	**12**
Reportedday of week	Day of year crime was reported	**13**
reportedhour	Hour crime was reported	**14**
occurrenceyear	Year crime occurred	**15**
occurrencemonth	Month crime occurred	**16**
occurrenceday	Day crime occurred	**17**
occurrencedayofyear	Day of year crime occurred	**18**
Occurrencedayofweek	Day of week crime occurred	**19**
occurrencehour	Hour crime occurred	**20**
MCI	Major crime incident category	**21**
Division	Division where event occurred	**22**
Hood-Id	Neighbourhood identifier	**23**
Neighbourhood	Neighbourhood name	**24**
Long	Longitude	**25**
Lat	Latitude	**26**

After cleaning the data, we end up with 205321 observations of 27 variables. The structure of the data frame showing the features and their data types is printed below:

```
> str(EDAfilter)
'data.frame':205321 obs. of 27 variables:
 $ ï.Index_          : int 101 102 103 104 105 106 107 108 109 110...
 $ event_unique_id   : Factor w/ 178892 levels "GO-20141035797", 4468 6014
 $ Occurrence date     : num 1.39e+12 1.40e+12
 $ Reported date       : num 1.39e+12 1.40e+12
 $ Premisetype        : Factor w/ 5 levels "Apartment","Commercial": 3 1 1 1
 $ ucr_code           : int 2132 1430 1420 2120 2130 1610 1430 1430 1430
 $ ucr_ext            : int 200 100 100 220 210 200 100 100 100 210
 $ Offence            : Factor w/ 49 levels "Administering Noxious Thing": 39
 $ Reportedyear       : int 2014 2014 2014 2014 2014 2014 2014 2014 2014 2014
 $ Reportedmonth      : Factor w/ 12 levels "April","August": 8 1 1 1 1 8 8 4
 $ Reportedday        : int 9 1 1 1 1 16 16 15 1 1
 $ Reporteddayofyear: int 68 91 91 91 91 75 75 46 1 1
 $ Reporteddayofweek: Factor w/ 7 levels "Friday    ","Monday    ": 4 6 6 6 6
 $ Reportedhour       : int 18 16 16 16 20 7 2 3 16 14.
 $ Occurrenceyear     : int 2014 2014 2014 2014 2014 2014 2014 2014 2014 2014
 $ Occurrencemonth    : Factor w/ 12 levels "January","February": 2 4 4 4 4 3
 $ Occurrenceday      : int 27 1 1 1 1 16 16 15 1 1
 $ Occurrencedayofyear: int 58 91 91 91 91 75 75 46 1 1
 $ Occurrencedayofweek: Factor w/ 7 levels "Monday","Tuesday": 4 2 2 2 2 7 7
 $ Occurrencehour     : int 16 16 16 16 12 7 2 1 11 3.
 $ MCI                : Factor w/ 5 levels "Assault","Auto Theft": 5 1 1 3 5
 $ Division           : Factor w/ 17 levels "D11","D12","D13": 15 10 10 10 8
 $ Hood_ID            : int 101 121 121 121 34 85 113 131 120 75
 $ Neighbourhood      : Factor w/ 140 levels "Agincourt North (129)": 45 92 9
 $ Long               : num -79.4 -79.3 -79.3 -79.3 -79.5.
 $ Lat                : num 43.7 43.7 43.7 43.7 43.8
 $ ObjectId           : int 1 2 3 4 5 6 7 8 9 10
```

Figure 1 Dataset Structure

Approach

- The first step was to pre-process i.e. examine and clean the data
- Exploratory data analysis was conducted to better understand the data and answer the research questions
- The most influencing features were selected for data modeling. Class imbalance was addressed using Synthetic Minority Oversampling Techniques (SMOTE)
- Four algorithms were tested to identify the best performing model
- Comparison of these methods along with detailed evaluation is presented to understand next steps and conclude the project.
- The entire analysis and implementation of machine learning algorithms were conducted in R.

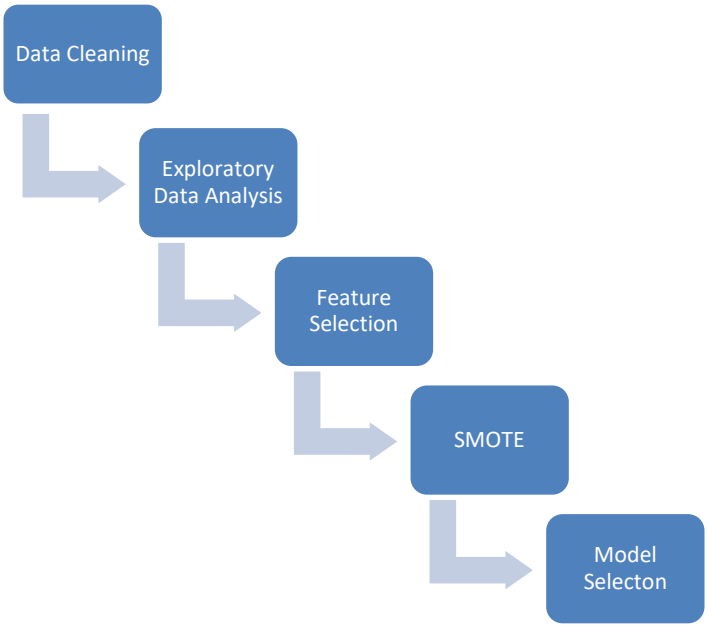


Figure 2 Approach Steps

Step 1: Data Cleaning/Pre-processing

Cleaning up of the dataset involved the following steps:

- Total number of records in the dataset was 206435
- Missing values, about 59, were omitted from the raw data
- In addition, some crime data records (about 1055) going back to 2000 was also in the dataset. Therefore the dataset was filtered to include only the occurrences from 2014 to 2019. After all this cleaning, we ended up with 205321 observations
- Factor variables such as occurrence month and occurrence day of week were converted to order factors as the levels earlier were arranged by alphabetic order. We wanted to avoid the possibility of this causing issues during the exploratory phase.

R code available here. (The first 30 lines of code in that file contain the cleaning and data pre-processing steps).

Step 2: Exploratory Data Analysis

R was used to explore the processed data to identify crime trends, and patterns. The exploratory analysis helps us to answer key questions such as top crimes committed by type, crime trends over time, which crimes are frequently committed, and crime hotspots in terms of location among others. A detailed report (containing the key findings) is available here.

R code available here.

Step 3: Feature Selection

The dataset is split into 70% training set and 30% test set. We use information gain, a filter-based feature selection method which ranks variables according to the information gain with respect to the outcome. Information gain technique is implemented by utilizing the FSelector package in R on the training dataset. Manual feature selection was also performed to drop off the attributes that were considered redundant or irrelevant for the analysis. For example, the dataset has both occurrence and reported day, month year and weeks and we decided to keep only the occurrence related time variables. Index id, and hood id were irrelevant while ucr codes and extensions were already captured under offences and MCI and were considered redundant.

The most influencing variables to predict crime are premisetype, occurrencehour, division, neighbourhood, longitude and latitude.

R code available here.

Step 4: SMOTE

The training dataset is seen to have class imbalances within the MCI category. The data is heavily skewed to the 'Assault' class with more than 50% of the observations belonging to this class within MCI. This class imbalance ought to be addressed in the training set prior to model building.

SMOTE was implemented in R using the UBL library to balance the classifications in the training dataset.

R code available here. (Lines 9-16 in that file contain the R code to implement SMOTE in the training dataset).

Step 5: Model Selection

Finally, a supervised machine learning algorithm (classification-decision tree) was implemented on the dataset to build a model to predict crime categories in Toronto given the inputs. In addition, we compared the performance of the model against other algorithms such as KNN classification, Naïve Bayes, and Random Forest to select the best performing model and identified next steps to conclude the project.

R code available here.

Results

Exploratory/Descriptive Analysis

R code available here.

Below are the key insights gleaned from the exploratory analysis conducted on the Toronto open dataset containing the crimes that occurred between 2014 and 2019.The supporting graphs are presented following the findings

- In Toronto, the average number of crime incidents is 33685 per year, 2852 per month, and 92 per day. Overall, crime has seen an increasing trend since 2014.
- Crime occurs the most during summer and fall (May-October). The first and the last week of any month is seen to have the most incidents. Notably, the peak observed is on the first day of every

month. During weekdays, peak crime is observed at noon and then starts to gradually build from 3PM onwards and steadily increases into the night. During weekends, peak is observed at midnight. In addition, there are more crimes on Fridays and weekends than any other day in the week.

- Most number of crimes happen outside followed by apartments and commercial establishments.
- Assault is the most prevalent form of crime in Toronto followed by home/commercial break and enter and auto theft. All crime types have seen an increasing trend since 2014 with the exception of robbery
- Most assault incidents happen between May and July, auto theft between July and November while other types are fairly consistent across all months. Maximum number of Assault incidents occur usually on weekends, while other types increasingly occur on Fridays. Assault, Break and Enter, Theft over peaks are observed at both midnight and noon, auto theft at 10PM, and robbery tops at 9PM.
- Auto theft happens mostly outside and at houses while other crime types are common across all premise types
- Top offenses within the assault category include - assault with weapon, bodily harm, and assault peace officer while top offenses within robbery include - mugging, other, robbery with weapons, and robbery-business.
- The most dangerous neighborhood is the Waterfront. The other two most dangerous hoods are the Bay Street Corridor and Yonge-Church Corridor. The safest hoods are Lambton Baby Point, Woodbine-Lumsden, Maple Leaf, and Guildwood
- Besides assaults, Bay Street Corridor, Church-Yonge Corridor and Waterfront had the most break and enter crimes while West Humber-Clairville had the most vehicle stolen crimes. Assaults, and Break and Enter occur all over the city, with a concentration in the Waterfront areas. Other crimes, such as Auto Theft, have more points on the west side than the east side. Robbery and Theft Over are primarily in the Waterfront areas.

In Toronto, the average number of crime incidents is 33685 per year, 2852 per month, and 92 per day.

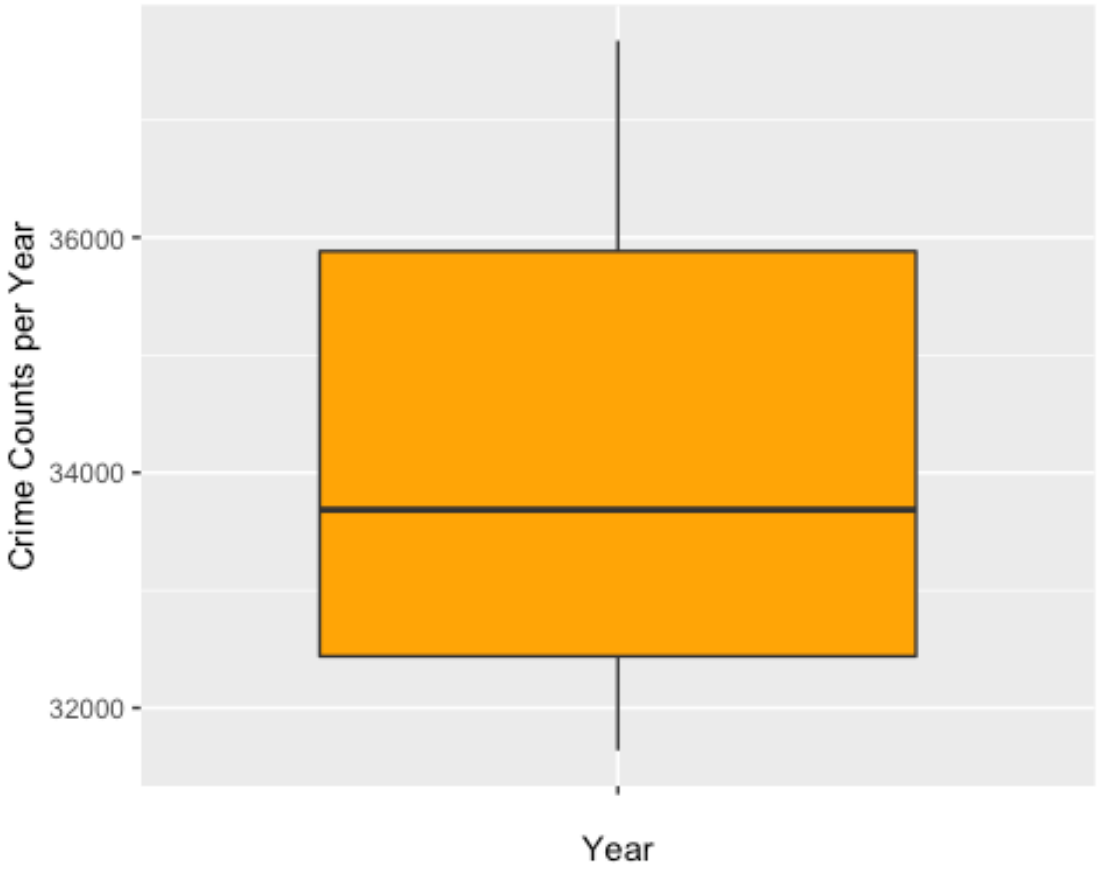


Figure 3 Crime Distribution per year in Toronto

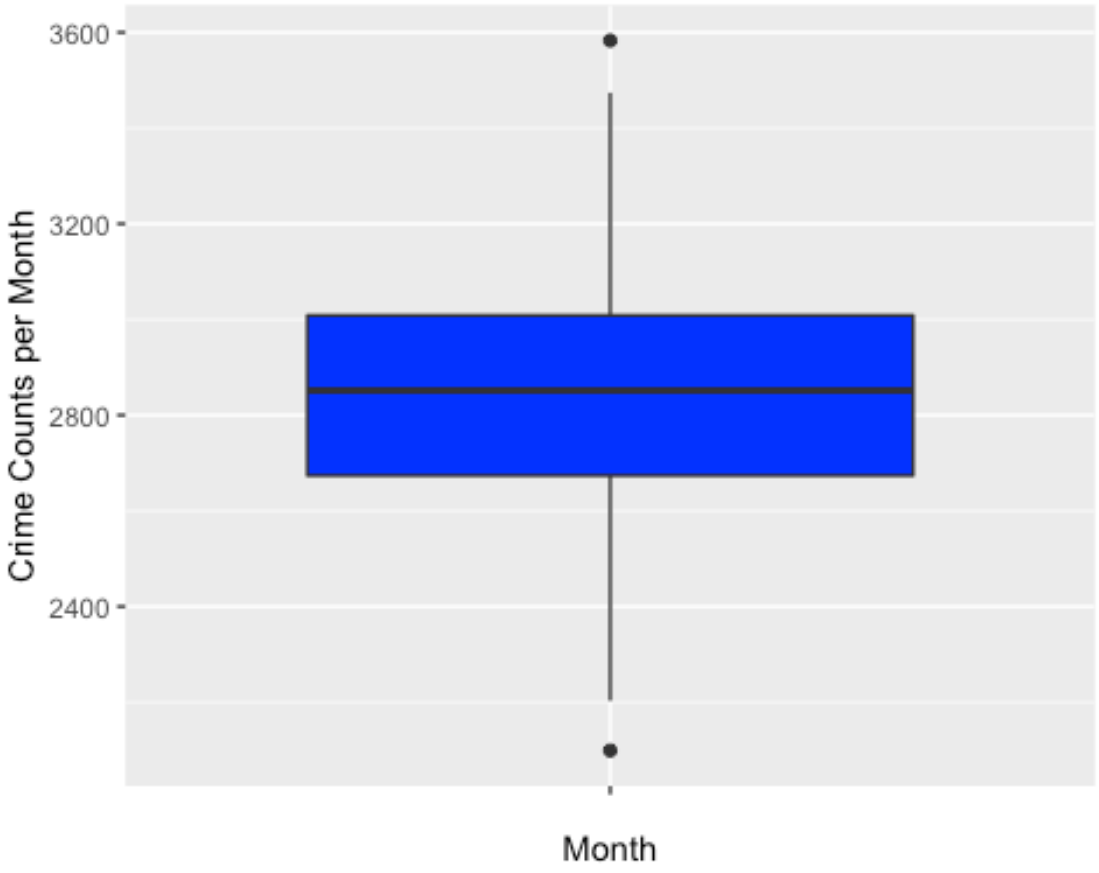


Figure 4 Crime Distribution per Month

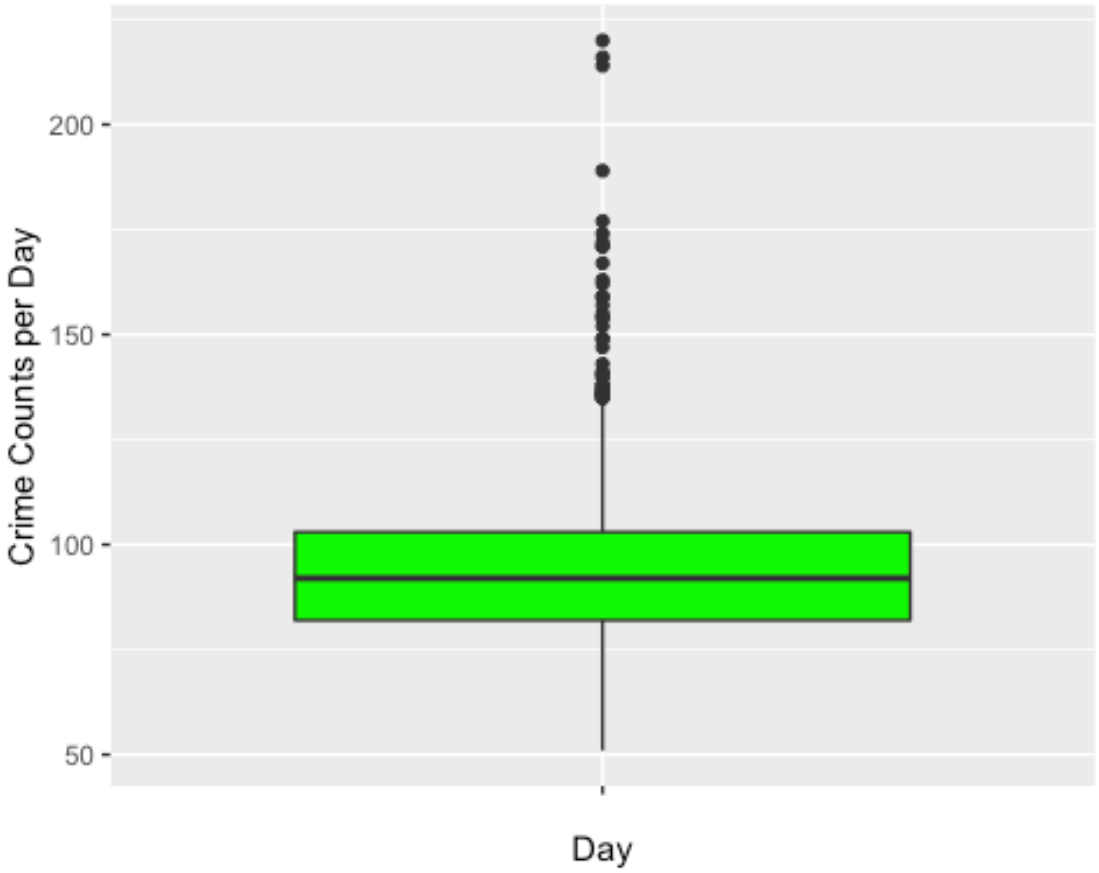


Figure 5 Crime Distribution per Day

Overall, crime has seen an increasing trend since 2014.

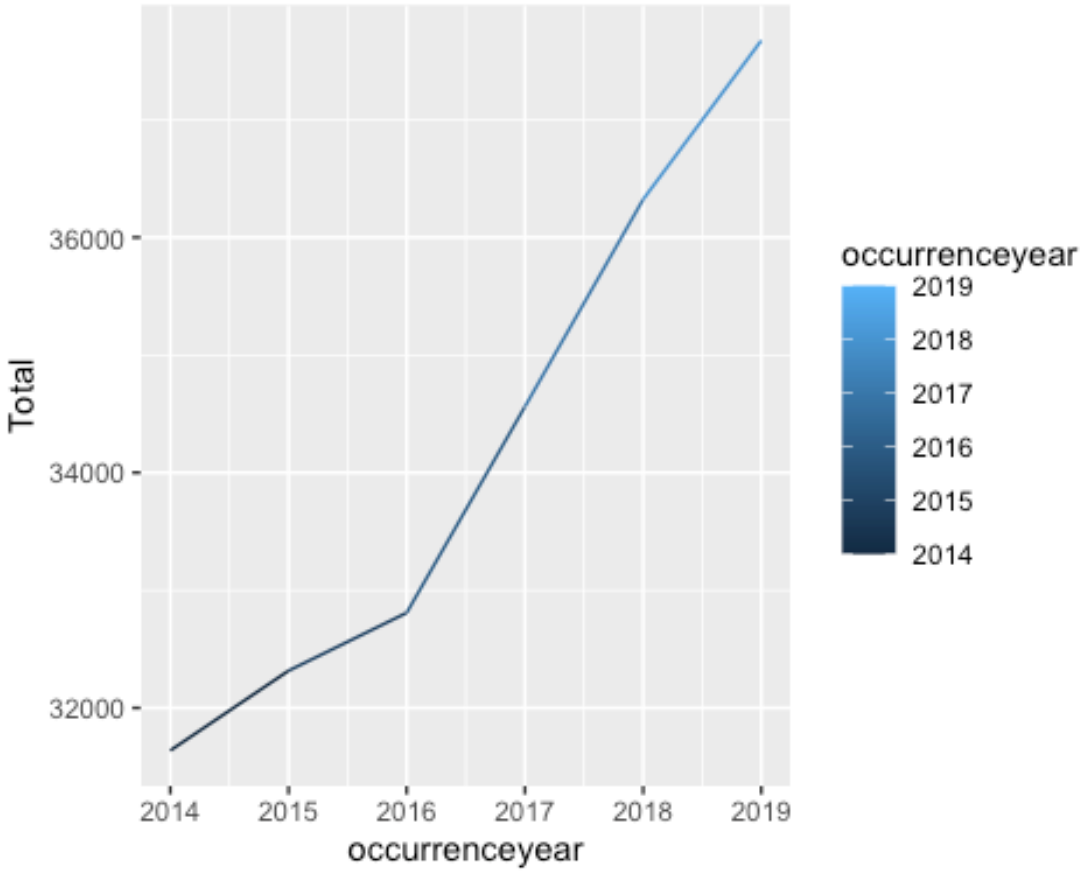


Figure 6 Crime Trends by Year

Crime incidents are most during summer and fall (May-October), with frequencies peaking on the first of every month. Crimes are also most around Fridays and weekends.

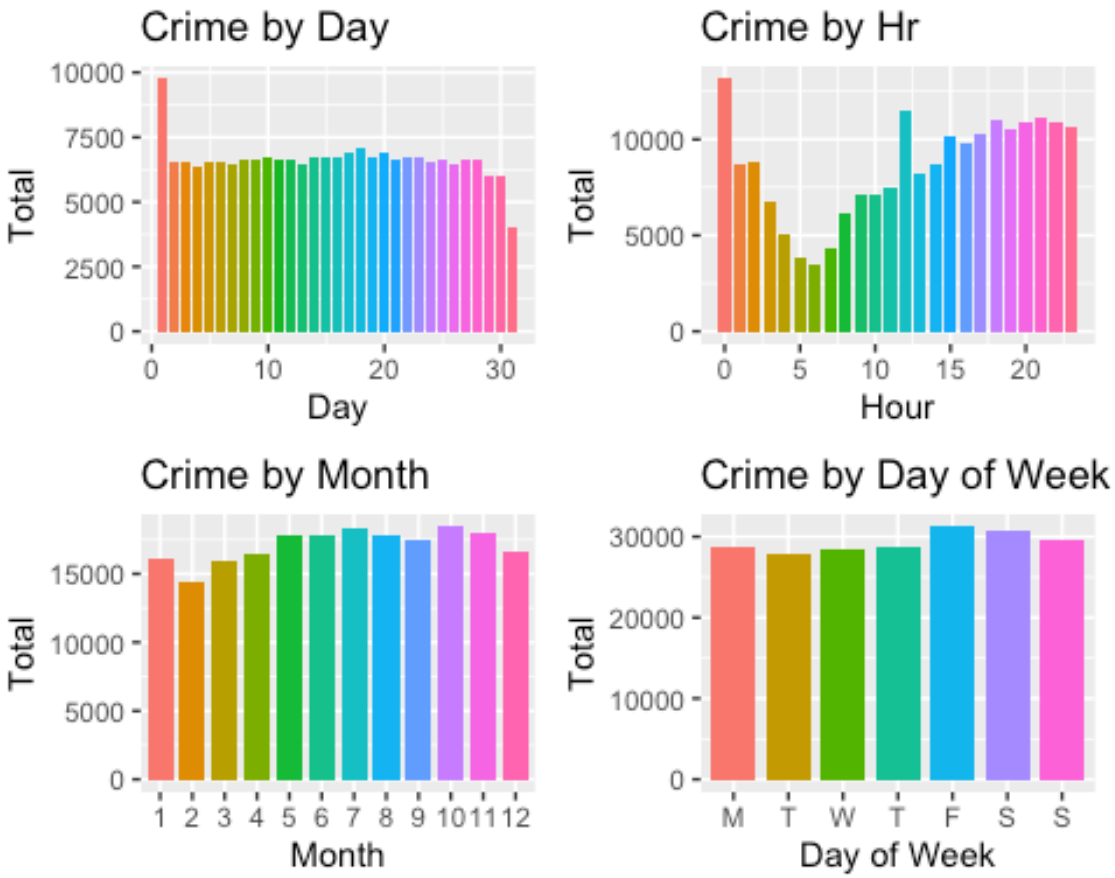


Figure 7 Crime Trends Summary

Crime peaks at noon, tapers off and gradually picks up steam to continuously progress through evening and late into night.

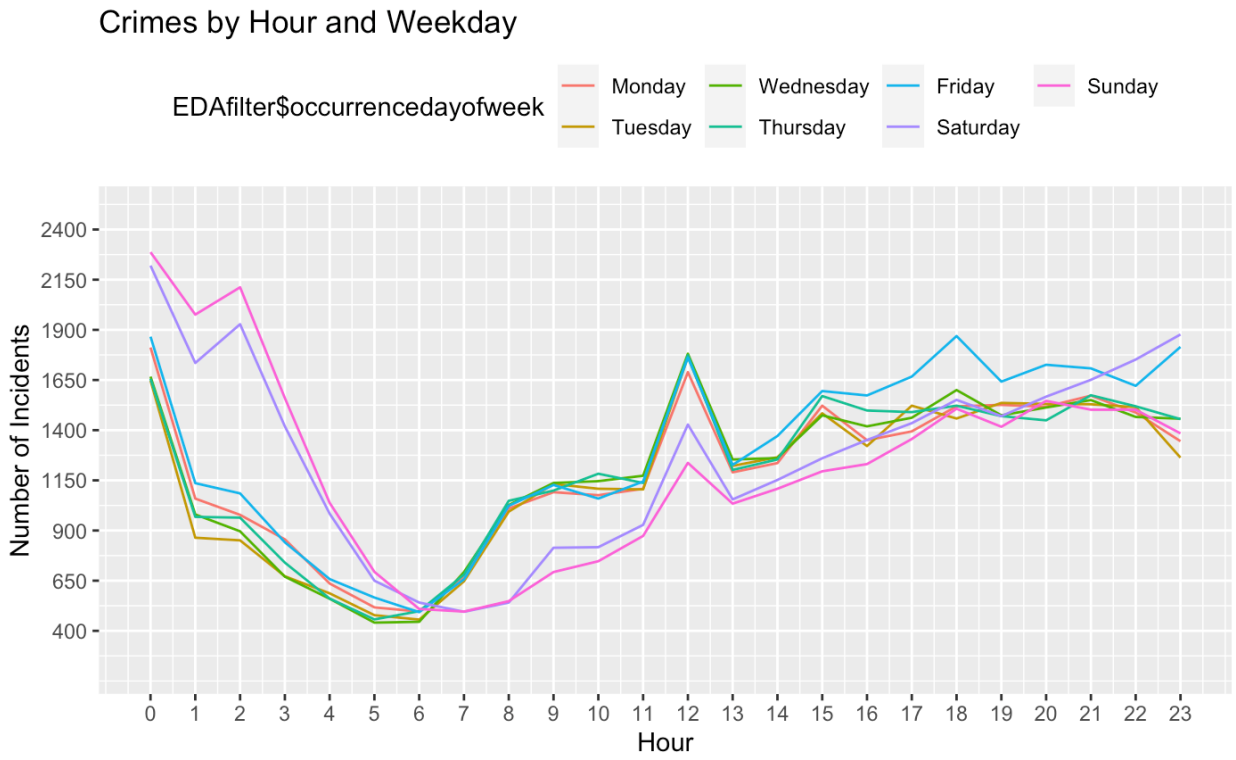


Figure 8 Crimes by Hour and Weekday

The first and the last week of any month is seen to have the most incidents. The peak observed is on the first day of every month.

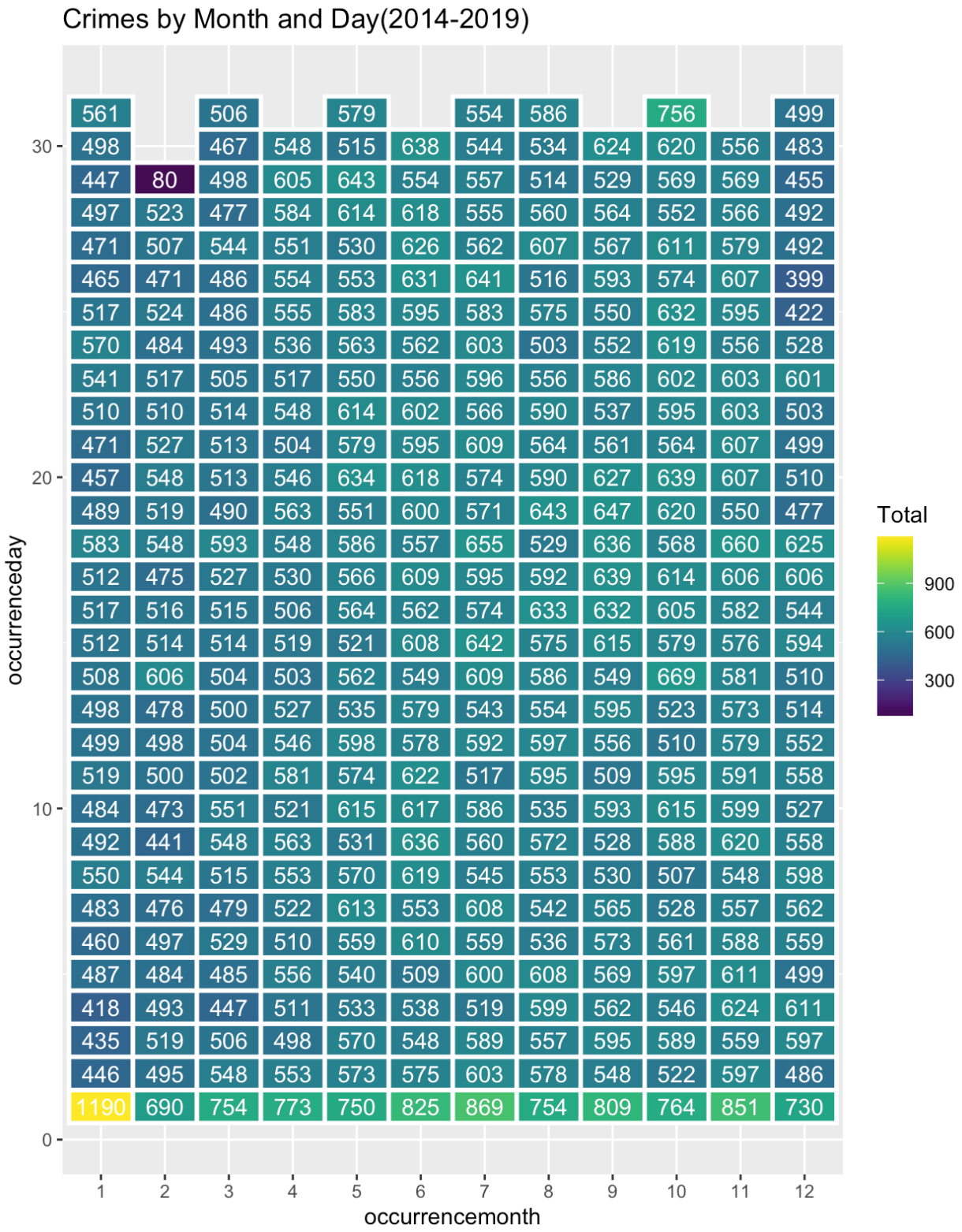


Figure 9 Crime Heat map Month and Day

During weekdays, peak is observed at noon and then starts to gradually build from 3PM. During weekends, peak is observed at midnight.

Figure 10 Crime Heat map Day and Hour

Most number of crimes happen outside followed by apartments and commercial establishments.

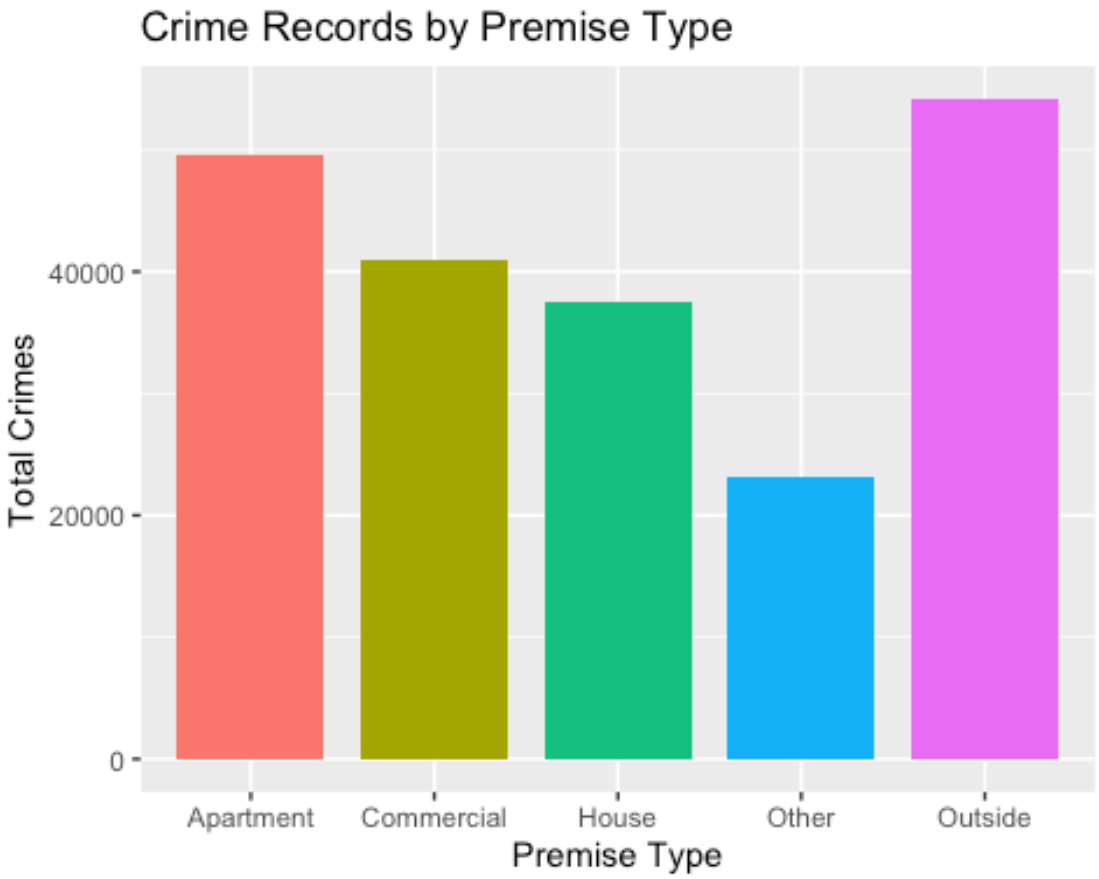


Figure 11 Crimes by Premise Type

Assault is the most prevalent form of crime in Toronto followed by home/commercial break and enter and auto theft.

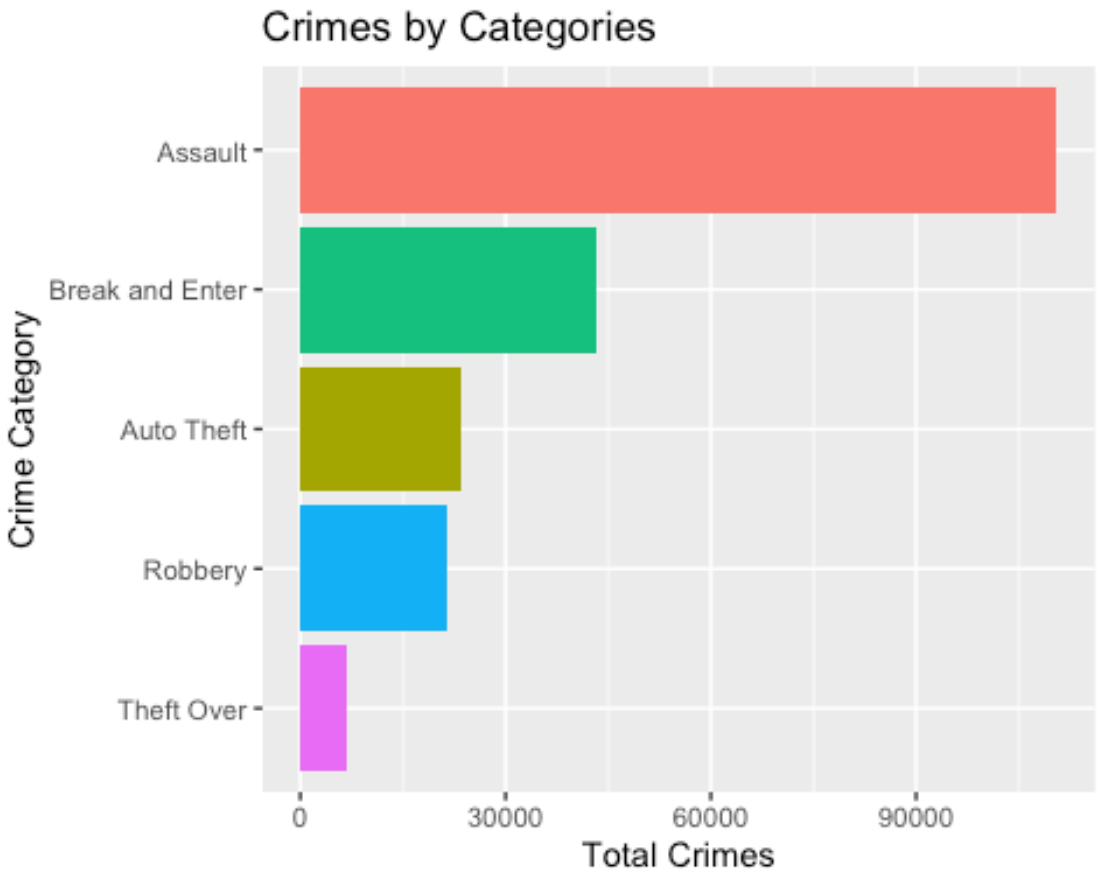


Figure 12 MCI Categories

All crime types have seen an increasing trend since 2014 with the exception of robbery

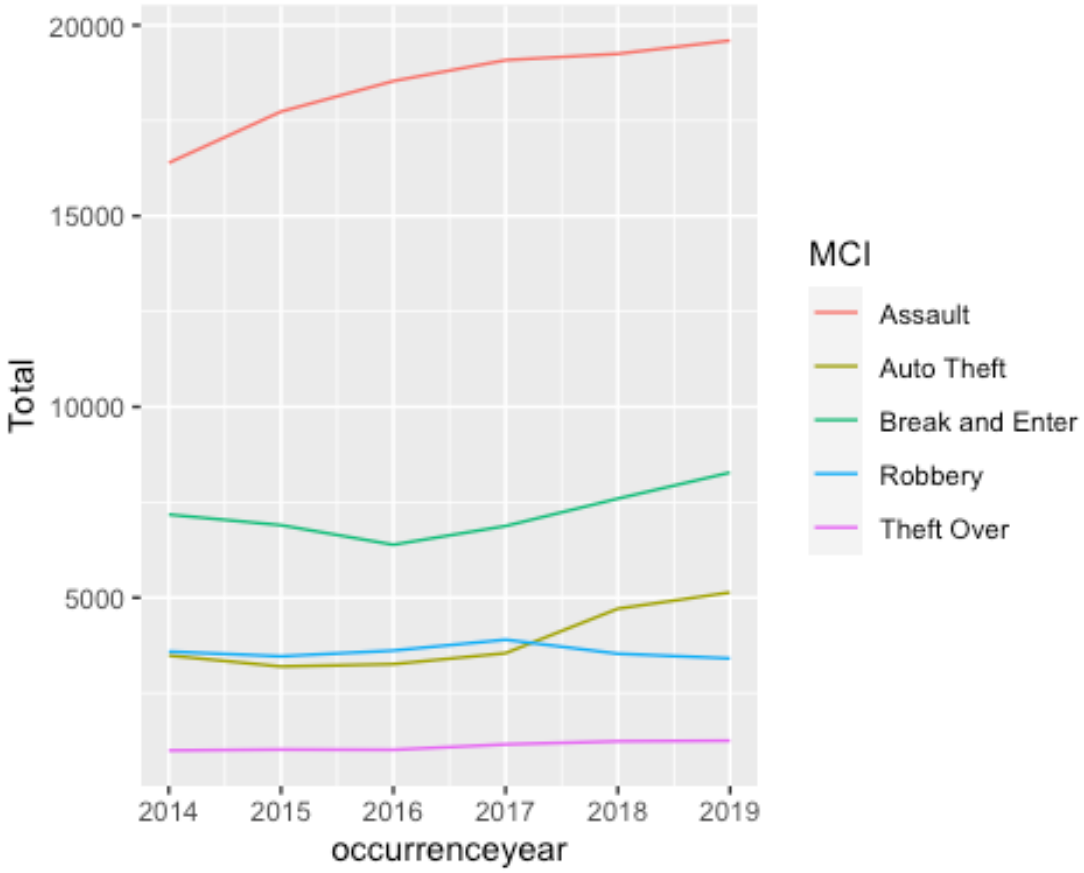


Figure 13 MCI Trends by Year

Most assault incidents happen between May and July, auto theft between July and November while other types are fairly consistent across all months.

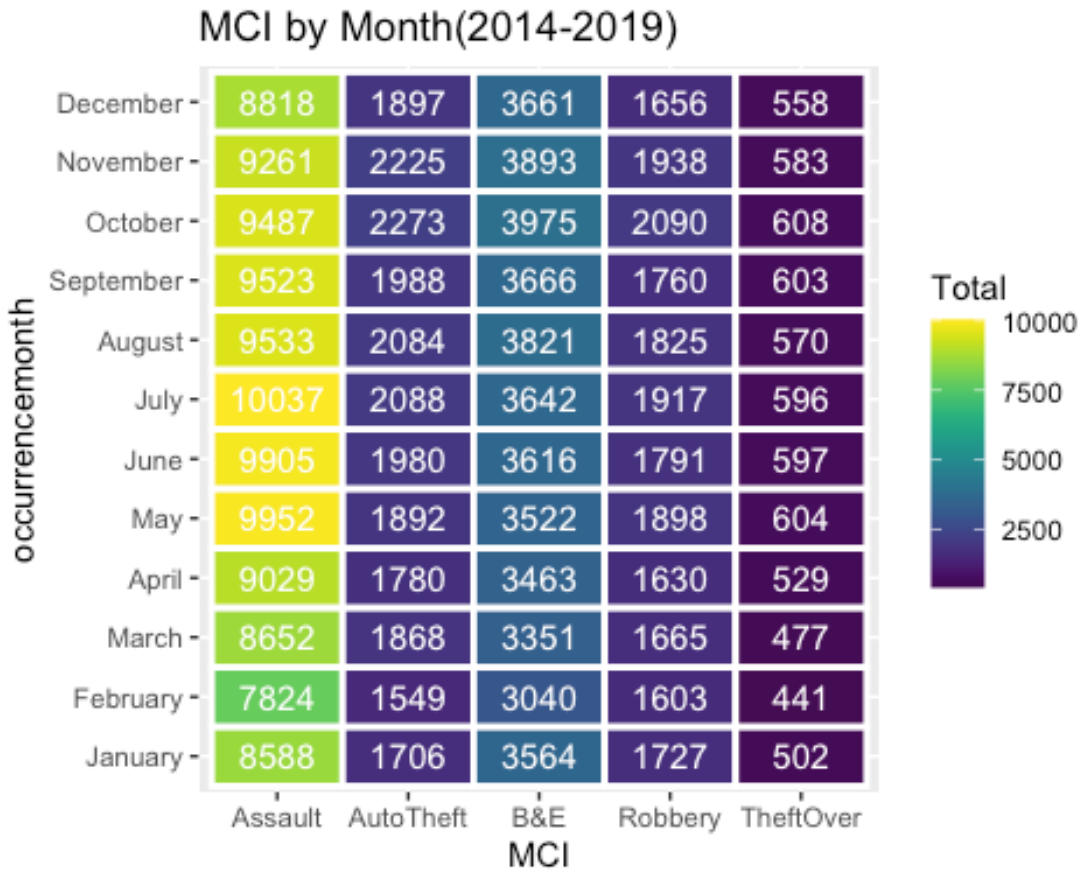


Figure 14 MCI Heat map by Month

Assault peak is observed usually on weekends, while other types peak on Fridays.

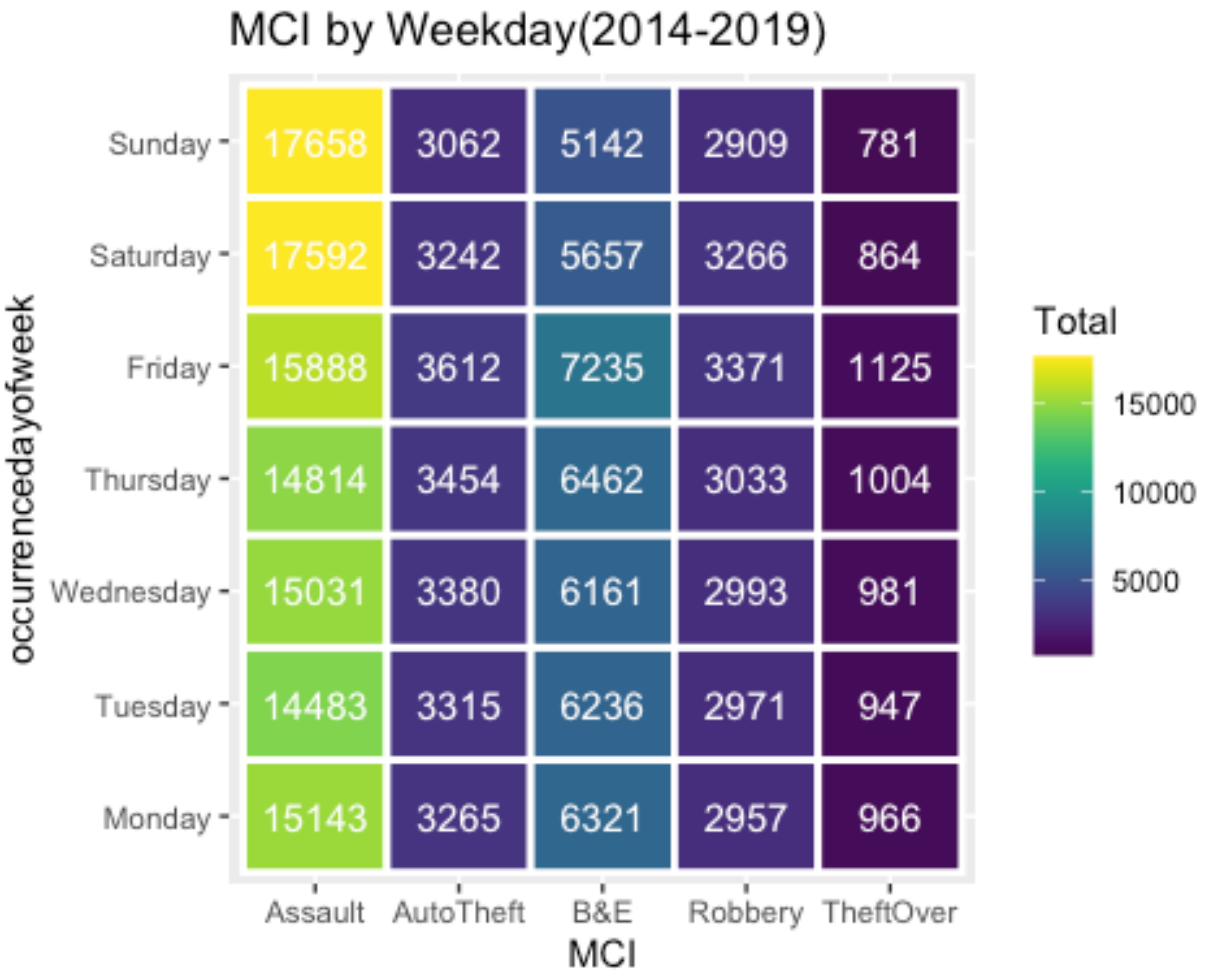


Figure 15 MCI Heat Map by Day of Week

Assault, Break and Enter, Theft over peaks are observed at both midnight and noon, auto theft at 10PM, robbery at 9PM.

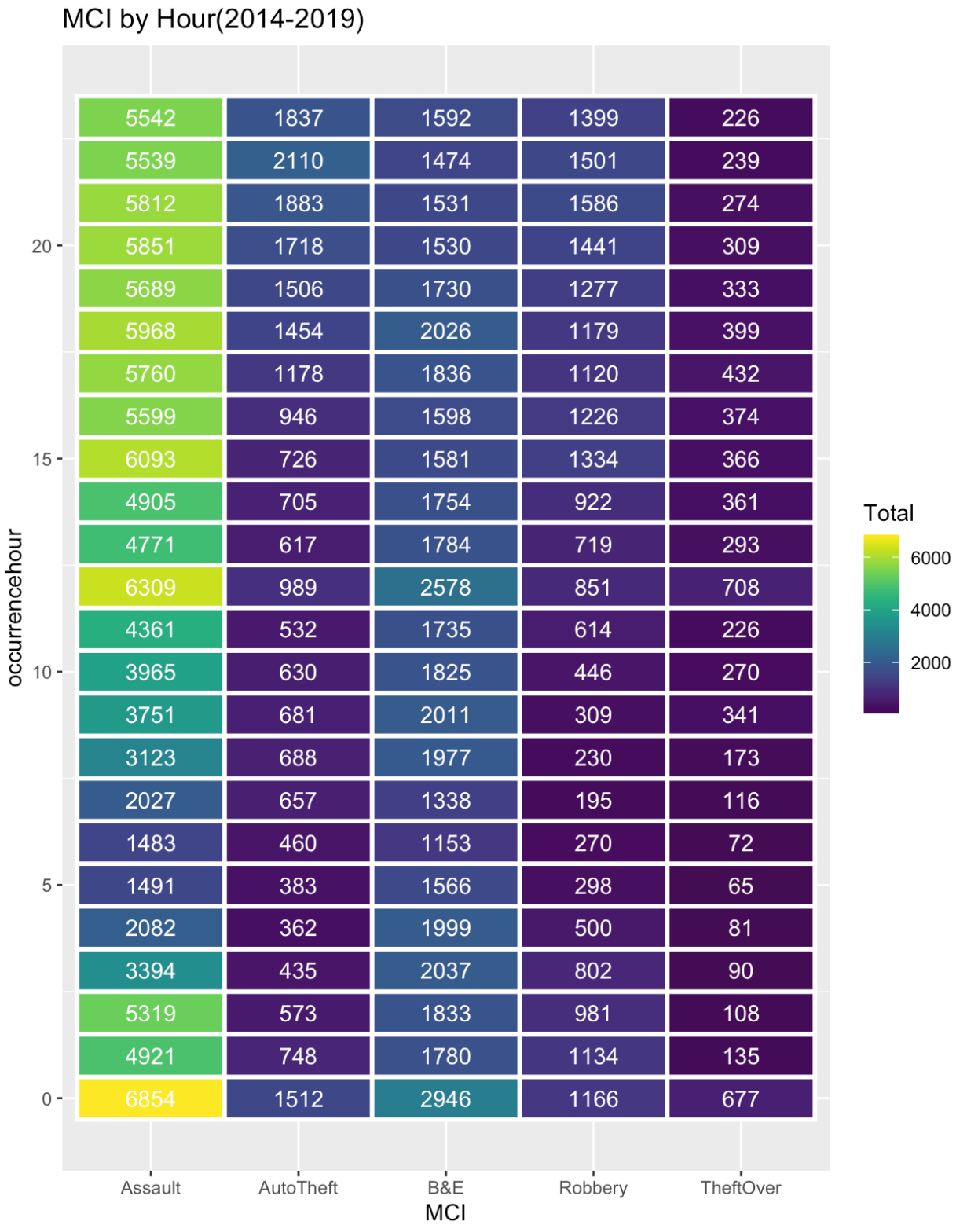


Figure 16 MCI Heat Map by Hour

Auto theft happens mostly outside and at houses while other crime types are common across all premise types.

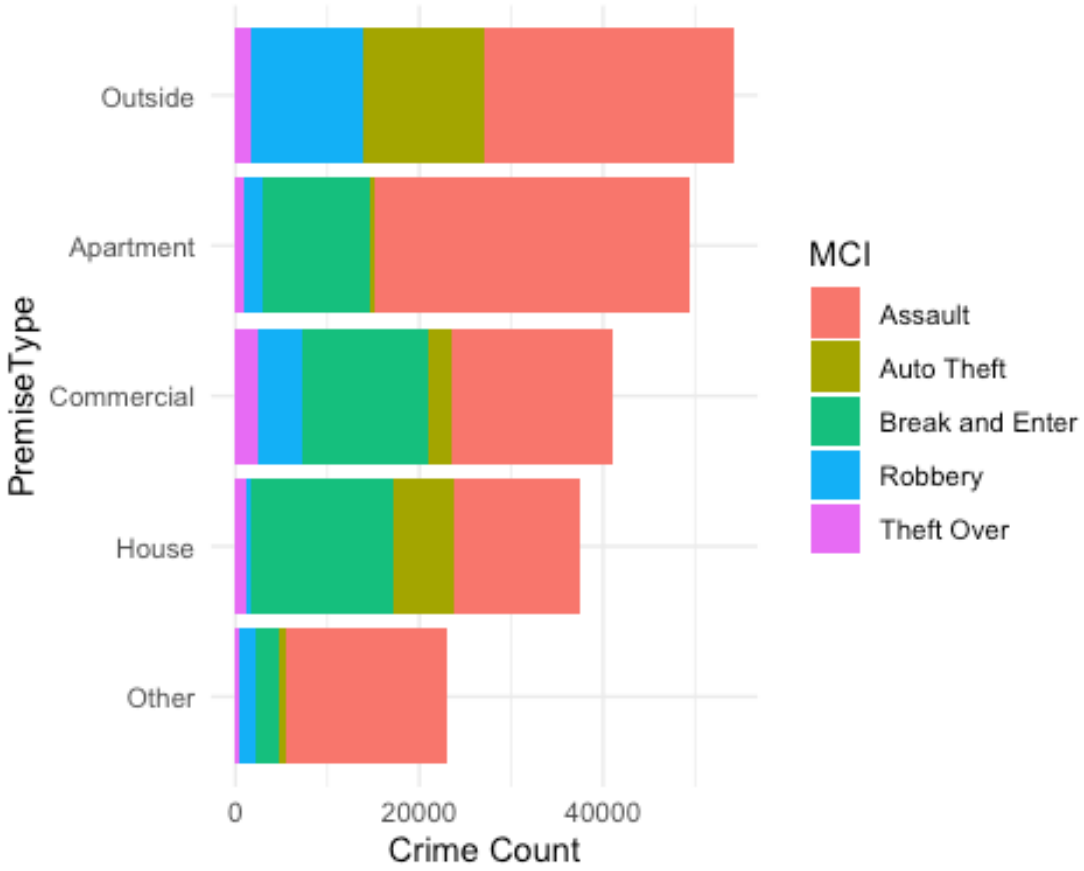


Figure 17 Crime Category by Premise Type

Top offences within the assault category include - assault with weapon, bodily harm, and assault peace officer while top offenses within robbery include - mugging, other, robbery with weapons, and robbery-business.

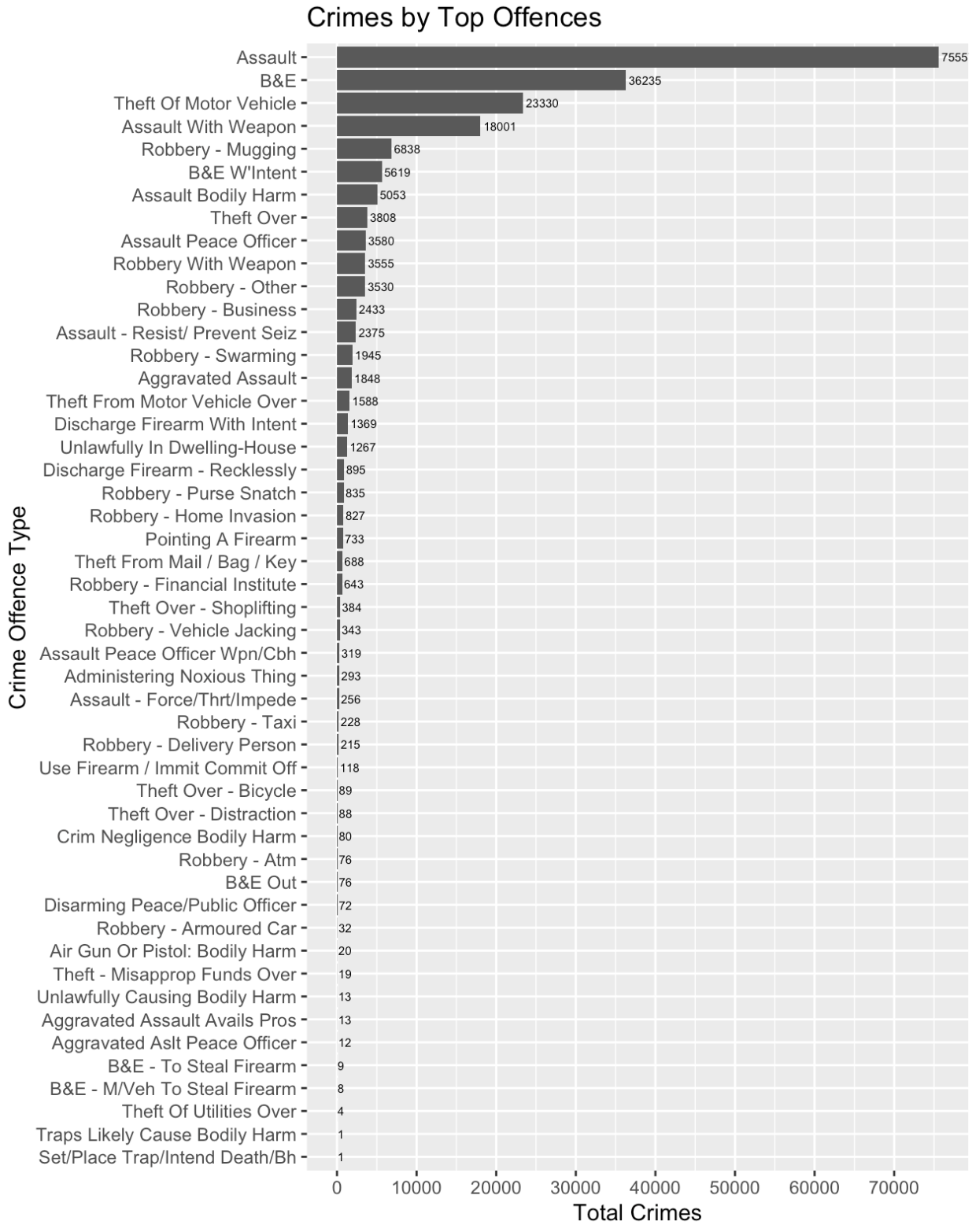


Figure 18 Top Offences

The most dangerous neighborhood is the Waterfront. The other two most dangerous hoods are the Bay Street Corridor and Yonge-Church Corridor.

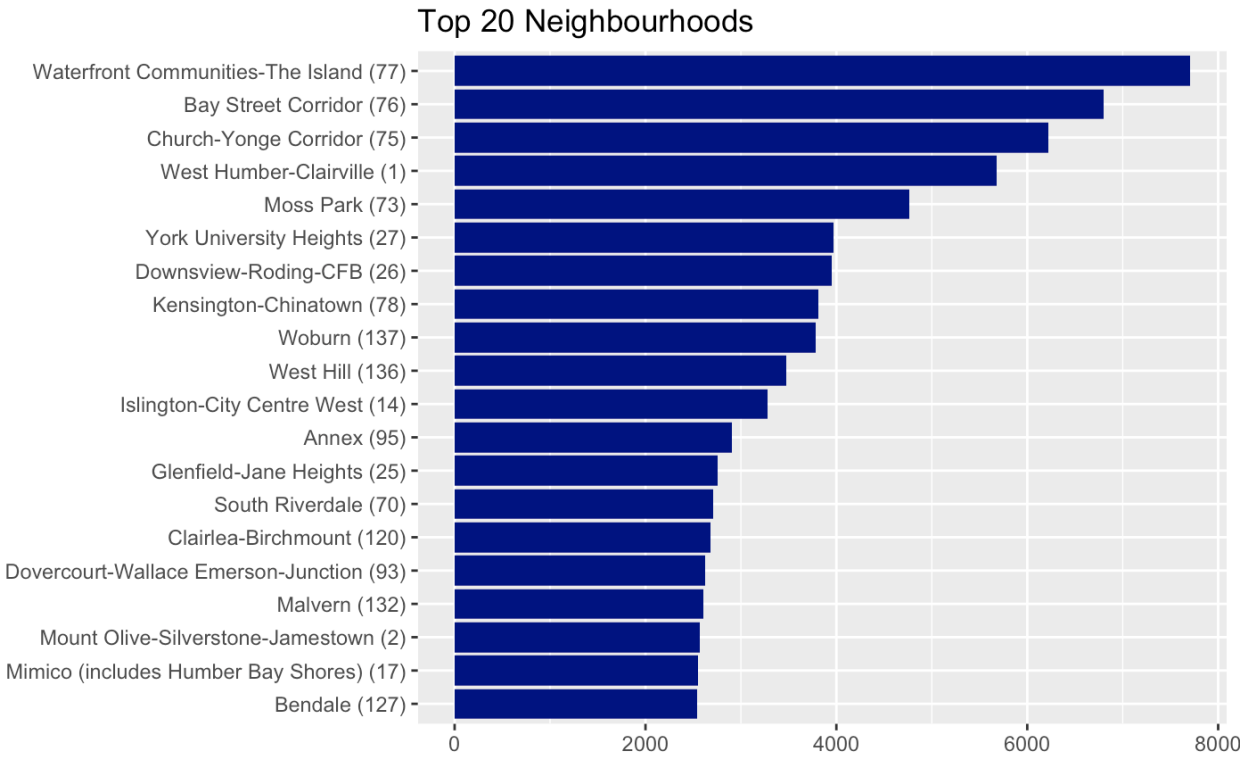


Figure 19 Top 20 Neighborhoods for Crime

The safest hoods are Lambton Baby Point, Woodbine-Lumsden, Maple Leaf, and Guildwood.

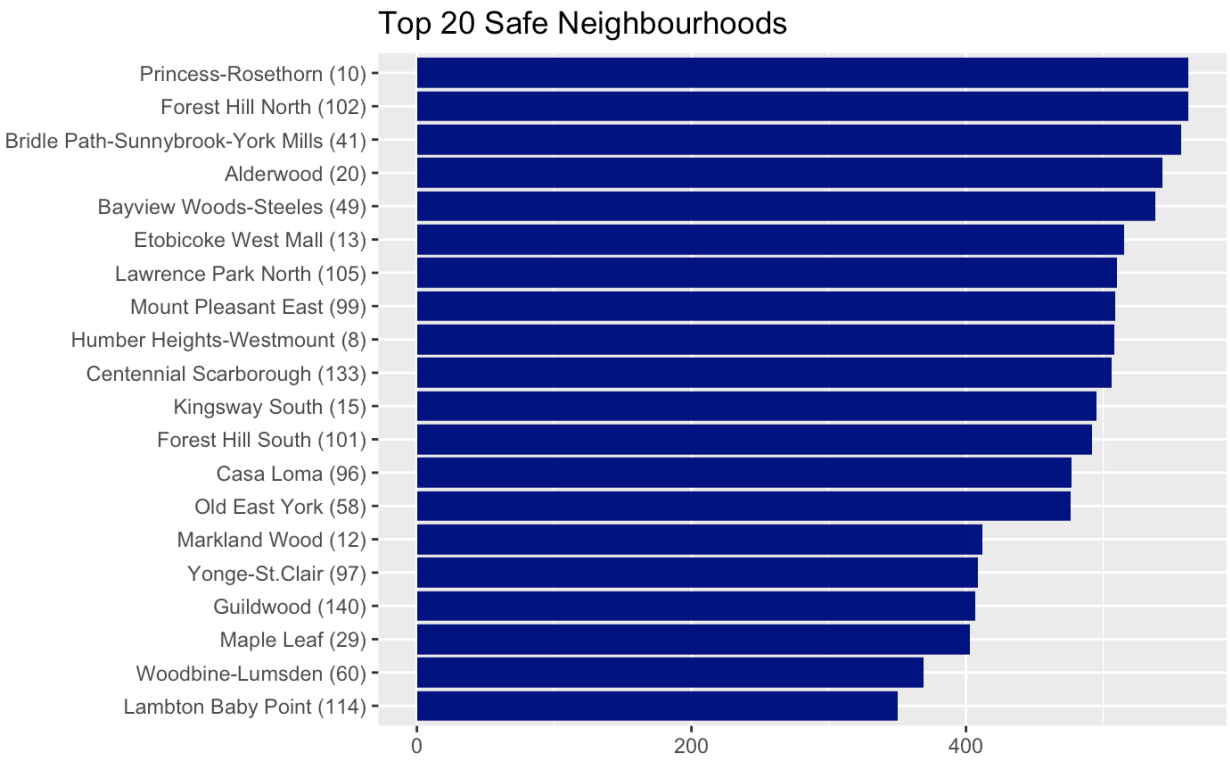


Figure 20 Top 20 Safe hoods in Toronto

Besides assaults, Bay Street Corridor, Church-Yonge Corridor and Waterfront had the most break and enter crimes while West Humber-Clairville had the most vehicle stolen crimes.

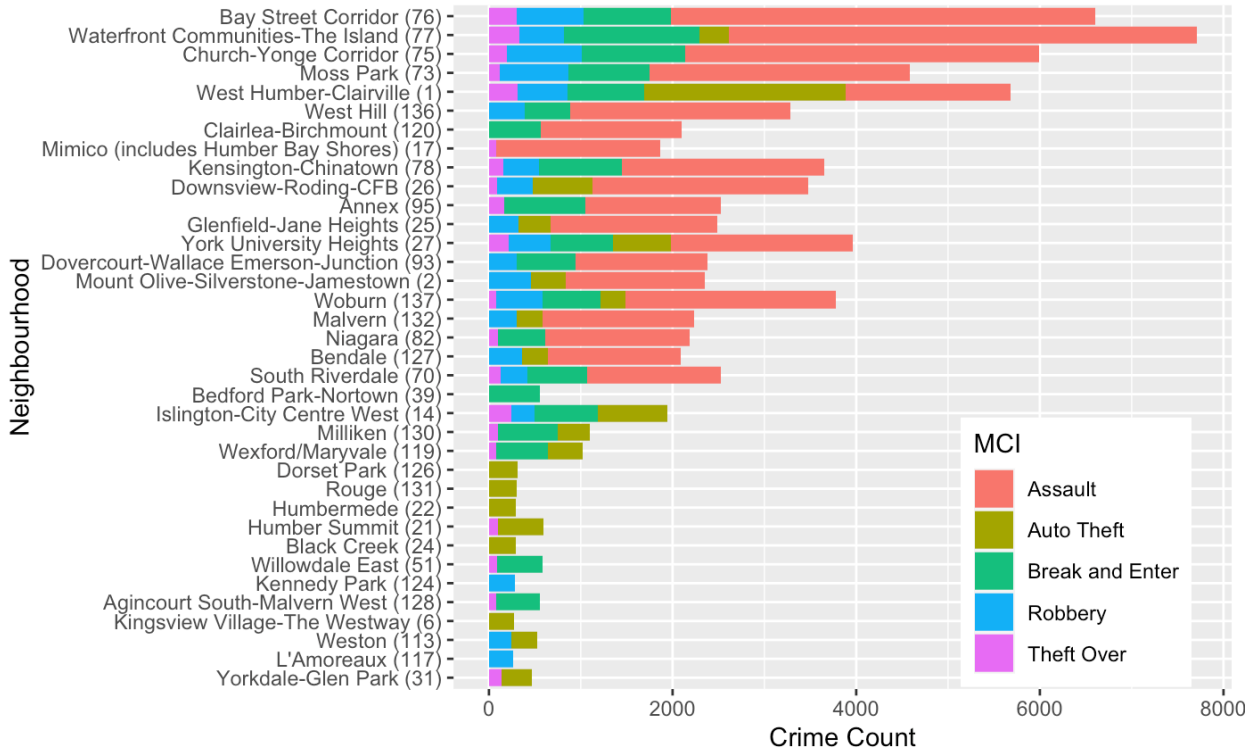


Figure 21 Top 20 Crime by Categories

Assaults, and Break and Enter occur all over the city, with a concentration in the Waterfront areas. Other crimes, such as Auto Theft has more points on the west side than the east side. Robbery and Theft Over are primarily in the Waterfront areas.

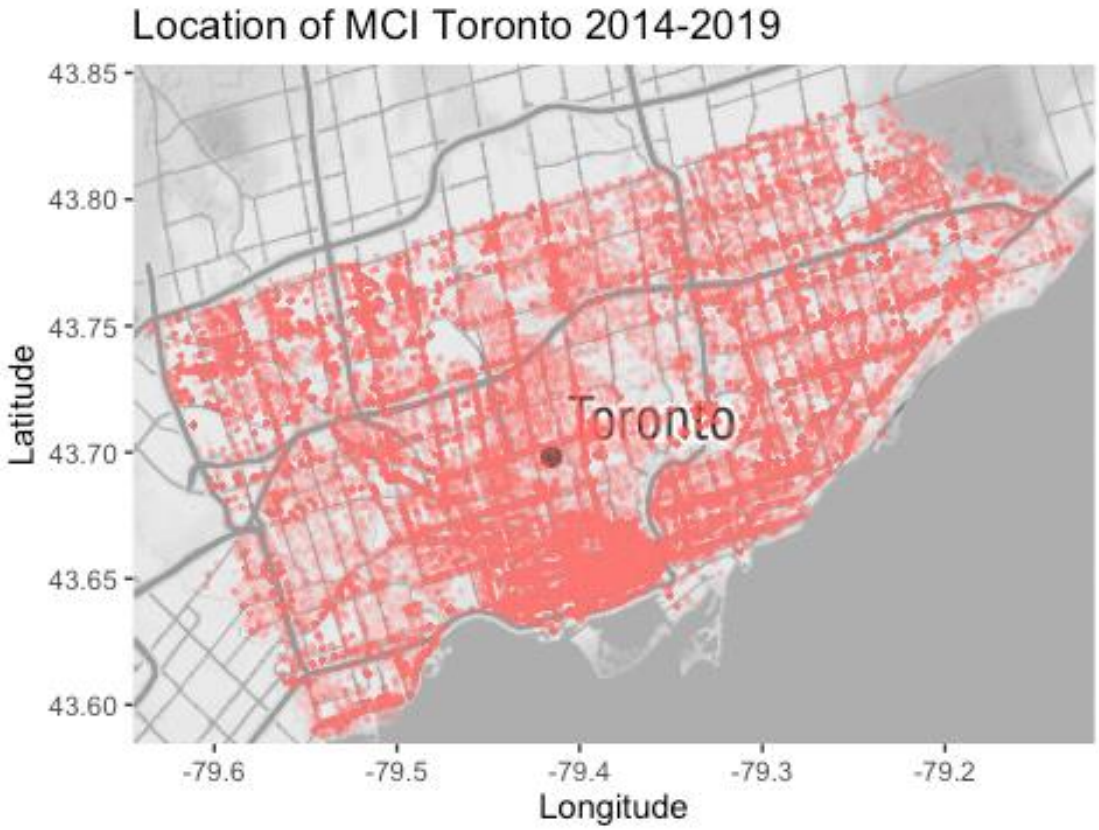


Figure 22 Map Visualization

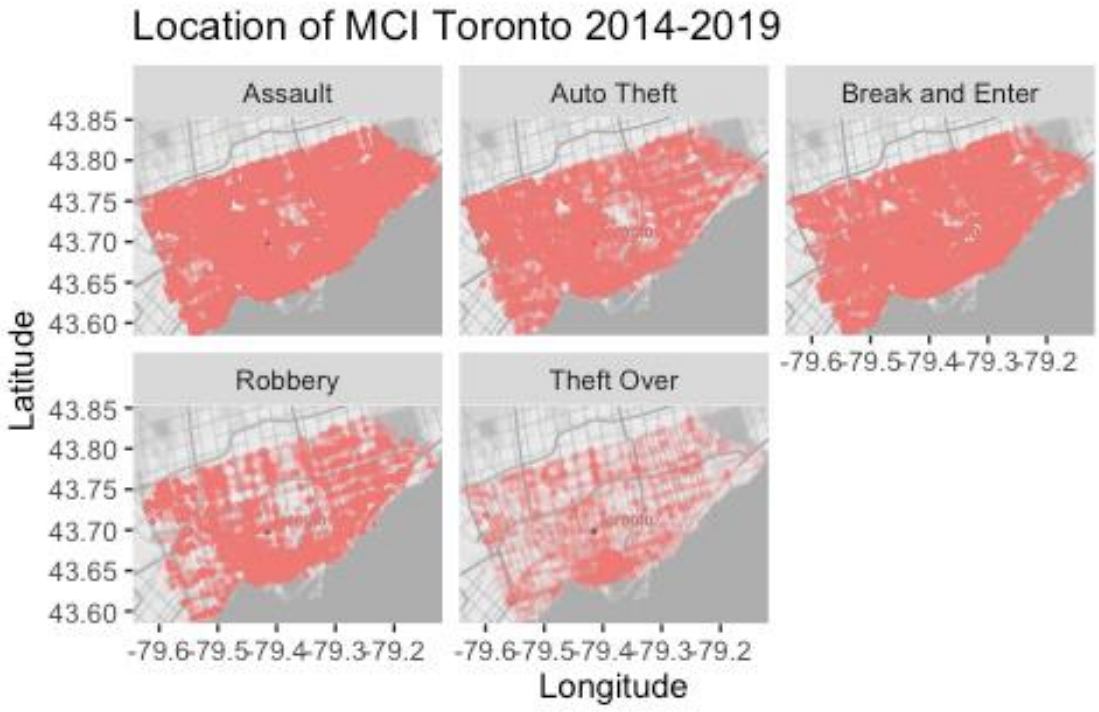


Figure 23 Location by MCI

Predictive Analysis

Feature Selection

Feature selection is a data preprocessing technique for selecting a subset of the best variables prior to constructing a model. It helps to remove irrelevant variables (i.e., variables that do not share a strong relationship with the target variable.

Although there are a wide variety of feature selection techniques i.e. filter-based, wrapper-based, and embedded families, in this project we attempt filter-based feature selection techniques to search for the best subset of variables prior to our model construction. Notably, the use of filter-based feature selection techniques is considered low cost. Of note, feature selection techniques do not necessarily mitigate correlated metrics [9].

There are many variants of filter-based feature selection techniques such as correlation-based, information gain, chi-square based and consistency based feature selection. In this project, we use information gain feature selection method which ranks variables according to the information gain with respect to the outcome.

Information gain technique is implemented by utilizing the FSelector package in R on the training dataset.

R code available here.

Feature Selection Results

Applying information gain filter based method:

Run using all variables

```
print(weights)

##                       attr_importance
## Ã¯..Index_              0.3600815277
## event_unique_id         1.2399845410
## occurrencedate          0.0022664022
## reporteddate            0.0023671202
## premisetype             0.1747254739
## ucr_code                1.2568644048
## ucr_ext                 0.8768522014
## offence                 1.2568644048
## reportedyear            0.0017143021
## reportedmonth           0.0007058475
## reportedday             0.0003986175
## reporteddayofyear       0.0007608399
## reporteddayofweek       0.0028992584
## reportedhour            0.0409393476
## occurrenceyear          0.0017017727
## occurrencemonth         0.0006335547
## occurrenceday           0.0018686390
## occurrencedayofyear     0.0013338778
## occurrencedayofweek     0.0021671882
## occurrencehour          0.0250131790
## Division                0.0293952571
## Hood_ID                 0.0491091607
## Neighbourhood           0.0543388334
## Long                    0.0374646908
## Lat                     0.0207975019
## ObjectId                0.3564296349
```

Table 2 Feature Selection Results

Manually remove variables that are not appropriate given the definitions and perform the same run again:

Data frame contains 27 variables. Drop the variables (14) that are not meaningful. This includes the following: #Index, event unique id, occurrence date and reported date in unix formats,ucr_code, ucr_ext, offence, reported year, month, day, day of year, day of week, hour, and objectid.

Applying information gain on the dataset after manually dropping these variables resulted in the following variables.

```
print(weights2)

##                      attr_importance
## premisetype             0.1737349122
## occurrenceyear          0.0018372615
## occurrencemonth         0.0006536711
## occurrenceday           0.0017255687
## occurrencedayofyear     0.0013449538
## occurrencedayofweek     0.0024220349
## occurrencehour          0.0246724036
## Division                0.0294742407
## Hood_ID                 0.0489552797
## Neighbourhood           0.0546586372
## Long                    0.0407133003
## Lat                     0.0202343335
```

Table 3 Feature Selection Final Run

Hood_id is merely an id, and the occurrence year, month, day, day of year, day of week are relatively weaker attributes compared to the others based on the importance scores.

We can, therefore, select the following variables for building the models based on the attribute importance scores. This includes: premisetype, occurrencehour, division, neighbourhood, longitude and latitude.

With these features selected, we can now attempt to predict the major crime incident categories.

SMOTE

A closer look at the distribution of crimes in the training dataset reveals the following:

```
> table(trainset.model$MCI)

      Assault      Auto Theft Break and Enter         Robbery      Theft Over
        77103           16428           30230           15137            4721

> prop.table(table(trainset.model$MCI))

  Assault      Auto Theft Break and Enter     Robbery      Theft Over
   0.5368         0.1143         0.2104       0.1053          0.0328
```

Table 4 Class Imbalance

The training dataset is heavily skewed to the 'Assault' category within MCI. Classification using class-imbalanced data is biased in favor of the majority class. Therefore, this class imbalance problem must be addressed in the training set prior to building the model.

SMOTE is an oversampling technique that generates synthetic samples from the minority class. It is used to obtain a synthetically class-balanced or nearly class-balanced training set, which is then used to train the classifier.

SMOTE was implemented in R using the UBL library to balance the classifications in the training dataset.

R code available here. (Lines 9-16 in that file contain the R code to implement SMOTE in the training dataset).

Model Selection

Since this is a classification problem, we chose to run the following algorithms in R using the RWeka package:

- Decision Tree
- KNN classification algorithm
- Naïve Bayes
- Random Forest

R code available here.

Training time Summary:

Table below shows the time taken to train the various algorithms using a 10 fold cross validation method on the training set.

10 fold cross validation	Training Time
J48 Decision Tree	60 secs
KNN Classifier	6.5 hours
Naïve Bayes	30 secs
Random Forest	65 mins

Table 5 Training Time

KNN, by far, took the longest time for training and evaluation followed by Random Forest. Naïve Bayes and Decision Tree algorithms took the shortest training time.

Model Results Summary:

Table below summarizes the overall results we were able to achieve after applying these 4 algorithms on both the training set and the test set respectively.

	Training Set		Test Set		
Algorithms	Accurac	Kappa	Accuracy	Kappa2	No Inf Rate
J48 Decision Tree	46.9%	0.3357	43.2%	0.2557	0.2637
KNN Classifier	60.4%	0.5047	48.8%	0.2807	0.3707
Naïve Bayes	40.3%	0.2532	40.0%	0.213	0.2593
Random Forest	63.7%	0.5466	51.7%	0.3275	0.351

Table 6 Model Results Summary

Ensemble classifiers perform typically better than individual classifiers i.e. random forest (homogenous ensemble) vs NB, KNN, and J48 [10] and we observed that this was true for the crime dataset as well.

Accuracy: This is the simplest scoring measure. It calculates the proportion of correctly classified instances. The Random Forest method scores better in terms of overall accuracy compared to the other methods on both the train and test sets. Training set vs test set accuracy differences for KNN and Random Forest could not be explained at this stage, however, must be investigated/addressed in future.

Kappa: The Kappa coefficient or statistic is a metric that compares an Observed Accuracy with an Expected Accuracy (random chance). It measures the reliability of a model by measuring the inter-rater agreement for qualitative items. It is an indicator of the agreement between two observers taking into account the possibility of agreement per chance. Of note, expected accuracy is defined as the accuracy that any random classifier would be expected to achieve based on the confusion matrix.

Interpreting the Kappa coefficient depends on the studied problem. In the literature there is a set of guidelines developed by Landis&Koch [11] to interpret Kappa values. We will use this convention below to interpret kappa values.

Kappa <0 – No agreement
Kappa 0-0.2 – Slight agreement
Kappa 0.21-0.4 – Fair agreement
Kappa 0.41-0.60 – Moderate agreement
Kappa 0.61-0.80 – Substantial agreement
Kappa 0.81-1 – Almost perfect agreement

Decision tree and Naïve Bayes achieved fair agreement on the Kappa metric while KNN and Random Forest achieved moderate agreement on the Kappa metric on the training set. In the test set, all algorithms achieved only fair agreement on the Kappa metric.

No Information Rate: This is the model's best guess given no information beyond the overall distribution of the classes we are trying to predict. For example, if we take all of the known classes in the test set, and just randomly guess the classes of the various records, we could get some of those right. Therefore, the No Information Rate is the proportion of classes that we could accurately guess if we randomly allocated them. Overall accuracy must be greater than the no information rate in general and this seems to be the case for all the algorithms.

RWeka output of algorithm implementations in both the training set and test sets are shown in the appendix.

Measurements for Performance Evaluation

The need for a Confusion Matrix

The confusion matrix helps to understand where the classifier is going wrong. The table below shows the confusion matrix after feeding a test set through the decision tree classifier. The examples and explanation covered here will establish the foundation for subsequent evaluation and discussion.

Decision Tree Test Set

Prediction	Assault	Auto Theft	Break and Enter	Robbery	Theft Over
Assault	12227	4043	6009	6534	4176
Auto Theft	307	3649	726	1625	729
Break and Enter	2554	1043	7192	609	1680
Robbery	886	1191	713	2968	726
Theft Over	267	333	468	348	583

Table 7 Confusion Matrix

As a simple rule, the more zeroes or smaller the numbers on all cells but the diagonal, the better the classifier is doing.

Computing Precision and Recall for Multi-Class Classification Problems

In evaluating multi-class classification problems, overall accuracy – percentage of correctly predicted labels over all predictions – of the algorithm is always not the only best measure to understand the performance of a classifier. We must compute precision and recall for each class label and analyze the individual performance on class labels or average the values to get the overall precision and recall.

Although RWeka provides both the confusion matrix and the summary statistics by Class, it is important to understand how to compute these values for a multi-class classification problem given the confusion matrix. A simple illustration is provided below to calculate these values for one of the classes in this analysis.

Precision (or positive predictive value (PPV)): Given all the predicted labels (for a given class say Assault), how many instances were correctly predicted? It refers to the percentage of results that are relevant.

Recall (or sensitivity or true positive rate (TPR))**:** For all instances that should have a label Assault, how many of these were correctly captured? It refers to the percentage of total relevant results correctly classified by the algorithm.

Recall (sensitivity) for Assault i.e. TPR = TP/TP+FN =
12227/(12227+307+2554+886+267) = 12227/16241 = 0.7528

Precision for Assault i.e. PPV = TP/TP+FP
12227/(12227+4043+6009+6534+4176) = 12227/32989 = 0.37

Both precision and recall are important model evaluation metrics and it must be noted that it is not possible to maximize both these metrics at the same time without going through a tradeoff i.e. one comes at the cost of another.

Classifier Evaluation Results on the Test Set – Summary

The table below captures the precision and recall for the various classes for each of the algorithms implemented on the test set. Of all the algorithms evaluated, the Random Forest method appears to

perform better on both precision and recall for all the classes studied within the crime category. That said, none of the algorithms seem to have good precision or recall scores for the 'Theft Over' class.

Decision Tree	Assault	Auto Theft	B & E	Robbery	Theft Over
Recall/Sensitivity	0.7528	0.35569	0.476	0.24561	0.073854
Precision/PPV	0.3706	0.51862	0.5499	0.45774	0.291646

KNN	Assault	Auto Theft	B & E	Robbery	Theft Over
Recall/Sensitivity	0.7212	0.35878	0.462	0.30356	0.077503
Precision/PPV	0.4923	0.48682	0.5365	0.45875	0.190036

Naïve Bayes	Assault	Auto Theft	B & E	Robbery	Theft Over
Recall/Sensitivity	0.7363	0.29172	0.4126	0.23047	0.06119
Precision/PPV	0.3517	0.53564	0.5031	0.34638	0.239856

Random Forest	Assault	Auto Theft	B & E	Robbery	Theft Over
Recall/Sensitivity	0.7605	0.40019	0.4942	0.3226	0.091859
Precision/PPV	0.4916	0.54318	0.6125	0.5255	0.198767

Table 8 Classifier Evaluation Summary

Conclusions

1. Assault is the most prevalent form of crime in Toronto followed by home/commercial break and enter and auto theft. All crime types have seen an increasing trend since 2014 with the exception of robbery.
2. Crime occurs the most during summer and fall (May-October). The first and the last week of any month is seen to have the most incidents. Notably, the peak observed is on the first day of every month. During weekdays, peak crime is observed at noon and then starts to gradually build from 3PM onwards and steadily increases into the night. During weekends, peak is observed at midnight. In addition, there are more crimes on Fridays and weekends than any other day in the week.
3. The most dangerous neighborhood is the Waterfront. The other two most dangerous hoods are the Bay Street Corridor and Yonge-Church Corridor. The safest hoods are Lambton Baby Point, Woodbine-Lumsden, Maple Leaf, and Guildwood.
4. Of all the algorithms evaluated and studied in this project, the Random Forest method shows the most promise and offers potential to predict crime in Toronto. This method also performs better on all evaluation measures such as overall accuracy, kappa as well as both precision and recall for all the classes studied within the crime category.
5. The results from this analysis could be used to elevate people's awareness regarding the dangerous locations in Toronto and attempt to help the Toronto Police Service to predict future crimes in a specific location within a particular time.
6. Overall, the capstone project report publication could start a trend of crimes prediction whereby we can anticipate that the law enforcing agencies throughout Canada can start to take

advantage of machine learning algorithms to effectively fight crime to keep our country safer for everyone.

Future Work

- In this project, we only attempted the information gain filter-based technique for feature selection. In future, we could not only try the other filter-based methods but also the wrapper-based and embedded techniques to see the features selected for this data set and analyze the performance of these algorithms.
- We balanced the dataset using SMOTE. In future, we could try under sampling methods and oversampling methods and see how these algorithms perform on the dataset.
- None of the algorithms studied had good precision or recall scores for the minority class within the crime category i.e. 'Theft Over' class. This must be investigated in future.
- Of all the algorithms evaluated, Random Forest observed the highest differences or gap in accuracy between the training set and testing set. Though the results are not provided here, testing and running the algorithm without balancing the classes i.e. running the algorithm on the imbalanced dataset and dropping the division variable we were able to close the gaps in accuracy between the training and test sets. That said, there still exists an opportunity to investigate this formally in future.
- Lastly, there exists an opportunity to secure conclusions by means of formal statistical testing procedures. This would be the application of nonparametric testing including the combination of a Friedman test with post-hoc test to compare the classifiers used in the project and statistically finalize the selection of the best performer.

References

[1] Y. L. Lin, L. C. Yu, and T. Y. Chen, "Using machine learning to assist crime prevention," IEEE 6th Intl. Congr on Advanced Appl. Inform. (IIAIAAI), Hamamatsu, Japan, Jul. 2017.

[2] Bogomolov, B. Lepri, J. Staiano, N. Oliver, F. Pianesi, and A. Pentland, "Once upon a crime: towards crime prediction from demographics and mobile data," Proc. of the 16th Intl. Conf. On Multimodal Interaction, pp. 427-434, 2014.

[3] V. Grover, R. Adderley, and M. Bramer, "Review of current crime prediction techniques," Intl. Conf. on Innovative Techn and Appl. Of Artificial Intel. pp. 233-237, Springer, London, 2007.

[4] R. Iqbal, M. A. A. Murad, A. Mustapha, P. H. Shariat Panahy, and N.Khanahmadliravi, "An experimental study of classification algorithms for crime prediction," Indian J. of Sci. and Technol., vol. 6, no. 3, pp. 4219-4225, Mar. 2013.

[5] L. McClendon and N. Meghanathan, "Using machine learning algorithms to analyze crime data," Mach. Learn and Appl.: an Intl. J. (MLAIJ), vol.2, no.1, Mar. 2015.

[6] H. W. Kang, H. B. Kang, "Prediction of crime occurrence from multimodal data using deep

learning," PLoS ONE, vol. 12, no. 4, Apr. 2017.
[7] M. V. Barnadas, Machine learning applied to crime prediction, Thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, Sep 2016.
[8] Kim, Suhong, Param Joshi, Parminder Singh Kalsi, and Pooya Taheri. "Crime Analysis through Machine Learning." In 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), pp. 415-420. IEEE, 2018.
[9] Jiarpakdee, Jirayus & Tantithamthavorn, Chakkrit & Treude, Christoph. (2018). AutoSpearman: Automatically Mitigating Correlated Metrics for Interpreting Defect Models.
[10] Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. European Journal of Operational Research, 247(1)
[11] Landis, J.R.; Koch, G.G. (1977). "The measurement of observer agreement for categorical data". Biometrics. 33 (1): 159–174. DOI: 10.2307/2529310. JSTOR 2529310. PMID 843571.

Appendix

Decision Tree:

Trainset

```
> evaluate_Weka_classifier (dtree.J48, numFolds = 10)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances       67288               46.8521 %
Incorrectly Classified Instances     76330               53.1479 %
Kappa statistic                          0.3357
Mean absolute error                      0.2632
Root mean squared error                  0.3647
Relative absolute error                 82.2584 %
Root relative squared error             91.1658 %
Total Number of Instances           143618

=== Confusion Matrix ===

   a    b    c    d    e   <-- classified as
 10762  3755  5641  4969  3597 |    a = Assault
  1200 15695  3348  5642  2839 |    b = Auto Theft
  5226  2192 16053  1326  3927 |    c = Break and Enter
  3931  5419  2810 12978  3585 |    d = Robbery
  3180  4511  5627  3605 11800 |    e = Theft Over

Number of Leaves:       989

Size of the tree:       1239
```

Test Set

```
> confusionMatrix (testset.model$MCI, dtree.pred)
Confusion Matrix and Statistics

                Reference
Prediction        Assault Auto Theft Break and Enter Robbery Theft Over
  Assault           12227       4043           6009    6534       4176
  Auto Theft          307       3649            726    1625        729
  Break and Enter    2554       1043           7192     609       1680
  Robbery             886       1191            713    2968        726
  Theft Over          267        333            468     348        583

Overall Statistics

               Accuracy : 0.4322
                 95% CI : (0.4283, 0.4361)
    No Information Rate : 0.2637
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2557

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: Assault Class: Auto Theft Class: Break and Enter Class: Robbery
Sensitivity                  0.7528          0.35569               0.4760       0.24561
Specificity                  0.5421          0.93401               0.8734       0.92897
Pos Pred Value               0.3706          0.51862               0.5499       0.45774
Neg Pred Value               0.8596          0.87883               0.8368       0.83456
Prevalence                   0.2637          0.16658               0.2453       0.19621
Detection Rate               0.1985          0.05925               0.1168       0.04819
Detection Prevalence         0.5357          0.11425               0.2124       0.10528
Balanced Accuracy            0.6475          0.64485               0.6747       0.58729
                     Class: Theft Over
Sensitivity                 0.073854
Specificity                 0.973627
Pos Pred Value              0.291646
Neg Pred Value              0.877305
Prevalence                  0.128178
Detection Rate              0.009466
Detection Prevalence        0.032459
```

Balanced Accuracy 0.523740

KNN Classifier:

Trainset

```
> evaluate_Weka_classifier (knn.classifier, numFolds = 10)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances       86707               60.3734 %
Incorrectly Classified Instances     56911               39.6266 %
Kappa statistic                          0.5047
Mean absolute error                      0.16
Root mean squared error                  0.3887
Relative absolute error                 49.998  %
Root relative squared error             97.1649 %
Total Number of Instances           143618

=== Confusion Matrix ===

    a     b     c     d     e   <-- classified as
13739  3223  5458  4023  2281 |     a = Assault
 3010 19489  1962  2560  1703 |     b = Auto Theft
 6541  2464 15234  1855  2630 |     c = Break and Enter
 4472  2638  1667 18423  1523 |     d = Robbery
 2529  1991  2706  1675 19822 |     e = Theft Over
```

Test Set

```
> confusionMatrix(testset.model$MCI, knn.pred)
Confusion Matrix and Statistics

                Reference
Prediction        Assault Auto Theft Break and Enter Robbery Theft Over
  Assault           16494       3631            6296    4729       2356
  Auto Theft         1272       3360             763     965        542
  Break and Enter    2988       1093            6966     780       1157
  Robbery            1608        950             537    2919        349
  Theft Over          508        331             515     223        370

Overall Statistics

               Accuracy : 0.488
                 95% CI : (0.484, 0.4919)
    No Information Rate : 0.3707
    P-Value [Acc > NIR] : < 2.2e-16
```

Kappa : 0.2807

Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

	Class: Assault	Class: Auto Theft	Class: Break and Enter	Class: Robbery
Sensitivity	0.7212	0.35878	0.4620	0.30356
Specificity	0.5619	0.93232	0.8709	0.93388
Pos Pred Value	0.4923	0.48682	0.5365	0.45875
Neg Pred Value	0.7739	0.89042	0.8335	0.87898
Prevalence	0.3707	0.15178	0.2444	0.15585
Detection Rate	0.2673	0.05446	0.1129	0.04731
Detection Prevalence	0.5430	0.11186	0.2104	0.10312
Balanced Accuracy	0.6416	0.64555	0.6665	0.61872

	Class: Theft Over
Sensitivity	0.077503
Specificity	0.972298
Pos Pred Value	0.190036
Neg Pred Value	0.926299
Prevalence	0.077372
Detection Rate	0.005997
Detection Prevalence	0.031555
Balanced Accuracy	0.524901

Naive Bayes:

Trainset

```
> evaluate_Weka_classifier(b.classifier, numFolds = 10)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances       57813               40.2547 %
Incorrectly Classified Instances     85805               59.7453 %
Kappa statistic                          0.2532
Mean absolute error                      0.272
Root mean squared error                  0.3819
Relative absolute error                 85.0135 %
Root relative squared error             95.4634 %
Total Number of Instances           143618

=== Confusion Matrix ===

    a     b     c     d     e   <-- classified as
10110  4289  6196  4373  3756 |     a = Assault
 1035 15577  3960  5909  2243 |     b = Auto Theft
 5927  3710 14503  1202  3382 |     c = Break and Enter
```

4036 7826 3007 10136 3718 | d = Robbery
3828 6527 7275 3606 7487 | e = Theft Over

Testset

```
> confusionMatrix(testset.model2$MCI, b.pred2)
Confusion Matrix and Statistics

                Reference
Prediction        Assault Auto Theft Break and Enter Robbery Theft Over
  Assault         11783       5085           7113    5188       4337
  Auto Theft        299       3697            976    1390        540
  Break and Enter  2700       1704           6532     524       1524
  Robbery           924       1776            695    2204        764
  Theft Over        296        411            516     257        467

Overall Statistics

               Accuracy : 0.4
                 95% CI : (0.3962, 0.4039)
    No Information Rate : 0.2593
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.213

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: Assault Class: Auto Theft Class: Break and Enter Class: Robbery
Sensitivity                  0.7363          0.29172                0.4126        0.23047
Specificity                  0.5247          0.93463                0.8593        0.92023
Pos Pred Value               0.3517          0.53564                0.5031        0.34638
Neg Pred Value               0.8504          0.83620                0.8091        0.86702
Prevalence                   0.2593          0.20539                0.2566        0.15499
Detection Rate               0.1910          0.05992                0.1059        0.03572
Detection Prevalence         0.5430          0.11186                0.2104        0.10312
Balanced Accuracy            0.6305          0.61318                0.6360        0.57535
                     Class: Theft Over
Sensitivity                 0.061190
Specificity                 0.972628
Pos Pred Value              0.239856
Neg Pred Value              0.880094
Prevalence                  0.123691
Detection Rate              0.007569
Detection Prevalence        0.031555
Balanced Accuracy           0.516909
```

Random Forest:

Trainset

```
> evaluate_Weka_classifier (rf.classifier, numFolds = 10)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances       91524               63.7274 %
Incorrectly Classified Instances     52094               36.2726 %
Kappa statistic                          0.5466
Mean absolute error                      0.1809
Root mean squared error                  0.3168
Relative absolute error                 56.5404 %
Root relative squared error             79.1904 %
Total Number of Instances           143618

=== Confusion Matrix ===

     a     b     c     d     e   <-- classified as
 13839  3106  5535  4302  1942 |     a = Assault
  2258 19733  2130  2853  1750 |     b = Auto Theft
  5274  2105 17496  1589  2260 |     c = Break and Enter
  3811  2747  1716 19026  1423 |     d = Robbery
  1644  1807  2400  1442 21430 |     e = Theft Over
```

Testset

```
> confusionMatrix(testset.model$MCI, rf.pred)
Confusion Matrix and Statistics

                Reference
Prediction        Assault Auto Theft Break and Enter Robbery Theft Over
  Assault         16473      3469          6351    5154      2059
  Auto Theft        953      3749           744     953       503
  Break and Enter  2430       939          7953     688       974
  Robbery          1365       873           491    3344       290
  Theft Over        439       338           555     228       387

Overall Statistics

               Accuracy : 0.5171
                 95% CI : (0.5131, 0.521)
    No Information Rate : 0.351
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3275

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: Assault Class: Auto Theft Class: Break and Enter Class: Robbery
Sensitivity                  0.7605          0.40019               0.4942       0.3226
Specificity                  0.5746          0.93975               0.8897       0.9412
```

Pos Pred Value	0.4916	0.54318	0.6125	0.5255
Neg Pred Value	0.8160	0.89746	0.8329	0.8731
Prevalence	0.3510	0.15183	0.2608	0.1680
Detection Rate	0.2670	0.06076	0.1289	0.0542
Detection Prevalence	0.5430	0.11186	0.2104	0.1031
Balanced Accuracy	0.6676	0.66997	0.6919	0.6319

Class: Theft Over	
Sensitivity	0.091859
Specificity	0.972864
Pos Pred Value	0.198767
Neg Pred Value	0.935972
Prevalence	0.068280
Detection Rate	0.006272
Detection Prevalence	0.031555
Balanced Accuracy	0.532361