Temporal and Spatial Analysis of Street Crime in Inner London

Blae Quayle

7/20/2017

1. Introduction

The Metropolitan Police operate across 32 boroughs within Greater London and aim to make London the safest global city. To this end, an accessible method allowing both residents and tourists to check where crime hotspots are located would allow them to make informed decisions about where they go and encourage preventative action. This would be a useful service for the Met Police to promote with the aim of reducing opportunistic crimes by raising awareness of the highest risk areas.

For locals, this may be in the context of checking incidences of bicycle theft to identify where is best to lock up your bike, or hotspots for burglaries when looking to rent or purchase property. This may also be useful for tourists visiting the city to flag if they are in a higher risk area.

This project aims to address several key questions:

1. Is crime on the increase or decrease in central London?
2. Is the frequency of specific crimes reducing or increasing on an annual basis?
3. Is there any seasonal variation in the frequency of crimes?
4. Are certain crimes more prevalent in certain areas - where are these 'hotspots'?
5. Create a visualisation allowing easy identification of risk level to a certain crime
6. Can correlations be made between crime occurrences and other indicators e.g. employment levels?
7. Can time series forecasting be used to estimate occurence of crime in the future?

2. Data acquisition

Datasets for street level crime in England, Wales and Northern Ireland are available at (https://data.police.uk/docs/method/crime-street/). Data is reported monthly and is available from December 2010 to March 2017 at present. All the data on this site is made available under the Open Government License v3.0.

There is an associated API which can efficiently and rapidly provide a rich data source of information. The API is implemented as a standard JSON web service using HTTP GET and POST requests.

Example API Request https://data.police.uk/api/crimes-street/all-crime?poly=52.268,0.543:52.794,0.238:52.130,0.478&date=2013-01

To keep the size of the dataset manageable an area over central London was initially defined. The *poly* parameter identifies the polygon within which data is extracted and is formatted in lat/lng pairs, separated by colons:

poly=[lat],[lng]:[lat],[lng]:[lat],[lng]:[lat],[lng]

The date also needs to be specified or the most recent month will be returned. The limit of a custom search area is 10,000 crimes, if it contains more than this the API will return a 503 status code.

Location data is only referencing the approximate location using 'anonymous' map points. This is to protect the privacy of the victim. The date of occurence is also truncated to record only month and year for the same reason.

The API was initially used to acquire the full dataset (*crime11to17*) from December 2011 to March 2017 for central London as defined by a polygon with corners: 51.514,-0.225 - Shepherds Bush 51.484,-0.225 - South Hammersmith 51.484,-0.105 - Kennington 51.514,-0.105 - Temple

It was subsequently used to acquire the 2016 crime data for a significantly larger area covering all the inner London boroughs (*central2016*). These are Camden, Greenwich, Hackney, Hammersmith and Fulham, Islington, Kensington and Chelsea, Lambeth, Lewisham, Southwark, Tower Hamlets, Wandsworth and Westminster. 17 individual polygons were requird to prevent the call limit of 10,000 being exceeded. There were then combined during the data wrangling process and filtered to remove the crimes that fell outside those boroughs.

3. Important fields and information

Variable names, types and number of observations were reviewed using `glimpse()` on the original crime11to17 dataframe. This reveals 11 variables - 6 characters, 3 numerics (1 integer, 2 double precision) and 2 factors. There are 498,628 observations in total.

The following variables are returned: **category**, location_type, context, persistent_id, id, location_subtype, **month**, **latitude**, **longitude**, outcome_status_category, outcome_status_data. The key fields for this study are in **bold**.

There are 15 different types of crime recorded in the **category** variable. Month and year are available but not day due to the police anonymisation process. Latitude and longitude are provided as geographical coordinates in decimal degrees. Location data is only referencing the approximate location using 'anonymous' map points.

The central2016 dataframe had 696,844 observations prior to filtering by the inner boroughs, and 447,418 observations post filtering. It contains two additional variables: **borough** and **census code**.

4. Data limitations

The main limitations that have been identified with the dataset are:

- Anonymistation of the data means the locations are fixed to a 'map point' and jittering is required to make all the events visible, although as the dataset gets larger it is not possible to create an effective map visualisation of individual crimes.
- Anonymisation also means no day/time is provided so it is not possible to analyse potential relationships on a weekly scale with certain crimes.
- It has been difficult to integrate other datasets relating to indicators at a Borough level as the data was downloaded using the API over a polygon area of central London which encompasses parts of Westminster, Camden, Lambeth and Southwark. As the project progressed, it became clear that a dataset covering a large, clearly defined area would be needed. GIS data was used to filter the large dataset, and remove all crimes which did not fall within one of the inner boroughs.
- City of London is not included in this study as it is has its own police force and therefore the data is stored separately. The white blob in the centre of plots indicates its location.

5. Data cleaning and wrangling

The following packages were loaded in preparation for data wrangling: `readr`, `dplyr`, `tidyr` and `lubridate`.

They key data wrangling steps were applied to crime11to17 and central2016:

- Convert `category` and `location_type` to factor using `as.factor()`.

- Several of the variables contained no useful data for a civilian user. The variables `context`, `persistent_id`, `id` and `location_subtype` were removed with `select()` and variable names for `category` and `location_type` were changed to `crime_type` and `service` respectively, to make the dataset clearer using `rename()`.
- Checked columns for missing values using `sapply(crime11to17, function(x) sum(is.na(x)))`. Only the `outcome_status_category` and `outcome_status_date` contain NA values and there is no plan to use these in the scope of the current project.
- Use the `lubridate` functions `year()` and `month()` to create new columns called `year` and `month` from the original 'month' variable which was in format y-m.
- Create dataframe for monthly crime statistics. Abbreviate the `crime_type` into new variable called `crime_abb` to improve data visualisations using `case_when()` function nested within `mutate()`. The `group_by()` function is then used to sort by `year`, `month` and `crime_abb`. `summarize()` is used to get a count for each of the new groupings.
- For crime11to17, the process was repeated for annual crime statistics from 2011 to 2016, with 2017 removed as only a partial year.
- Dataframes for seasonal crime were created using case_when() to assign Spring (March to May), Summer (June to August), Autumn (September to November) or Winter (December to February) to a new **season** variable. Removed Mar-2017 from analysis as not sufficient data to constitute a full season.
- The 2016 inner borough dataset was subset to include only the crimes which occured within the 12 borough outlines using the `sp` package and GIS files of borough outlines.
- Save dataframes created using saveRDS for loading into analysis section.

6. Data Investigation

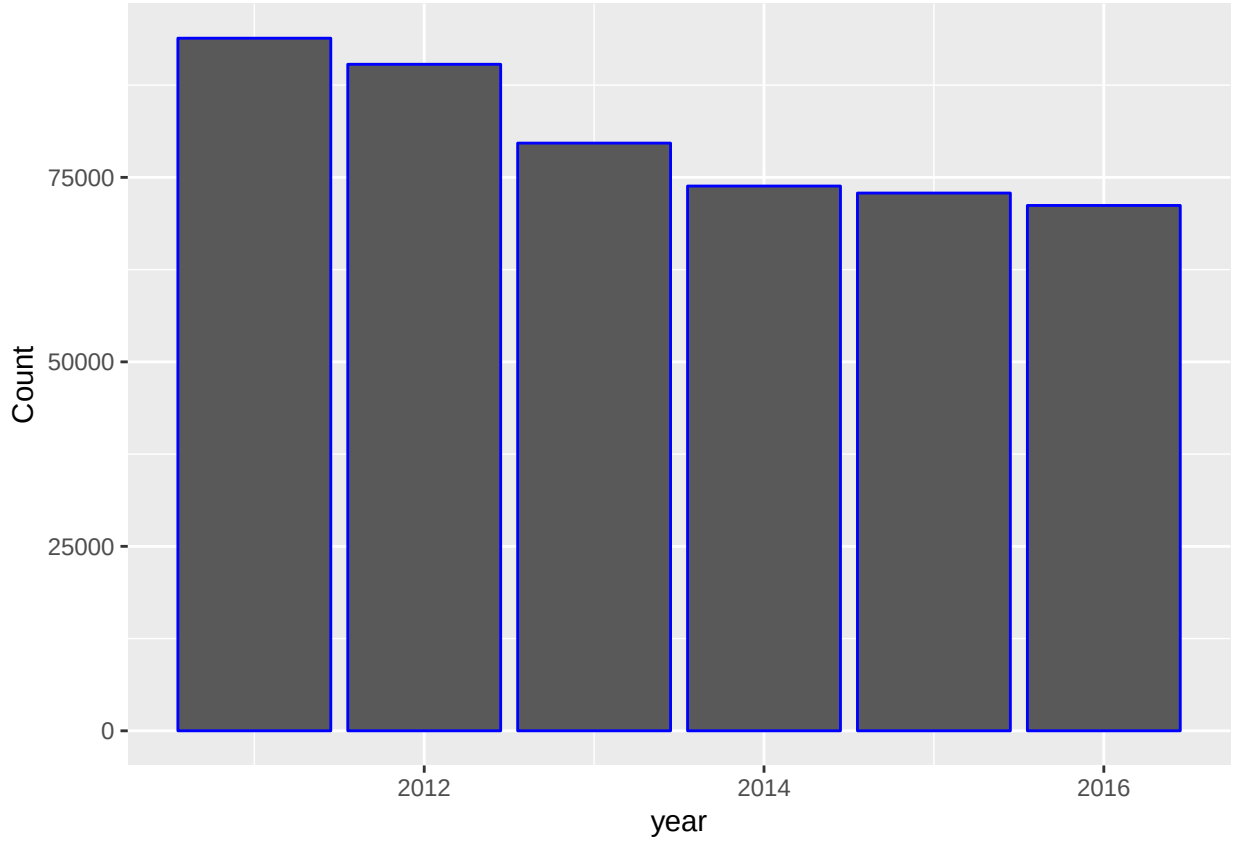
6.1 Central 2011-2017 Dataset

Initially the 2011-2017 dataset for central London was explored.The most prevalent crime in this period was anti-social-behaviour. This accounted for 23% of total recorded crime. The majority of the records are from the Police Force but a small portion were submitted by the British Transport Police who operate on the railways. These records account for only 4.3%. It would be interesting to explore the annual variation in street crimes using a bar chart. The total crimes per year are calculated using `group_by(year)` and `summarize(Count=n())`. 2017 is removed using `filter()`.

```
library(readr)
library(ggplot2)
library(dplyr)
library(tidyr)
library(lubridate)
library(scales)
library(rgdal)
library(broom)
library(classInt)
library(KernSmooth)
library(RColorBrewer)
library(leaflet)
library(sp)
library(maptools)
library(rgdal)
library(rgeos)

crime11to17 <- readRDS("crime11to17.rds")
```

```
monthly.crime <- readRDS("monthlycrime.rds")
annual.crime <- readRDS("annualcrime.rds")
seasonal.crime <- readRDS("seasonalcrime.rds")
total.year <- crime11to17 %>%
  group_by(year) %>%
  summarize(Count=n()) %>%
  filter(year != 2017) %>%
  as.data.frame()
```

```
ggplot(total.year, aes(x = year, y = Count)) +
  geom_col(col = "blue")
```



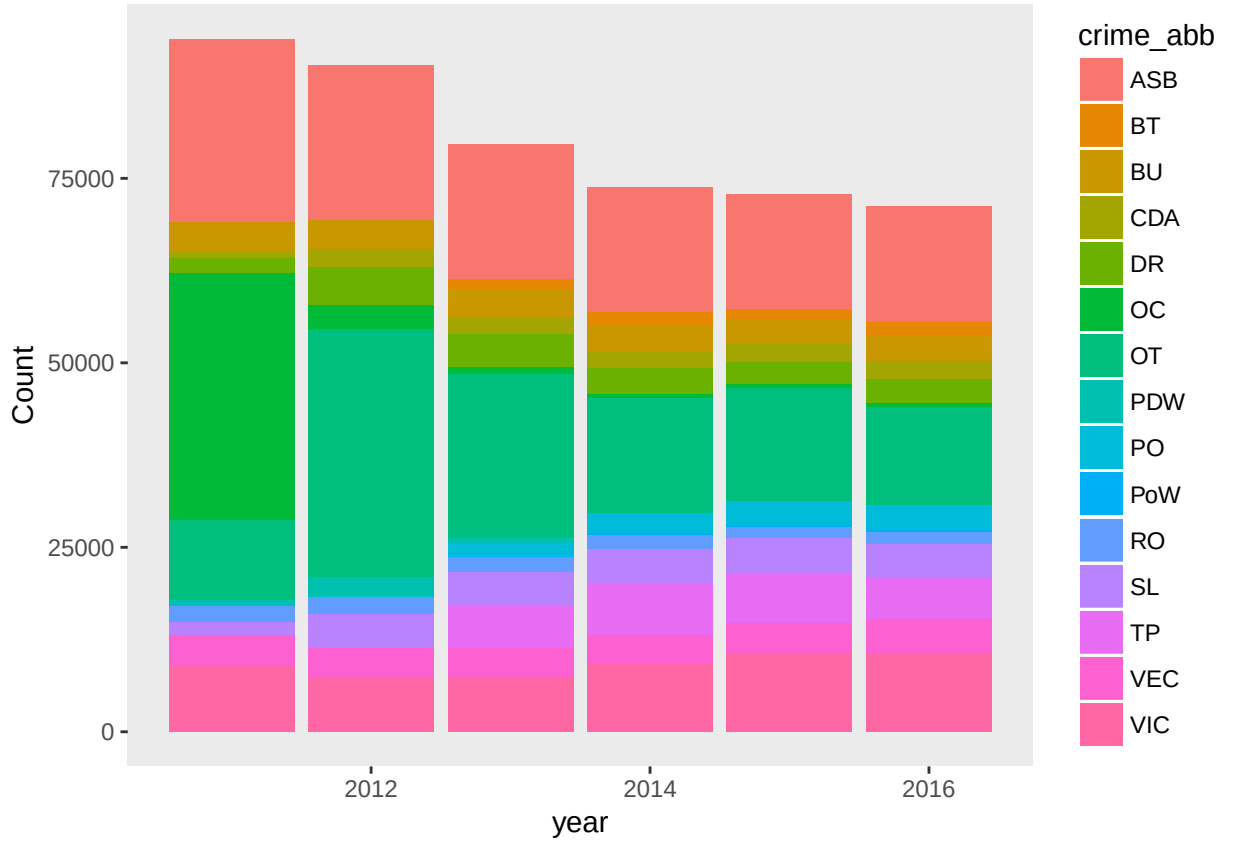
The abbreviations used are as follows:

Abbreviation	Crime
ASB	Anti-social behaviour
BT	Bicycle theft
BU	Burglary
CDA	Criminal damage and arson
DR	Drugs
OC	Other crime
OT	Other theft
PDW	Public disorder weapons
PO	Public order
PoW	Possesion of weapons
RO	Robbery

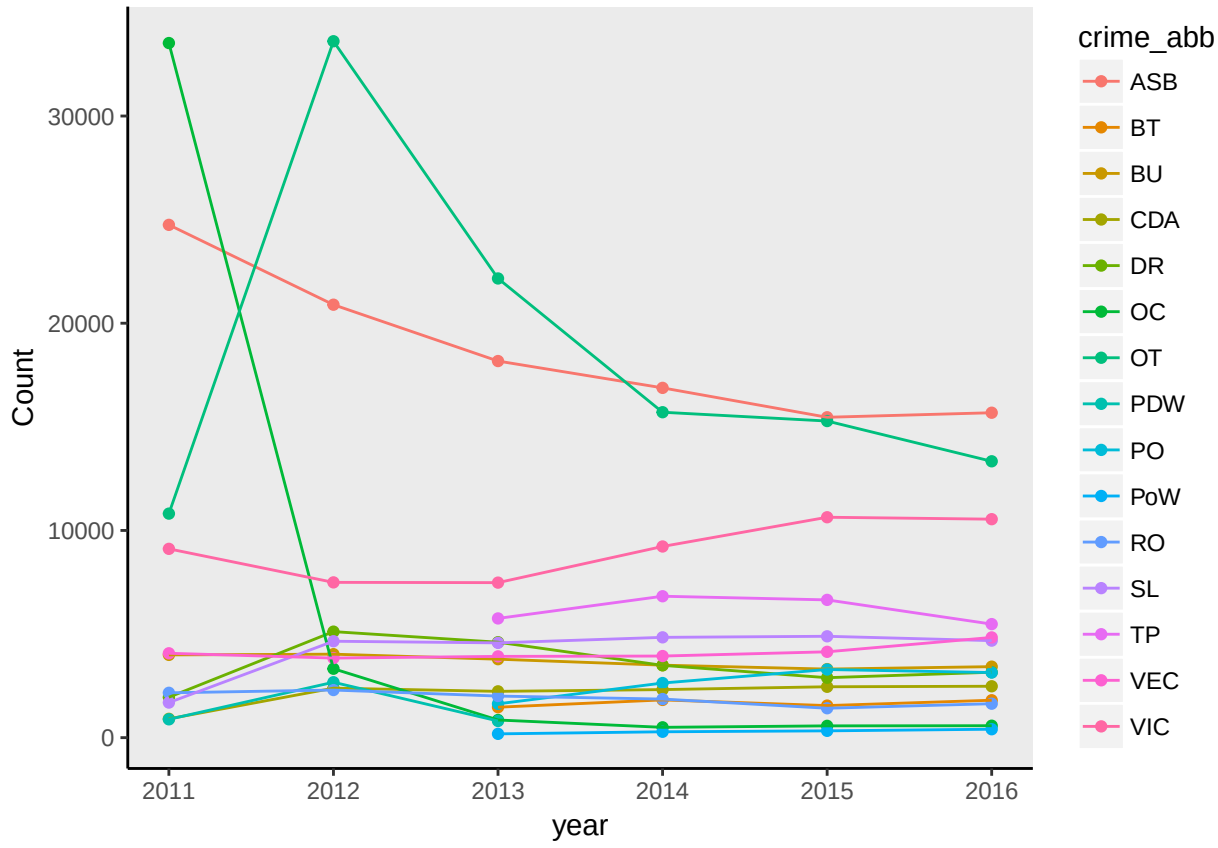
Abbreviation	Crime
SL	Shoplifting
TP	Theft from the person
VEC	Vehicle crime
VIC	Violent crime

Recorded annual street crime has been decreasing each year in London since 2011. There was a significant decrease between 2012 and 2013 of 11.8% and subsequent years show a reduction with a shallower negative gradient. The variation of specific crime frequency over time is displayed using a bar chart and a line plot with the `annual.crime` dataframe as input.

```
ggplot(annual.crime) +
  geom_bar(stat="identity",aes(x = year, y = Count,fill = crime_abb)) +
  theme(panel.grid = element_blank())
```




```
ggplot(annual.crime, aes(x = year, y = Count)) +
  geom_point(aes(col = crime_abb)) +
  geom_line(aes(col = crime_abb)) +
  theme(panel.grid = element_blank(), axis.line = element_line(colour = "black"))
```



Three categories have shown a strong decrease between 2011 and 2016 - other crime, other theft and anti-social behaviour. The anonymously large drop between 2011 and 2012 for other crime, and large increase of other theft between 2012 and 2013 may suggest a change in recording/reporting. Worryingly, violent crime displays a steady increase from 2013 onwards.

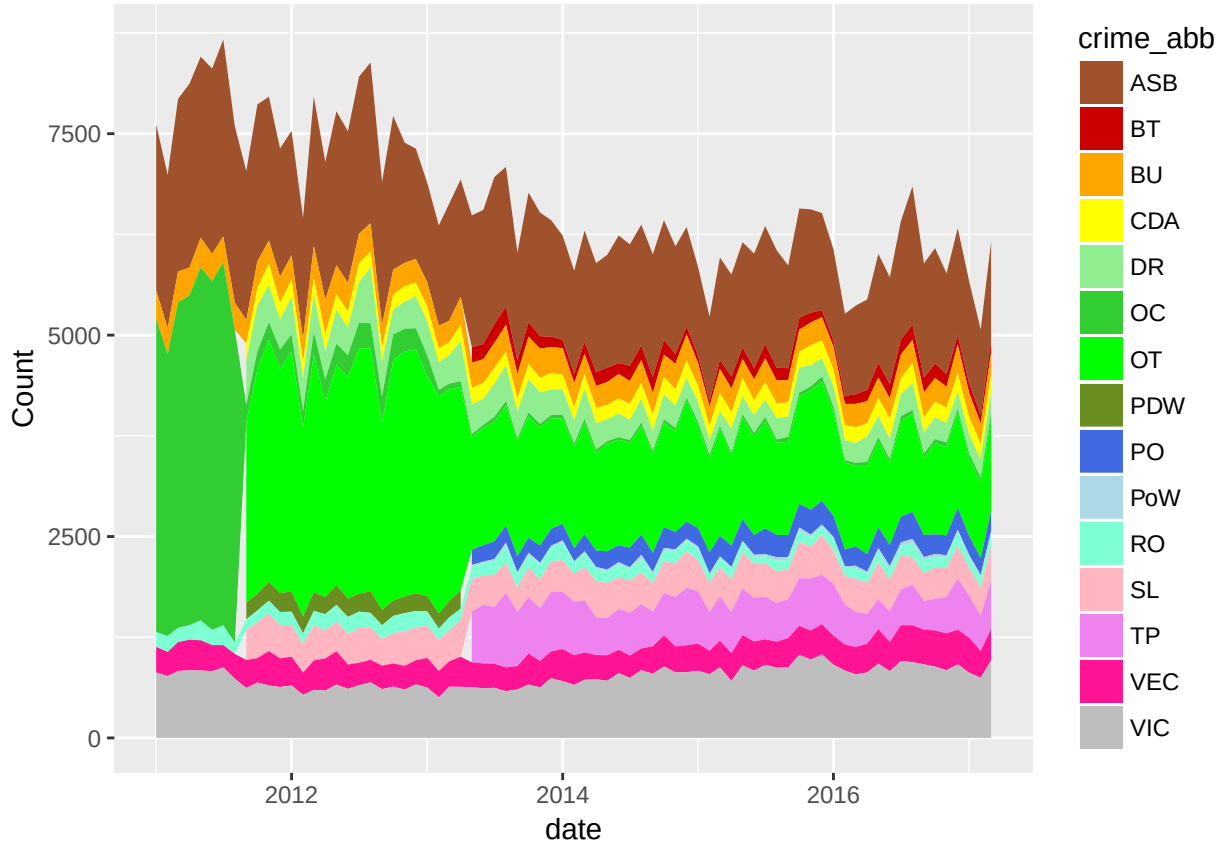
The `monthly.crime` dataframe is used to investigate time series. Initially, the date column needs to be set to the correct format using Lubridate.

```
monthly.crime <- monthly.crime %>%
  unite(date, month, year, sep = "-", remove = FALSE)
monthly.crime$date <- paste0("01-", monthly.crime$date)
monthly.crime$date <- dmy(monthly.crime$date)
```

A time series is plotted using geom_area() which stacks time series at each point. The colour scale for the crime categories is manually set using a vector due to the large number of categories.

```
crime.col <- c("ASB" = "sienna", "BT" = "red3", "BU" = "orange", "CDA" = "yellow", "DR" = "lightgreen",
```

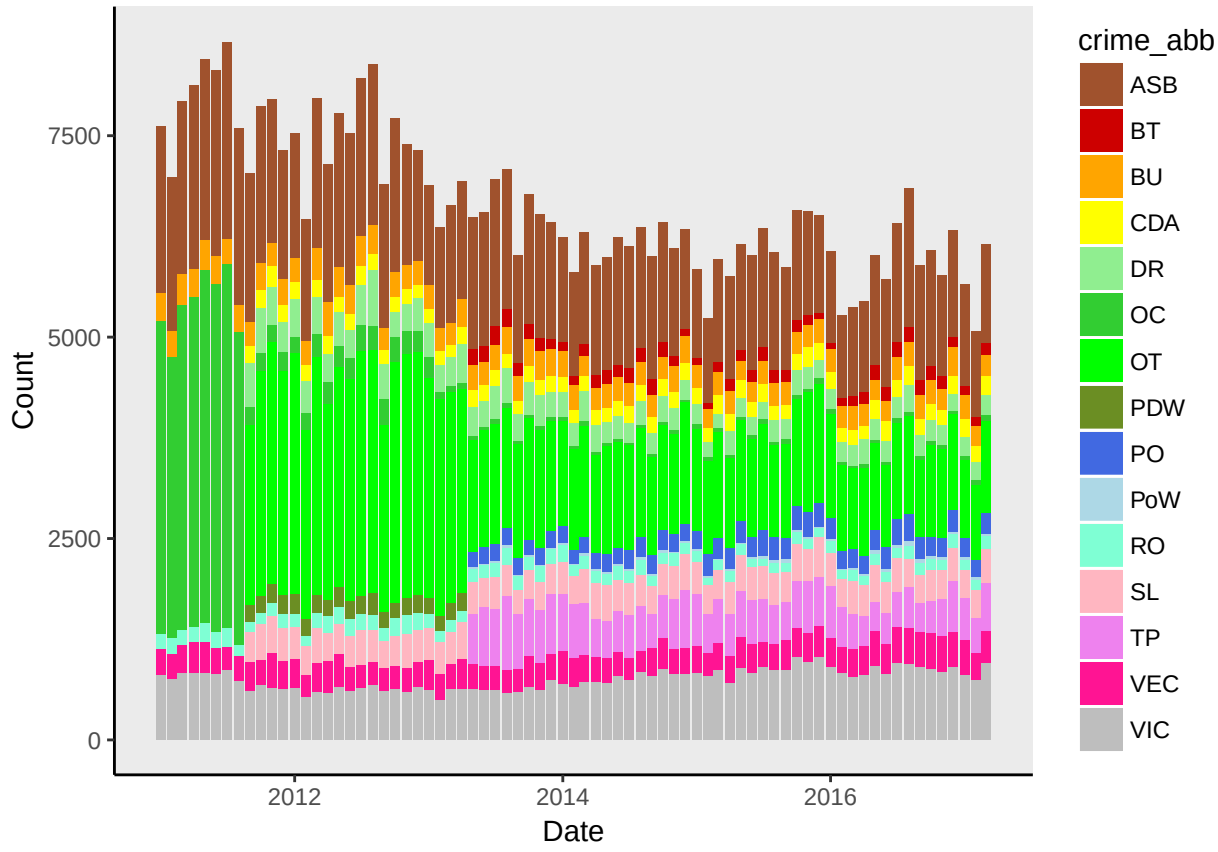
```
ggplot(monthly.crime, aes(x = date,  y = Count, fill = crime_abb)) +
  geom_area() +
  scale_fill_manual(values = crime.col)
```



This illustrates the changing classification of crimes over time and suggests there was a drive to classify 'other crimes' into different categories. 'Other theft' did not exist prior to late 2011 and seems to increase by a similar degree to the decrease in 'other crime'. Several other categories are added at this point: drugs, shoplifting, criminal damage and arson, and public disorder weapons. The final tranche of categories were introduced in early 2013 - public order, bicycle theft and theft from the person. By early 2017, the other-crime category was used only minimally.

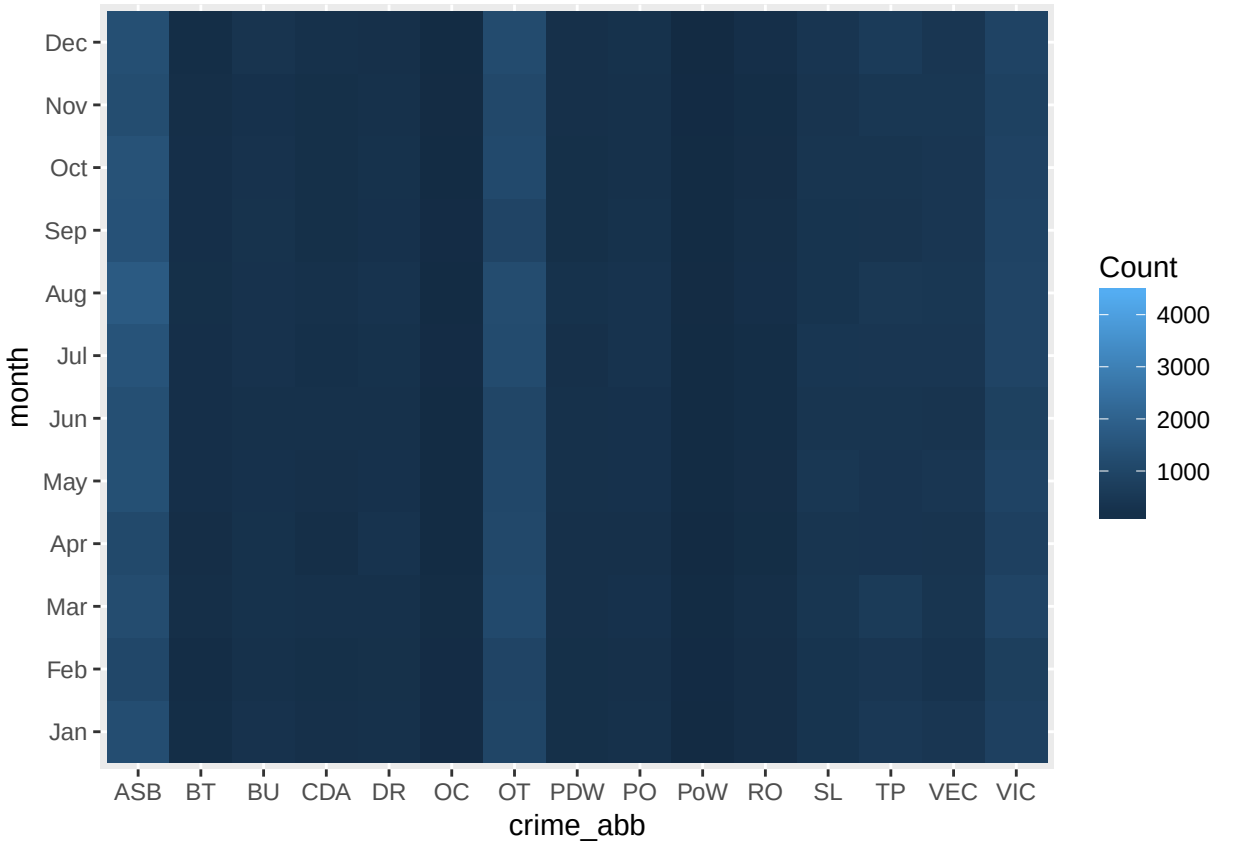
A plot is created using geom_bar() which suggests anti-social behaviour peaks each summer, but due to the scale used it is difficult to analyse it further.

```
ggplot(monthly.crime) +
  geom_bar(stat="identity",aes(x = date, y = Count,fill = crime_abb)) +
  theme(panel.grid = element_blank(), axis.line = element_line(colour = "black")) +
  xlab("Date") +
  ylab("Count") +
  scale_fill_manual(values = crime.col)
```



A heatmap can be used to get an overview of seasonal variation across all crime categories.

```
ggplot(monthly.crime,aes(x=crime_abb,y=month,fill=Count))+
  geom_tile(aes(fill=Count))
```

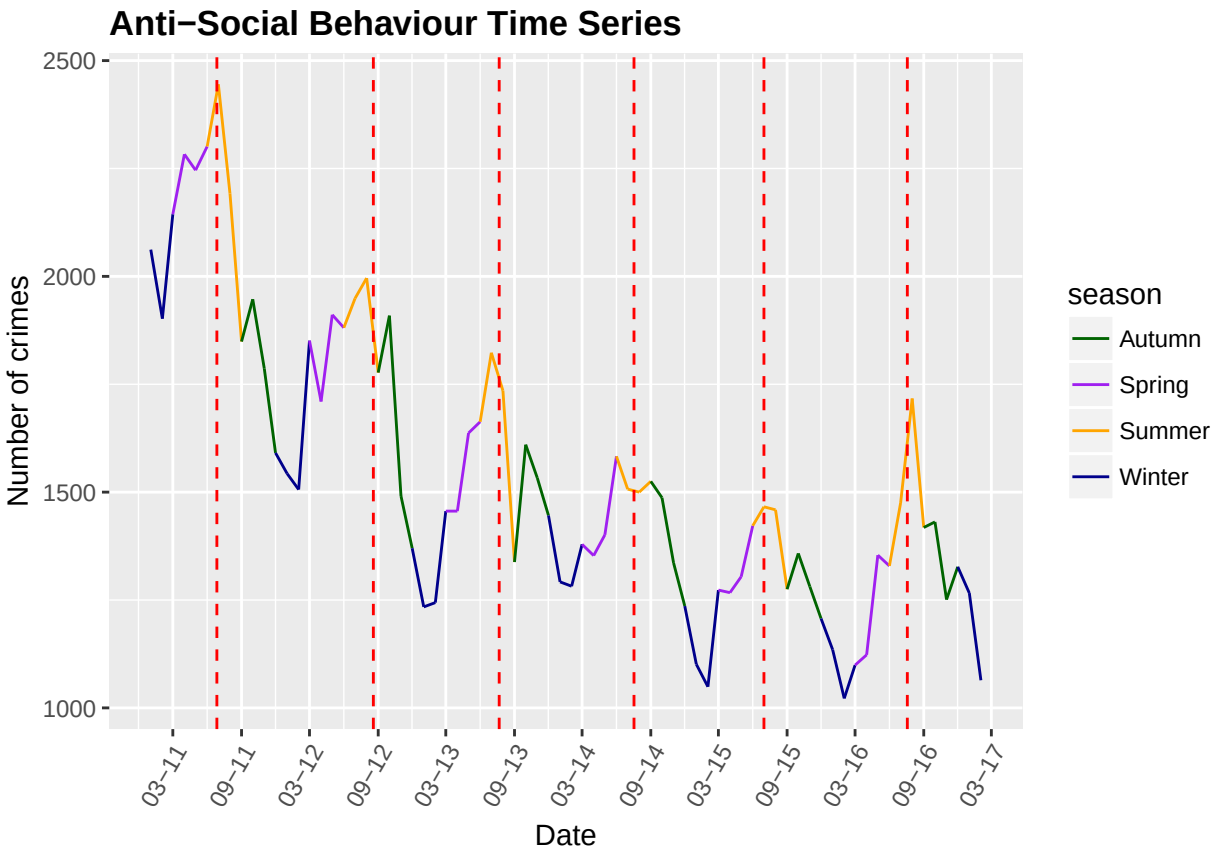


This clearly demonstrates the seasonal fluctuations in anti-social behaviour, with peaks each summer. Bicycle crime and violent crime also appear to be more common in May-July. Most other crimes show minimal seasonal variation.

These variations will be analysed in more detail using the `seasonal.crime` dataframe. Seasonal colours are set using a vector so the line segments can be individually coloured. Date is converted for time series use. A line plot is created with vertical lines to indicate the day with the highest temperature (red lines). This is created using a dataframe with the maximum London temperature for each year (`hottest.day`).

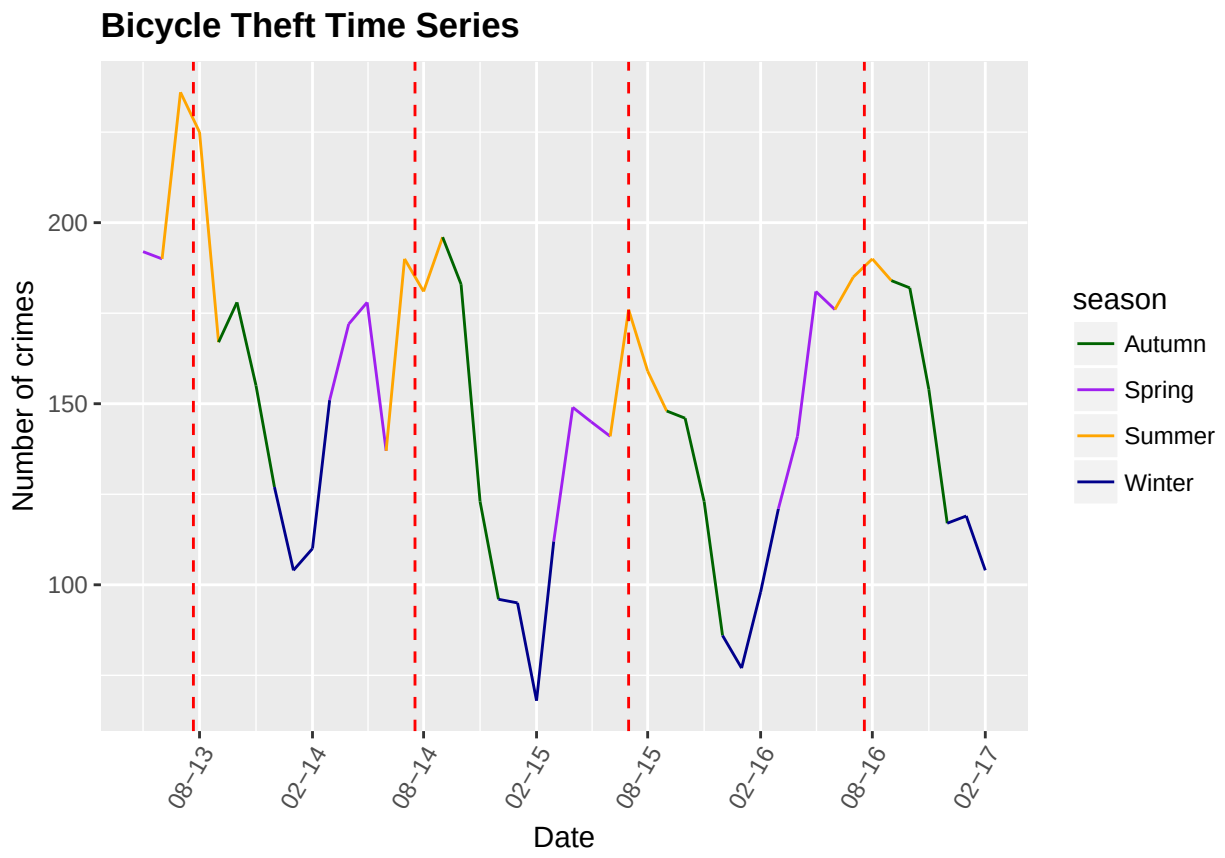
```
season.cols <- c("Summer" = "orange", "Autumn" = "darkgreen", "Winter" = "darkblue", "Spring" = "purple"
seasonal.crime$date <- paste0("01-", seasonal.crime$date)
seasonal.crime$date <- dmy(seasonal.crime$date)
ASB.season <- filter(seasonal.crime, crime_type == "anti-social-behaviour")

TDates <- c("2011-06-27", "2012-08-19", "2013-07-22", "2014-07-18", "2015-07-01", "2016-07-19")
HTemp <- c(30, 30, 33, 30, 35, 33)
hottest.day <- data.frame(x = TDates, y = HTemp)
```

```
ggplot(ASB.season, aes(x = date, y = Count)) +
  geom_line(aes(group = crime_type, col = season)) +
  scale_colour_manual(values = season.cols) +
  scale_x_date(date_breaks = "6 month", date_labels = "%m-%y") +
  ylab("Number of crimes") +
  xlab("Date") +
  ggtitle("Anti-Social Behaviour Time Series") +
  theme(axis.text.x=element_text(angle=60, hjust=1), plot.title = element_text(lineheight=.8, face="bol
  geom_vline(xintercept = as.numeric(as.Date(TDates)), linetype=2, col = "red")
```

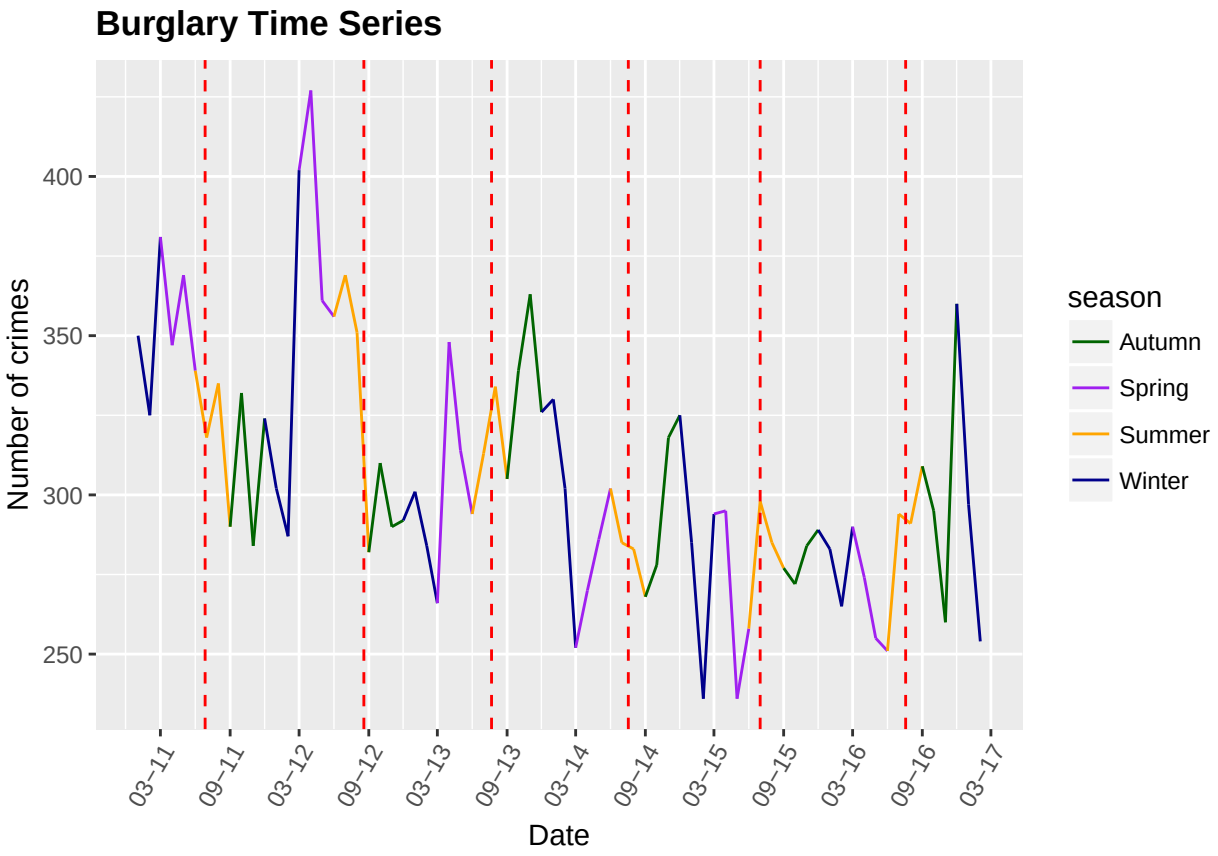


There has been a decrease in anti-social behaviour over time, with the exception of 2016 which showed a significant increase. Clear peaks are visible in the summer (orange segments), often coinciding with August when the school summer holidays occur. The peaks frequently coincide very closely to the hottest day of the year, with the exception of 2014. Minimum occurences are consistently just after 1st January. This process will be repeated for bicycle theft and burglaries.

```
BT.season <- filter(seasonal.crime, crime_type == "bicycle-theft")
ggplot(BT.season, aes(x = date, y = Count)) +
  geom_line(aes(group = crime_type, col = season)) +
  scale_colour_manual(values = season.cols) +
  scale_x_date(date_breaks = "6 month", date_labels = "%m-%y") +
  ylab("Number of crimes") +
  xlab("Date") +
  ggtitle("Bicycle Theft Time Series") +
  theme(axis.text.x=element_text(angle=60, hjust=1), plot.title = element_text(lineheight=.8, face="bol
  geom_vline(xintercept = as.numeric(as.Date(TDates)), linetype=2, col = "red")
```




```
BU.season <- filter(seasonal.crime, crime_type == "burglary")

ggplot(BU.season, aes(x = date, y = Count)) +
  geom_line(aes(group = crime_type, col = season)) +
  scale_colour_manual(values = season.cols) +
  scale_x_date(date_breaks = "6 month", date_labels = "%m-%y") +
  ylab("Number of crimes") +
  xlab("Date") +
  ggtitle("Burglary Time Series") +
  theme(axis.text.x=element_text(angle=60, hjust=1), plot.title = element_text(lineheight=.8, face="bol
    geom_vline(xintercept = as.numeric(as.Date(TDates)), linetype=2, col = "red")
```



Bicycle theft statistics are only available from 2013 and they have been fairly steady from 2014 onwards. The time series is cyclical with peaks generally corresponding reasonably (within 3 weeks) to the hottest day of the year, and minimum recorded incidences always occuring in the winter.

Burglaries have generally decreased since 2012, though there was a surge in Spring 2012 and another peak in Autumn 2016. There is less of a pattern visible in burglary occurences, though there is frequently a peak in the run up to Christmas followed by a sharp decrease over New Year. There is no clear pattern between high temperatures and burglary.

The Leaflet package can be used to visualise the location of crime occurences. A reduced dataset is required and is only useful if the data is filtered to a single year and single crime, due to overlapping or proximal datapoints. Jittering could be used to improve this.

```
VEC.2016 <- crime11to17 %>%
filter(year == 2016, crime_type == 'vehicle-crime')

VEC.2016 %>%
  leaflet() %>%
  addTiles() %>%
  addMarkers(~longitude, ~latitude)
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed, pleas
```

It would be valuable to be able to visualise the density of crimes rather than individual occurences. A dataframe is created from the original dataset containing only bicycle theft. Jitter is then added to the latitude and longitude. Leaflet can be used to visualise the output.

```
bike <- crime11to17 %>%
  filter(crime_type == 'bicycle-theft')
```

```
for(i in 1:length(bike$latitude)){
  bike$longitude[i] <- bike$longitude[i]+runif(1,0.001,0.004)
  bike$latitude[i] <- bike$latitude[i]+runif(1,0.001,0.004)}
```

```
leaflet(bike) %>%
  addTiles() %>%
  addCircles(~longitude,~latitude)
```

An improvement is visible but it's still difficult to interpret. To identify crime hotspots, geographical areas of higher than average crime, kernel density estimation (KDE) or chloropleth maps may produce a more useful output.

Kernel Density Estimation (KDE) is a way to estimate the probability density function of a random variable, inferences about the population are made based on a finite data sample. This method requires the `sp`,`KernSmooth` and `leaflet` packages. The latitude and longitude for bicycle thefts are saved in a separate dataframe. KDE is carried out using bkde2D() with the dataframe, bandwidth and gridsize as input. The contour lines created are then converted to spatial polygons.

```
latlong <- select(bike, longitude, latitude)
kde <- bkde2D(latlong,
              bandwidth=c(.0055, .0048), gridsize = c(75,75))
cl <- contourLines(kde$x1 , kde$x2 , kde$fhat)
levs <- as.factor(sapply(cl, `[[`, "level"))
nlev <- length(levels(levs))
pgons <- lapply(1:length(cl), function(i) Polygons(list(Polygon(cbind(CL[[i]]$x, CL[[i]]$y))), ID=i))
spgons <- SpatialPolygons(pgons)
leaflet(spgons) %>% addTiles() %>%
  addPolygons(color = heat.colors(nlev, NULL)[levs])
```

This heat map shows a concentration of bicycle thefts in Covent Garden and on the South Bank, two of the most popular tourist areas in London. Whilst this method communicates the information effectively, it would be far more valuable if it covered a larger area. Weakness of KDE include the smooth which occurs between areas and crimes indicated in areas where it is not possible for crimes to occur [1].

6.2 Inner Boroughs 2016 Dataset

The next step is to carry out analysis of the 2016 datasets.

```
crime16 <- readRDS("crime16.rds")
monthlycrime16 <- readRDS("monthlycrime16.rds")
seasonalcrime16 <- readRDS("seasonalcrime16.rds")
```

To continue with the KDE map creation process, bicycle theft will be repeated again using the more extensive dataset.

```
bike16 <- crime16 %>%
  filter(crime_type == 'bicycle-theft')

for(i in 1:length(bike16$latitude)){
  bike16$longitude[i] <- bike16$longitude[i]+runif(1,0.001,0.004)
  bike16$latitude[i] <- bike16$latitude[i]+runif(1,0.001,0.004)}

latlong <- select(bike16, longitude, latitude)
kde <- bkde2D(latlong,
              bandwidth=c(.0055, .0048), gridsize = c(75,75))
cl <- contourLines(kde$x1 , kde$x2 , kde$fhat)
```

```
levs <- as.factor(sapply(cl, `[[`, "level"))
nlevs <- length(levels(levs))
pgons <- lapply(1:length(cl), function(i)
  Polygons(list(Polygon(cbind(cl[[i]]$x, cl[[i]]$y))), ID=i))
spgons = SpatialPolygons(pgons)

leaflet(spgons) %>% addTiles() %>%
  addPolygons(color = heat.colors(nlevs, NULL)[levs])
```

The heat map shows the highest concentration of crimes around Russell Square and Covent Garden, in close proximity to three universities: University College Longon, Birkbeck and School of Oriental and Asian Studies. Shoreditch and Dalston are also centres for bike theft. With the more extensive dataset, the South Bank does not stand out as a hotspot for bike theft.

This process was subsequently repeated for burglaries in 2016.

```
burglary16 <- crime16 %>%
  filter(crime_type == 'burglary')
for(i in 1:length(burglary16$latitude)){
  burglary16$longitude[i] <- burglary16$longitude[i]+runif(1,0.001,0.004)
  burglary16$latitude[i] <- burglary16$latitude[i]+runif(1,0.001,0.004)}
latlong <- select(burglary16, longitude, latitude)
kde <- bkde2D(latlong,
              bandwidth=c(.0055, .0048), gridsize = c(75,75))
cl <- contourLines(kde$x1 , kde$x2 , kde$fhat)
levs <- as.factor(sapply(cl, `[[`, "level"))
nlevs <- length(levels(levs))
pgons <- lapply(1:length(cl), function(i)
  Polygons(list(Polygon(cbind(cl[[i]]$x, cl[[i]]$y))), ID=i))
spgons = SpatialPolygons(pgons)

leaflet(spgons) %>% addTiles() %>%
  addPolygons(color = heat.colors(nlevs, NULL)[levs])
```

This KDE map shows a high concentration of burglaries in Fitzrovia and into Soho, with Brixton, Paddington and Shoreditch also showing minor hotspots.

Repeat the KDE map process for theft from the person

```
TFTP16 <- crime16 %>%
  filter(crime_type == 'theft-from-the-person')
for(i in 1:length(TFTP16$latitude)){
  TFTP16$longitude[i] <- TFTP16$longitude[i]+runif(1,0.001,0.004)
  TFTP16$latitude[i] <- TFTP16$latitude[i]+runif(1,0.001,0.004)}
latlong <- select(TFTP16, longitude, latitude)
kde <- bkde2D(latlong,
              bandwidth=c(.0055, .0048), gridsize = c(75,75))
cl <- contourLines(kde$x1 , kde$x2 , kde$fhat)
levs <- as.factor(sapply(cl, `[[`, "level"))
nlevs <- length(levels(levs))
pgons <- lapply(1:length(cl), function(i)
  Polygons(list(Polygon(cbind(cl[[i]]$x, cl[[i]]$y))), ID=i))
spgons = SpatialPolygons(pgons)
```

```
leaflet(spgons) %>% addTiles() %>%
  addPolygons(color = heat.colors(nlevs, NULL)[levs])
```

The KDE map shows a very concentrated hotspot around Soho and Covent Garden, and a smaller one over Shoreditch. These three areas contain the high density of nightclubs and bars in London.

Returning to general analyis, it would be useful to normalise crime occurences by the resident population in each borough. Population figures are taken from the 2013 census data and loaded from csv.

```
employment <- read_csv("employment_london.csv")
```

```
## Parsed with column specification:
## cols(
##   Borough = col_character(),
##   `Working age pop` = col_number(),
##   `EP number` = col_number(),
##   `SE number` = col_number(),
##   `Area (sq mi)` = col_double(),
##   Population = col_integer(),
##   `Median Salary 14-15` = col_number(),
##   `Less LLW 15` = col_integer()
## )
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 26 parsing failures.
## row # A tibble: 5 x 5 col     row   col  expected     actual                     file expected   <int
## ... ................. ... ........................................................... ........ .....
## See problems(...) for more details.
```

```
employment <- employment %>%
  as.data.frame() %>%
  na.omit()
```

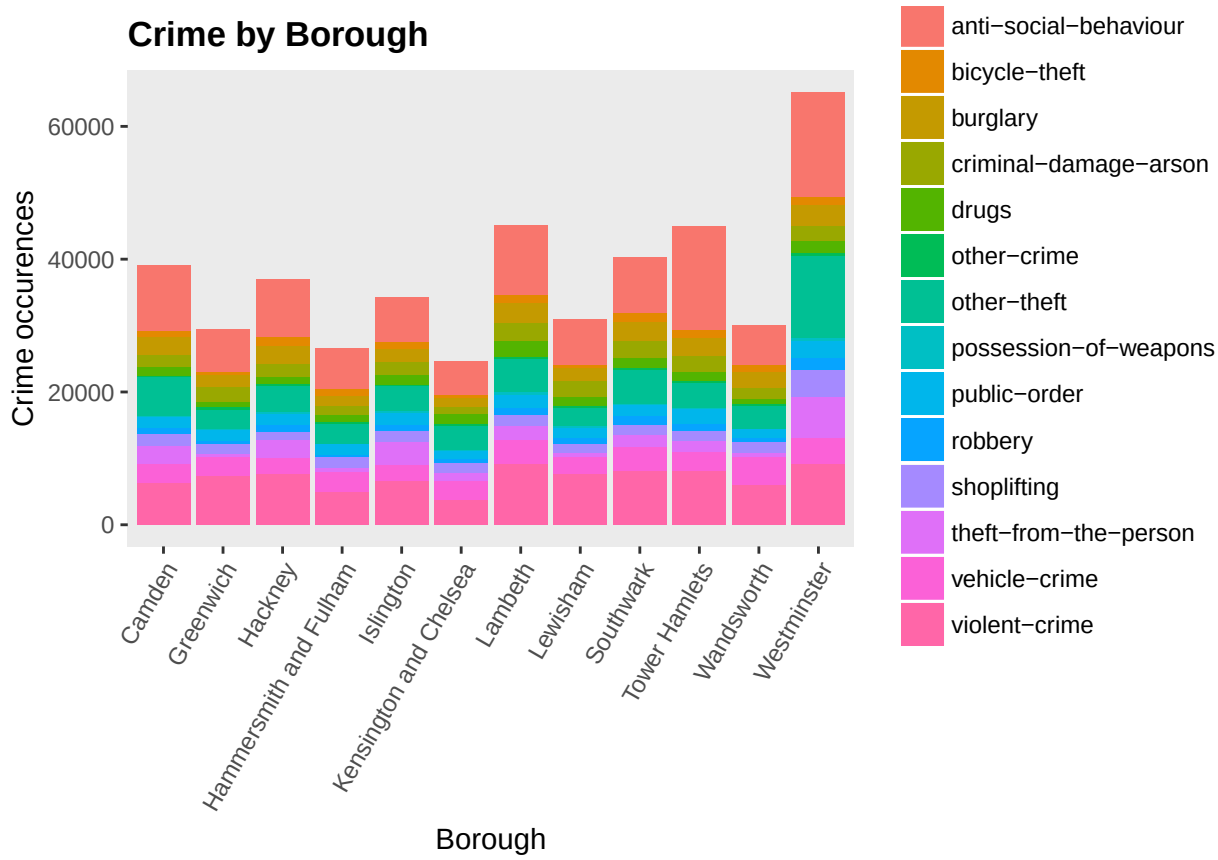
Frequency of crimes by borough can now be determined and normalised by the population (per 100 people). This is carried out by adding the `Population` variable from the `employment` dataframe as a new variable to the `borough.rate` dataframe, then using `mutate()` to create a normalisation variable (`norm`) by dividing count by population and multiplying by 100.

```
borough.rate <- crime16 %>%
  group_by(borough) %>%
  summarize(Count=n())

borough.rate$pop <- employment$Population
borough.rate <- borough.rate %>%
  mutate(norm = Count/pop*100) %>%
  arrange(desc(norm))
```

Overall, the borough with fewest crimes per 100 people in 2016 was Wandsworth with 9.66. Westminster had the highest crime rate with 28.7 crimes per 100 people. This is likely to be highly skewed due to both the number of tourists and workers who are not accounted for in this normalisation. For specific crime occurences, let's first look without the normalisation using a bar plot.

```
crime.rate.by.borough <- crime16 %>%
  group_by(borough, crime_type) %>%
  summarize(Count=n())
```

```
ggplot(crime.rate.by.borough) +
  geom_bar(stat="identity",aes(x = borough, y = Count,fill = crime_type)) +
  theme(panel.grid = element_blank(), axis.text.x=element_text(angle=60, hjust=1), plot.title = element_
  ggtitle("Crime by Borough") +
  ylab("Crime occurences") +
  xlab("Borough")
```



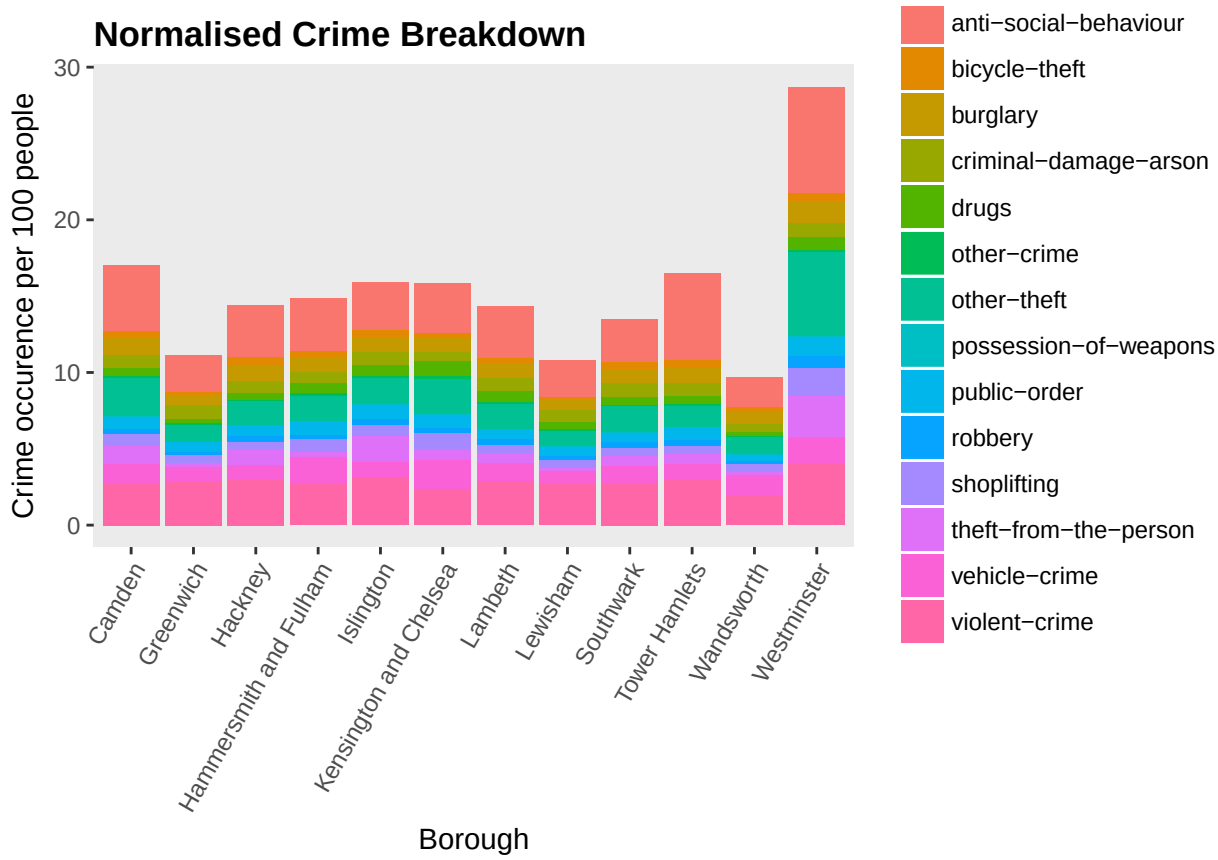
There is significant variation between crime in the boroughs with Westminster having the most recorded crimes particularly for other-theft and theft-from-the-person, and Kensingston and Chelsea the least.

The normalisation process will be repeated to the crime categories. Colours are specified for the boroughs to make identification easier.

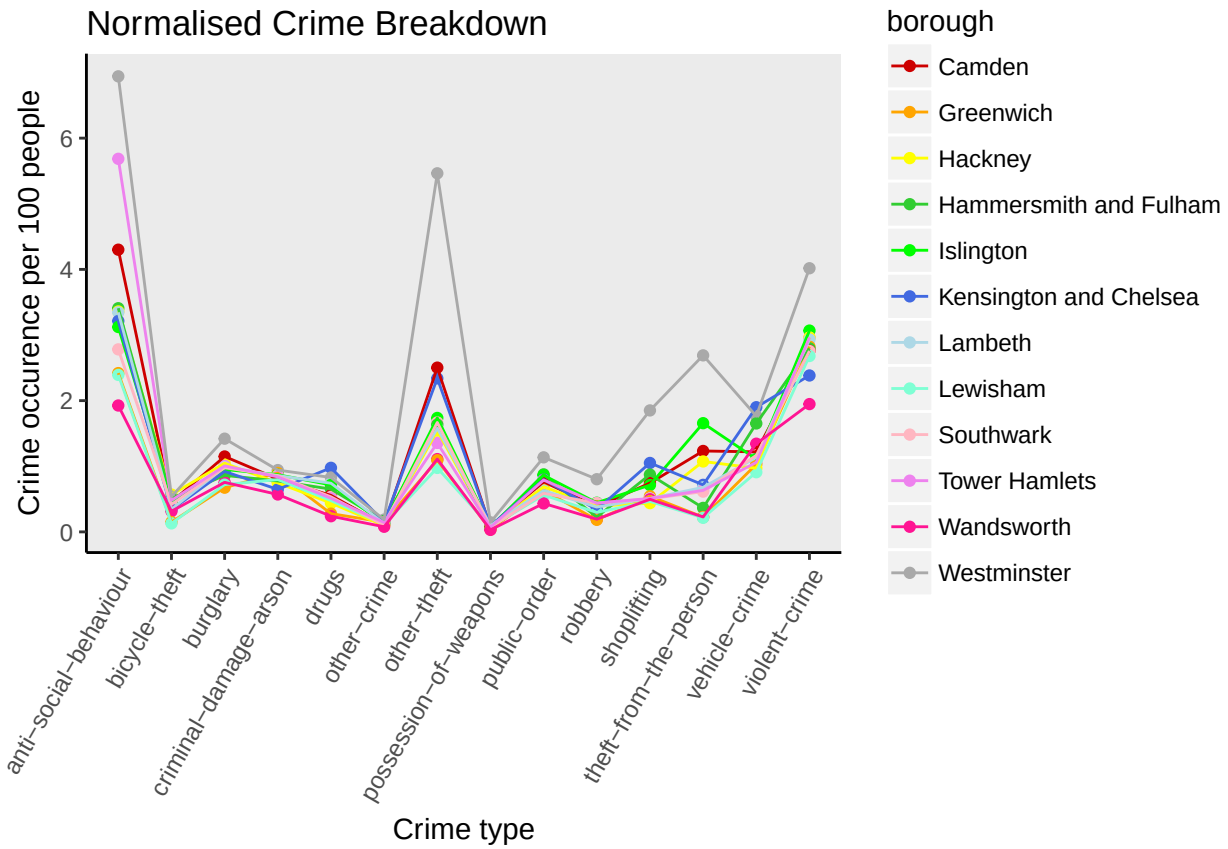
```
boroughpop <- select(borough.rate, borough, pop)
crime.borough.norm <- left_join(crime.rate.by.borough, boroughpop) %>%
  mutate(norm = Count/pop*100) %>% arrange(desc(norm))
```

```
## Joining, by = "borough"
```

```
borough.col <- c("Camden" = "red3", "Greenwich" = "orange", "Hackney" = "yellow", "Hammersmith and Fulha

ggplot(crime.borough.norm) +
  geom_bar(stat="identity",aes(x = borough, y = norm, fill = crime_type)) +
  theme(panel.grid = element_blank(), axis.text.x=element_text(angle=60, hjust=1), plot.title = element_
  ggtitle("Normalised Crime Breakdown") +
  ylab("Crime occurence per 100 people") +
  xlab("Borough")
```



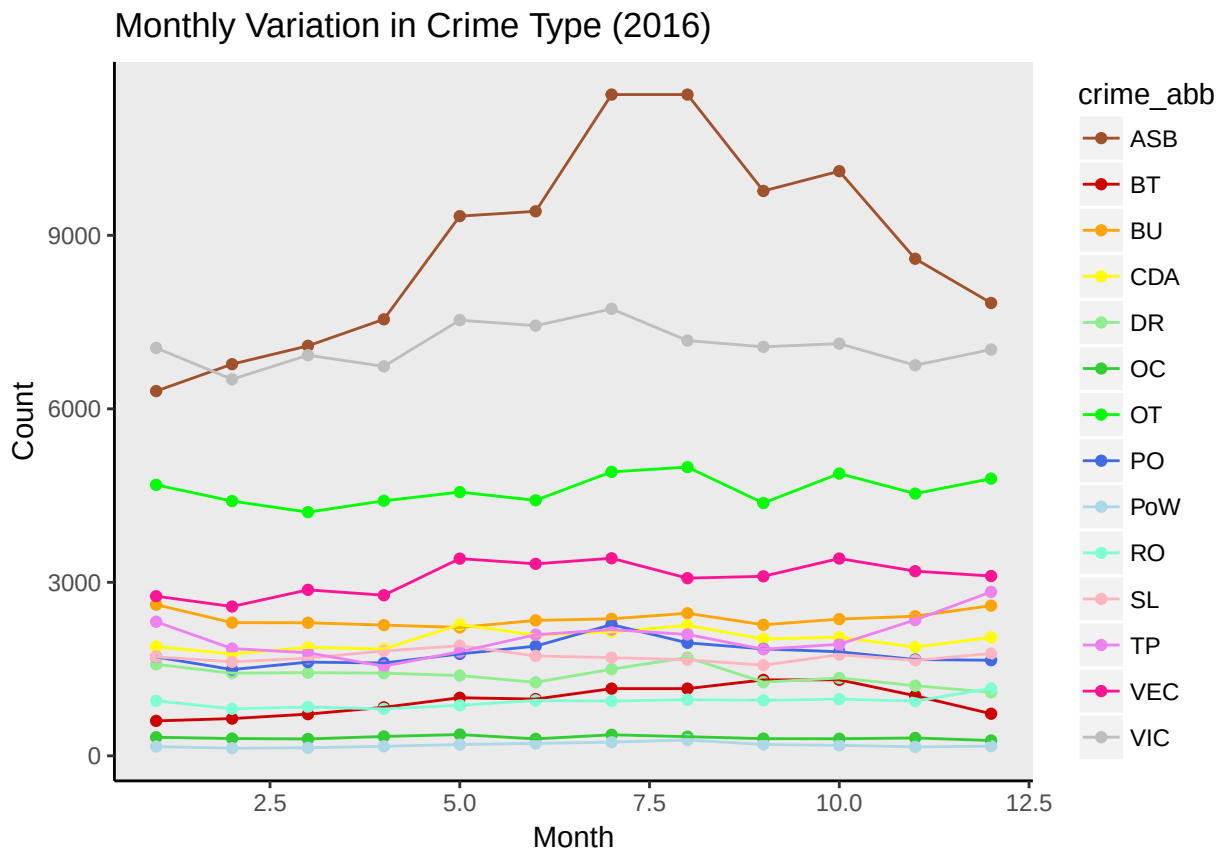

```
ggplot(crime.borough.norm, aes(x = crime_type, y = norm)) +
  geom_point(aes(col = borough)) +
  geom_line(aes(col = borough, group = borough)) +
  xlab("Crime type") +
  ylab("Crime occurence per 100 people") +
  ggtitle("Normalised Crime Breakdown") +
  scale_colour_manual(values = borough.col) +
  theme(panel.grid = element_blank(), axis.line = element_line(colour = "black"), axis.text.x=element_t
```



Relative to its population, Wesminster has even more crimes per 100 residents.This does take the large number of tourists and workers. This is three times the number of crimes recorded in Wandsworth per 100 residents, which has consistently low records all crimes. The line plot shows Westminster is considerably elevated above the main trend for other-theft, shoplifting, theft-from-the-person and violent-crime. Interestingly, this plot also shows that Kensington and Chelsea has the largest number of drug offences once normalised.

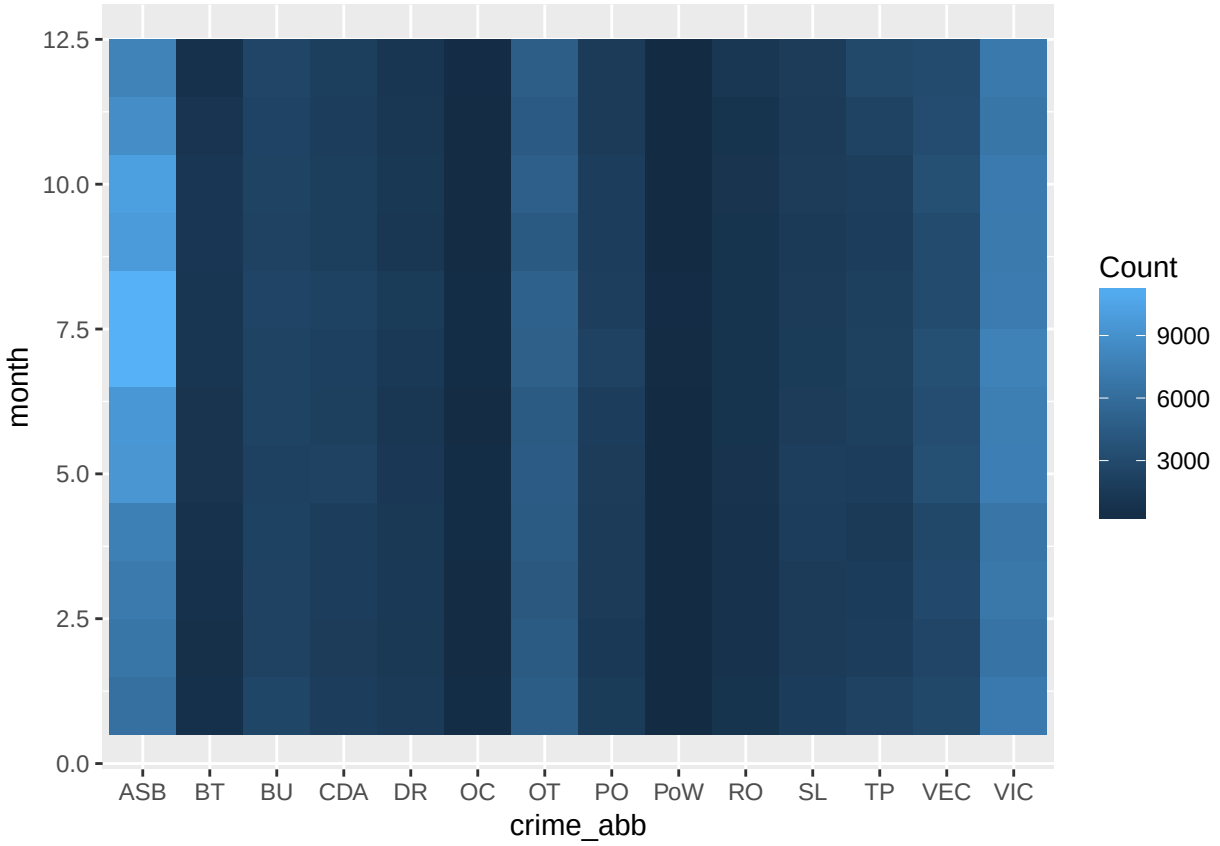
The analysis of this dataset has so far focussed on the full dataset, a quick analysis of the `monthlycrime16` dataset will be carried out.

```
ggplot(monthlycrime16, aes(x = month, y = Count)) +
  geom_point(aes(col = crime_abb)) +
  geom_line(aes(col = crime_abb)) +
  xlab("Month") +
  theme(panel.grid = element_blank(), axis.line = element_line(colour = "black")) +
  scale_colour_manual(values = crime.col) +
   ggtitle("Monthly Variation in Crime Type (2016)")
```



The anti-social behaviour peak in summer is very clear still across the inner borough dataset, as is a peak in violent crime and vehicle crime from May through to August. A tiled heatmap also allows these seasonal trends to be easily identified.

```
ggplot(monthlycrime16,aes(x=crime_abb,y=month,fill=Count))+
  geom_tile(aes(fill=Count))
```



Another option for interactively displaying crime density is to use chloropleth mapping. The borough polygons needs to be loaded and then subset to include the 12 inner boroughs of interest.

```
boroughs <- readOGR(dsn = "statistical-gis-boundaries-london/ESRI", "London_Borough_Excluding_MHW", verb
central <- boroughs %>%
  subset(NAME %in% c('Camden','Greenwich','Hackney', 'Hammersmith and Fulham','Islington', 'Kensington a
```

The `crime16` data is then converted to a SpatialPointsDataFrame with latitude - longitude projection and then the borough boundaries are transformed to geographic coordinates.

```
coords <- SpatialPoints(crime16[,c("longitude","latitude")])
crime16 <- SpatialPointsDataFrame(coords, crime16)
proj4string(crime16) <- CRS("+init=epsg:4326")
central <- spTransform(central, CRS("+init=epsg:4326"))
central@proj4string
```

```
## CRS arguments:
##  +init=epsg:4326 +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84
## +towgs84=0,0,0
```

Bicycle theft will be looked at first by subsetting those offences and counting the number of points within boroughs using `over()`, then the occurences of bicycle theft are added to the central SpatialPolygons dataframe using `unlist()`.

```
bicycle_theft <- crime16[crime16$crime_type == "bicycle-theft",]
proj4string(crime16) <- proj4string(central)
proj4string(bicycle_theft) <- proj4string(central)
pointsinpolygon <- over(SpatialPolygons(central@polygons), SpatialPoints(bicycle_theft), returnList = TR
```

```
central$bicycle_theft <- unlist(lapply(pointsinpolygon, length))
```

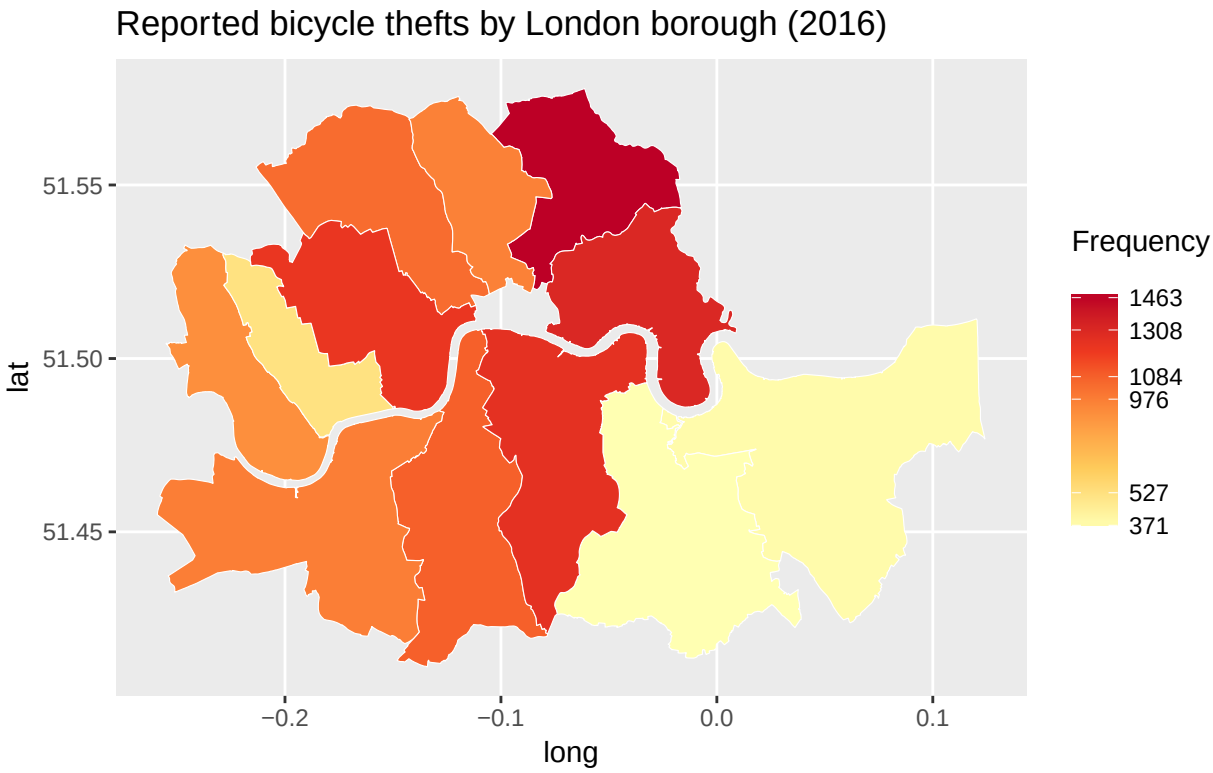
The thematic ranges are calulcated using natural breaks with `classIntervals()` and the central shapefile is fortified for plotting using ggplot2.

```
classes <- classIntervals(central$bicycle_theft, n=5, style="jenks")
central_f <- fortify(central, region="NAME")
central_f <- left_join(central_f, central@data, by = c("id" = "NAME"))
```

```
## Warning: Column `id`/`NAME` joining character vector and factor, coercing
## into character vector
```

A chloropleth map can then be plotted for bicycle thefts in 2016 [2].

```
ggplot() +
  geom_polygon(data = central_f, aes(x = long, y = lat, group = group, fill = bicycle_theft), color = "
  coord_map() +
  scale_fill_gradientn('Frequency\n', colours = brewer.pal(5,"YlOrRd"), breaks = classes$brks) +
  labs(title = "Reported bicycle thefts by London borough (2016)")
```



This would be better if it was interactive. First, the text for a pop-up box needs to be set using html, then the colour table using `findColours()` and the labels for the legend.

```
central_popup <- paste0("<strong>Borough: </strong>",
                        central$NAME,
                        "<br><strong>Offences (2016): </strong>",
                        central$bicycle_theft)
colcode2 <- findColours(classes, c("darkgrey", "yellow", "orange", "red", "sienna"))
```

```
breaks <- round(classes$brks, 1)
labels = matrix(1:(length(breaks)-1))
for(j in 1:length(labels)){labels [j] =
  paste(as.character(breaks[j]),"-",as.character(breaks[j+1]))}
```

An interactive chloropleth plot can then be made using OpenStreetMap raster layer in Leaflet. When you roll over the borough it provides the number of records of that offence in a pop up box.

```
leaflet(data = central) %>%
  addProviderTiles("OpenStreetMap.BlackAndWhite") %>%
  addPolygons(data = central,
              fillColor = colcode2,
              fillOpacity = 0.6,
              color = "#636363",
              weight = 2,
              popup = central_popup)  %>%
  addLegend(position = "bottomright",
            colors = c("darkgrey", "yellow", "orange", "red", "sienna"),
            labels = labels,
            opacity = 0.8,
            title = "Number of bike thefts")
```

Hackney is the borough with the highest recorded number of bicycle thefts at 1463, Lewisham is the lowest at 371. This is repeated for violent crime and for burglaries.

Violent crimes peak in Westminster and Lambeth. There is a clear split between higher values in central, south and east London, with lower values in north and west london. Again, burglaries are most prevalent in Westminster and Lambeth, with the lowest in Kensington and Chelsea and Hammersmith and Fulham.

In terms of tourists, the crime which may potentially impact them directly is theft from the person, so one more interactive chloropleth will be built.

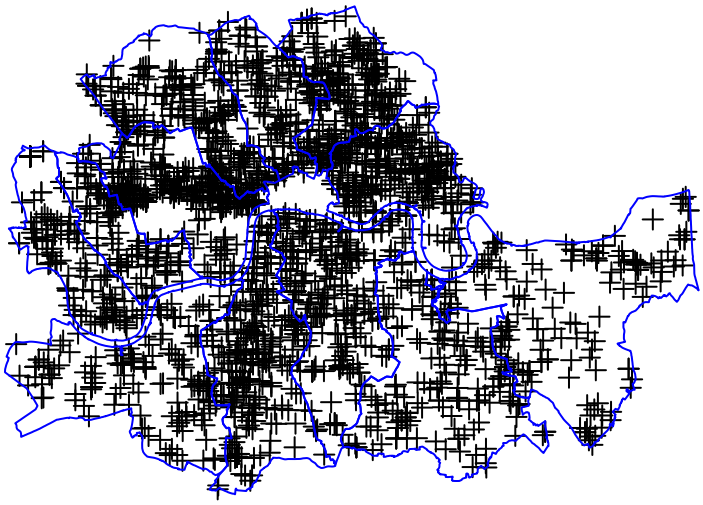
The highest occurences of theft from a person are in Westminster, 6101 records in 2016. This borough contains many of the capitals major tourist attractions including Buckingham Palace, London Zoo and Covent Garden.

6.3 Spatial Analysis

Point pattern analysis is the evaluation of the pattern, or distribution, of a set of points on a surface. It can refer to the actual spatial or temporal location of these points or also include data from point sources [3].

The first steps are to load addional libraries `raster`, `gridExtra`, `spatstat` and `tidyverse`, then ensure that the same coordinate reference system (CRS) is used for both crime points and borough polygons - WGS84 was selected. The crime data for the inner boroughs for 2016 is filtered to include only burglaries in November 2016 as this method did not function with the full unfiltered dataset. The coordinates for the crime locations and boroughs need to be obtained, these are then plotted to check the process has worked. The bulk of the workflow in this section is based on [4].

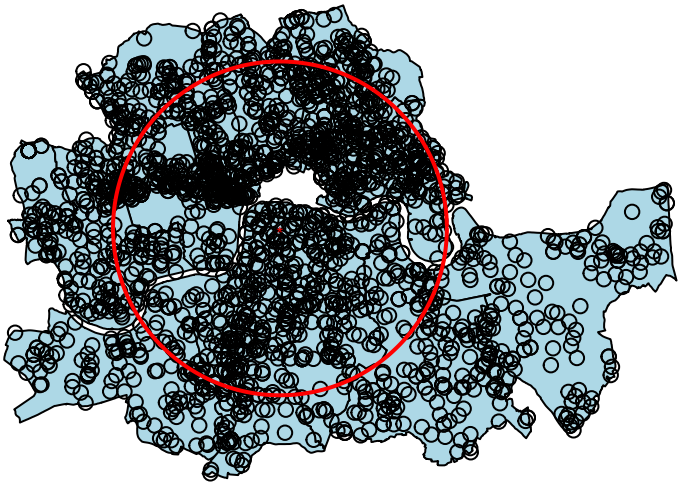
```
## Warning: package 'spatstat' was built under R version 3.4.1
```



```
## [1] 2414    2
```

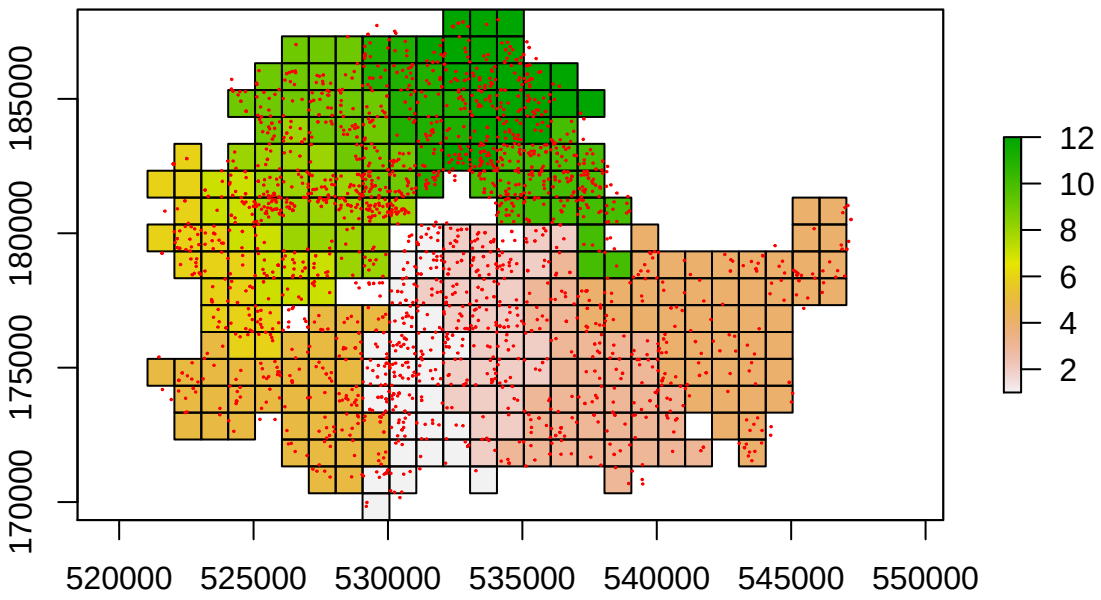
```
## [1] 1925    2
```

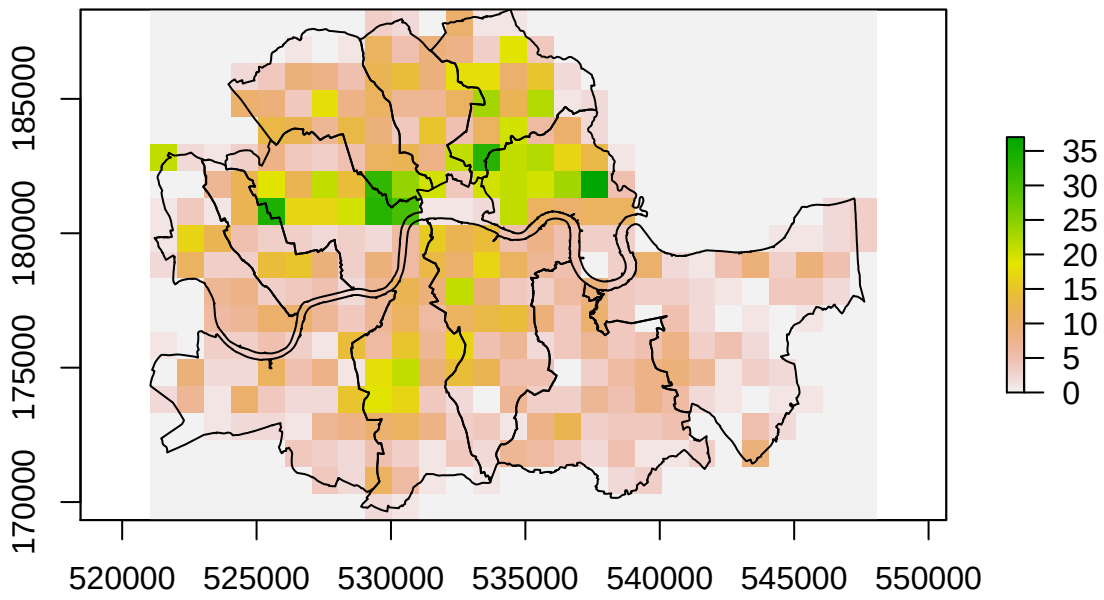
Duplicated crime locations must be removed for this technique to be correctly executed. Due to the 'map points' previously mentioned, many locations are duplications. When these are removed the dataset reduces from 2414 to 1925 unique sets of coordinates. Basic statistics are calculated, mean centre and standard distance for the crime locations, and plotted with a summary circle created by dividing the circle into 360 points and computing the bearing in radians. The crime location density is then calculated as 6.4 per km2.



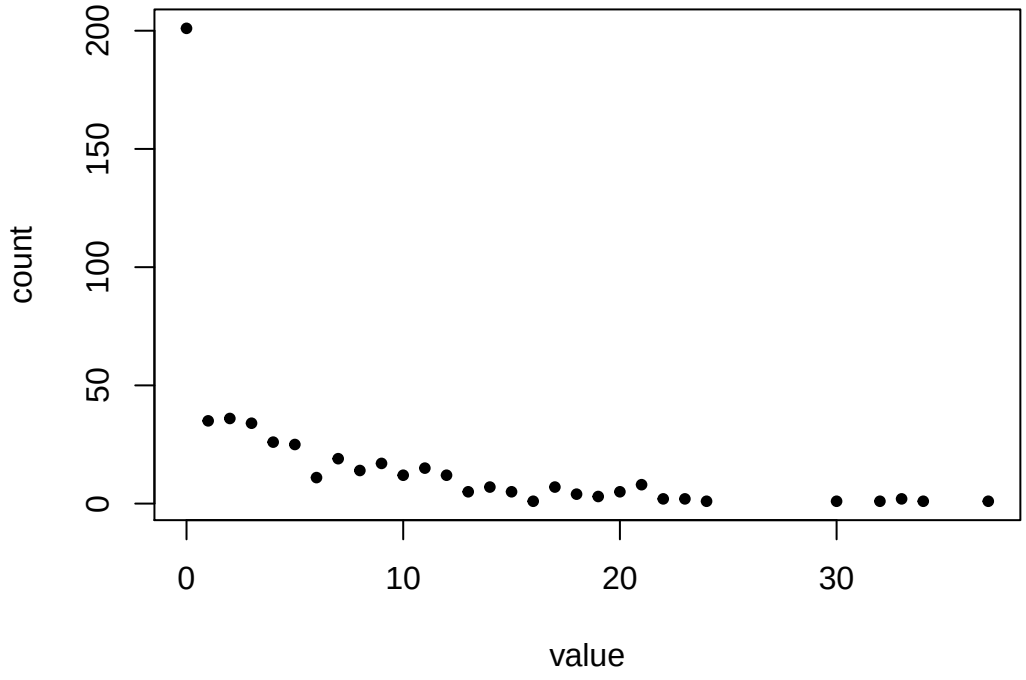
Quadrat analysis was then applied whereby you divide your study area into subsections of equal size, count the frequency of points in each subsection and then calculate the frequency of points in each subsection. Extent for the raster is obtained from the borough polygon, and then an arbitrary resolution is assigned of 1000. Cells are identified which lie within the borough boundary, then rasterize is used to determine frequency of locations in each quadrat.

```
## class       : RasterLayer
## dimensions  : 19, 27, 513  (nrow, ncol, ncell)
## resolution  : 1000, 1000  (x, y)
## extent      : 521054.9, 548054.9, 169327.4, 188327.4  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.999601272 +x_0=400000 +y_0=-100000 +datum=OSGB36 +
```


185000
180000
175000
170000
520000
525000
530000
535000
540000
545000
550000
12
10
8
6
4
2




```
##      value count
## [1,]     0   201
## [2,]     1    35
## [3,]     2    36
## [4,]     3    34
## [5,]     4    26
## [6,]     5    25
```




```
## [1] 4.705653
```

```
##   K   X       Kmu        Kmu2       XKmu2
## 1 0 201 -4.705653 22.14317036 4450.777242
## 2 1  35 -3.705653 13.73186432  480.615251
## 3 2  36 -2.705653  7.32055827  263.540098
## 4 3  34 -1.705653  2.90925223   98.914576
## 5 4  26 -0.705653  0.49794619   12.946601
## 6 5  25  0.294347  0.08664014    2.166004
```

This indicates that the 201 quadrats with no crime locations at all include the null ones that lie outside the boroughs. There are 5 quadrats with over 30 different crime locations. The average number of crime locations per quadrat is 4.7. Statistics can also be calculated using the quadrat data.

```
s2 <- sum(ff$XKmu2) / (sum(ff$X)-1)
VMR <- s2 / mu
```

The observed variance s2 is 41.5362375 and the variance to mean ratio (VMR) is 8.8268806. This is a unit-less statistic describing the spatial arrangement of points. In general stratified distributions ~ 0, random distributions = 1 and clustered distributions a VMR of above 1. This distribution is highly clustered.

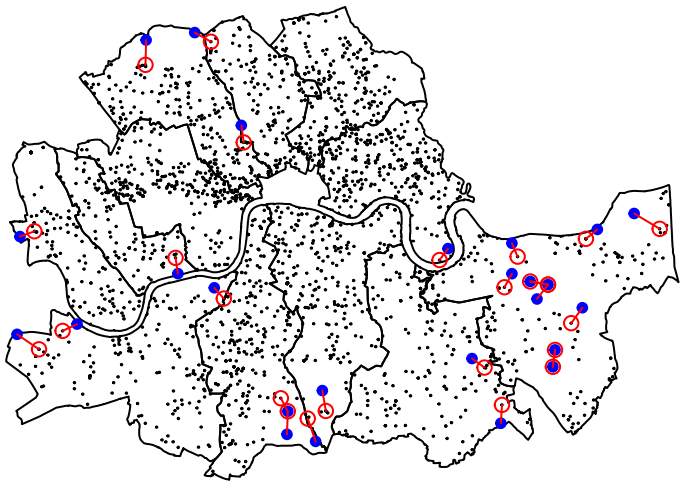
Distance based measurements will be carried out requiring a planar coordinate based system. A `dist` object is created and then coerced to a matrix, the distances from each point to itself are removed. To get the minimum distance to another event `apply` can be used.

```
##          1         2         3         4         5
## 1    0.000 2328.9164 3490.7732 3418.9726 2069.8328
## 2 2328.916    0.0000 1171.4764 1120.9239  828.9643
## 3 3490.773 1171.4764    0.0000  155.9122 1798.5964
```

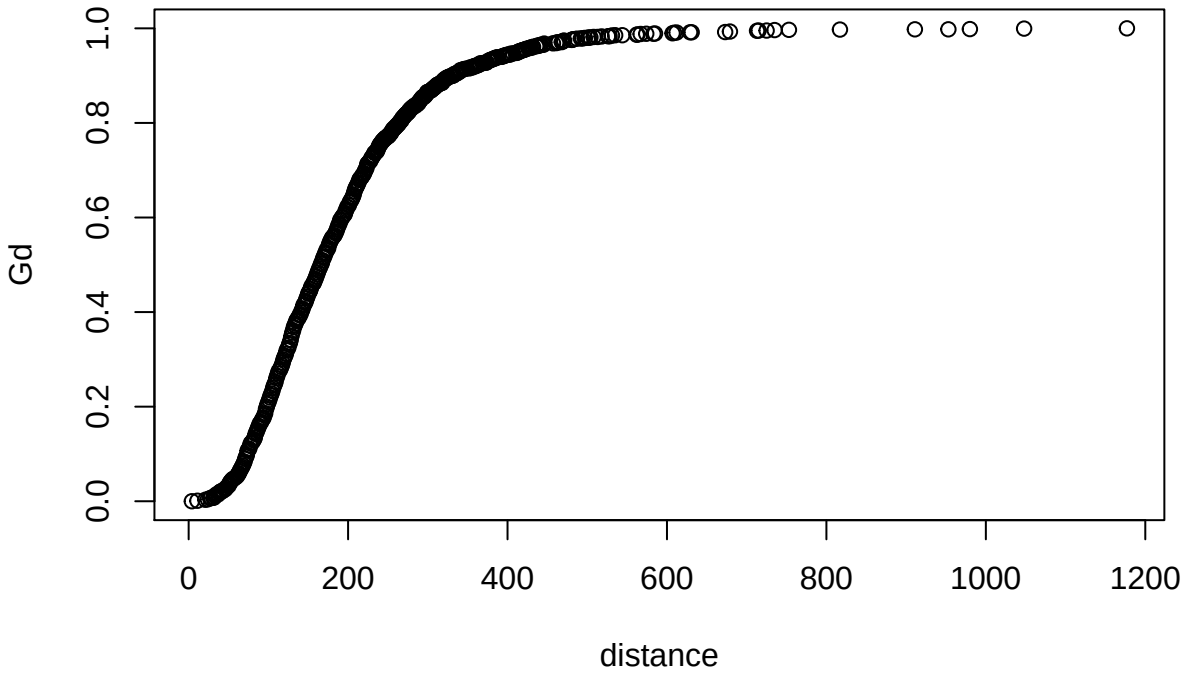
```
## 4 3418.973 1120.9239  155.9122    0.0000 1802.5113
## 5 2069.833  828.9643 1798.5964 1802.5113    0.0000
```

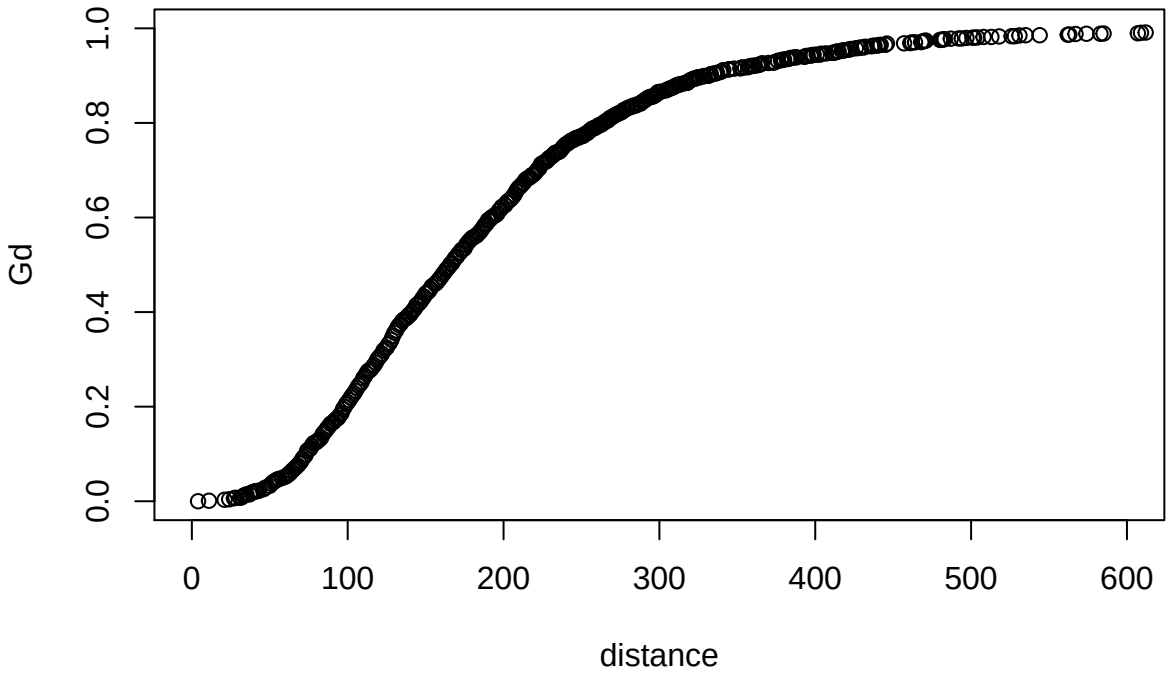
```
##          1         2         3         4         5
## 1       NA 2328.9164 3490.7732 3418.9726 2069.8328
## 2 2328.916        NA 1171.4764 1120.9239  828.9643
## 3 3490.773 1171.4764        NA  155.9122 1798.5964
## 4 3418.973 1120.9239  155.9122        NA 1802.5113
## 5 2069.833  828.9643 1798.5964 1802.5113        NA
```

The mean nearest neighbour distance is 190.0353488 metres. The top 25 most isolated (from their nearest neighour) cases can be identified and plotted.



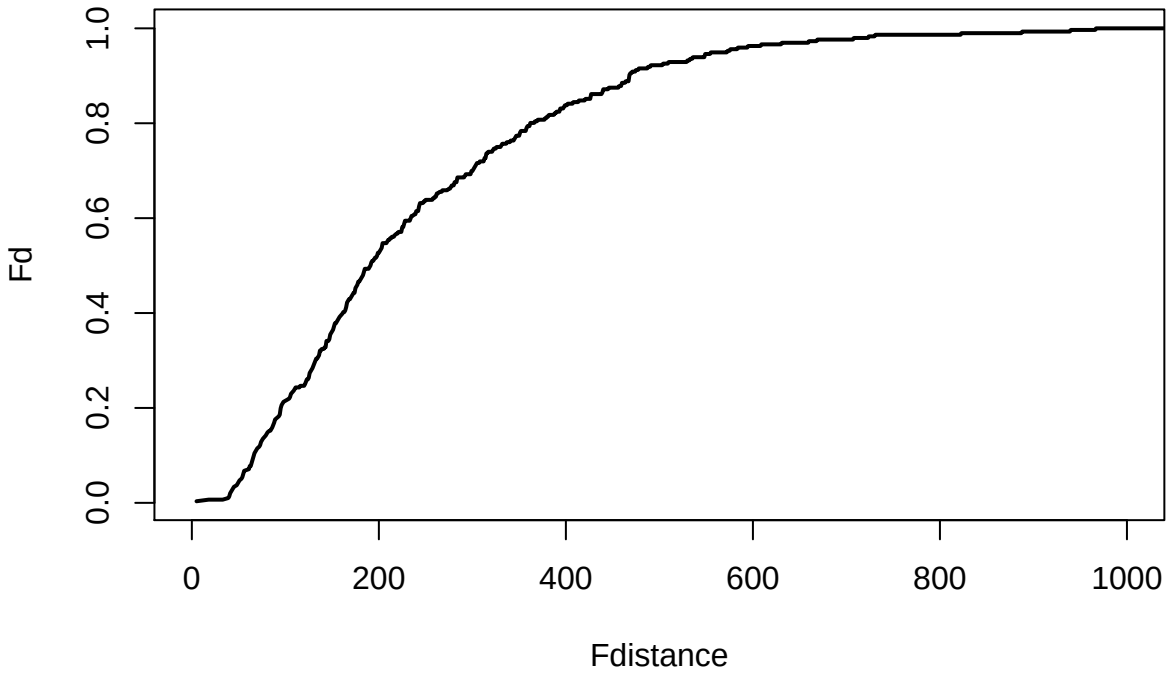
The G function is a cumulative frequency distribution of the nearest neighbor distance. It is the probability for a specified distance, that the nearest neighbor distance to another event in the pattern will be less than the specified distance. The maximum distance to a nearest neighbour is 1176.9604373 metres. The unique distances need to be calculated for the x-axis and how many cases there are with distances smaller than each x. These are then normalised between 0 and 1.


Gd
0.0
0.2
0.4
0.6
0.8
1.0
0
200
400
600
800
1000
1200
distance

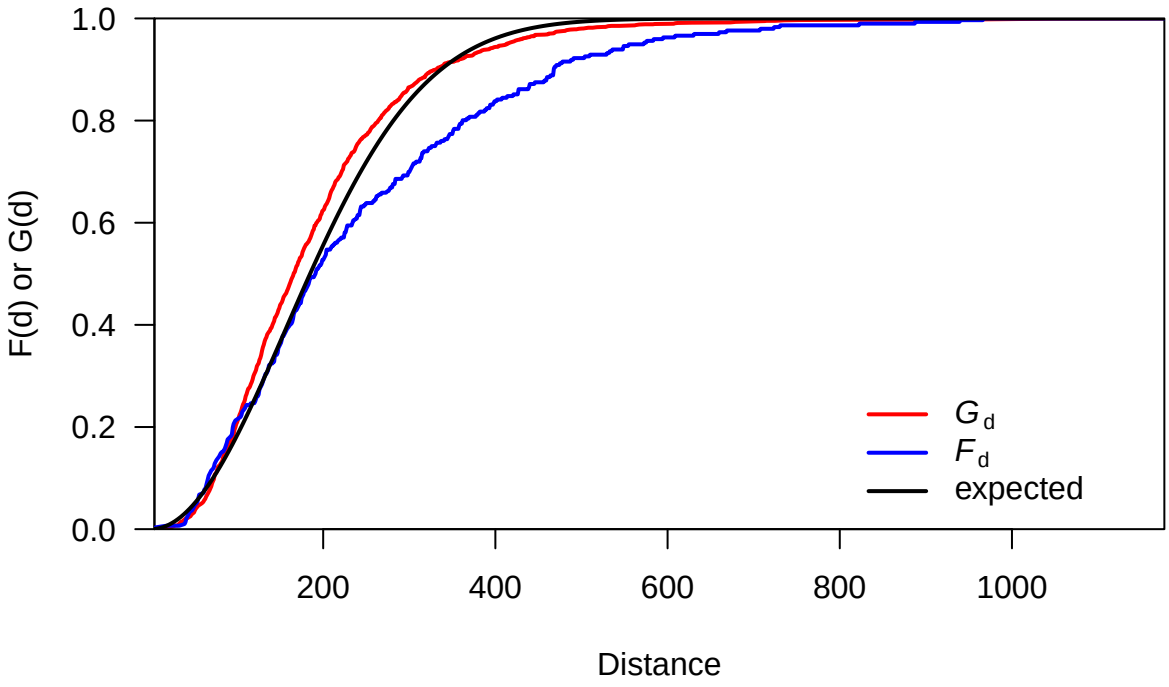


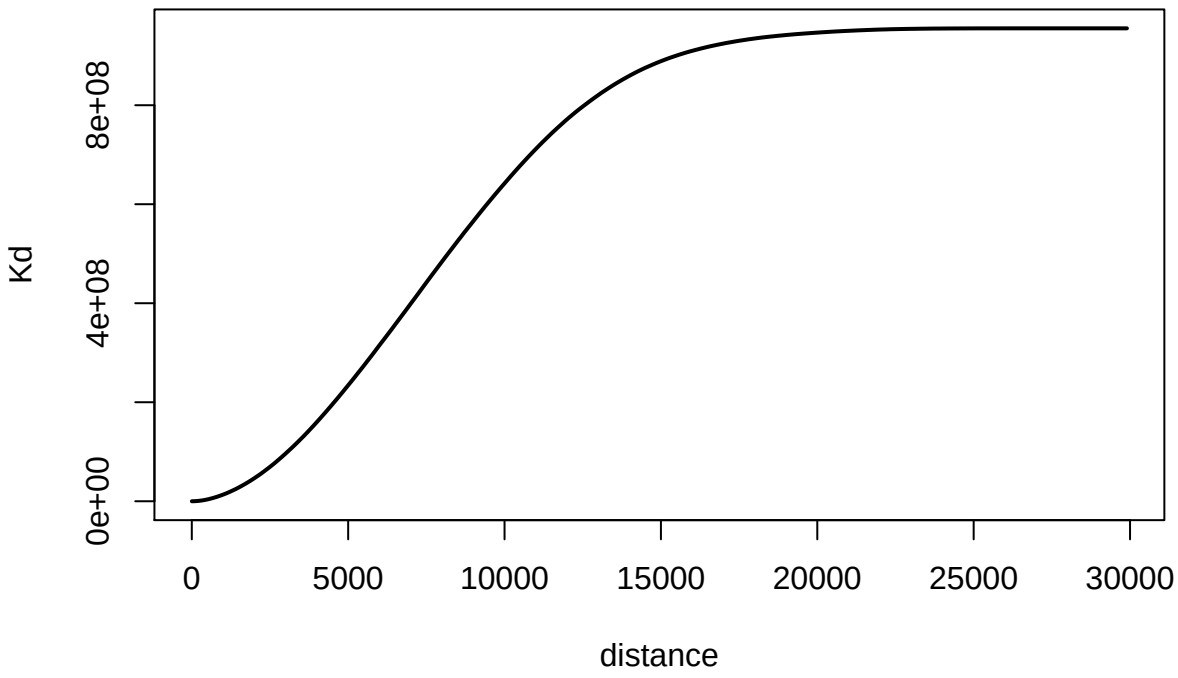
With evenly-spaced events G(d) should rise gradually up to the distance at which most events are spaced, and then increase rapidly. For clustered events G(d) rises rapidly at short distances, and then levels off at larger d-values. The distribution is more clustered than evenly spaced.

The centres of previously defined raster cells are used to compute the F function. The distance from all crime locations to these cell centres is calculated, then a similar process to the G function is followed.



For Evenly-spaced events F(d) should rise rapidly up to the distance at which most events are spaced and then level off (more nearest neighbors at small distances from randomly placed points). For clustered events F(d) rises rapidly at short distances, and then levels off at larger d-values; again indicating a highly clustered distribution in the dataset. The expected distribution can be computed, then all three functions plotted together. K is calculated using the original distance matrix `d`.


1.0
0.8
0.6
0.4
0.2
0.0
F(d) or G(d)
200
400
600
800
1000
Distance
G_d
F_d
expected



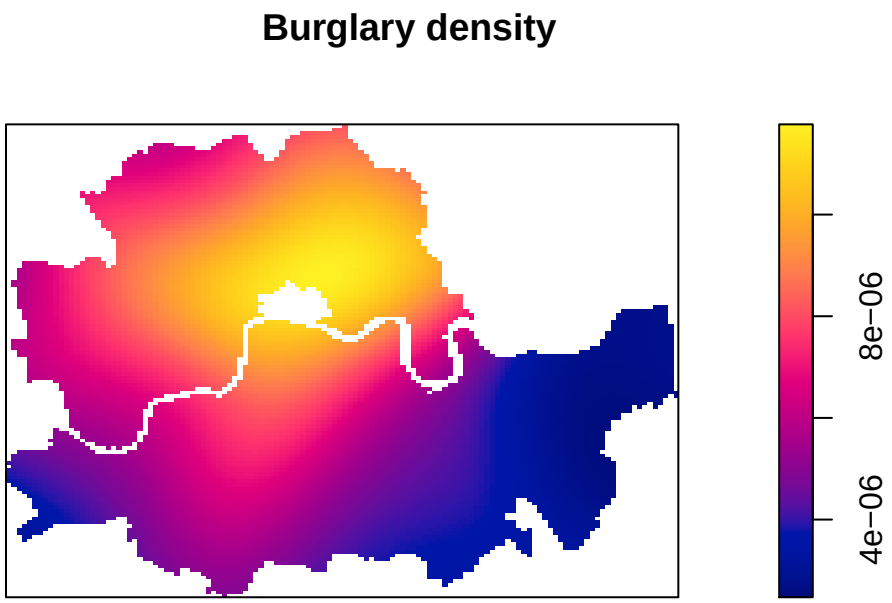
Next the spatstat package is used to make a Kernel Density raster. SpatialPolygons are coerced to an object of class "owin" (observation window) and coordinates extracted from SpatialPointsDataFrame. The kernel density was calculated - the number of points per km2 - then plotted. A marked point pattern object (ppp) is created, the marks must be coerced to a factor variable.

```
## [1] "owin"
```

```
## Planar point pattern: 1925 points
## window: polygonal boundary
## enclosing rectangle: [521054.9, 547612] x [169648, 188327.4] units
```

p

```
## [1] "im"
```



K-plots with an envelope were produced for burglary.

```
spatstat.options(checksegments = FALSE)
kburglary <- Kest(spp$"burglary")
keburglary <- envelope(spp$"burglary", Kest)

par(mfrow=c(1,2))
plot(kburglary, xlim=c(0,2500))
png("kburglary.png")
plot(keburglary)
png("keburglary.png")
```

In order to determine if population density is a good predictor of burglary, a Kolmogorov-Smirnov ('kstest') test was carried out using population density as a covariate:

```
KS.burglary <- kstest(spp$burglary, ds)
KS.burglary
```

This gives the following result: Spatial Kolmogorov-Smirnov test of CSR in two dimensions data: covariate 'ds' evaluated at points of 'spp$burglary' and transformed to uniform distribution under CSR D = 0.21068, p-value < 2.2e-16 alternative hypothesis: two-sided

This function performs a goodness-of-fit test of a Poisson point process model fitted to point pattern data. The observed distribution of the values of a spatial covariate at the data points, and the predicted distribution of the same values under the model, are compared. The null hypothesis is that 2 independent samples are drawn from the same continuous distribiton. The output reports maximum difference between the two cumulative distributions (D), and calculates a P value from that and the sample sizes [5]. The P value is small (2.2e-16) so it is concluded that the two groups were sampled from populations with different distributions

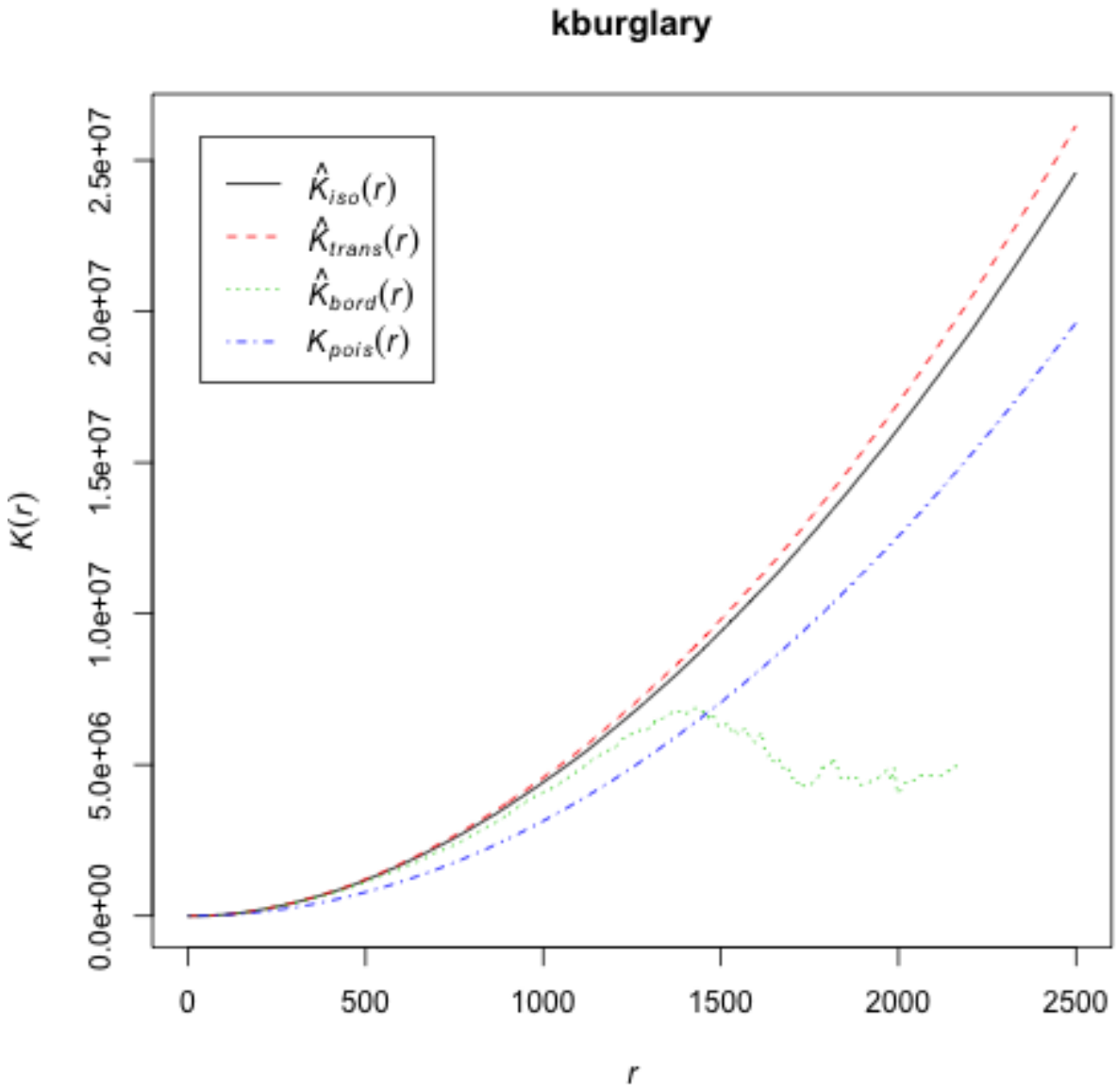


Figure 1: kburglary plot

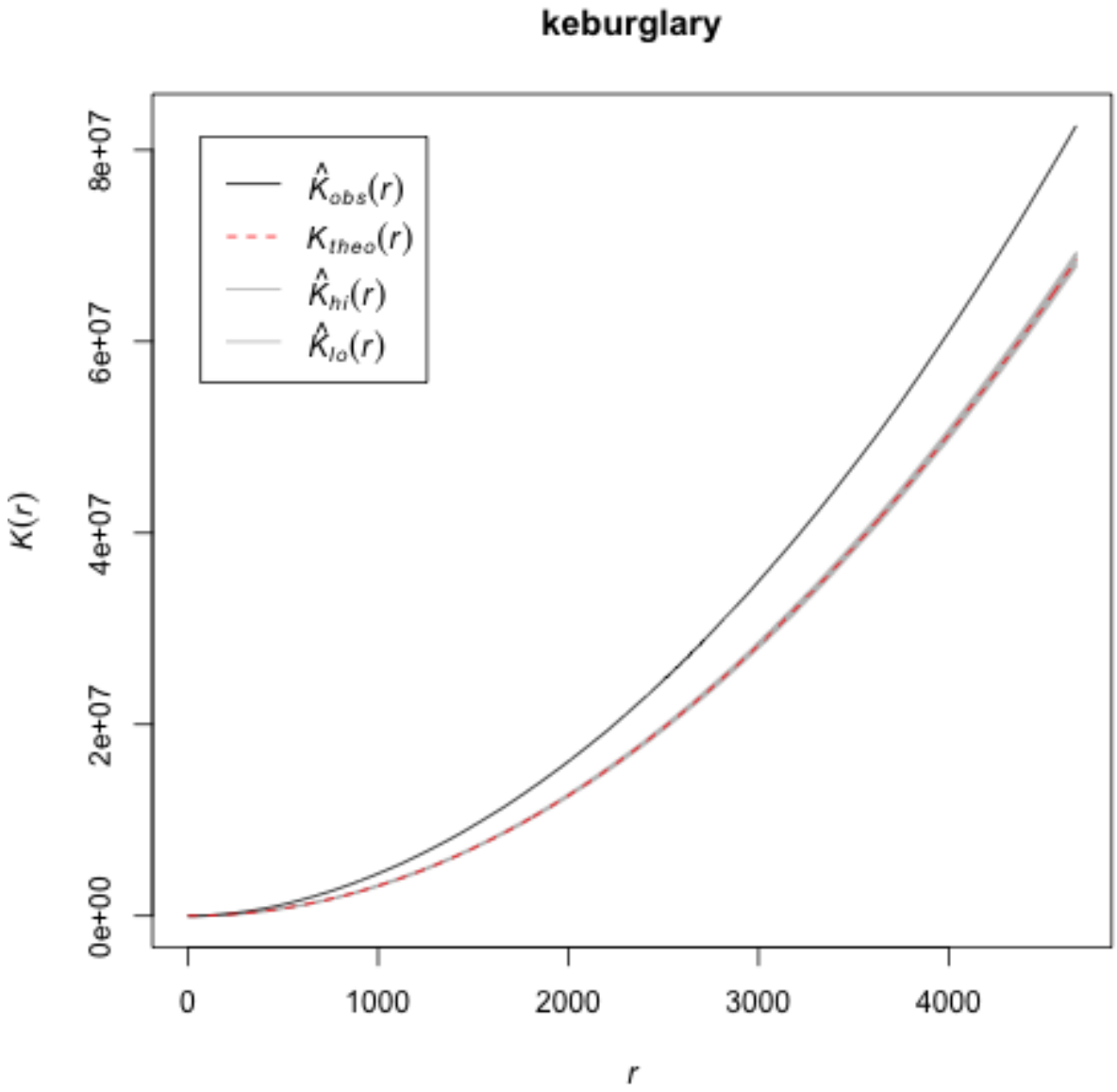


Figure 2: keburglary plot

and the null hypothesis is rejected.

7. Data Presentation

A combination of interactive chloropleth and KDE maps have been used to populate a Shiny App for easy access on a smartphone or tablet with a user friendly interface that allows for simple, easy access and availability to a large audience. The data used is for bicycle theft only for the inner London boroughs in 2016. It includes the facility to click on a map of the boroughs and a coloured circle appears indicating the density of bicycle thefts which occured in that area.

8. Conclusions

Crime rates have been dropping consistently in London from 2011 onwards, with a anonymously high decrease between 2012 and 2013 noted. However, certain crimes have shown an increase over that time e.g. violent crime. There are seasonal patterns in crime present, as evidenced by peaks of anti-social behaviour correlating to the highest temperature peaks in the summer season. Bicycle theft is also clearly cyclical with peaks in the summer and troughs in the winter, against a background of fairly constant thefts. Burglary has been broadly decreasing since 2012 but a peak is often visible in the run up to Christmas followed by a sharp drop in the New Year.

KDE is a valuable method for visualising the density of crimes when displaying a very large dataset, particular due to the location anonymisation meaning many crime observations lie immediately on top of each other. Russel Square and Shoreditch were identified as bicycle theft hotspots, with Soho and Fitzrovia as burglary hotspots, and Covent Garden and Soho for theft from the person.

The distribution of crimes across across the inner London boroughs varies widely. Westminster, with one of the lowest populations, displays the largest number of crimes at 28.7 per 100 people. Violent crimes peak in Westminster and Lambeth. There is a clear split between higher values in central, south and east London, with lower values in north and west london. Burglaries are also most prevalent in Westminster and Lambeth, with the lowest in Kensington and Chelsea and Hammersmith and Fulham.

Spatial analysis using point pattern analysis has shown that the most isolated occurences of burglary tend to occur on the periphery of the inner London boroughs or within the parks. The dataset is highly clustered based on the G and F functions, this is further confirmed by the VMR. The mean nearest neighbour distance is 83m and the maximum nearest neighbour is 854m. A main constraint of the dataset used is the lack of unique geographic location resulting in more than half the records being discounted from the analysis as they are non-unique. The P value found using the Kolmogorov-Smirnov test is very small (2.2e-16) so it is concluded that the two groups used were sampled from populations with different distributions and the null hypothesis of a continuous distribution is rejected.

9. Further Work and Recommendations

A potential feature which could be added to the KDE maps in the Shiny application would be loction services so user could identify current position automatically and return a risk rating from 1 to 5 for example, elaborating on the green, orange and red circles. Ideally, the chloropleth maps should be adjusted to display data normalised by population, so the effect of population is removed from the records.

If data download constraints were not an issue, a larger dataset covering all of the London boroughs for the full period from 2011-2017 could be used. This would provide a more thorough overview of crime patterns across London. Ideally, the dataset would include the day of the month the crime occurred as a minimum, and ideally the time of day too, plus the specific location, as made available within crime datasets in other cities. This additional granularity would widen the scope for machine learning methods such as forecasting to be used.

The client focus was threefold for this project, the local residents, the tourists and the Met Police. There are several recommendations which can be made as a result of the work carried out:

1. The Shiny could be made publicly available to encourage both residents and tourists to check crime hotspots. This would support their aim of making London the safest global city and puts data directly into the hands of the general public, allowing them to take informed decisions.
2. For tourists, always be on high alert for pickpockets when visiting tourist attractions in Westminster, especially the areas around Soho and Covent Garden, and when visiting Shoreditch.
3. For local residents, think twice before locking your bike up in the area around the universities in Russell Square or Shoreditch - there's a far greater risk of it getting stolen here than elsewhere, always make sure to lock up your car in the summer and if they're looking for the 'safest' inner London borough then move to Wandsworth which had only 9.66 crimes per 100 people in 2016, compared to Westminster's 28.7 crimes per 100 people.

10. References

[1]Identifying hotspots:an assessment of common techniques, Spencer Chainey, UCL Jill Danda Institue of Crime Science http://www.ucl.ac.uk/jdi/events/int-CIA-conf/ICIAC11_Slides/ICIAC11_1C_SChainey

[2] Making Maps with R, Eric Anderson http://eriqande.github.io/rep-res-web/lectures/making-maps-with-R.html

[3] An Introduction to Point Pattern Analysis http://gispopsci.org/point-pattern-analysis/

[4] Point Pattern Analysis from R Spatial http://rspatial.org/analysis/rst/8-pointpat.html

[5] Interpreting results: Kolmogorov-Smirnov test from GraphPad Software https://www.graphpad.com/guides/prism/7/statistics/interpreting_results_kolmogorov-smirnov_test.htm