AOSD - Final Assignment

Rajasirpi Subramaniyan

2023-03-08

ANALYSIS AND VISUALISATION OF UK ROAD ACCIDENT DATA (2021)

UNIVERSITY OF MUENSTER

ABSTRACT:

Road accidents are a frequent example of both spatial and temporal problems, and they involve a number of different aspects. Road accident data analysis will provide insights into the causes of these collisions and assist in preventing/reducing accidents in the future. This study is done to locate accident hotspots and determine how different types of roads and weather conditions affect accidents. According to this study, the majority of accidents take place on single-carriageway roads as compared to other kinds of roads and when the weather is “fine with no winds” as compared to other weather conditions. Also, it has been discovered that urban areas experience more accidents than rural ones.Kernel density estimation for accidents that occur within the boundary of the Metropolitan Police Force reveals that accidents are more concentrated in the center of the jurisdiction and decrease as they move outward. Accidents that result in fatalities also follow a similar pattern but have a wider range of higher density estimates.

INTRODUCTION:

One of the most significant daily difficulties that cannot be avoided and occurs at random times and places are road accidents. Road traffic accidents are the biggest cause of death for children and young adults, according to the World Health Organization, killing 1.35 million people annually. Since 2010, the United Kingdom has made open data in a variety of fields available for free to everyone in order to increase transparency between the government and the general public and to make it simple for the general public to access common information using their portal. Road safety statistics and other accident data are also included in these categories.Road accidents occur for a variety of reasons, including different types of roads, climatic circumstances, driver behavior, location characteristics, etc. This significant diversity of the factors makes it difficult to predict road accidents. Road accidents cannot entirely be prevented, but if appropriate steps are taken, the effects, the frequency of accidents, and the number of fatalities can all be reduced. Understanding the nature of these accidents is crucial in order to implement the necessary measures.

Research questions:

How different types of roads and weather are involved in these accidents?
How are the accidents distributed between urban and rural areas?
What are the hotspots for the recorded accidents within the Metropolitan force boundary?

Hypothesis:

More accidents may have happened in urban than rural areas and during winter months due to bad weather conditions.

Method:

Data Used:

The United Kingdom's 2021 road accident data were used in this analysis. This information was collected from the GDI data gov portal in the United Kingdom (https://www.data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data (https://www.data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data)).There are three different types of data in the overall zip file containing data on accidents, vehicles, and casualties. All accidents with accident indices are included in the accident data, along with information on the victims and vehicles involved in the accidents. However, only accident data were used and examined in this study. In this study, the data was downloaded separately as a csv file and then imported into the software using the read.csv function. This data may also be downloaded by script using the stats19 package.The Metropolitan Police force's boundary, which was exported from the police boundary and transformed into simple feature geometry using QGIS, is another set of data that is used. It was imported during the study.

Bar plots

Bar plots are a useful visualisation technique which aids better understanding of a problem when there are categorical variables present in the data. The first two questions are answered using representations of bar plots.

Cross tabulation matrix

Correlation between the road types was found using the cross tabulation matrix between the different road types in data and to see the relationships between different roads

Hotspot analysis

A mapping approach called "hotspot analysis" is utilized to find spatial point clusters that can be used for spatial analysis [1]. Hotspots are areas with higher point densities than would be expected based on the number of density distributions [1]. By contrasting the density of points in the given space with a random spatial model in which points appear at random, point patterns are examined[1]. These analyses typically make use of vector data, grouping points into polygons or converging points that are close to one another based on a determined distance [2] to locate statistically significant hot spots in the data. There are a number of techniques for locating hotspots, however in this study kernel density estimation is utilized because it has been successfully applied to the analysis of data from road crashes [3].

```
library(ggrepel) #FOR PLOTTING LABELS
```

```
## Loading required package: ggplot2
```

```
library(ggplot2) #FOR PLOTS
library(lubridate) #FOR WORKING ON DATE
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(stats19) #FOR PROCESSING AND FORMATTING DATA
```

```
## Data provided under OGL v3.0. Cite the source and link to:
## www.nationalarchives.gov.uk/doc/open-government-licence/version/3/
```

```
library(dplyr) #FOR DATA RESHAPING ANND SUMMARISING
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(sf) #FOR SPATIAL WORKS
```

```
## Linking to GEOS 3.9.1, GDAL 3.2.1, PROJ 7.2.1; sf_use_s2() is TRUE
```

```
library(viridis) #FOR COLOUR PALETTE
```

```
## Loading required package: viridisLite
```

```
library(spatstat)
```

```
## Loading required package: spatstat.data
```

```
## Loading required package: spatstat.geom
```

```
## spatstat.geom 3.0-3
```

```
## Loading required package: spatstat.random
```

```
## spatstat.random 3.0-1
```

```
## Loading required package: spatstat.explore
```

```
## Loading required package: nlme
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
## spatstat.explore 3.0-5
```

```
## Loading required package: spatstat.model
```

```
## Loading required package: rpart
```

```
## spatstat.model 3.0-2
```

```
## Loading required package: spatstat.linnet
```

```
## spatstat.linnet 3.0-3
```

```
##
## spatstat 3.0-2
## For an introduction to spatstat, type 'beginner'
```

```
library(GGally) #FOR VISUALISING CORRELATION
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(purrr)  #FOR JOINING TABLES
library(sp)  #FOR SPATIAL PROCESSES
library(tidyr)
library(mapview)  #FOR CREATING INTERACTIVE MAPS
library(leaflet)
library(rgdal)  #FOR PROJECTION AND TRANSFORMATIONS
```

```
## Please note that rgdal will be retired by the end of 2023,
## plan transition to sf/stars/terra functions using GDAL and PROJ
## at your earliest convenience.
##
## rgdal: version: 1.5-32, (SVN revision 1176)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.4.1, released 2021/12/27
## Path to GDAL shared files: C:/Users/User/Documents/R/win-library/4.1/rgdal/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 7.2.1, January 1st, 2021, [PJ_VERSION: 721]
## Path to PROJ shared files: C:\Program Files\PostgreSQL\14\share\contrib\postgis-3.3\proj
## PROJ CDN enabled: FALSE
## Linking to sp version:1.5-0
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading sp or rgdal.
```

```
library(adehabitatHR)  #FOR COORDINATES
```

```
## Loading required package: deldir
```

```
## deldir 1.0-6      Nickname: "Mendacious Cosmonaut"
```

```
##
##      The syntax of deldir() has had an important change.
##      The arguments have been re-ordered (the first three
##      are now "x, y, z") and some arguments have been
##      eliminated.  The handling of the z ("tags")
##      argument has been improved.
##
##      The "dummy points" facility has been removed.
##      This facility was a historical artefact, was really
##      of no use to anyone, and had hung around much too
##      long.  Since there are no longer any "dummy points",
##      the structure of the value returned by deldir() has
##      changed slightly.  The arguments of plot.deldir()
##      have been adjusted accordingly; e.g. the character
##      string "wpoints" ("which points") has been
##      replaced by the logical scalar "showpoints".
##      The user should consult the help files.
```

```
## Loading required package: ade4
```

```
##
## Attaching package: 'ade4'
```

```
## The following object is masked from 'package:spatstat.geom':
##
##     disc
```

```
## Loading required package: adehabitatMA
```

```
## Registered S3 methods overwritten by 'adehabitatMA':
##   method                       from
##   print.SpatialPixelsDataFrame sp
##   print.SpatialPixels          sp
```

```
## Loading required package: adehabitatLT
```

```
## Loading required package: CircStats
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:spatstat.geom':
##
##     area
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: boot
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:spatstat.explore':
##
##     envelope
```

```
##
## Attaching package: 'adehabitatLT'
```

```
## The following object is masked from 'package:dplyr':
##
##     id
```

```
library(raster)   #FOR RASTER ANALYSIS
```

```
##
## Attaching package: 'raster'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following object is masked from 'package:nlme':
##
##     getData
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

DATA PREPARATION

```
# Read the CSV file into a data frame
accident_raw <- read.csv("Uk-accident-2021.csv")
#head(accident_raw)

# Check for missing values in the data frame
any(is.na(accident_raw))
```

```
## [1] FALSE
```

```
#To understand the data distribution, relationships between variables
summary(accident_raw)
```

```
##  ï..accident_index  accident_year  accident_reference location_easting_osgr
##  Length:101087      Min.   :2021   Length:101087      Length:101087
##  Class :character   1st Qu.:2021   Class :character   Class :character
##  Mode  :character   Median :2021   Mode  :character   Mode  :character
##                     Mean   :2021
##                     3rd Qu.:2021
##                     Max.   :2021
##  location_northing_osgr  longitude           latitude          police_force
##  Length:101087          Length:101087      Length:101087      Min.   : 1.00
##  Class :character       Class :character   Class :character   1st Qu.: 4.00
##  Mode  :character       Mode  :character   Mode  :character   Median :21.00
##                                                               Mean   :27.06
##                                                               3rd Qu.:44.00
##                                                               Max.   :99.00
##  accident_severity number_of_vehicles number_of_casualties     date
##  Min.   :1.00      Min.   : 1.000     Min.   : 1.000       Length:101087
##  1st Qu.:3.00      1st Qu.: 1.000     1st Qu.: 1.000       Class :character
##  Median :3.00      Median : 2.000     Median : 1.000       Mode  :character
##  Mean   :2.76      Mean   : 1.844     Mean   : 1.268
##  3rd Qu.:3.00      3rd Qu.: 2.000     3rd Qu.: 1.000
##  Max.   :3.00      Max.   :13.000     Max.   :22.000
##   day_of_week        time           local_authority_district
##  Min.   :1.000   Length:101087      Min.   : -1.0000
##  1st Qu.:2.000   Class :character   1st Qu.: -1.0000
##  Median :4.000   Mode  :character   Median : -1.0000
##  Mean   :4.139                      Mean   : -0.3518
##  3rd Qu.:6.000                      3rd Qu.: -1.0000
##  Max.   :7.000                      Max.   :480.0000
##  local_authority_ons_district local_authority_highway first_road_class
##  Length:101087                Length:101087           Min.   :1.000
##  Class :character             Class :character        1st Qu.:3.000
##  Mode  :character             Mode  :character        Median :4.000
##                                                       Mean   :4.205
##                                                       3rd Qu.:6.000
##                                                       Max.   :6.000
##  first_road_number   road_type      speed_limit junction_detail
##  Min.   :   0.0    Min.   :1.000   Min.   :20   Min.   :-1.000
##  1st Qu.:   0.0    1st Qu.:6.000   1st Qu.:30   1st Qu.: 0.000
##  Median :  34.0    Median :6.000   Median :30   Median : 2.000
##  Mean   : 789.6    Mean   :5.257   Mean   :36   Mean   : 4.386
##  3rd Qu.: 532.0    3rd Qu.:6.000   3rd Qu.:40   3rd Qu.: 3.000
##  Max.   :9480.0    Max.   :9.000   Max.   :70   Max.   :99.000
##  junction_control second_road_class second_road_number
##  Min.   :-1.000   Min.   :-1.000    Min.   :  -1.0
##  1st Qu.:-1.000   1st Qu.: 0.000    1st Qu.:  -1.0
##  Median : 2.000   Median : 3.000    Median :   0.0
##  Mean   : 1.771   Mean   : 3.088    Mean   : 223.7
##  3rd Qu.: 4.000   3rd Qu.: 6.000    3rd Qu.:   0.0
##  Max.   : 9.000   Max.   : 6.000    Max.   :9176.0
##  pedestrian_crossing_human_control pedestrian_crossing_physical_facilities
##  Min.   :-1.0000                   Min.   :-1.000
##  1st Qu.: 0.0000                   1st Qu.: 0.000
##  Median : 0.0000                   Median : 0.000
##  Mean   : 0.3621                   Mean   : 1.168
##  3rd Qu.: 0.0000                   3rd Qu.: 0.000
```

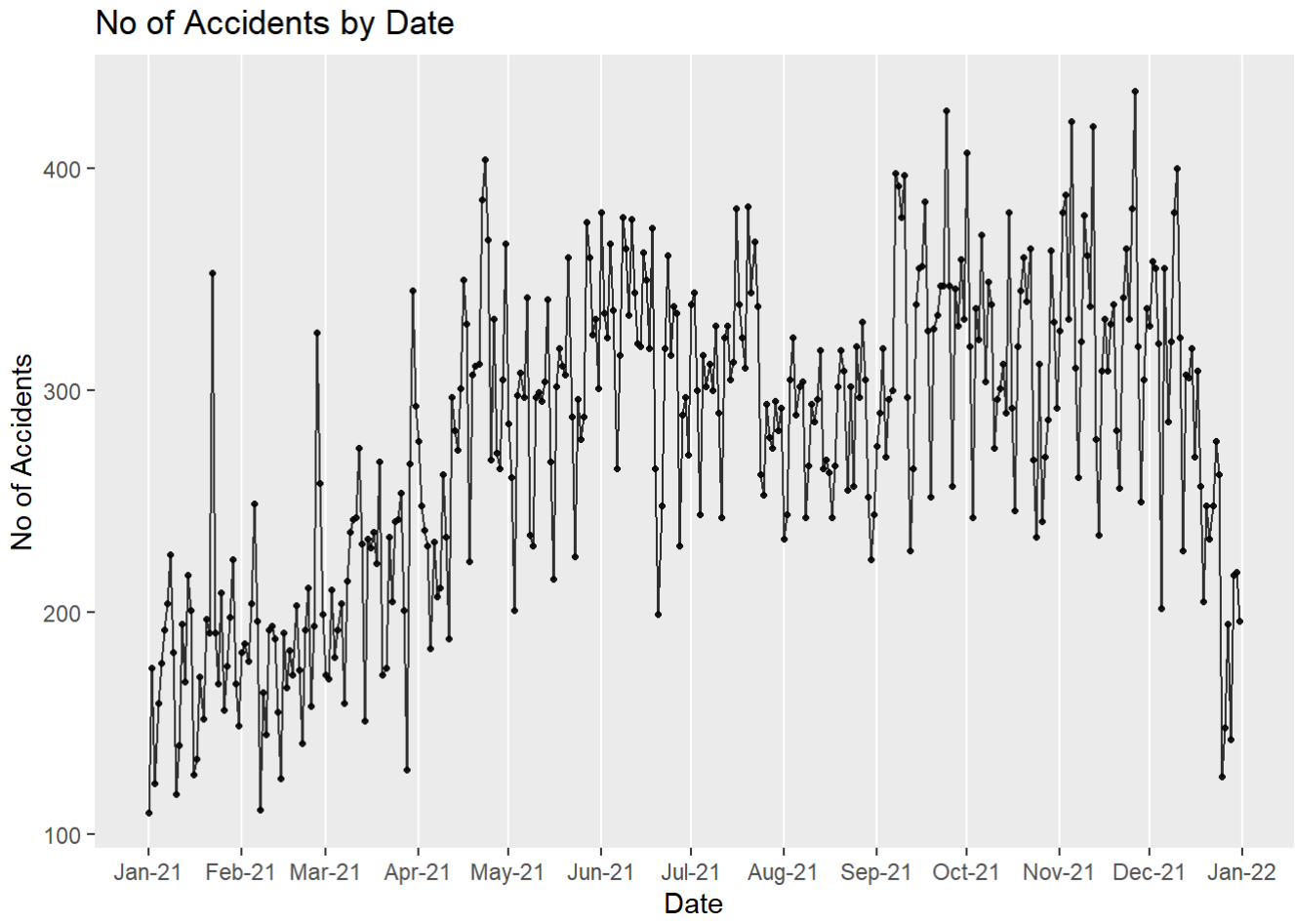
```
##  Max.   : 9.0000                  Max.   : 9.000
##  light_conditions weather_conditions road_surface_conditions
##  Min.   :-1.000   Min.   :-1.000     Min.   :-1.000
##  1st Qu.: 1.000   1st Qu.: 1.000     1st Qu.: 1.000
##  Median : 1.000   Median : 1.000     Median : 1.000
##  Mean   : 1.977   Mean   : 1.651     Mean   : 1.347
##  3rd Qu.: 4.000   3rd Qu.: 1.000     3rd Qu.: 2.000
##  Max.   : 7.000   Max.   : 9.000     Max.   : 9.000
##  special_conditions_at_site carriageway_hazards urban_or_rural_area
##  Min.   :-1.0000            Min.   :-1.0000     Min.   :1.00
##  1st Qu.: 0.0000            1st Qu.: 0.0000     1st Qu.:1.00
##  Median : 0.0000            Median : 0.0000     Median :1.00
##  Mean   : 0.2531            Mean   : 0.1966     Mean   :1.32
##  3rd Qu.: 0.0000            3rd Qu.: 0.0000     3rd Qu.:2.00
##  Max.   : 9.0000            Max.   : 9.0000     Max.   :3.00
##  did_police_officer_attend_scene_of_accident trunk_road_flag
##  Min.   :1.000                               Min.   :-1.000
##  1st Qu.:1.000                               1st Qu.: 2.000
##  Median :1.000                               Median : 2.000
##  Mean   :1.451                               Mean   : 1.722
##  3rd Qu.:2.000                               3rd Qu.: 2.000
##  Max.   :3.000                               Max.   : 2.000
##  lsoa_of_accident_location
##  Length:101087
##  Class :character
##  Mode  :character
##
##
##
```

```
accident_raw$Day_number<- wday(accident_raw$date)
accident_raw$N_Date <- dmy(as.character(accident_raw$date))
accident_raw$Month_Number <- factor(month(as.character(accident_raw$N_Date)))

#This function was used to name the week of the day
name_the_days <- function(week_number){
  factor(
    ifelse(week_number==1,"Sunday",
           ifelse(week_number==2,"Monday",
                  ifelse(week_number==3,"Tuesday",
                         ifelse(week_number==4,"Wednesday",
                                ifelse(week_number==5,"Thursday",
                                       ifelse(week_number==6,"Friday",
                                              ifelse(week_number==7,"Saturday",NA
                                              )))))),
    levels=c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"))}

accident_raw$Week_Name <- name_the_days(accident_raw$Day_number)
```

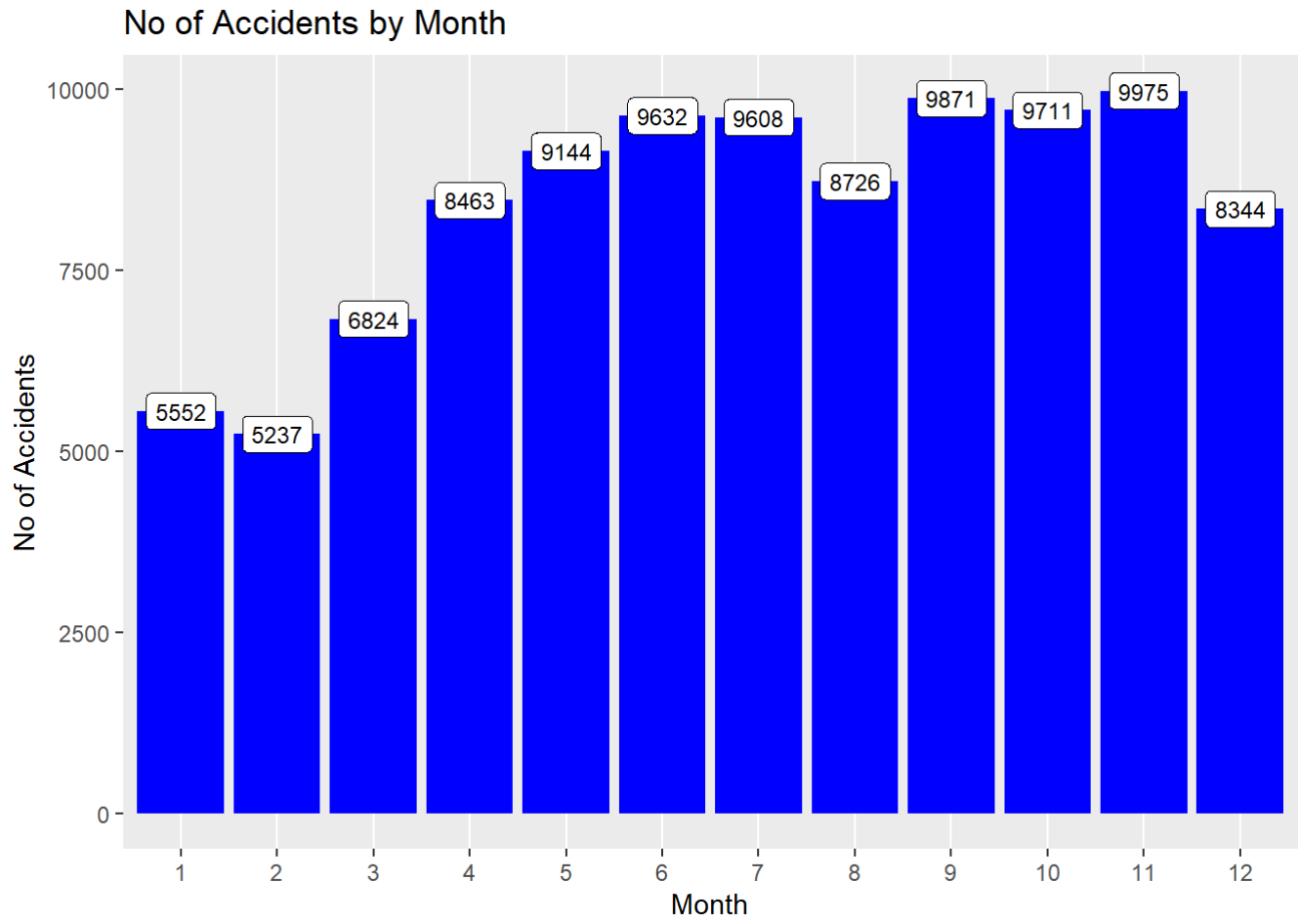
The data consists of 101,087 observations which represents individual accidents that occurred all over the United kingdom throughout the year 2021. There are a total of 36 columns which were present in the data and 4 columns which are added to the data for this study. The data includes various information about accidents such as latitude, longitude, road type, weather conditions, light conditions, road surface conditions, speed limit etc.

TOTAL ACCIDENTS IN 2021

```
accident_raw %>%
  group_by(N_Date) %>%
  summarise("n"=n()) %>%
  ggplot(data=., aes(N_Date,n)) +
  geom_point(size=0.9,alpha=0.9) + geom_line(alpha=0.8)  +
  scale_x_date(date_breaks="1 month", date_labels="%b-%y") +
  theme(panel.grid.minor = element_blank(),panel.grid.major.y = element_blank()) +
  labs(title="No of Accidents by Date",
       x="Date",y="No of Accidents")
```



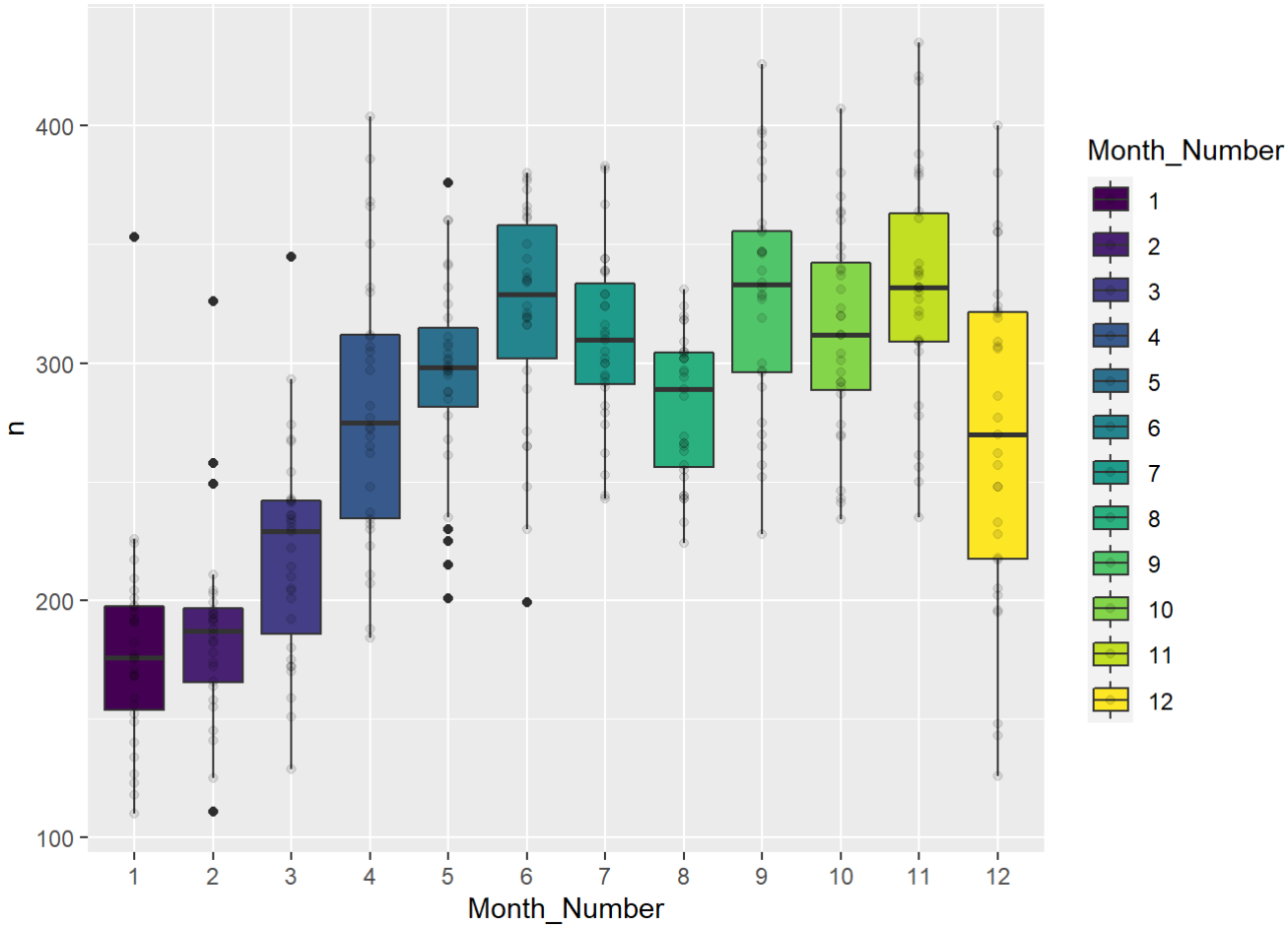

```
accident_raw %>%
  group_by(Month_Number) %>%
  summarise("n"=n()) %>%
  ggplot(data=., aes(Month_Number,n)) +
  geom_col(fill='blue') +
  theme(panel.grid.minor = element_blank(),panel.grid.major.y = element_blank()) + geom_label
(aes(label=n),size=3)+
  labs(title="No of Accidents by Month",
       x="Month",y="No of Accidents")
```



Preliminary time series exploration is done in these steps. The data was categorized based on date and month to have a basic understanding of the accidents and to see the number of accidents that has happened every month in the year 2021. It was found that the month of November has the highest amounts of recorded accidents with a number of 9975 accidents followed by the month of september, 9871 accidents. It can also be seen that the second half of the year (i.e July - December) has a higher number of accidents compared to the first half (i.e January - June)

```
accident_raw %>%
  group_by(N_Date,Month_Number) %>%
  summarise("n"=n()) %>%
  ggplot(data=., aes(Month_Number,n, fill=Month_Number))  +
  geom_boxplot() + scale_fill_viridis(option="viridis", discrete=T) +   geom_point(alpha=0.1)
```

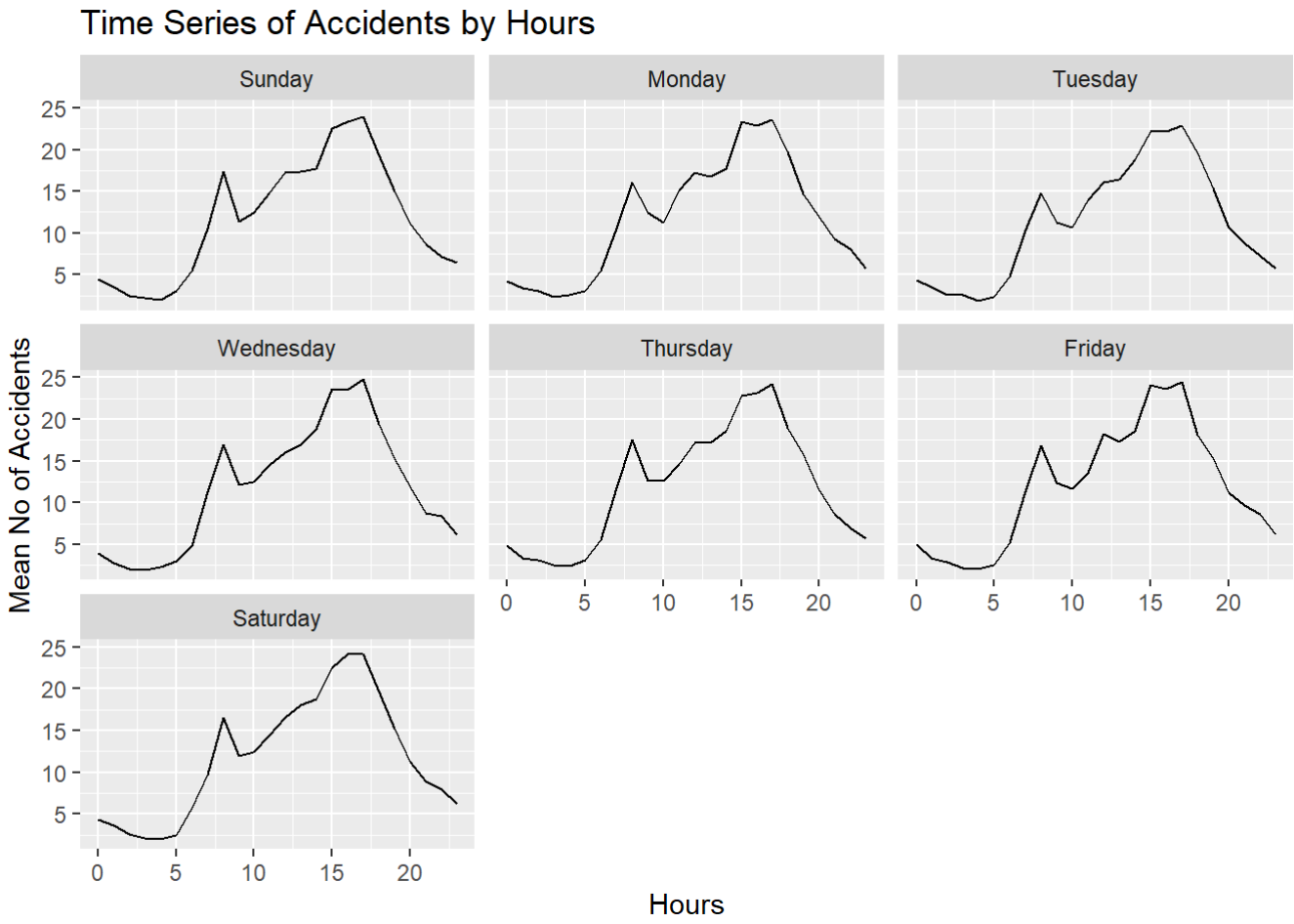
```
## `summarise()` has grouped output by 'N_Date'. You can override using the
## `.groups` argument.
```




```
weekday_summary <- accident_raw %>%
  mutate("Hour"=as.numeric(substr(as.character(time),1,2))) %>%
  group_by(N_Date,Week_Name,Hour) %>%
  summarise(
    "n"=n()
  ) %>%
  group_by(Week_Name,Hour) %>%
  summarise("mean_number_of_accidents"=mean(n))
```

```
## `summarise()` has grouped output by 'N_Date', 'Week_Name'. You can override
## using the `.groups` argument.
## `summarise()` has grouped output by 'Week_Name'. You can override using the
## `.groups` argument.
```

```
ggplot(weekday_summary, aes(Hour,mean_number_of_accidents)) +
  geom_line() +
  facet_wrap(~Week_Name) +
  labs(title="Time Series of Accidents by Hours",
       x="Hours",y="Mean No of Accidents")
```



Here the data is plotted based on both Months and Dates using a box plot. Each point in the box plot represents each date of a month. The next plot shows the time series of mean number of accidents categorised based on the week of the day. From this it can be seen that the mean number of accidents reach their peak during 15-18 hour everyday although they vary in numbers and also have significant peaks during 7-10 everyday.

ACCIDENTS CATEGORIZED BASED ON LOCATIONS, ROADS AND WEATHER CONDITIONS

```
formatted_accidents<-format_accidents(accident_raw)
```

```
## date and time columns present, creating formatted datetime column
```

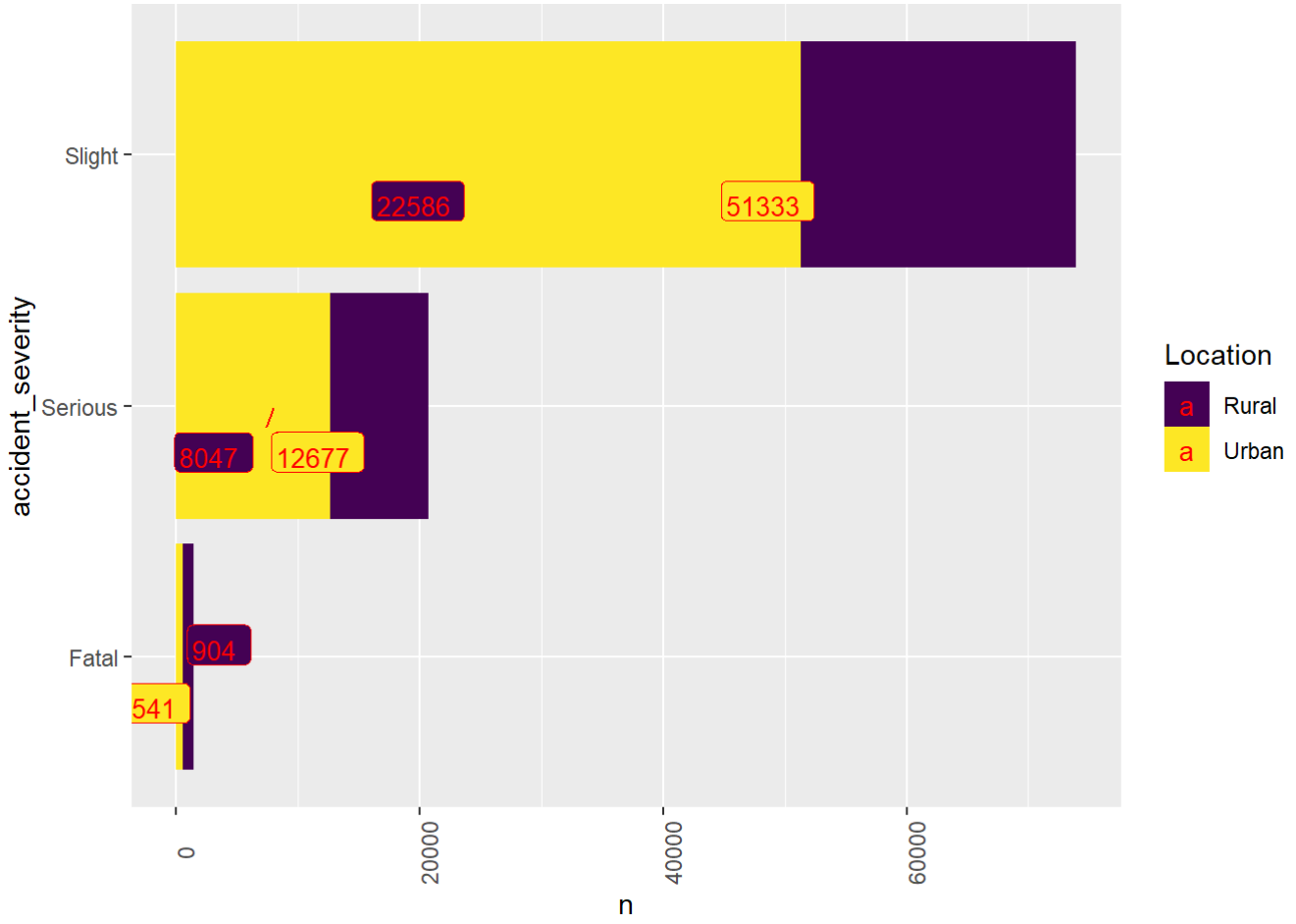
```
glimpse(formatted_accidents)
```

```
## Rows: 101,087
## Columns: 41
## $ ï..accident_index                              <chr> "2021010287148", "20210102~
## $ accident_year                                  <int> 2021, 2021, 2021, 2021, 20~
## $ accident_reference                             <chr> "010287148", "010287149", ~
## $ location_easting_osgr                          <chr> "521508", "535379", "52970~
## $ location_northing_osgr                         <chr> "193079", "180783", "17039~
## $ longitude                                      <chr> "-0.246102", "-0.050574", ~
## $ latitude                                       <chr> "51.623425", "51.509767", ~
## $ police_force                                   <chr> "Metropolitan Police", "Me~
## $ accident_severity                              <chr> "Slight", "Serious", "Seri~
## $ number_of_vehicles                             <int> 3, 2, 2, 1, 4, 2, 2, 2, 1,~
## $ number_of_casualties                           <int> 1, 3, 4, 1, 1, 1, 1, 1, 1,~
## $ date                                           <date> 2021-01-01, 2021-01-01, 2~
## $ day_of_week                                    <chr> "Friday", "Friday", "Frida~
## $ time                                           <chr> "02:05", "03:30", "04:07",~
## $ local_authority_district                       <chr> NA, NA, NA, NA, NA, NA, NA~
## $ local_authority_ons_district                   <chr> "E09000003", "E09000030", ~
## $ local_authority_highway                        <chr> "E09000003", "E09000030", ~
## $ first_road_class                               <chr> "Unclassified", "A", "B", ~
## $ first_road_number                              <chr> "first_road_class is C or ~
## $ road_type                                      <chr> "Single carriageway", "Dua~
## $ speed_limit                                    <int> 30, 30, 30, 30, 20, 20, 20~
## $ junction_detail                                <chr> "Other junction", "More th~
## $ junction_control                               <chr> "Give way or uncontrolled"~
## $ second_road_class                              <chr> "Unclassified", "A", "C", ~
## $ second_road_number                             <chr> "first_road_class is C or ~
## $ pedestrian_crossing_human_control              <chr> "None within 50 metres", "~
## $ pedestrian_crossing_physical_facilities        <chr> "No physical crossing faci~
## $ light_conditions                               <chr> "Darkness - lights lit", "~
## $ weather_conditions                             <chr> "Fog or mist", "Fine no hi~
## $ road_surface_conditions                        <chr> "Frost or ice", "Dry", "Dr~
## $ special_conditions_at_site                     <chr> "Auto traffic signal - out~
## $ carriageway_hazards                            <chr> "None", "None", "None", "N~
## $ urban_or_rural_area                            <chr> "Urban", "Urban", "Urban",~
## $ did_police_officer_attend_scene_of_accident    <chr> "Yes", "Yes", "Yes", "Yes"~
## $ trunk_road_flag                                <chr> "Non-trunk", "Non-trunk", ~
## $ lsoa_of_accident_location                      <chr> "E01000263", "E01004303", ~
## $ day_number                                     <dbl> 7, 7, 7, 7, 7, 7, 7, 7, 7,~
## $ n_date                                         <date> 2021-01-01, 2021-01-01, 2~
## $ month_number                                   <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ week_name                                      <fct> Saturday, Saturday, Saturd~
## $ datetime                                       <dttm> 2021-01-01 02:05:00, 2021~
```

Since the data has not yet been formatted, the majority of its variables contain values that are not entirely understandable, making the data that was read still more of a raw data. When the data is formatted, the values that are contained in the columns can be decoded, and some of the columns can display the data category. The stats19 package's format accidents function was used to do this. And it is evident that the previously available data is now easier to read and comprehend. It is easy to see the change when compared to accident raw data.For example, in accident_raw, the column weather conditions has values 1,2 etc, which was not easy to understand unless you have the look up table but after the formatting it changed to Fine, Fine no high winds etc, which is comprehensible and understandable.

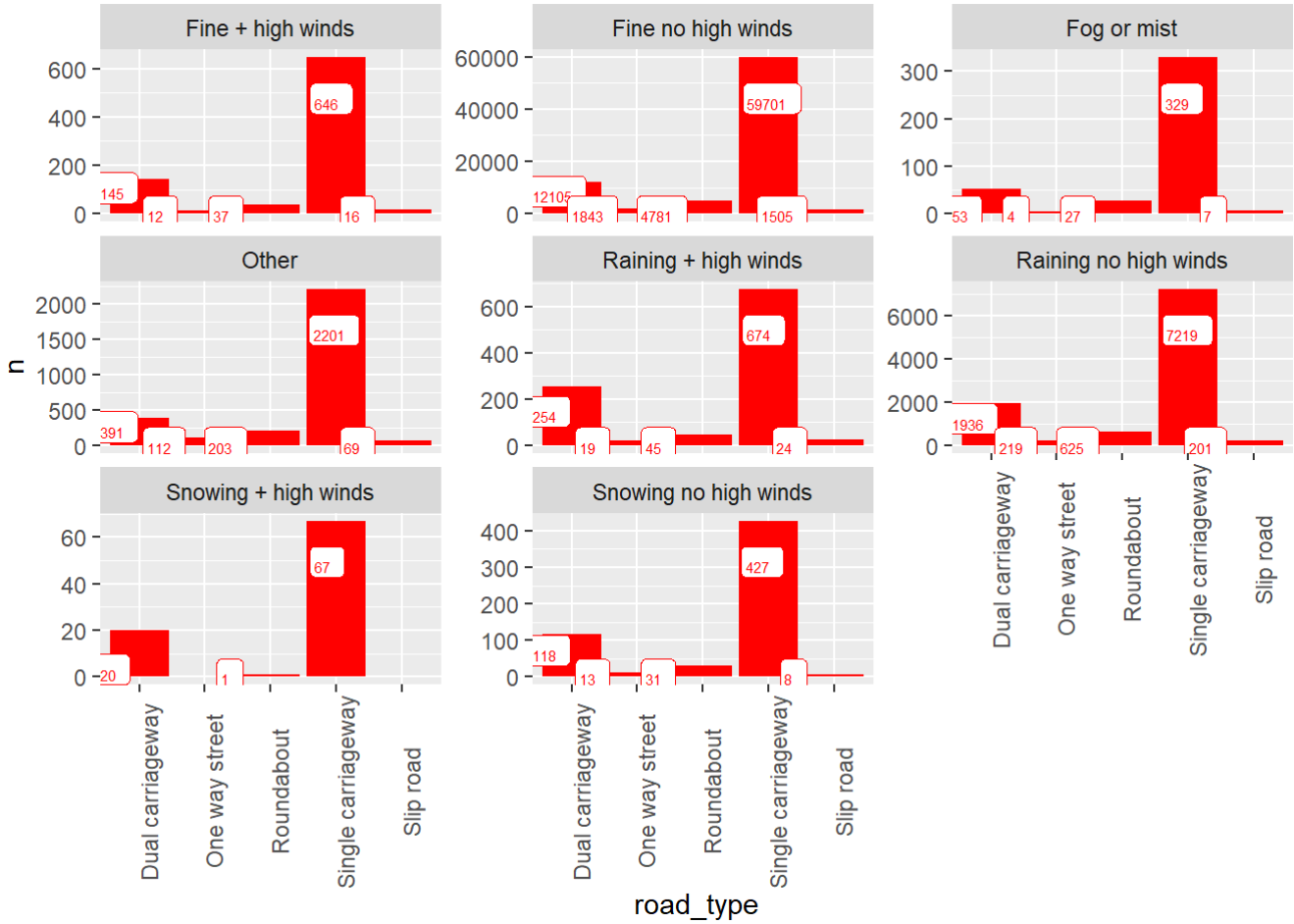
```
filtered= formatted_accidents%>%filter(urban_or_rural_area =="Urban" | urban_or_rural_area =
="Rural")%>%filter(road_type!="Unknown"&weather_conditions!="Data missing or out of range"&we
ather_conditions!="Unknown"& road_surface_conditions!="Data missing or out of range")

filtered %>% group_by(accident_severity,urban_or_rural_area)%>% summarise("n"=n()) %>% ggplot
(data=., aes(accident_severity,n,fill= urban_or_rural_area)) + geom_col() +
  labs(fill = "Location")+scale_fill_viridis(option="viridis", discrete=T)  +
  theme(axis.text.x = element_text(angle = 90)) + coord_flip()+ geom_label_repel(aes(label=
n),size=3.5,colour="red", vjust=0,hjust=0)
```

```
## `summarise()` has grouped output by 'accident_severity'. You can override using
## the `.groups` argument.
```




```
filtered %>% group_by(road_type,weather_conditions)%>% summarise("n"=n()) %>% ggplot(data=.,
aes(road_type,n)) + geom_col(fill='red')+ facet_wrap(vars(weather_conditions), scales = "free
_y") + geom_label_repel(aes(label=n),size=2,colour="red", vjust=0,hjust=0)+
  theme(axis.text.x = element_text(angle = 90))
```

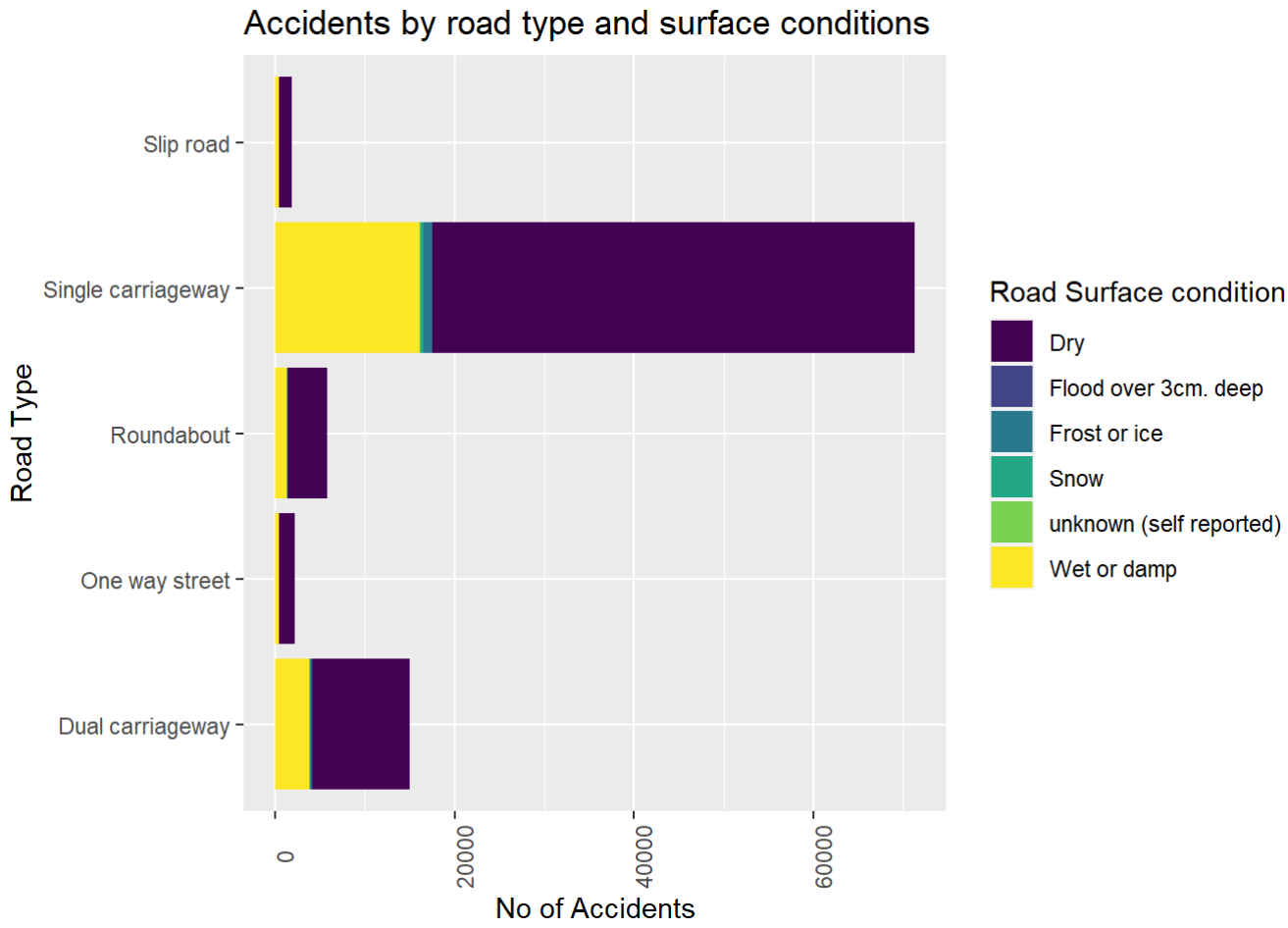
```
## `summarise()` has grouped output by 'road_type'. You can override using the
## `.groups` argument.
```



Filtering was done to eliminate observations with unknown or missing data in order to move on. Based on whether the accident occurred in an urban or rural area, the data was filtered. Moreover, road type, weather conditions, and road surface conditions observations with uncertain or missing data are filtered out. Plots are then developed after this. Based on the severity of the incidents and whether they occurred in urban or rural areas, the first plot classifies the accidents. As predicted, there are more accidents in urban areas than in rural ones, but surprisingly, there were more fatal incidents in rural areas than in urban ones. The second plot demonstrates the distribution of accidents on various types of roads in various weather conditions.Single carriageway roads are the roads where most number of accidents are recorded.

```
filtered %>%
  group_by(road_type,road_surface_conditions) %>%
  summarise("n"=n()) %>%
  ggplot(data=.,aes(road_type,n,fill=factor(road_surface_conditions))) +
  geom_col()+
  scale_fill_viridis(option="viridis", discrete=T) +
  labs(title="Accidents by road type and surface conditions",
       x="Road Type",y="No of Accidents",fill="Road Surface condition")+    theme(axis.text.x
= element_text(angle = 90))+coord_flip()
```

```
## `summarise()` has grouped output by 'road_type'. You can override using the
## `.groups` argument.
```



The following graph illustrates how various road surface conditions affect accidents based on the kind of road. As previously stated, single carriageway roads contribute to the largest number of accidents among the various road categories. Dry and Wet or Damp contribute to more accidents than other surface conditions.

CORRELATION BETWEEN ROAD TYPES

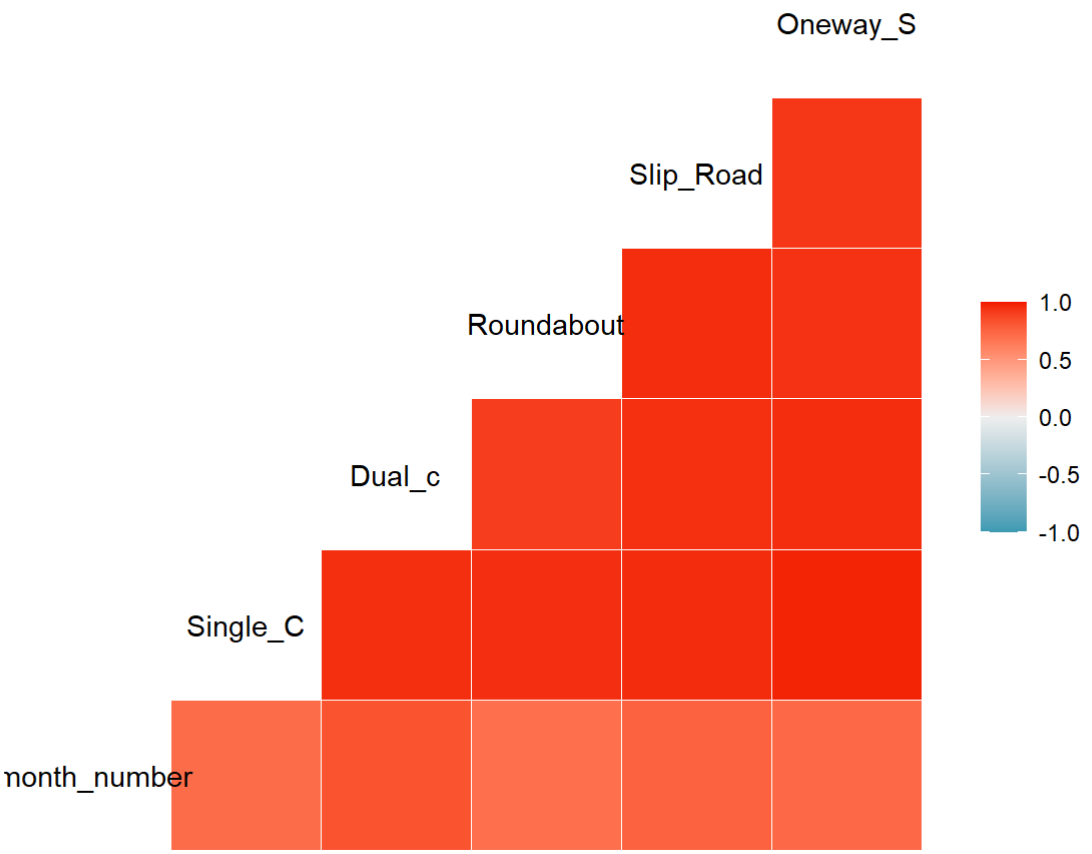
```
##FILTERING DIFFERENT ROAD TYPES FROM THE DATA
SingleC<-filtered%>%filter(road_type=='Single carriageway')
Sac<-SingleC%>%group_by(month_number)%>%summarise('n'=n())

DualC<-filtered%>%filter(road_type=='Dual carriageway')
Dac<-DualC%>%group_by(month_number)%>%summarise('n'=n())

Rabout<-filtered%>%filter(road_type=='Roundabout')
Rab<-Rabout%>%group_by(month_number)%>%summarise('n'=n())

SlipR<-filtered%>%filter(road_type=='Slip road')
Slroad<-SlipR%>%group_by(month_number)%>%summarise('n'=n())

Oneway<-filtered%>%filter(road_type=='One way street')
Onest<-Oneway%>%group_by(month_number)%>%summarise('n'=n())

month_number<-filtered%>%group_by(month_number)%>%summarise('n'=n())

road_cor<-list(Sac,Dac,Rab,Slroad, Onest)%>%reduce(left_join, by='month_number')
road_cor$month_number<-as.numeric(road_cor$month_number)
names(road_cor)[2]<-"Single_C"
names(road_cor)[3]<-"Dual_c"
names(road_cor)[4]<-"Roundabout"
names(road_cor)[5]<-"Slip_Road"
names(road_cor)[6]<-"Oneway_S"
head(road_cor)
```

```
## # A tibble: 6 x 6
##   month_number Single_C Dual_c Roundabout Slip_Road Oneway_S
##          <dbl>    <int>  <int>      <int>     <int>    <int>
## 1            1     3864    850        341       105      109
## 2            2     3686    805        348        79      104
## 3            3     4924    970        384       110      149
## 4            4     6036   1193        523       162      179
## 5            5     6453   1380        475       159      206
## 6            6     6931   1306        534       178      206
```

```
cor(road_cor)
```

```
##              month_number  Single_C    Dual_c Roundabout Slip_Road  Oneway_S
## month_number    1.0000000 0.7186804 0.8301812  0.7018670 0.7628843 0.7293647
## Single_C        0.7186804 1.0000000 0.9547131  0.9563787 0.9615973 0.9817747
## Dual_c          0.8301812 0.9547131 1.0000000  0.9109051 0.9518807 0.9611369
## Roundabout      0.7018670 0.9563787 0.9109051  1.0000000 0.9606231 0.9429141
## Slip_Road       0.7628843 0.9615973 0.9518807  0.9606231 1.0000000 0.9332589
## Oneway_S        0.7293647 0.9817747 0.9611369  0.9429141 0.9332589 1.0000000
```

```
ggcorr(road_cor)
```

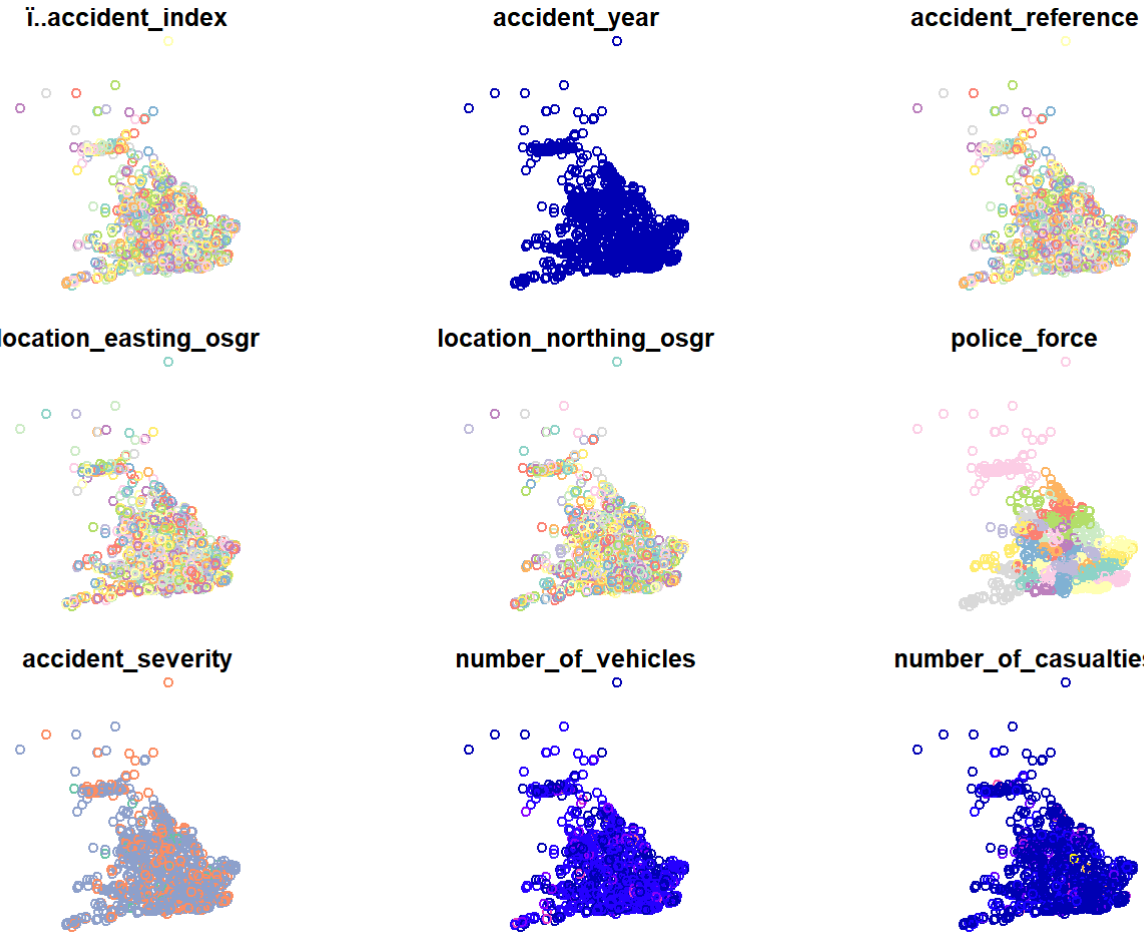


UNDERSTANDING

The correlation matrix was used to calculate correlation between various road types in this above phase. The many relationships between variables—in this case, road types—are sorted out using this matrix. A correlation's value might be either negative or positive, or it can vary from -1 to +1. When there is a negative correlation, one variable rate falls relative to an increase in another variable rate. Positive correlation means that the rate of one variable rises in relation to the rate rise of another variable. It appears that accidents are increasing month over month and are positively correlated with all different types of roads. Also, it was discovered that there is no apparent negative correlation between the variables and that all road types appear to have high correlation. Month number has positive correlation with every variable which means the mean of accidents increases over time.

HOTSPOT ANALYSIS

```
formatted_accidents$longitude <- coalesce(as.numeric(formatted_accidents$longitude), NA)
formatted_accidents$latitude <- coalesce(as.numeric(formatted_accidents$latitude), NA)
formatted_accidents <- formatted_accidents %>%
                        drop_na(longitude, latitude)
SF_accidents = st_as_sf(formatted_accidents,coords = c("longitude", "latitude"),crs = 27700 )
samplePlot<-SF_accidents %>% sample_n(2000)
plot(samplePlot)
```




```
names(SF_accidents)[1]<-"accident_index"
```

In order to identify the accident hotspots. For this, a sample of 2000 observations was plotted based on the factors after the data was initially processed using format sf to exclude observations with no coordinates. The data was then transformed into spatial points after coordinates were established based on the latitude and longitude of the accidents. After that, the spatial points are plotted. The plot makes it evident that the data contains a considerable number of points.

```
police_boundaries = sf::st_read('boundary_shape/Police_Force_Areas_UK.shp')
```

```
## Reading layer `Police_Force_Areas_UK' from data source
##   `F:\WWU\Winter_2022\AOSD\AOSD_Final_Assignment\boundary_shape\Police_Force_Areas_UK.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 43 features and 10 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -6.360599 ymin: 49.88575 xmax: 1.763151 ymax: 55.81107
## Geodetic CRS:  WGS 84
```

```
police = st_transform(police_boundaries, 27700)
names(police_boundaries)
```

```
##  [1] "OBJECTID"   "pfa16cd"    "pfa16nm"    "bng_e"      "bng_n"
##  [6] "long"       "lat"        "GlobalID"   "SHAPE_Leng" "SHAPE_Area"
## [11] "geometry"
```

```
names(police)
```

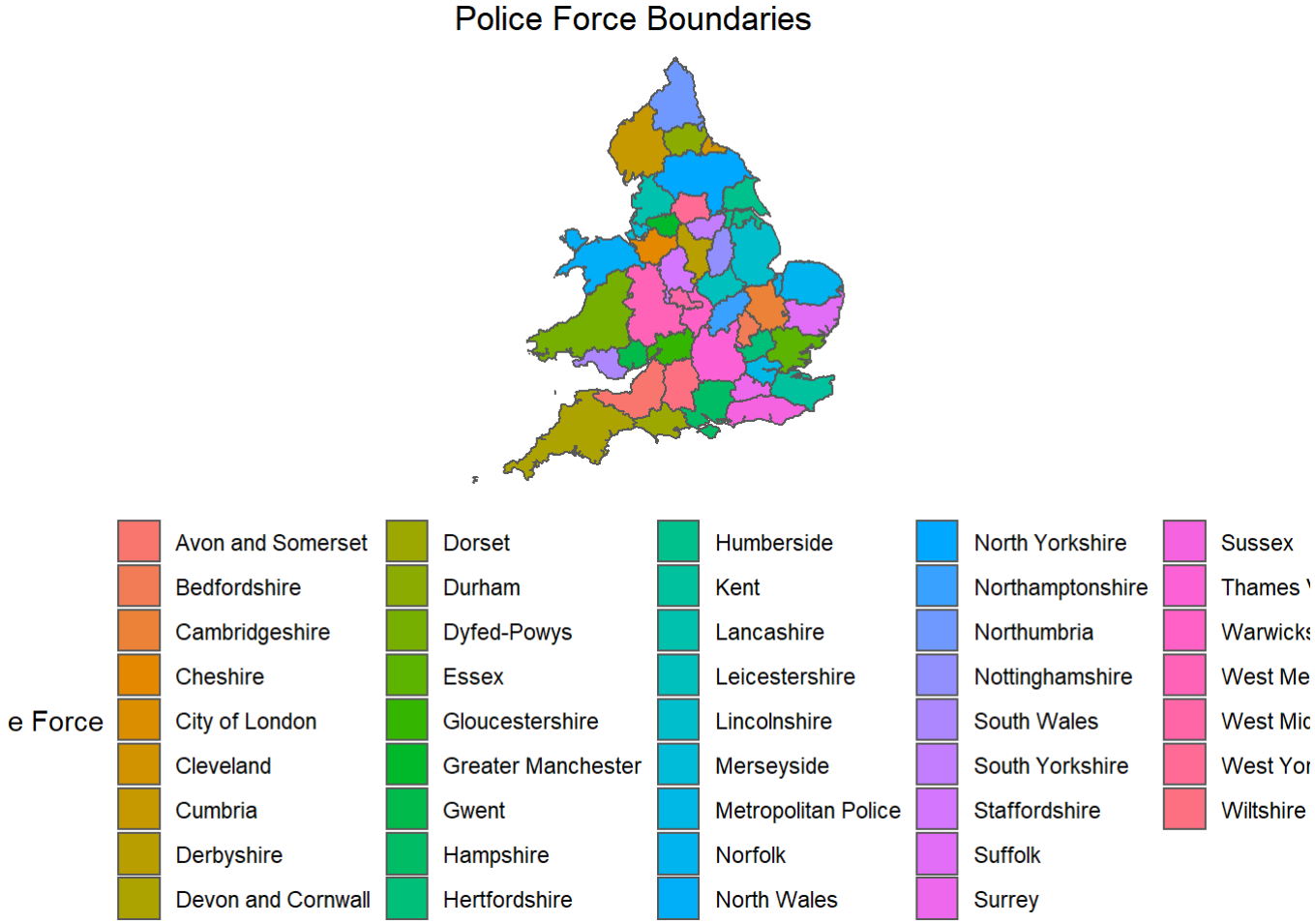
```
##  [1] "OBJECTID"   "pfa16cd"    "pfa16nm"    "bng_e"      "bng_n"
##  [6] "long"       "lat"        "GlobalID"   "SHAPE_Leng" "SHAPE_Area"
## [11] "geometry"
```

```
police = police[c("pfa16cd", "pfa16nm")]
glimpse(police)
```

```
## Rows: 43
## Columns: 3
## $ pfa16cd  <chr> "E23000001", "E23000002", "E23000003", "E23000004", "E2300000~
## $ pfa16nm  <chr> "Metropolitan Police", "Cumbria", "Lancashire", "Merseyside",~
## $ geometry <MULTIPOLYGON [m]> MULTIPOLYGON (((537545.6 19..., MULTIPOLYGON (((~
```

There are data available in (https://hub.arcgis.com/datasets/ons (https://hub.arcgis.com/datasets/ons)::police-force-areas-december-2016-full-clipped-boundaries-in-england-and-wales) about police boundaries that provides a variety of details on the England police limits. The boundaries of each England police force are one of the details. Also, as the accident data covers the whole of England and can be utilized for this study, it can be regarded as one of the significant data because it includes the police jurisdiction of each accident. So I have download the boundary data from here but the shapefile size large, hence I have simplified the geometry and stored as shapefile with Rgdal library as “Police_Force_Areas_uk.shp” which used here.

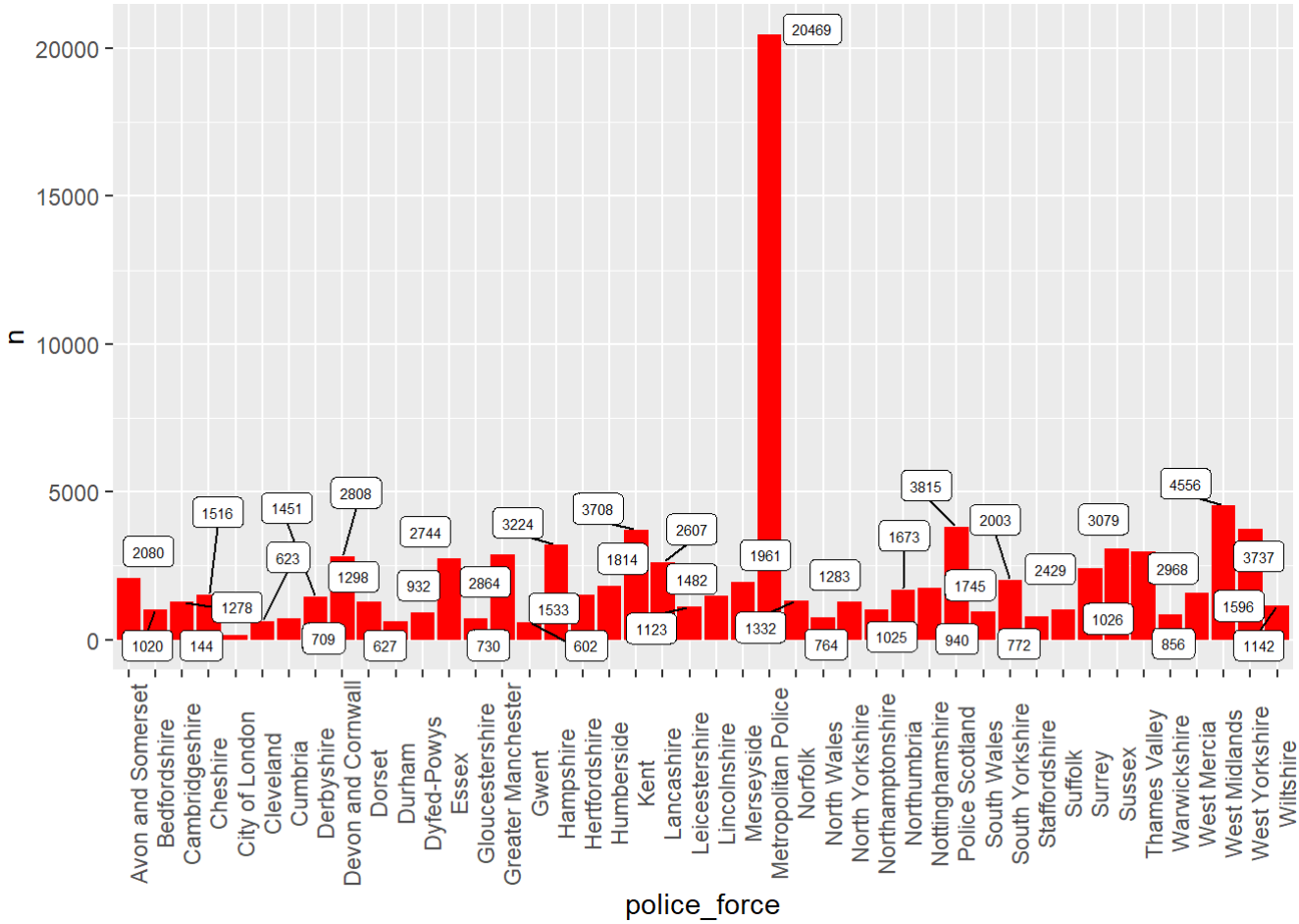
```
# Plotting police force boundaries
ggplot() +
  geom_sf(data = police, aes(fill = pfa16nm)) +
  scale_fill_discrete(name = "Police Force") +
  labs(title = "Police Force Boundaries") +
  theme_void() +
  theme(legend.position = "bottom")
```




```
  geom_sf_text(aes(label = police_force), size = 3, color = "black")
```

```
## mapping: label = ~police_force
## geom_text: parse = FALSE, check_overlap = FALSE, na.rm = FALSE
## stat_sf_coordinates: na.rm = FALSE, fun.geometry = NULL
## position_identity
```

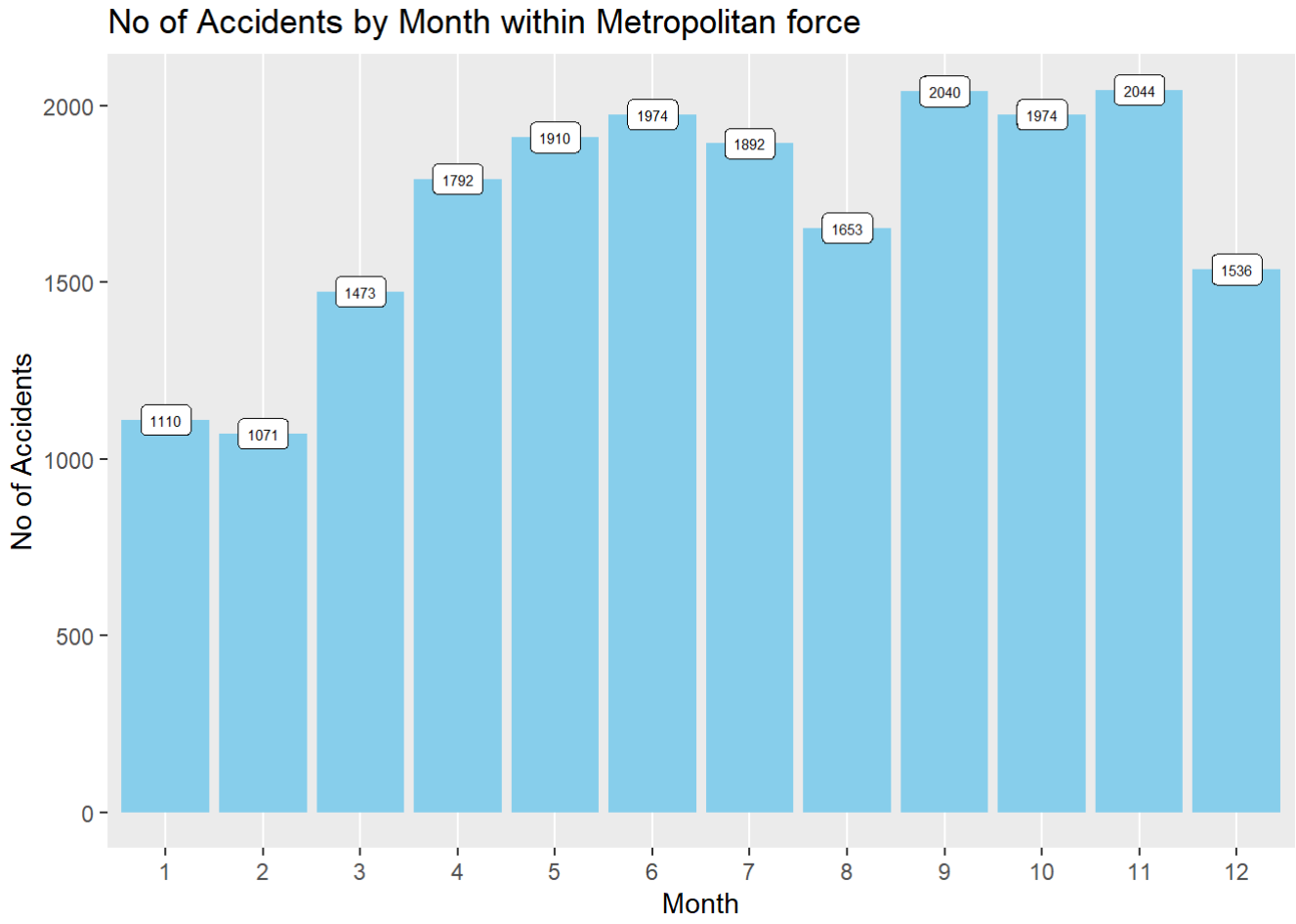
```
filtered %>%
  group_by(police_force) %>%
  summarise("n"=n()) %>%
  ggplot(data=.,aes(police_force,n)) +
  geom_col(fill='red')+theme(axis.text.x = element_text(angle = 90))+
  geom_label_repel(aes(label=n),size=2)
```




```
fatal_counts <- SF_accidents %>%
                filter(accident_severity == "Fatal") %>%
                group_by(police_force) %>%
                summarise(count = n())
fatal_counts
```

```
## Simple feature collection with 44 features and 2 fields
## Geometry type: GEOMETRY
## Dimension:     XY
## Bounding box:  xmin: -6.411079 ymin: 50.08539 xmax: 1.738033 ymax: 59.08816
## Projected CRS: OSGB 1936 / British National Grid
## # A tibble: 44 x 3
##    police_force       count                                          geometry
##    <chr>              <int>                                    <GEOMETRY [m]>
##  1 Avon and Somerset     43 MULTIPOINT ((-3.649171 51.13259), (-3.527527 51.201~
##  2 Bedfordshire          16 MULTIPOINT ((-0.642223 51.96342), (-0.59308 52.1586~
##  3 Cambridgeshire        43 MULTIPOINT ((-0.486782 52.59481), (-0.480107 52.636~
##  4 Cheshire              26 MULTIPOINT ((-2.944491 53.24264), (-2.865651 53.263~
##  5 City of London         1                          POINT (-0.075176 51.51109)
##  6 Cleveland              9 MULTIPOINT ((-1.391481 54.61662), (-1.36116 54.5969~
##  7 Cumbria               22 MULTIPOINT ((-3.488816 54.70888), (-3.391363 54.664~
##  8 Derbyshire            29 MULTIPOINT ((-1.778282 53.28819), (-1.703443 52.881~
##  9 Devon and Cornwall    48 MULTIPOINT ((-5.473567 50.17899), (-5.410937 50.190~
## 10 Dorset                24 MULTIPOINT ((-2.933471 50.72458), (-2.761224 50.829~
## # ... with 34 more rows
```

```
filt_Metro<-filtered%>%filter(police_force=="Metropolitan Police")

filt_Metro %>%
  group_by(month_number) %>%
  summarise("n"=n()) %>%
  ggplot(data=., aes(month_number,n)) +
  geom_col(fill='skyblue') +
  theme(panel.grid.minor = element_blank(),panel.grid.major.y = element_blank()) + geom_label
(aes(label=n),size=2)+
  labs(title="No of Accidents by Month within Metropolitan force",
       x="Month",y="No of Accidents")
```

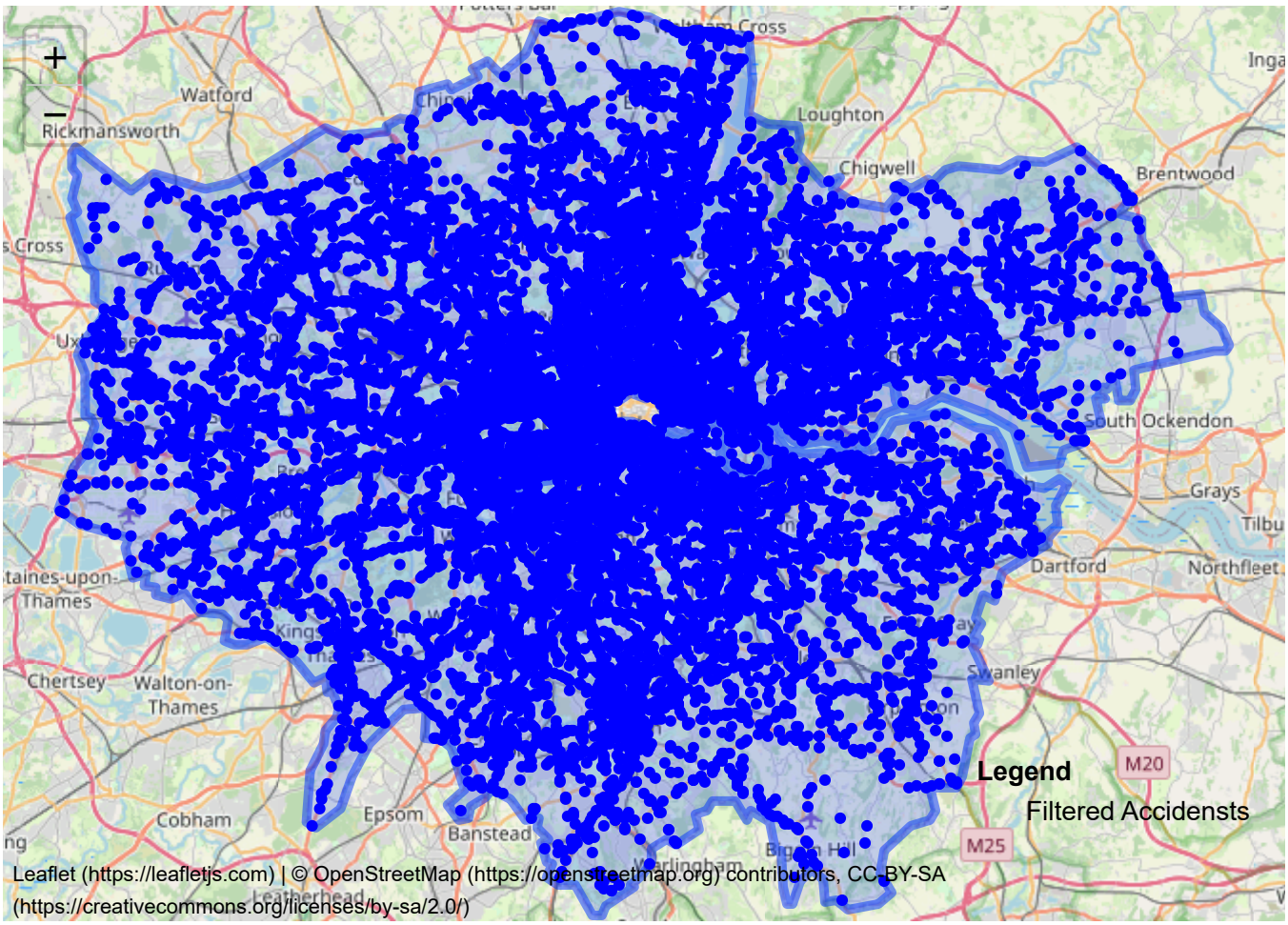


It was observed that the majority of fatal accidents and the majority of total accidents occurred within the jurisdiction of the Metropolitan police force. As the boundary data doesn't define the boundaries for Police Scotland, where the first-highest number of fatal accidents occur, I did an analysis for the Metropolitan Police. As a result, the data was then spatially filtered based on the accidents that took place within this jurisdiction. A month-by-month visualization of the number of accidents has been plotted

```
Metro_bound <- st_read("boundary_shape/222.shp")
```

```
## Reading layer `222' from data source
##   `F:\WWU\Winter_2022\AOSD\AOSD_Final_Assignment\boundary_shape\222.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 1 feature and 9 fields
## Geometry type: LINESTRING
## Dimension:     XY
## Bounding box:  xmin: -0.5097014 ymin: 51.28676 xmax: 0.3339026 ymax: 51.69188
## Geodetic CRS:  WGS 84
```

```
filtered_Metro <- filtered %>% filter(police_force == "Metropolitan Police")
filtered_Metro_sf <- st_as_sf(filtered_Metro, coords = c("longitude", "latitude"), crs = 432
6)
filtered_Metro_sf <- st_transform(filtered_Metro_sf, st_crs(Metro_bound))
metropolitan <- police %>% filter(pfa16nm == "Metropolitan Police")
leaflet() %>%
  addTiles() %>%
  addPolygons(data = Metro_bound, fillOpacity = 0.2) %>%
  addCircleMarkers(data = filtered_Metro_sf, radius = 3,
                   color = "blue", stroke = FALSE, fillOpacity = 1) %>%
  addLegend("bottomright",
            title = "Legend",
            colors = "blue",
            labels = "Filtered Accidensts")
```



The points are then displayed on a mapview interactive leaflet so that you can pinpoint specific accidents as well as the border of the metropolitan force. The Metropolitan police police force boundary was extracted from the police force boundary, processed in QGIS, and then transformed back into a shapefile with only

metropolitan police boundary. The boundary and the points are both changed into the same projection and made into sf objects.

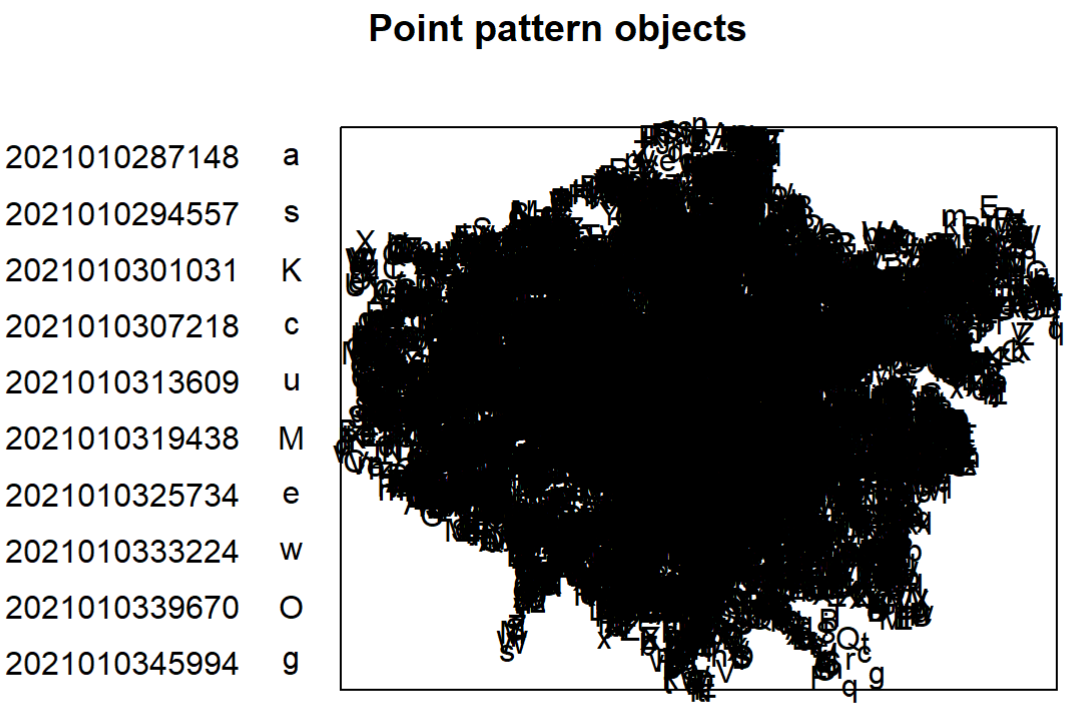
DENSITY ESTIMATES FOR ALL ACCIDENTS

```
filtered_Metro_sf_proj <- st_transform(filtered_Metro_sf, 27700)
de_points <- as.ppp(filtered_Metro_sf_proj)
```

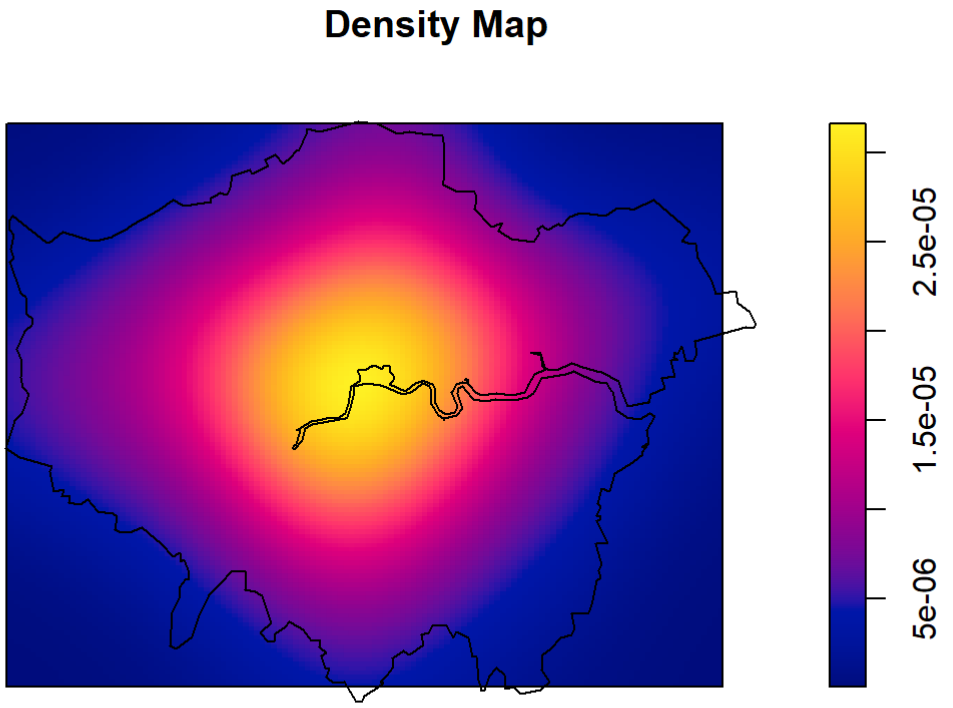
```
## Warning in as.ppp.sf(filtered_Metro_sf_proj): only first attribute column is
## used for marks
```

```
metro_poly <- st_cast(Metro_bound, "POLYGON")
metro_poly_proj <- st_transform(metro_poly, 27700)
de_window <- as.owin(metro_poly_proj)
de_ppp <- as.ppp(de_points, de_window)
plot(de_ppp, main = "Point pattern objects")
```

```
## Warning in default.charmap(ntypes, chars): Too many types to display every type
## as a different character
```

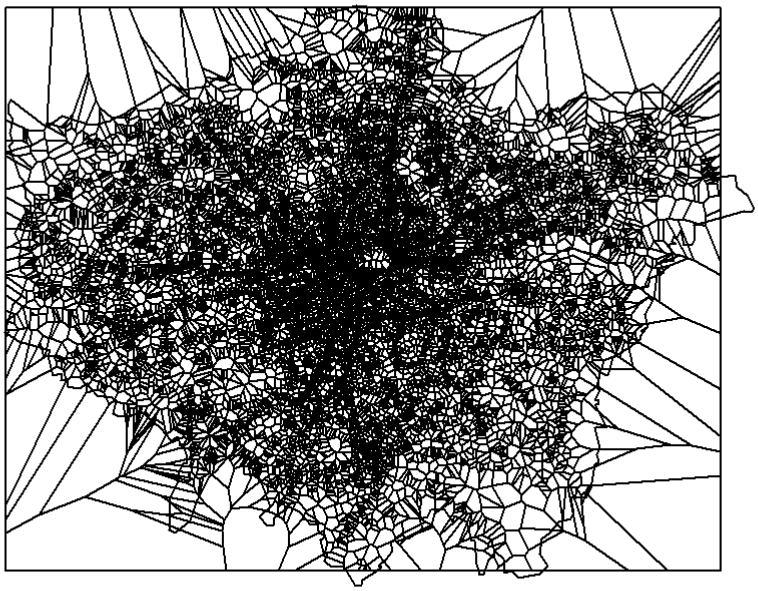


```
# compute the density over de_points and de_window_new
dens <- density.ppp(de_points, window = de_window)

# plot the density
plot(dens, main = "Density Map")
plot(de_window, add = TRUE)
```




```
# image(dens, col = terrain.colors)
# contour(dens, add = TRUE)
# plot(de_window, add = TRUE)

voronoi <- dirichlet(de_points)
plot(voronoi, main = "Voronoi Map")
plot(de_window, add = TRUE)
```

Voronoi Map

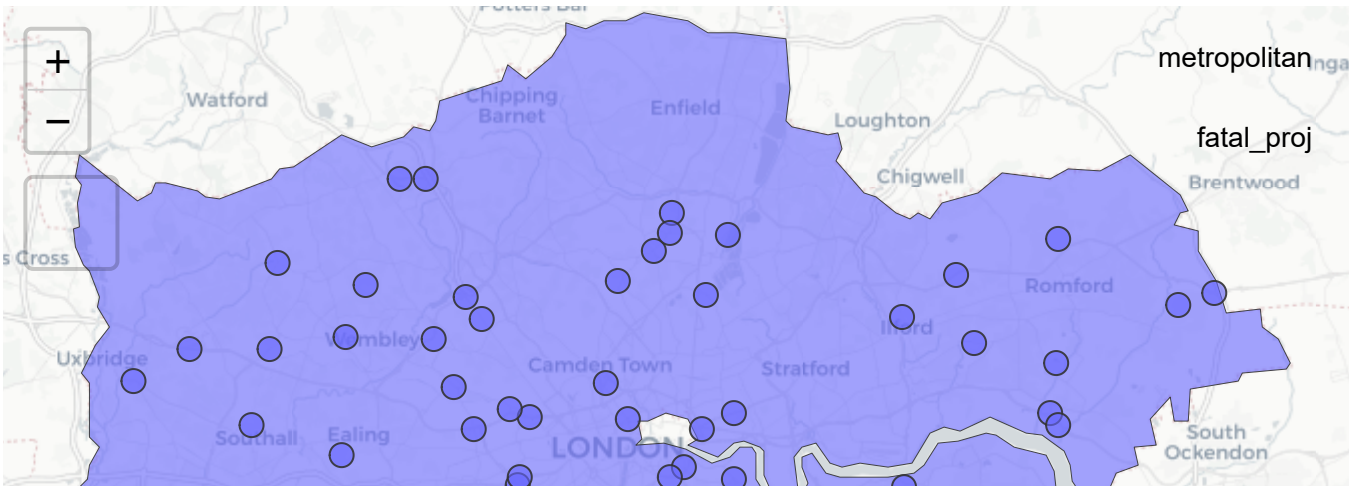


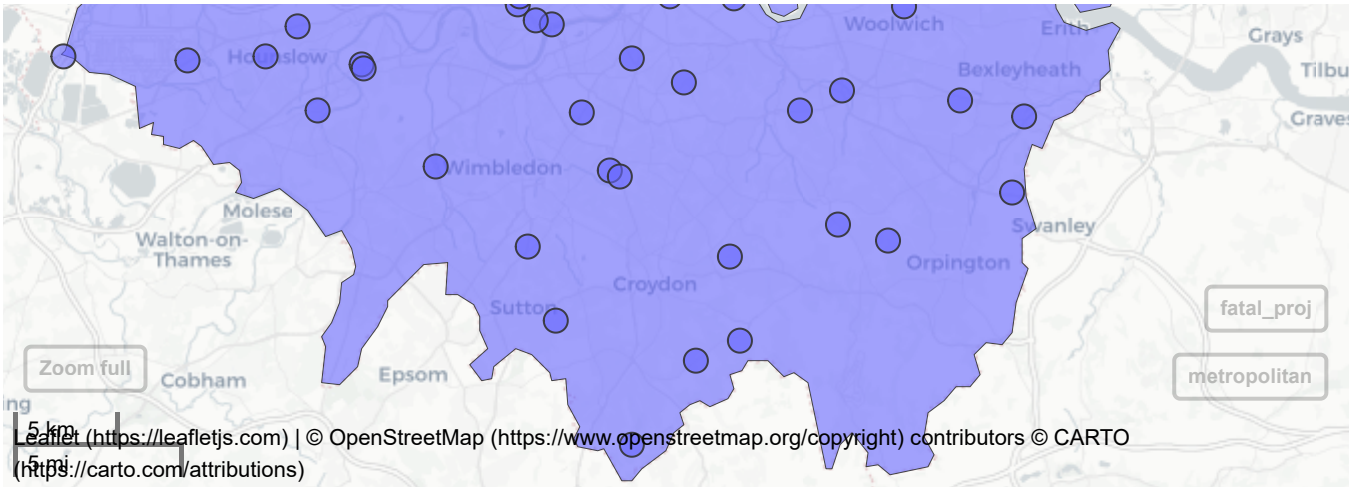
The density estimates within the Metropolitan force's boundary are high in the center of the boundary and zero or below virtually everywhere along the border, indicating that accidents happen more frequently in the region's center than along its boundaries.

Voronoi tessellation: You can create a Voronoi tessellation of the point pattern using the dirichlet function from the spatstat package. This will create polygons around each point that show the area of the plane that is closest to that point.

DENSITY ESTIMATES FOR FATAL ACCIDENTS

```
fatal <- filtered %>% filter(accident_severity == 'Fatal') %>% filter(police_force == "Metrop
olitan Police")
fatal_sf <- st_as_sf(fatal, coords = c("longitude", "latitude"), crs = 4326)
fatal_proj <- st_transform(fatal_sf, 27700)
mapview(metropolitan)+mapview(fatal_proj)
```

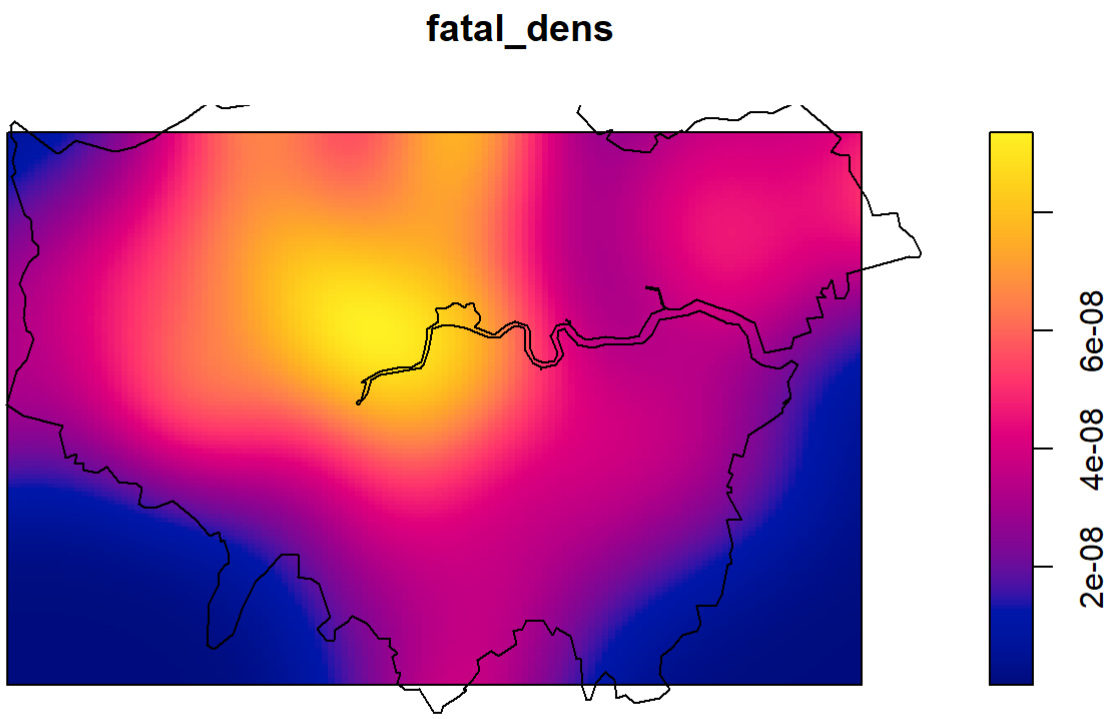





```
fatal_points <- as.ppp(fatal_proj)
```

```
## Warning in as.ppp.sf(fatal_proj): only first attribute column is used for marks
```

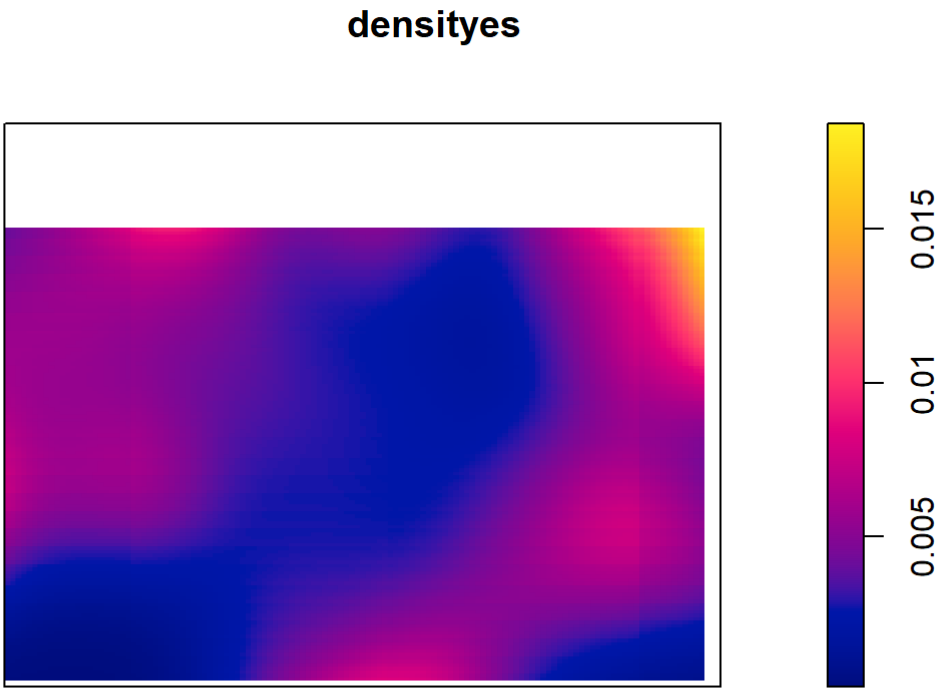
```
fatal_dens <- density.ppp(fatal_points)
plot(fatal_dens)
plot(de_window, add = TRUE)
```




```
densityes<-fatal_dens/dens
```

```
## Warning: the images 'e1' and 'e2' were not compatible
```

```
densityes[densityes<0|densityes>1]<-NA
plot(densityes)
```



UNDERSTANDING

Finally, densities are estimated for fatal accidents that occur within the same boundary. The findings indicate that the higher density estimates for fatal accidents are more widespread than the higher density estimates for all incidents within the same region. This suggests that fatal accidents under the Metropolitan force's jurisdiction are spread out rather than occurring in just one or two locations. This was accomplished by dividing the fatal density grid by the density grid for all accidents and fitting the density values in the 0 to 1 range.

CONCLUSION

This study explores accidents according to various categories and the density estimation was done and visualized the accidents inside the Metropolitan Police force's jurisdiction. As per analysis the higher density estimates, the frequency of accidents increases in the boundary's center and decreases as it moves outside. The same is true for fatal accidents, however compared to higher density estimates for all incidents, there is a wider spread for fatal accidents. It was also found that urban areas have a higher amount of recorded accidents than rural but rural areas have more fatal accidents. Using a cross tabulation matrix to calculate the correlation between different types of roads, it was also discovered that single carriageway roads contributed to more accidents. This study shows where there is a necessity for implementing road safety rules in order to avoid future accidents within the boundaries of Metropolitan force jurisdiction. There are other parameters/variables that are not included in the study because it only focused on using the different types of roads and the weather. Better analysis and prediction of road accident trends will result from a more detailed investigation that includes all the characteristics as well as additional information on road safety.

References

1. https://www.mailman.columbia.edu/research/population-health-methods/hot-spot-detection (https://www.mailman.columbia.edu/research/population-health-methods/hot-spot-detection)
2. https://glenbambrick.com/2016/01/21/what-is-hotspot-analysis/ (https://glenbambrick.com/2016/01/21/what-is-hotspot-analysis/)
3. Thakali, L., Kwon, T. J., & Fu, L. (2015). Identification of crash hotspots using kernel density estimation and kriging methods: a comparison. Journal of Modern Transportation, 23(2), 93-106.
4. https://cran.r-project.org/web/packages/stats19/index.html (https://cran.r-project.org/web/packages/stats19/index.html)
5. Wang, Meina, et al. “Spatial and Temporal Distribution Analysis of Traffic Accidents Using GIS-Based Data in Harbin.” Journal of Advanced Transportation 2021 (2021): 1-10.
6. Prasannakumar, V., et al. “Spatio-temporal clustering of road accidents: GIS based analysis and assessment.” Procedia-social and behavioral sciences 21 (2011): 317-325.
7. Hazaymeh, Khaled, Ali Almagbile, and Ahmad H. Alomari. “Spatiotemporal analysis of traffic accidents hotspots based on geospatial techniques.” ISPRS International Journal of Geo-Information 11.4 (2022): 260.
8. Lovelace R., (2019).Reproducible road safety research: an exploration of the shifting spatial and temporal distribution of car-pedestrian crashes https://github.com/Robinlovelace/stats19-gisruk (https://github.com/Robinlovelace/stats19-gisruk)