Crimes of London: Perspective Analysis on Crimes within England's Capital

Mackenzie Ripari

7/22/2021

Abstract

Crime occurs throughout the world. As such the ability to predict the occurrence or instances of crimes would redoubtably be of worth to discover. Using crimes in the city of London dataset, aptly called london_crimes, this report aims to predict crimes with an accuracy of 70%. The creation of a model to predict the occurrence of crime is ultimately developed after exploration and minimal algorithm production of a recommendation model that does not quite work well enough. Effects within the dataset are also explored, such as the occurrence of crime and the prevalence of location effecting not just crime report rates but their being effected on by other factors. Once done a look at various other models is done, with a final model being developed as a result of the best predicting model looked at. With said final model final predictions are made and the goal of a minimal 70% accurate model is completed. Afterward a discussion on improvements and afterthoughts is done along with a final closing on this report.

1.0 Introduction

1.1 Preliminary Outline

Crime exists. Defined by Cornell Law as "any act or omission in violation of a law prohibiting the action or omission" crime has an impact on society at large (Cornell, 2021). It would stand to reason then that crime as a whole is documented and analyzed in order to reduce rates or see why it occurs. Indeed, there are a multitude of resources detailing the various counts and records of crime around the world. One such record details the crime within London from the years 2008 to 2016. This record, aptly named London Crimes, holds a plethora of information that can be used to analyze crime not only to see incidence but also to see where crime occurs. This can then be used to predict crimes.

This report will showcase a multitude of methods in the prediction of crime rates. Primarily, this will be shown using a method unsuited towards this type of prediction and a set of methods suited towards the prediction. As a method of motivation, this report will have a goal of creating a method that predicts with at minimum 70% accuracy crime occurrences within a given month. To begin there will be an introduction to the dataset itself; background on column meaning and in general what the dataset entails. From there there will be a breakdown on various exploratory and analysis done to see trends and importance within the dataset. Moving on from there a breakdown on the three methods earlier will be shown. From there the best method will be utilized with a portioned off set of the London Crimes dataset to predict crime occurrences. To end there will be a recap of the report as a whole, summarizing key points within the data, before ending with final thoughts and improvement opportunities.

Before moving onward to the description portion of this report the required libraries, repositories of code and functionality, and memory size increase to handle code throughout this report need to be loaded.

```r
##INCREASES MEMORY LIMIT WITHIN R
memory.limit(size=100000)
```

```
## [1] 1e+05
```

1.2 London Crimes: Dataset Description

Before this report documents its exploration of the London Crime dataset some explanation of pertinent background is required. To begin this London Crimes dataset, this record obtained via Kaggle, holds information on the monthly incidence of crimes within London and its surrounding boroughs (Boysen, 2017). For a breakdown of columns within this dataset before wrangling see below.

1. lsoa_code: code for Lower Super Output Area in Greater London. For those unfamiliar with this terminology think of it as akin to cenus data denoting smaller population areas.
2. borough: Common name for London borough. Akin in many ways to a city/district within a city.
3. major_category: High level categorization of crime. For instance Murder and Theft would be two
4. minor_category: Low level categorization of crime within major category. Examples include stealing vehicles or attempted assault.
5. value: monthly reported count of categorical crime in given borough. Of note here is that there may in fact be no reported crime within a given month, giving a value of 0.
6. year: Year that the value was given ranging from 2008 to 2016.
7. month: Month that the value was given in numeric form, meaning from 1 to 12.

As noted above this is before wrangling occurs. Due to hardware limitations this report necessitates the sampling of data and selecting of what data to analyze. Before moving onto that however one final thing to note about this report. As this report is documenting the reported counts of crimes as a monthly value it is likely that more unreported crimes occurred during the various time-frames than were reported. As mentioned earlier the goal of this report concerns itself with a method of at least 70% accuracy. While unable to accounted for undocumented data this report will strive to utilize what data it can to the best of its ability.

2.0 London Crimes Dataset Overview:

2.2 Dataset Wrangling

Mentioned above hardware limitations necessitate some wrangling that occurs within this dataset. Of note this report will concern itself with crime from the year 2010 onward. Moreover due to the incidence rates of certain values, which will be explained below, a categorical column of crime Occurrence will be added, noting if a crime occurred with either a "Yes" or "No" value. The months column will also be converted to an abbreviated name for convenience as well. Finally due to the size of the data only 30,000 rows of data will be usable to create the model due to vector size limitations which will be discussed later. Altogether there are now 8 columns within the dataset shown below with a glimpse of how the dataset is setup.

One final note: the creation of this selection was done twofold. Firstly, while only sampling 1% of the data may seem small the overall size of the data makes it so that this report feels confident in gleaming information from said sample. Besides this this report needs a manageable size of data to upload to a githhub repository.

1. lsoa_code: code for Lower Super Output Area in Greater London. For those unfamiliar with this terminology think of it as akin to census data denoting smaller population areas.
2. borough: Common name for London borough. Akin in many ways to a city/district within a city.
3. major_category: High level categorization of crime. For instance Murder and Theft would be two
4. minor_category: Low level categorization of crime within major category. Examples include stealing vehicles or attempted assault.
5. value: monthly reported count of categorical crime in given borough. Of note here is that there may in fact be no reported crime within a given month, giving a value of 0.
6. year: Year that the value was given ranging from 2012 to 2016.
7. month: Month that the value was given in an abbreviated month format.
8. crime_occurred: Categorical column determining whether a crime occurred in the specified location at the specified time.

The above code is meant to demonstrate how the used csv file loaded below was created via the code acquired online. For the purposes of this report lonond_crime_curated is ultimately what this report will be concerned with analyzing. Also, for demonstrative purposes, below is a small preview of the data within it.

```
##   lsoa_code                borough             major_category
## 1 E01003031                Lambeth Violence Against the Person
## 2 E01004197          Tower Hamlets          Theft and Handling
## 3 E01001959 Hammersmith and Fulham          Theft and Handling
## 4 E01003344               Lewisham Violence Against the Person
## 5 E01004701            Westminster          Theft and Handling
##                    minor_category value year month crime_occurred
## 1               Offensive Weapon     0 2016   Aug             No
## 2                    Other Theft     6 2013   Jun            Yes
## 3 Theft/Taking Of Motor Vehicle     1 2014   Dec            Yes
## 4            Assault with Injury     0 2014   Apr             No
## 5   Theft/Taking of Pedal Cycle     0 2015   Jan             No
```

With the above done this report can now partition the data into two parts, a final test set for use in determining a model with an accuracy of 70% and a training set for developing the multitude of methods. These sets, final_test and london respectively, come from the london crimes dataset and are split 10% for the final_test set and 90% for the london set along with a final check to ensure data within the final_test may occur within the larger london set. With this done this report can now move onward to the exploration of the data within it.

```
#sample 1% of the data for use


#set aside 10% of data for final validation and a set for training
#set seed to ensure replicability
set.seed(2021, sample.kind="Rounding")

#split for a final test set and observation set, henceforth london
test_index <- createDataPartition(y = london_crimes$value, times = 1, p = 0.1, list = FALSE)
london <- london_crimes[-test_index,]
temp <- london_crimes[test_index,]

# Make sure following are found in both final test and london
#borough major and minor cat year month crime occur value
final_test <- temp %>%
  semi_join(london, by = "borough")%>%
  semi_join(london, by = "major_category")%>%
  semi_join(london, by = "lsoa_code")%>%
  semi_join(london, by = "minor_category")%>%
  semi_join(london, by = "year")%>%
  semi_join(london, by = "month")%>%
  semi_join(london, by = "value")%>%
  semi_join(london, by = "crime_occurred")


# Add rows removed from final set back into london
removed <- anti_join(temp, final_test)
london <- rbind(london, removed)

##cleanup
rm(removed, temp, test_index, london_crimes)
```

2.3 Exploration of London Crimes

To recap here is a glimpse at the first 5 records of the london dataset, which will be used to train and create the upcoming models in this report.

```
#head of data for quick overview
head(london, 5)
```

```
##   lsoa_code                borough             major_category
## 1 E01003031                Lambeth Violence Against the Person
## 2 E01004197          Tower Hamlets          Theft and Handling
## 3 E01001959 Hammersmith and Fulham          Theft and Handling
## 4 E01003344               Lewisham Violence Against the Person
## 6 E01001619              Greenwich          Theft and Handling
##                   minor_category value year month crime_occurred
## 1              Offensive Weapon     0 2016   Aug             No
## 2                   Other Theft     6 2013   Jun            Yes
## 3 Theft/Taking Of Motor Vehicle     1 2014   Dec            Yes
## 4             Assault with Injury     0 2014   Apr             No
## 6          Handling Stolen Goods     0 2015   Apr             No
```

With this in mind here is a breakdown of unqiue values within each column of the dataset to see the general nuance within the data. For the sake of space only the first 5 of each category will be shown though.

```
#now check distinct values for specific cat data
# lsoa code, borough, major cat, minor cat
n_distinct(london$lsoa_code)
```

```
## [1] 4826
```

```
#districts distinct and names thereof
n_distinct(london$borough)
```

```
## [1] 33
```

```
head(distinct(london[2]),5)
```

```
##                  borough
## 1                Lambeth
## 2          Tower Hamlets
## 3 Hammersmith and Fulham
## 4               Lewisham
## 6              Greenwich
```

```
#now breakdown of major cat
n_distinct(london$major_category)
```

```
## [1] 9
```

```
head(distinct(london[3]),5)
```

```
##                 major_category
## 1  Violence Against the Person
## 2           Theft and Handling
## 10                     Robbery
## 13   Other Notifiable Offences
## 19             Criminal Damage
```

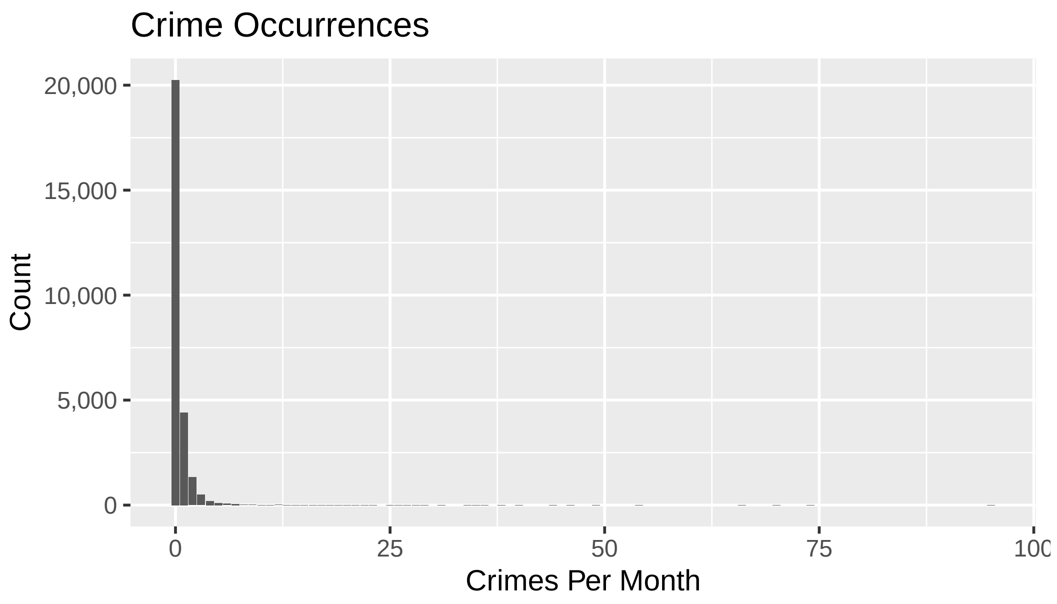
```
#minor cat breakdown
n_distinct(london$minor_category)
```

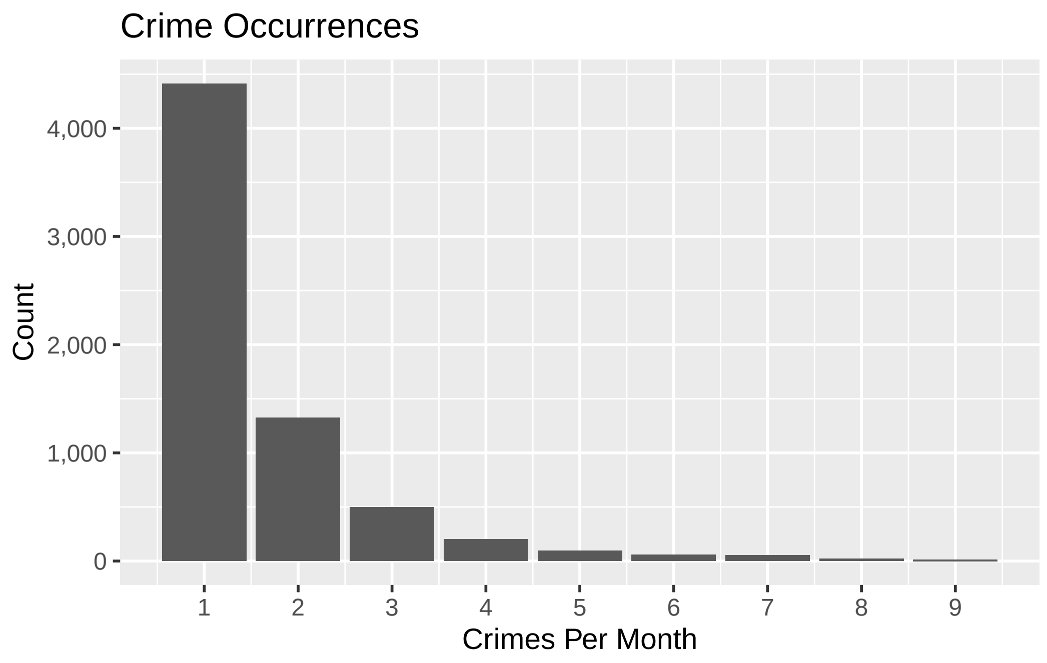
```
## [1] 32
```

```
head(distinct(london[4]),5)
```

```
##                  minor_category
## 1              Offensive Weapon
## 2                   Other Theft
## 3 Theft/Taking Of Motor Vehicle
## 4            Assault with Injury
## 6           Handling Stolen Goods
```

As shown above there is quite a bit of depth to the data. Also of note is the values column, which notes the occurrence of a crime within a month. Below is a side by side of two plots depicting the occurrence of crimes by the values per month, one including all types of occurrences and the other filtering the values from 1 to 10.



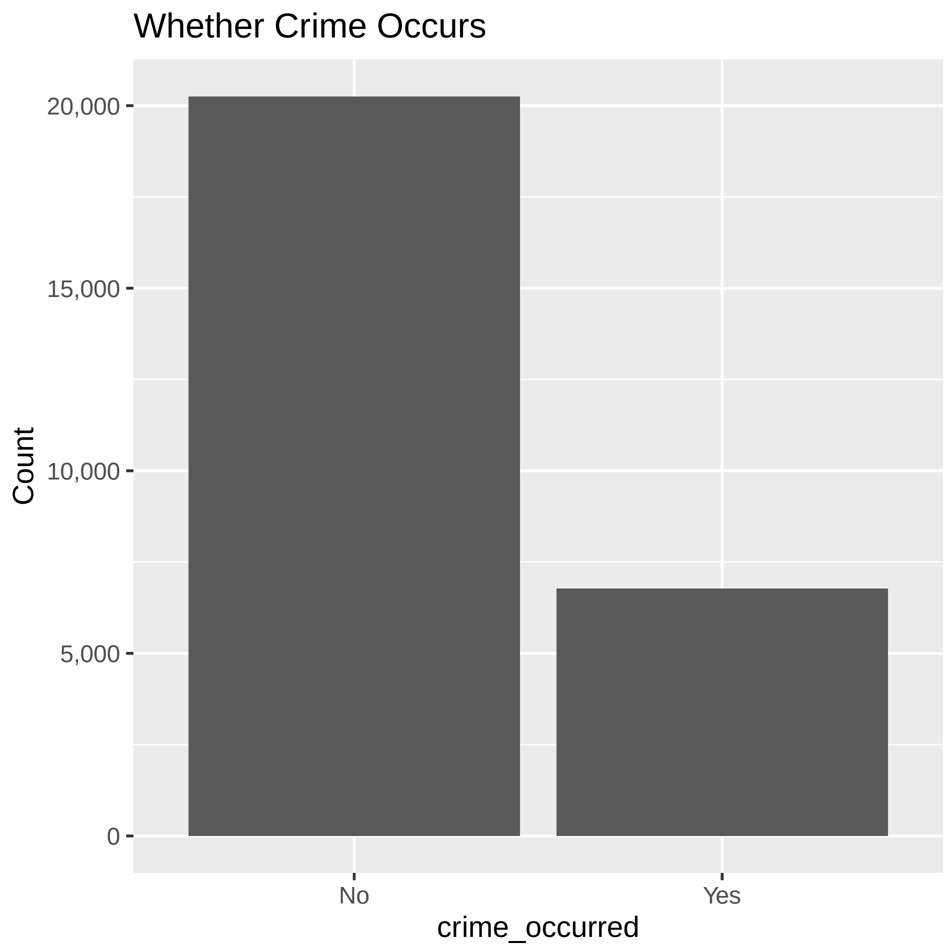


For a more numeric view of the data below is a short table showcasing values 0 to 9. Of note here is the prominence of the values 0, 1, and 2.

```
#more meaningful as summarize
london %>% group_by(value) %>%
  summarise(n=n()) %>% head(10)
```

```
## # A tibble: 10 x 2
##    value     n
##    <int> <int>
##  1     0 20245
##  2     1  4413
##  3     2  1325
##  4     3   496
##  5     4   201
##  6     5    96
##  7     6    58
##  8     7    52
##  9     8    23
## 10     9    14
```

As seen above there is a prominent skew of the data towards no crimes being reported within a month. As such it may be prudent to view the data in categorical terms of a binary yes or no response. Shown below is a short plot of such, with the incidence of whether crime occurs with a more numeric viewpoint provided as well.

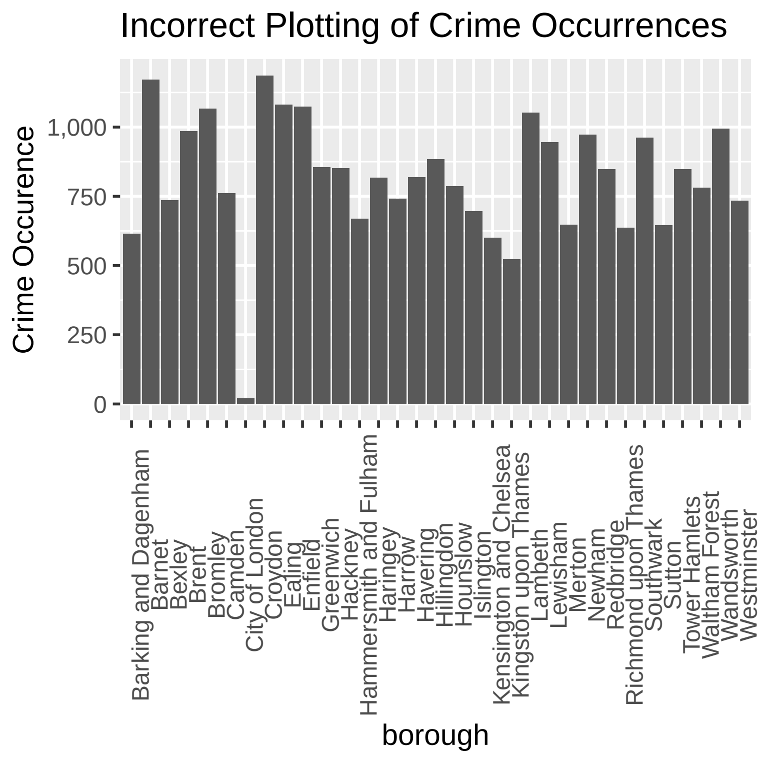


```
## `summarise()` ungrouping output (override with `.groups` argument)
```

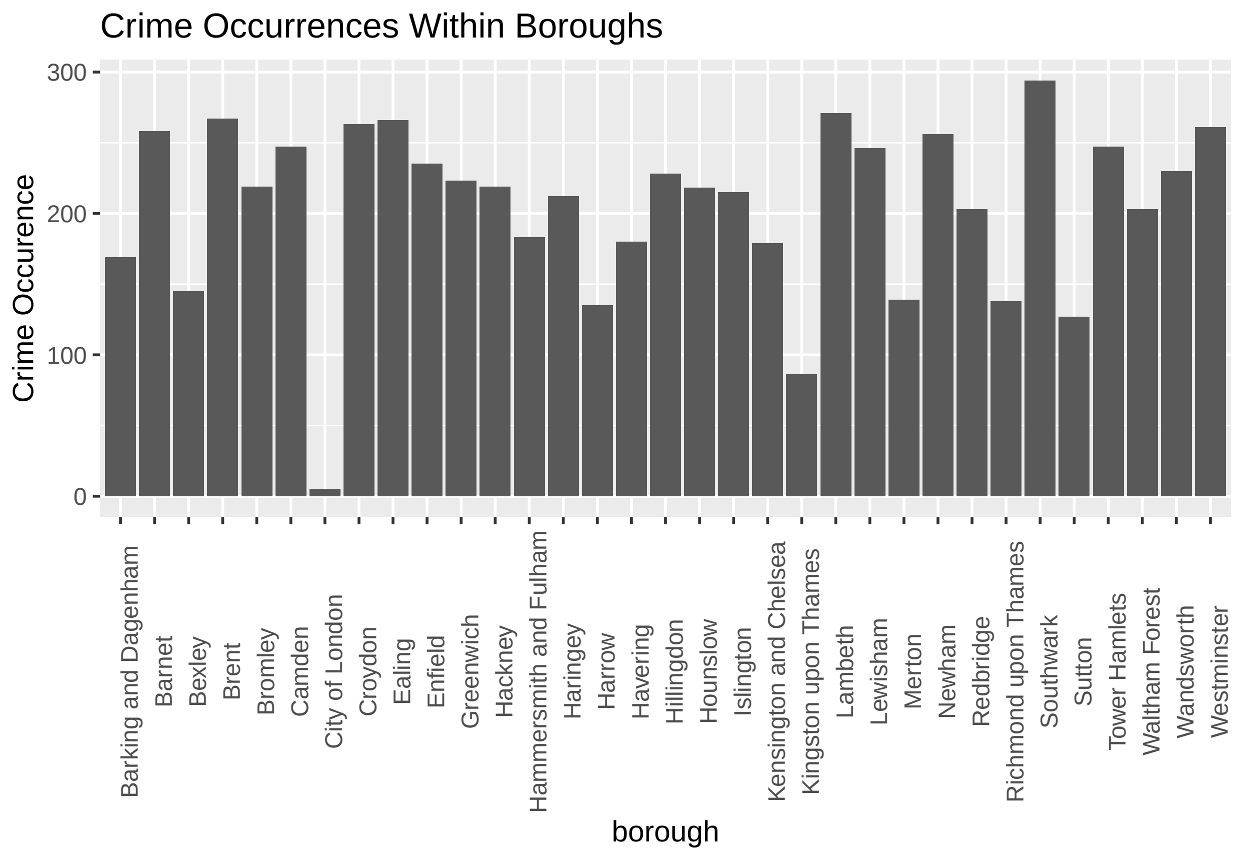
```
## # A tibble: 2 x 2
##   crime_occurred     n
##   <chr>          <int>
## 1 No             20245
## 2 Yes             6767
```

With this in mind here is a look at the occurence, not the amount, of crimes within each of the cities. To show the difference in this below are two plots, one counting the occurences with the value of zero being accounted for, erronously in this case, and one accouting and ignoring these values.

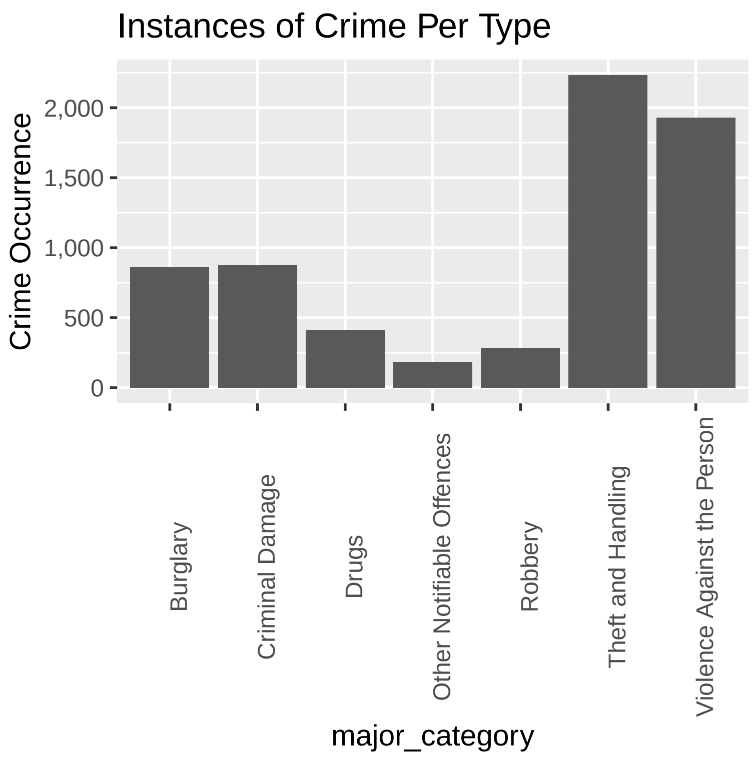
```
#note, need to look at values greater than 0 when summarizing or via crime occurred
#0 here means nothing counted for specific crime in given month

#demonstrate below, comparing borough crime occurences per month
#note, just seeing if crime occured, not amount,

#incorrect plot,
london %>%
  group_by(borough)%>%
  ggplot(aes(borough))+
  geom_bar()+
  theme(axis.text.x = element_text(angle = 90))+
  scale_y_continuous(name="Crime Occurence", labels = comma)+
  labs(title="Incorrect Plotting of Crime Occurrences")
```



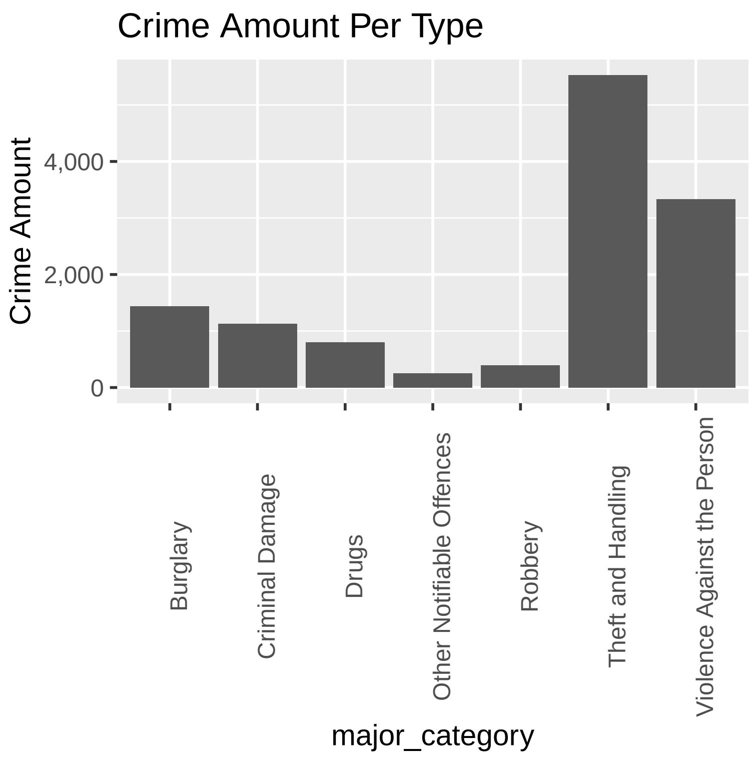
```
#correct plot
london %>%
  filter(crime_occurred=="Yes") %>%
  group_by(borough)%>%
  ggplot(aes(borough))+
  geom_bar()+
  theme(axis.text.x = element_text(angle = 90))+
  scale_y_continuous(name="Crime Occurence", labels = comma)+
  labs(title="Crime Occurrences Within Boroughs")
```

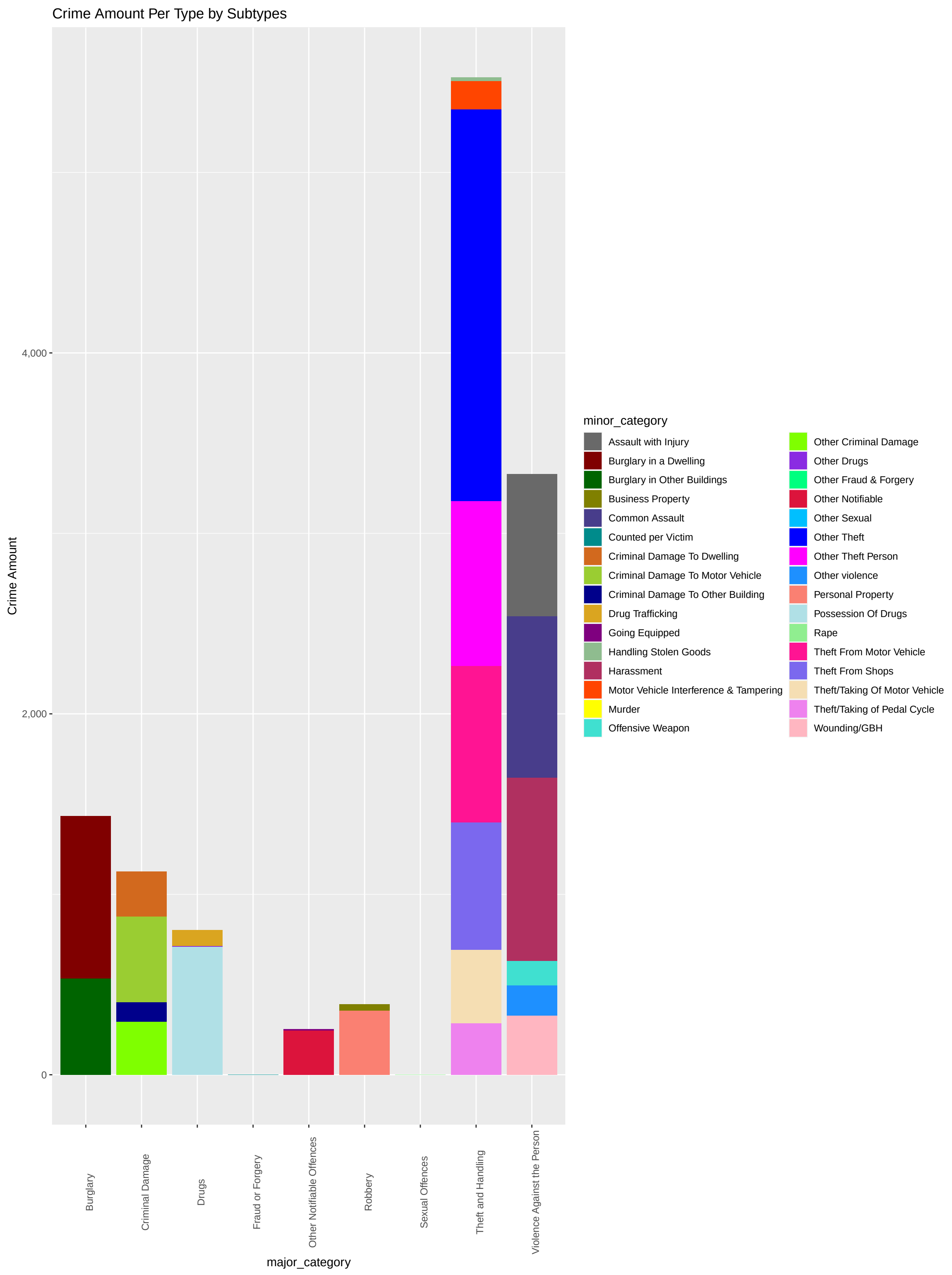


Notably to the above two plots are how significant the lack of a report has on the plots, with a significant decrease across the board for all borough crime occurrences. When looking at the instances of crime however the need to account for zero is already handled, and when seen below a marked incidence with the borough of Westminster with almost 3,000 crime instances.



In conjuncture with the point of crime instances to the above is the marked proclivity towards two types of crime, along with their sub-types, the major_category and minor_category respectively. Shown below are plots showcasing this marked increase in crimes in Thievery and Violence Against Persons along with a breakdown of the sub-types on the next page.




Crime Amount Per Type by Subtypes
Crime Amount
0
2,000
4,000
major_category
Burglary
Criminal Damage
Drugs
Fraud or Forgery
Other Notifiable Offences
Robbery
Sexual Offences
Theft and Handling
Violence Against the Person
minor_category
Assault with Injury
Burglary in a Dwelling
Burglary in Other Buildings
Business Property
Common Assault
Counted per Victim
Criminal Damage To Dwelling
Criminal Damage To Motor Vehicle
Criminal Damage To Other Building
Drug Trafficking
Going Equipped
Handling Stolen Goods
Harassment
Motor Vehicle Interference & Tampering
Murder
Offensive Weapon
Other Criminal Damage
Other Drugs
Other Fraud & Forgery
Other Notifiable
Other Sexual
Other Theft
Other Theft Person
Other violence
Personal Property
Possession Of Drugs
Rape
Theft From Motor Vehicle
Theft From Shops
Theft/Taking Of Motor Vehicle
Theft/Taking of Pedal Cycle
Wounding/GBH

The codes within the london dataset, the lsoa_code, are also pertinent to this analysis not just in the distribution of crime occurrences but also due to how they are related to other pieces of data within the dataset. A subset is best to describe the distributions of the former, shown below with a set of tibbles.

```
#better to see as summary perhaps
lsoa_Table <- london %>% group_by(lsoa_code) %>%
  summarise(n=n())

#more distinct values
top_n(as.data.frame(lsoa_Table), 3)
```

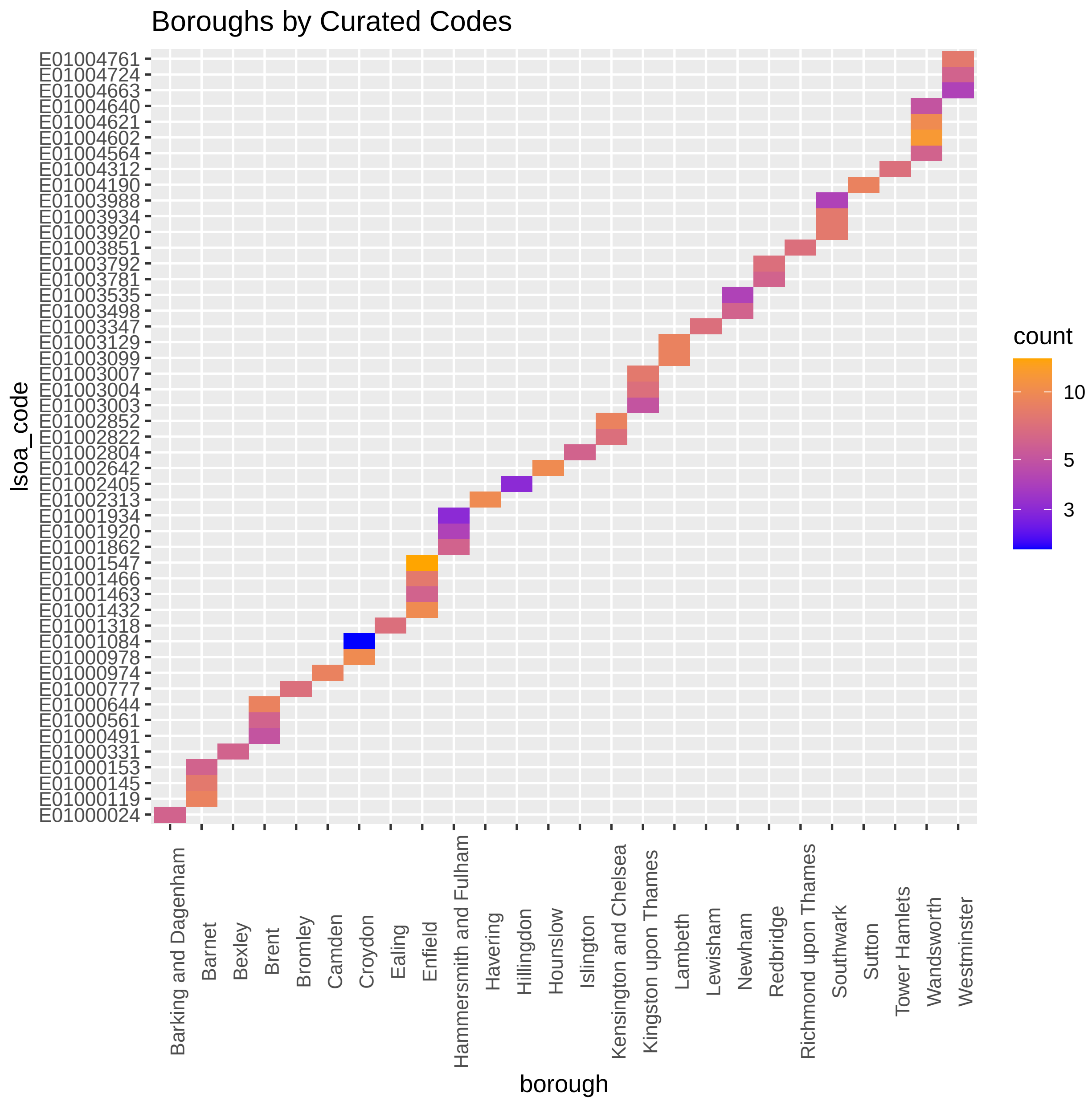
```
##   lsoa_code  n
## 1 E01000090 15
## 2 E01000661 18
## 3 E01002361 15
```

```
sample_n(lsoa_Table,5)
```

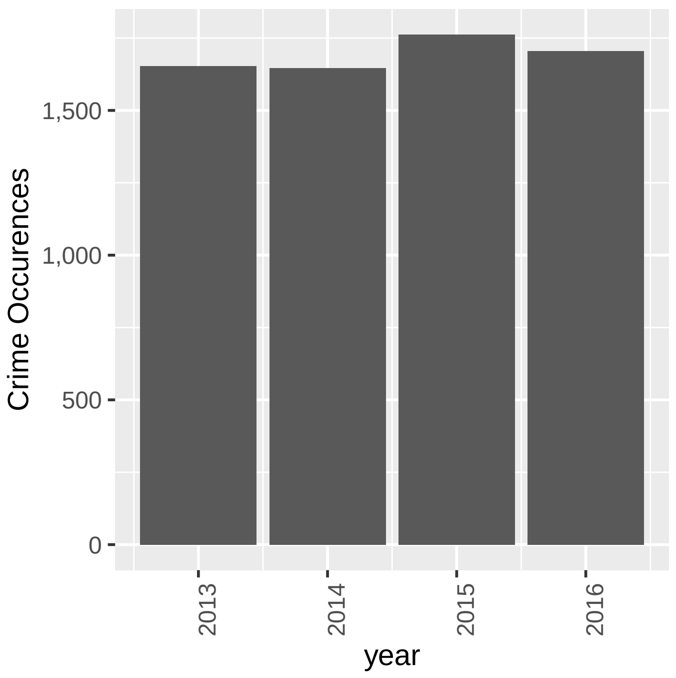
```
## # A tibble: 5 x 2
##   lsoa_code     n
##   <chr>     <int>
## 1 E01002845     5
## 2 E01001423     7
## 3 E01001702     3
## 4 E01003715     3
## 5 E01033491     7
```

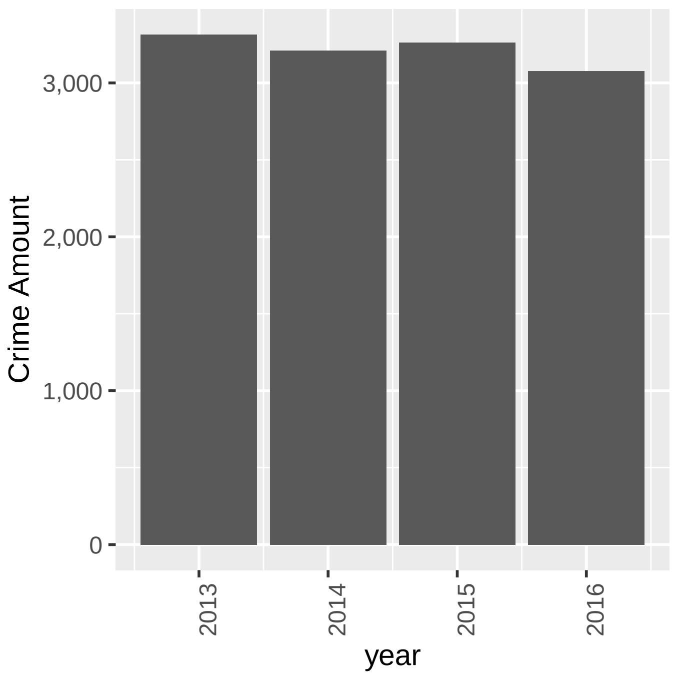
Besides this the effect of the lsoa_codes in relation to other variables within the london dataset. In particular is a look at the boroughs of which these codes typically refer to. A heatmap of approximately 30 of these codes, while only a subset of the data, can be informative on their relations.

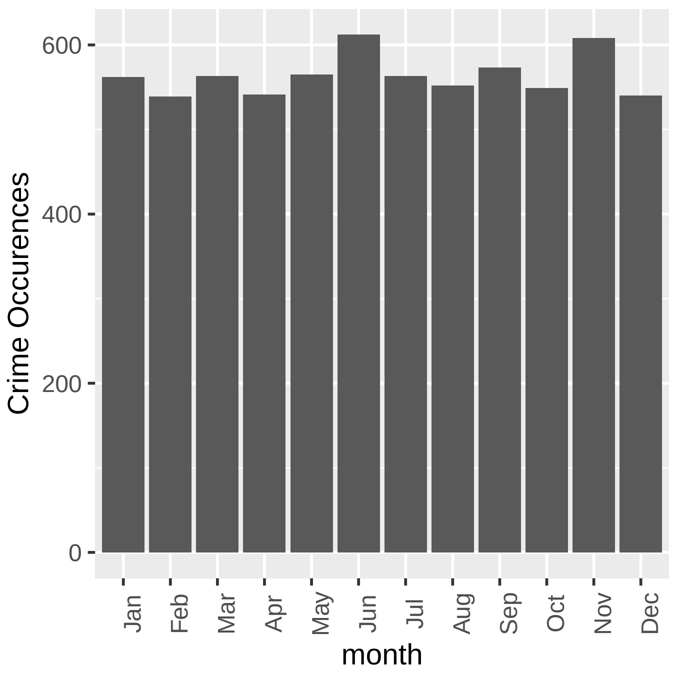
```
#get index of lsoa codes as char vector for filtering
set.seed(2021, sample.kind="Rounding")
lsoa_index <- as.vector(sample(london$lsoa_code,50))

#sample low amount to show readable lsoa codes
london%>%
  filter(lsoa_code %in% lsoa_index)%>%
  ggplot(aes(x=borough, y=lsoa_code))+
  geom_bin2d() +
  scale_fill_gradient(low="blue",high="orange",trans="log10")+
  labs(title = "Boroughs by Curated Codes")+
  theme(axis.text.x = element_text(angle = 90))
```

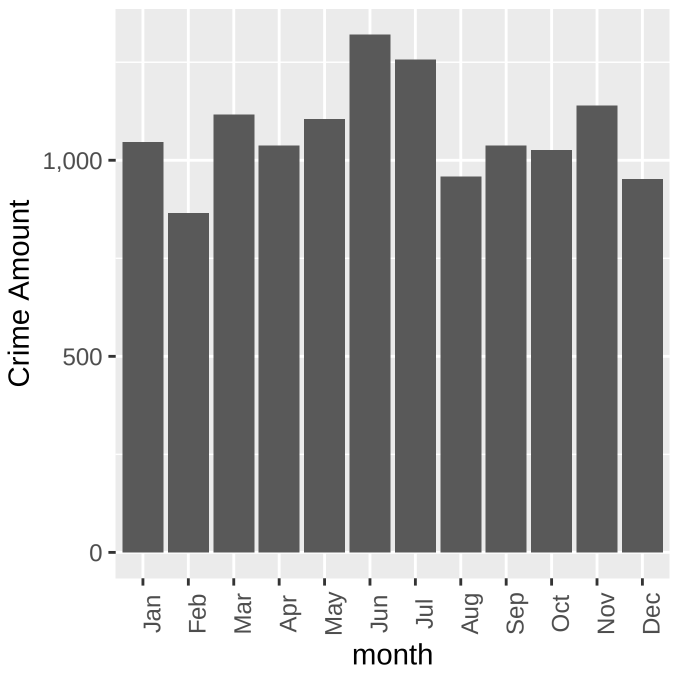


Before moving on a final look at the two last variables in the data, the month and the year. Looking towards both the occurrence and instances there is a general level of similar results, with only small deviations in the instances in the year 2012 and the months February and July.









2.4 London Crimes Recap: Thoughts and Goals

With what has previously been shown it is important to go over this report's preliminary findings. Crime occurrence values skew towards 0, inviting the prospect of a categorical viewpoint of occurrence of crimes. Lsoa_codes have a differing relation between the boroughs in the dataset, hinting at bias towards specific borough. Generally similar values in the month and year variables barring a few outlier only in the instances of reported crime, showing that there are minor effects. The methods discussed below will need to account for this. As a general step, due to the various values, there are two direct directions this report can take, one view as continuous and one as categorical. Finally as a reminder to the goal of this report is to have a minimum accuracy of 70% in predicting crime.

3.0 London Crimes: Recommendation Model

3.1 Recommendation Setup

One possible method of analyzing this dataset would be to utilize a recommendation based model due to the size and amount of data within the dataset. The idea here being to recommend the values of what may occur based on the dataset. In order to do so an RMSE, or residual mean squared error, may be used to do so as it acts very much akin to a standard deviation (Irizarry, 2021, Loss function). Notable in this dataset though is the fact that values within the dataset are whole numbers, as one cannot partially have an instance of a crime. This will then have to be taken into account when deriving the recommendation model. Before developing the model however a split in the training set, london, needs to occur for training purposes. Not only that, but to ensure accuracy data found within the test set must be found within the train set.

```r
#make rmse function
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}


set.seed(2021, sample.kind="Rounding")
#since modeling for final test must separate the training set into two, with 10% as training test set
test_index <- createDataPartition(y = london$value, times = 1, p = 0.1,
                                  list = FALSE)
train_set <- london[-test_index,]
test_set <- london[test_index,]

##ensure for samples data in test set has same borough, major, and code as train set sample
test_set <- test_set %>%
  semi_join(train_set, by = "borough")%>%
  semi_join(train_set, by = "major_category")%>%
  semi_join(train_set, by = "lsoa_code")%>%
  semi_join(train_set, by = "minor_category")%>%
  semi_join(train_set, by = "year")%>%
  semi_join(train_set, by = "month")%>%
  semi_join(train_set, by = "value")%>%
  semi_join(train_set, by = "crime_occurred")
```

With this done the recommendation model can be developed in the following section.

3.2 Averaging the Dataset

To begin the recommendation model here is a quick base RMSE to improve from, derived via predicting all variation in the data is by chance as shown in the model below from Professor Irazarry's githhub (Irizarry, 2021, A first model). This is done via Y, the predicted value, being generated from mu, all the values in the dataset, and e, the assumed errors.

$$Y_{u,i} = \mu + \varepsilon_{u,i}$$

```r
##simple way to predict,
##using only mu, the average of random variance
set.seed(2021, sample.kind="Rounding")

mu_hat <- mean(train_set$value)
mu_hat
```

```
## [1] 0.4755245
```

```r
naive_rmse <- RMSE(test_set$value, mu_hat)

##put result into table for later use
rmse_results <- tibble(method = "Assumed Average", RMSE = naive_rmse)
rmse_results
```

```
## # A tibble: 1 x 2
##   method           RMSE
##   <chr>           <dbl>
## 1 Assumed Average  1.20
```

As seen above this assumption is highly inaccurate and has much room for improvement. One method of improving this model is to account for the variations being from the effects of the variables. Said variables being the codes, borough, major category, minor cateogy, month, and finally year. Demonstrated below is a model of this effect. These effects can be approximated with the model below based on one shown by Professor Irizarry (Irizarry, 2021, Modeling movie effects). Notably it is similar to the model above but with the addition of v, the effects defined as average values. For the sake of brevity, please refer to the R script provided with this code for reference to code utilized

$$Y_{u,i} = \mu + v_i + \varepsilon_{u,i}$$

```
## [1] 1.255076
```

As noted previously however there are ways to possibly improve the predicted values. For instance values in the dataset are whole integers. As there are numerous records close to either zero, one, or two this report feels it safe to possibly round predicted values downward. Another possibility are the occurrences of negative values is impossible as there cannot be negative crimes. Therein this report can safely assume all negative values can be counted as zero values. Taking into these possibilities into account here are the predicted values with these assumptions.

```r
#now the effects rounded downward
all_effects_rounded <- RMSE(floor(predicted_values), test_set$value)
all_effects_rounded
```

```
## [1] 1.387981
```

```r
#effects but all negative numbers rounded to zero
predicted_values_no_negative <- ifelse(predicted_values <= 0, 0, predicted_values)

all_effects_added_no_negatives <- RMSE(predicted_values_no_negative, test_set$value)
all_effects_added_no_negatives
```

```
## [1] 1.245732
```

```
rmse_results <- rmse_results %>% add_row(method = "All Effects",
                                         RMSE = all_effects_added_no_negatives)
```

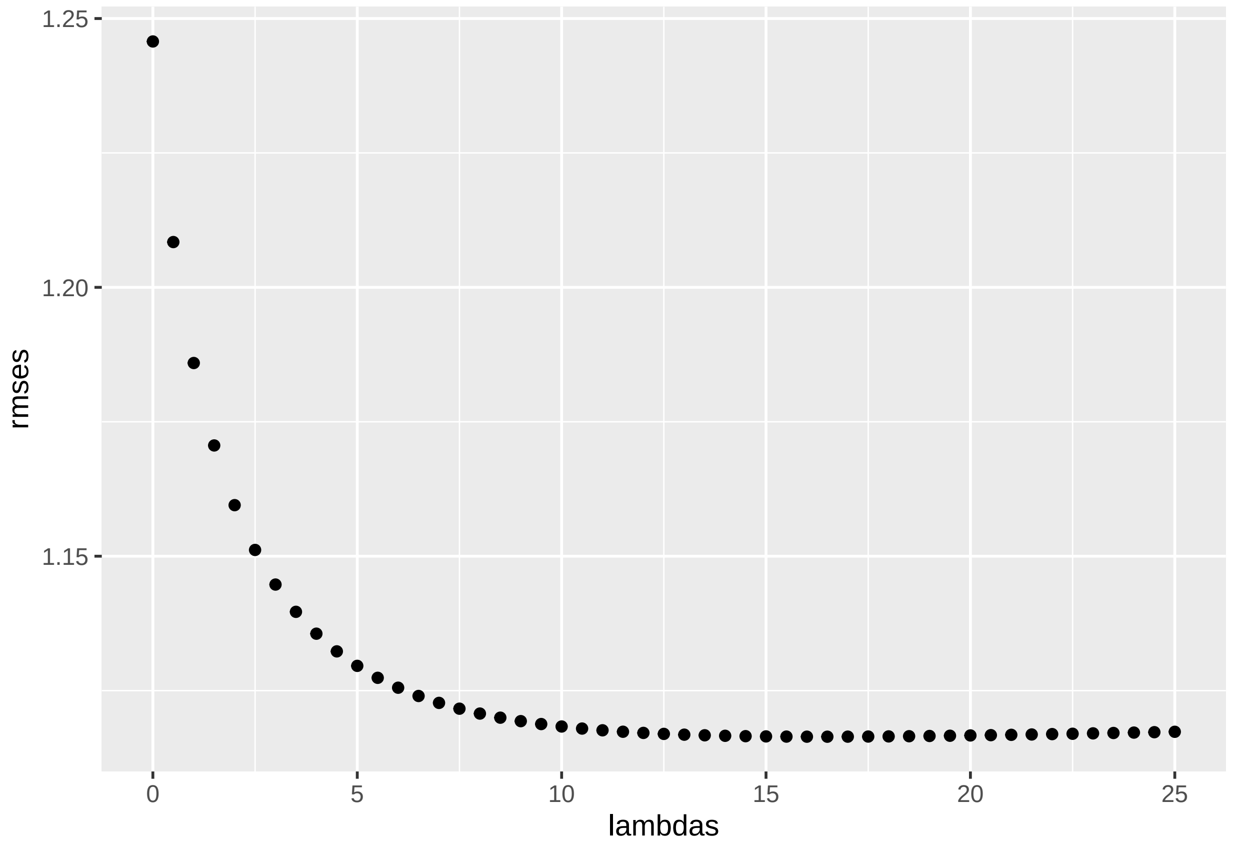
As seen above by taking the negative values and assuming they were in fact zero has improved the accuracy of the predicted values. Moving onward regularization needs to be taken into account.

3.3 Regularizing the Dataset

Regularization takes into account the fact that some places simply have more records or instances of reports than other. Seen in a model below based on Professor Irizarry's model in his Data Science course, with the idea here is to use a tuning parameter to account for these differences (Irizarry, 2021, Penalized least squares).

$$\hat{v}_i(\lambda) = \frac{1}{\lambda + n_i} \sum_{u=1}^{n_i} (Y_{u,i} - \hat{\mu})$$

Now shown below are many possible tuning parameters to use with a plot to demonstrate the tuning parameters. Of note is that this report also takes into account the need to set negative values to zero before applying the prediction. Note that for the sake of brevity only the code utilizing the best tuning parameter will be shown below.



With this done the best tuning parameter can be applied to the to a single version of the model shown above. As mentioned earlier here is the method utilizing the best lambda found previously with the model.

the only difference being that there is no application of multiple tuning parameters to see which one works best.

```r
#obtain best lambda
best_lambda <- lambdas[which.min(rmses)]

#now apply to the model above
set.seed(2021, sample.kind="Rounding")
mu <- mean(train_set$value)

b_c <- train_set %>%
    group_by(lsoa_code) %>%
    summarize(b_c = sum(value - mu)/(n()+best_lambda))

b_b <- train_set %>%
    left_join(b_c, by="lsoa_code") %>%
    group_by(borough) %>%
    summarize(b_b = sum(value - b_c - mu)/(n()+best_lambda))

b_ma <- train_set %>%
    left_join(b_c, by="lsoa_code")%>%
    left_join(b_b, by="borough") %>%
    group_by(major_category) %>%
    summarize(b_ma = sum(value - b_c - b_b - mu)/(n()+best_lambda))

b_mi <- train_set%>%
    left_join(b_c, by="lsoa_code")%>%
    left_join(b_b, by="borough") %>%
    left_join(b_ma, by="major_category") %>%
    group_by(minor_category) %>%
    summarize(b_mi = sum(value - b_c - b_b - b_ma - mu)/(n()+best_lambda))

b_y <- train_set%>%
    left_join(b_c, by="lsoa_code")%>%
    left_join(b_b, by="borough") %>%
    left_join(b_ma, by="major_category") %>%
    left_join(b_mi, by='minor_category') %>%
    group_by(year) %>%
    summarize(b_y = sum(value - b_c - b_b - b_ma - b_mi - mu)/(n()+best_lambda))

b_m <- train_set%>%
    left_join(b_c, by="lsoa_code")%>%
    left_join(b_b, by="borough") %>%
    left_join(b_ma, by="major_category") %>%
    left_join(b_mi, by='minor_category') %>%
    left_join(b_y, by='year') %>%
    group_by(month) %>%
    summarize(b_m = sum(value - b_c - b_b - b_ma - b_mi - b_y - mu)/(n()+best_lambda))

predicted_values <-
    test_set %>%
    left_join(b_c, by="lsoa_code")%>%
    left_join(b_b, by="borough") %>%
    left_join(b_ma, by="major_category") %>%
    left_join(b_mi, by='minor_category') %>%
    left_join(b_y, by='year') %>%
    left_join(b_m, by='month') %>%
    mutate(pred = mu + b_c + b_b + b_ma + b_mi + b_y + b_m) %>%
    pull(pred)
```

```
#now set predicted values for negative to be zero and predict
predicted_values <- ifelse(predicted_values <= 0, 0, predicted_values)
values_regularized_effects <- RMSE(predicted_values, test_set$value)

rmse_results <- rmse_results %>% add_row(method = "Regularized All Effects",
                                         RMSE = values_regularized_effects)

rmse_results %>% knitr::kable()
```

method	RMSE
Assumed Average	1.197242
All Effects	1.245732
Regularized All Effects	1.116433

3.4 Reccomendation Model Results

At the start of the recommendation model the RMSE was value was rather poor but did improve when taking into account the effects and regularizing those effects. With that said the goal of an accuracy of minimally 70% was not met. Ultimately the recommendation model developed cannot be recommended, being ill-suited towards this type of analysis in its current form. The reason for this is likely due to the prevalence of values 0, 1, and 2. Now taking into account the possibility of analyzing the data in a categorical viewpoint as discussed above there may be an alternate way to create the model. For this however this report will go towards a model more proven to working with multitudes of categorical variables.

4.0 Categorically-Oriented Methods

4.1 Training Setup

Before beginning to test the various categorical methods with this dataset there remains one possible issue. Due to hardware limitations as well as speed/performance issues time to process in training can be strenuous. To solve this a sample of the training and testing sets, 10%, will be utilized to help the performance. This will allow predictive findings to be made with the data while keeping the processing time swift, allowing for more fine tuning to be done. With this done the report may now move onto other methods utilized.

```
#for training purposes only
#sample for faster usage
train_set_sample <- sample_frac(train_set, 0.1)
test_set_sample <- sample_frac(test_set, 0.1)

test_set_sample <- test_set_sample %>%
  semi_join(train_set_sample, by = "borough")%>%
  semi_join(train_set_sample, by = "major_category")%>%
  semi_join(train_set_sample, by = "lsoa_code")%>%
```

```r
  semi_join(train_set_sample, by = "minor_category")%>%
  semi_join(train_set_sample, by = "year")%>%
  semi_join(train_set_sample, by = "month")%>%
  semi_join(train_set_sample, by = "value")%>%
  semi_join(train_set_sample, by = "crime_occurred")
```

4.1 Nearest Neighbor Method

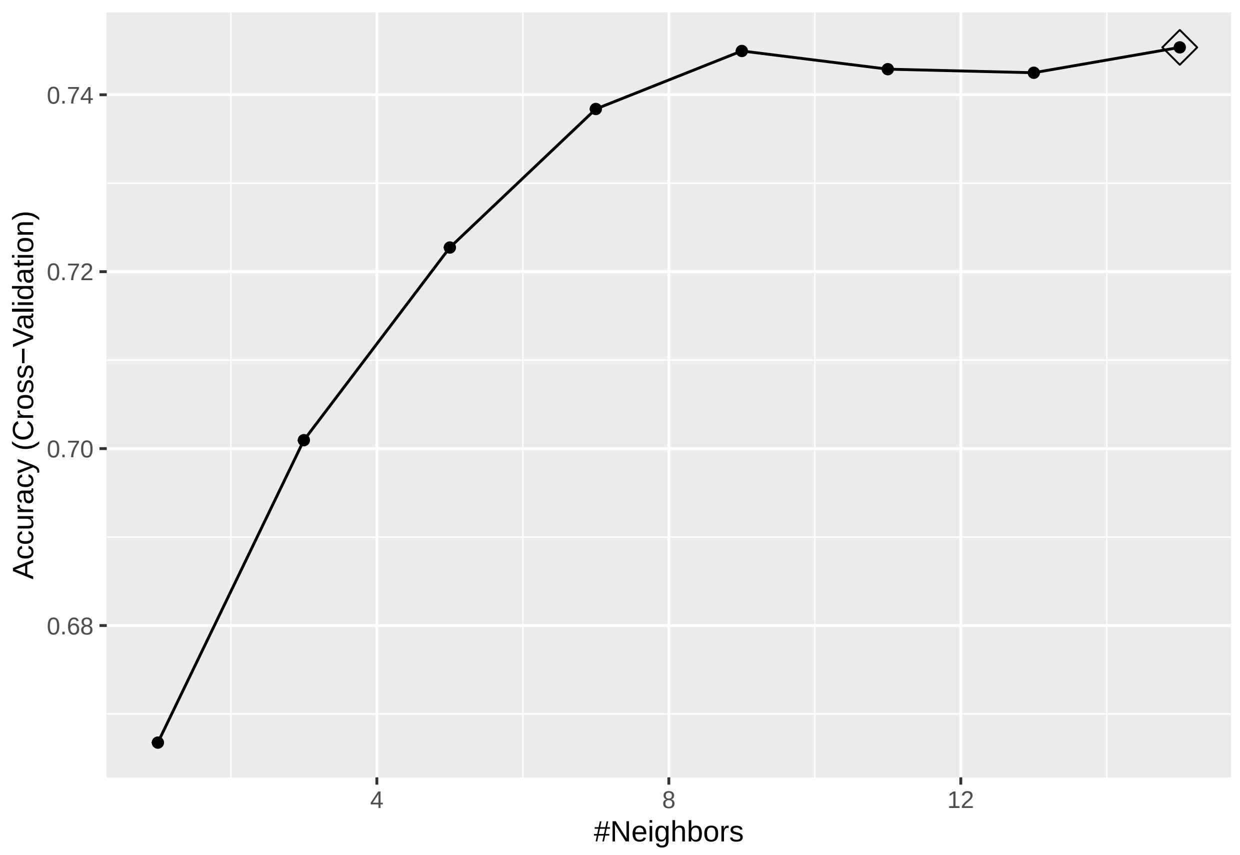
One method of analyzing the data in another method, though not one predicated on categorical variables, is to use kNN, or k-nearest neighbors, which involves using the observations in the data to make estimates via the nearest observation points as talked about on Professor Irizarry's online githhub (Irizarry, 2021, Motivation with k-nearest neighbors). In implementing kNN this report will be using the crime_occurred column created earlier and leaving out the values column. The idea here being that, based on a grid, values that are of similar crime_occurred status will be near one another as that will have similar factors. For example having similar lsoa_code, borough, major_category, etc. From there by utilizing the similarities this model should then be able to predict whether a crime occurred or did not occur. Besides this, as mentioned earlier, performance has been a key component of this report. As such the k tuning value will only tune to a maximum of 15 to keep computation time to a minimum.

```r
#due to computation time, this will be used within the markdown
train_knn_cv <- train(crime_occurred ~ lsoa_code + borough + major_category +
                        minor_category + year + month,
                      method = "knn",
                      data = train_set_sample,
                      tuneGrid = data.frame(k = seq(1, 15, 2)),
                      trControl = trainControl(method = "cv", number = 10, p = .9))

#prediction
knn_cv_preds <- predict(train_knn_cv, test_set_sample[c(1:4,6:7)])
knn_prediction <- mean(knn_cv_preds == test_set_sample$crime_occurred)

#copy prediction to table for future comparisons
prediction_results <- tibble(method = "kNN Method", Prediction = knn_prediction)

#see results
prediction_results %>% knitr::kable()
```

method	Prediction
kNN Method	0.7807018

Also, for visual reference, a plot of the possible prediction values with the number of neighbors utilized. With this the best k tuning parameter found may be used later in the final method should kNN be said method.

```r
#see best fit as number and within plot
ggplot(train_knn_cv, highlight = TRUE)
```



```
#also the best tune assigned
train_knn_cv$bestTune
```

```
##    k
## 8 15
```

```
best_k <- train_knn_cv$bestTune
```

4.2 Rpart Regression Tree

Another method of analyzing the data and predicting results would be to use rpart, recursive partitioning. Here data is partitioned to create a tree of possible outcomes recursively, meaning that the current partition is utilized in determining future partitions and so on (Irizarry, 2021, Regression Trees). By doing so a prediction can be made within a region or range of data that, when applied to the london dataset, may allow for the prediction of crime based on the various variables held within it. Again utilizing the crime_occurred column and not the values column and accounting for the effects of the other variables is shown below. Besides this to keep computation time to a minimum the cp tuning value will be capped at a value of 0.2 incrementing by 0.05 from 0 onward.

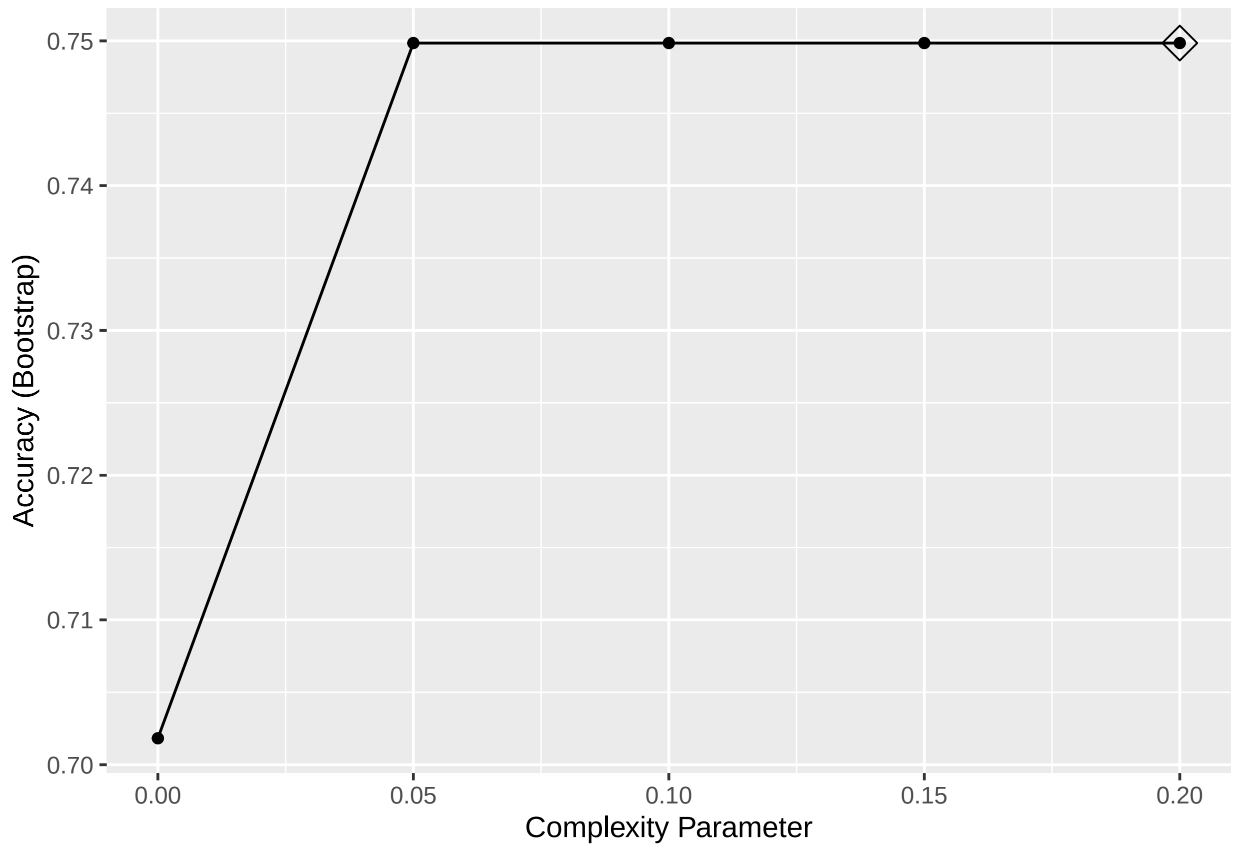
```
#train via rpart
train_rpart <- train(crime_occurred ~ lsoa_code + borough + major_category + minor_category + year + mor
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.2, 0.05)),
                     data = train_set_sample)
```

```r
#prediction
rpart_preds <- predict(train_rpart, test_set_sample[c(1:4,6:7)])
rpart_prediction <- mean(rpart_preds == test_set_sample$crime_occurred)

#add to table
prediction_results <- prediction_results %>% add_row(method = "RPart Method",
                                                     Prediction = rpart_prediction)

#see results
prediction_results %>% knitr::kable()
```

method	Prediction
kNN Method	0.7807018
RPart Method	0.7807018

As seen above rpart did not do as well as kNN, which is likely due to the fact that it more suited with continuous variables even if applicable with categorical ones. Also below is a visual reference for the possible prediction values calculated via the cp and the retaining of the best one for use if this model should be the best one to use.

```r
#see rpart tune and plot
ggplot(train_rpart, highlight = TRUE)
```



```
train_rpart$bestTune
```

```
##    cp
## 5 0.2
```

```
#set best cp
best_cp <- train_rpart$bestTune
```

4.3 Random Forest Method

The final independent method that this report will be looking at is the random forest method. This method, somewhat similar to rpart, involves the usage of decision trees but in addition to this the method averages decision trees together (Irizarry, 2021, Random Forest). From there these averages are then used to create the prediction. To help again with the computation time node size, which is partly how the averages of decision trees are made, will be set to 100 and the ntree will be set to 250. Respectively these values are the minimal size node of a tree, or put plainly how deep the tree can be, and the number of trees created. By doing so this limits how many trees are made, ntree, and how many divisions there can be per tree, nodesize. Shown below is the code in creating the random forest prediction. In addition a look at the most important variables in this random forest is shown as well.

```
#random forest

train_rf <- train(crime_occurred ~ lsoa_code + borough + major_category + minor_category + year + month
                  data = train_set_sample,
                  method = "rf",
                  ntree = 100,
                  nodesize = 250,
                  tuneGrid = data.frame(mtry = seq(1:3)))

#create prediction
rf_preds <- predict(train_rf, test_set_sample[c(1:4,6:7)])
randForest_prediction <- mean(rf_preds == test_set_sample$crime_occurred)


#add to table
prediction_results <- prediction_results %>% add_row(method = "Random Forest Method",
                                                     Prediction = randForest_prediction)
#see results
prediction_results %>% knitr::kable()
```

method	Prediction
kNN Method	0.7807018
RPart Method	0.7807018
Random Forest Method	0.7807018

```
#see best tuning and set
train_rf$bestTune
```

```
##   mtry
## 1    1
```

```r
best_mtry <- train_rf$bestTune

#see important variables
varImp(train_rf)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 1998)
##
##                                  Overall
## minor_categoryOther Theft         100.00
## minor_categoryOther violence       44.97
## boroughLewisham                    32.07
## minor_categoryPossession Of Drugs   31.91
## lsoa_codeE01003943                 29.83
## lsoa_codeE01004576                 23.68
## lsoa_codeE01001354                 23.60
## lsoa_codeE01001118                 23.38
## lsoa_codeE01003984                 23.27
## lsoa_codeE01001436                 18.55
## lsoa_codeE01033735                 18.45
## lsoa_codeE01002444                 17.99
## lsoa_codeE01004713                 17.86
## lsoa_codeE01000305                 17.80
## lsoa_codeE01004475                 17.53
## lsoa_codeE01003550                 17.48
## minor_categoryCounted per Victim   15.40
## lsoa_codeE01033593                 13.34
## boroughBromley                     13.29
## lsoa_codeE01003456                 12.67
```

4.4 Quick Comparison

Now that this is done all models can be compared to one another to see which model had the best predictive power. As shown below all models performed admirably. With that said the best overall model was the kNN model as it not only performed well but performed with speed which was necessary to this report. There is a touch of surprise that kNN performed better than random forest, as random forest's decision trees work nicely with the categorical nature of the dataset. Likely this is due to the performance cuts taken to increase overall performance time. Shown below is a quick once over of the previous models utilized and a cleanup of previous variables no longer needed.

method	Prediction
kNN Method	0.7807018
RPart Method	0.7807018
Random Forest Method	0.7807018

5.0 London Crimes: Final Testing

5.1 London Crimes: Final Test Method

With the above fine-tuning of the above methods and algorithms done, and the best predictive model that also performs quickly is the random forest method, this report can now apply it to the london and final_test datasets

5.2 End Results

To begin firstly this report must utilize kNN method to the london dataset. Once done this can then be compared to the final_test dataset. From there an accuracy mirroring what was found in the training set, perhaps a bit less due to working with more data, can be achieved. Below is the final model with the results. Of note here is that the k value which has been set to the best k tuning parameter found during training.

method	Prediction
kNN Method	0.7807018
RPart Method	0.7807018
Random Forest Method	0.7807018
Final Model: kNN	0.7620482

6.0 Conclusion

6.1 Report Summary

This report went over the thought process of analyzing the results of london crimes from the year 2010 onward and find a predictive model that reached at minimum 70%. Discarding a more linear regressive recommendation model and the values, the count of monthly crime incidences, for a broader look at the data as a binary "yes" or "no" viewpoint was implemented within the report. After testing more categorical methods a final method was created via utilizing three sub-methods to increase accuracy. With this done the final method ultimately met the goal of predicting with an accuracy of at least 70%.

6.2 Improvement Postulation

Looking back on this report there is much room for improvement. To begin better fine tuning could have been utilized, for instance with kNN and its k parameter and for random forest and its ntree parameter were lowered for the sake of performance and speed. Speed and performance were a crucial aspect of this report, often necessitating editing of sections in order to account for disproportionate runtimes of training methods. Undoubtedly there could have been more tuning the recommendation view of the data, however for this report it felt prudent to have a method that did not work quite as well for a comparative.

Exploration, while satisfactory overall, could have been improved with better nuisances in visuals, specifically with more diverse plots. Besides this some explanatory information was cut for the sake of keeping this report within a respectable length. Other than that minor changes in the overall structure of this report, such as choosing whether to sample training data or not and what year to begin trimming the data, while not harmful did take an unexpected amount of time when compounded with one another.

Hardware limitations were also a grave handicap for this report. Often processes on their own simply took too long to reasonably utilize due to speed issues that better hardware could have solved. Due to this training was sampled for the sake of speed and the overall size of the data taken from the original source, london_crimes, was shrunk from approximately 13 million records to 30,000 records for the utilized london_crime_curated.csv file. Most tragically the code to utilize an ensemble method ultimately remained unused due to the process time it'd take to utilize such a model. In general this report would have benefited greatly with better hardware, which as time moves on will become more available. With that said this report will mention that the handling of this is an aspect of data analysis and data science that practitioners will need to face.

6.3 Final Thoughts

On a final note, the size of the analysis also presents a problem. Currently this analysis only represents less than 1% of the data within the original dataset. Mentioned earlier this was done due to size constraints and hardware limitations yet as such it is doubtful to be holistically applicable to the overall dataset. So what was this report about then? In summation a great learning opportunity and lesson. The exploration through various methods, discarding unsuited and adjusting those that are until they work, and the wrangling of data along with the work-arounds usable when faced with diverse limitations. This report can certainly be utilized as a working point for future analysis, a means of going to the right direction while providing insights that others can use. The motivation of this report was a final model that had a minimal of 70% accuracy. This report, through said motivation goal, was meant to show how one can go about it.

In general this report was an interesting exploration into independent research into various methods to analyze data. The handicaps and struggles faced throughout it were likely also faced by others in their own analyzes as well. With that said, as a first stepping stone into the field of Data Science, this report hopes to have been informative and educational in its usage. I, the author, would personally like to thank Professor Irizarry and his staff in their Data Science course. Also thank you for reading this report and hopefully looking forward to more as well.

References

- Cornell (2021). Criminal Law https://www.law.cornell.edu/wex/criminal_law. Accessed 7/01/2021.
- Boysen, J (2017). Kaggle: London Crime Data, 2008-2016. https://www.kaggle.com/jboysen/london-crime. Accessed 6/24/2021.
- Irizarry, R (2021). Introduction to Data Science: Loss function. https://rafalab.github.io/dsbook/large-datasets.html#netflix-loss-function.
- Irizarry, R (2021). Introduction to Data Science: A first model. https://rafalab.github.io/dsbook/large-datasets.html#a-first-model.
- Irizarry, R (2021). Introduction to Data Science: Modeling movie effects. https://rafalab.github.io/dsbook/large-datasets.html#modeling-movie-effects.
- Irizarry, R (2021). Introduction to Data Science: Penalized least squares. https://rafalab.github.io/dsbook/large-datasets.html#penalized-least-squares.
- Irizarry, R (2021). Introduction to Data Science: Motivation with k-nearest neighbors. https://rafalab.github.io/dsbook/cross-validation.html#knn-cv-intro
- Irizarry, R (2021). Introduction to Data Science: Regression Trees. https://rafalab.github.io/dsbook/examples-of-algorithms.html#regression-trees
- Irizarry, R (2021). Introduction to Data Science: Random Forest. https://rafalab.github.io/dsbook/examples-of-algorithms.html#random-forests