Milestone 4: UK Road Traffic Accidents Data Analysis

Goal

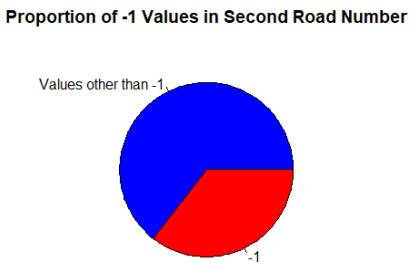
The goal of this project is to analyse the data and figure out various factors that are leading to accidents in the UK.

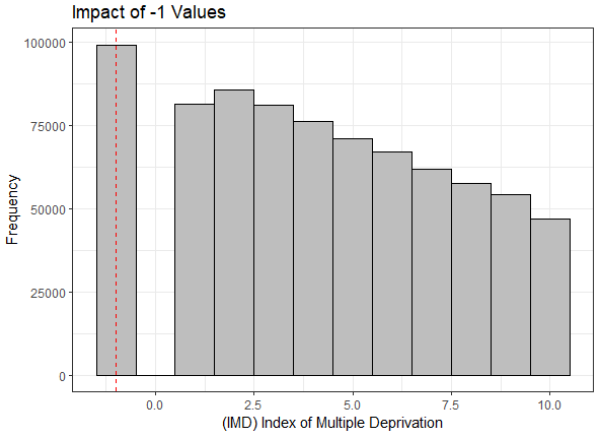
Data Source

The dataset used is an open data from the Department of Transport UK of road traffic accidents occurring between 2016 and 2020, taken from Kaggle. It consists of data related to accidents, collisions, victims, and vehicles involved in the incidents. The data is divided into three excel sheets which has Accident, Casualty and Vehicle data. We also have a lookup table to understand different codes used in the data.

Data Processing

For tidying the data available, the excel sheets were imported into the three different data frames. After analysing the data, it is evident that some of the columns in the data were not useful considering the goal of the project. So, these columns were removed from the data frames. Handling of NULL and -1 values is done by imputing mean or median values. In case of longitude and latitude values of the accident place we have used median values as there are less than 15 NULL values. For other data related to vehicle, weather, and speed, we have used mean of the columns while for casualty and victim details, we have used median of the columns as replacement. At the end we have merged the three data frames in one so that it is easier to handle it.





Analytical Plan

In the data, we observe the data and list out the factors that can be considered as causes of the accident. To get an idea of which values can be included in the factors contributing to accidents, different visualizations of data are used. Each of the column values in the data are plotted against the frequency of accident to understand the relationship between the variables.

In these plots, we found that following factors can be viewed for accidents' causes –

Column Name	Column Information
accident_year	Year in which accident occurred
road_type	Type of road
speed_limit	Speed limit set for the area
second_road_class	Class of road for vehicles
second_road_number	Road on which accident occurred
light_conditions	Day/Night light conditions at the time of accident
weather_conditions	Weather condition at the time of the accident
road_surface_condition	Road condition at the time of the accident
special_conditions_at_site	Any special issue at the time of the accident

carriageway_hazards	If the vehicle involved was carrying hazards
urban_or_rural_area	Area of the accident location
trunk_road_flag	If the road was managed by Highways of England
pedestrian_location	Location of pedestrian involved in the accident
vehicle_type	Type of vehicle
vehicle_manoeuvre	Position of vehicle
skidding_and_overturning	If there is skidding or overturning
age_of_driver	Age of Driver
sex_of_driver	Sex of Driver
age_of_vehicle	Age of Vehicle
driver_imd_decile	Index of Multiple Deprivation (IMD) decile of the driver involved in the accident

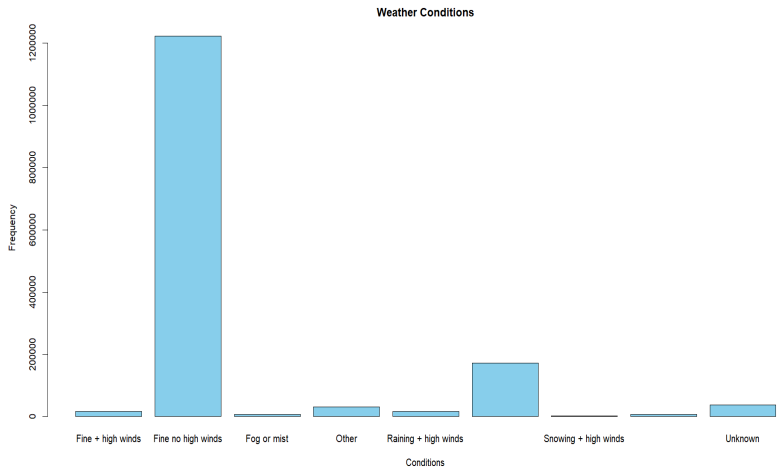
Here we use the Poisson Regression Analysis technique to find the aspects that influence accidents. In this technique, we count the number of occurrences of event that are independent to each other. It is like our case, where accident is an event that occurs independently.

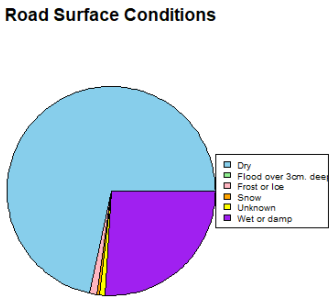
Poisson regression assumes that the counts are independent and have constant variance (mean equal to variance). It is also assumed that the counts are not over dispersed, meaning that the observed variance is approximately equal to the predicted variance. After plotting the various candidates for factors of accidents, it is clear that the data is spread out well and there are not many outliers. It allows you to assess the relationship between the predictors (such as road conditions, weather conditions, etc.) and the counts of accidents. It estimates the effects of the predictors on the expected number of accidents, accounting for their influence on the occurrence rate. So, it is suitable for our research question. Poisson regression models the rate of events (accidents) as a function of predictors. By considering the rate of accidents instead of the raw count, it will help us to account for the varying exposure or time-at-risk across different conditions.

The second exploratory direction is to know the year in which number of accidents took place in large numbers. For this purpose, we have a column as accident year that will be helpful in this analysis. We can plot accident year against the frequency of accidents to get this result.

The third exploratory direction is to deduce the area of UK where the occurrence of accidents is quite frequent. We have data related to longitude and latitude of the location that can be used in the analysis. For this purpose, we plan to show locations based on the above column data.

To get the location list from the available data, we use map view package, which will help us understand which part of UK has more accidents.





Results

The dataset after wrangling of data is quite large with 2,45895 rows and 20 columns. For feasibility and time efficiency, we take a sample of this dataset for modelling Poisson regression. Here to have a sample that is representative of the given dataset, we have used systematic sampling technique. It is a sampling technique that involves selecting elements from a population at regular intervals. The process begins by randomly selecting a starting point in the population, and then every kth element thereafter is included in the sample until the desired sample size is reached.

After taking a sample of 3500 records, we apply Poisson regression analysis on the data. Initially, we apply the regression analysis on the data with all dependent variables mentioned above. Later we apply Stepwise AIC (Akaike Information Criterion). It is commonly used in regression analysis to determine the best subset of predictor variables to include in the model.

The stepwise AIC procedure starts with an initial model that includes all predictor variables. It then iteratively adds or removes variables from the model based on their impact on the AIC value. In each step, the AIC is calculated for different models with different combinations of predictor variables. The model with the lowest AIC is selected as the best model. Since our model is currently have around 20 predictor variables, we reduce the number of these variables using stepwise AIC procedure.

```
Step:  AIC=6551
number_of_accidents ~ speed_limit + weather_conditions + carriageway_hazards +
    trunk_road_flag + pedestrian_location + sex_of_driver

                          Df Deviance    AIC
<none>                       1129.0 6551.0
+ road_type               1   1127.6 6551.7
+ accident_year            1   1127.8 6551.9
+ vehicle_type             1   1127.8 6551.9
- carriageway_hazards      1   1131.9 6551.9
+ skidding_and_overturning   1   1127.9 6552.0
+ vehicle_manoeuvre        1   1128.4 6552.4
+ special_conditions_at_site  1   1128.5 6552.5
+ second_road_number        1   1128.6 6552.6
+ road_surface_conditions    1   1128.7 6552.7
- trunk_road_flag           1   1132.7 6552.8
+ driver_imd_decile         1   1128.7 6552.8
+ second_road_class         1   1128.9 6552.9
+ light_conditions          1   1128.9 6552.9
+ age_of_driver             1   1128.9 6552.9
+ urban_or_rural_area       1   1128.9 6552.9
+ age_of_vehicle            1   1128.9 6553.0
- weather_conditions        1   1133.9 6554.0
- speed_limit               1   1137.7 6557.7
- sex_of_driver             1   1137.8 6557.8
- pedestrian_location       1   1148.8 6568.9
```

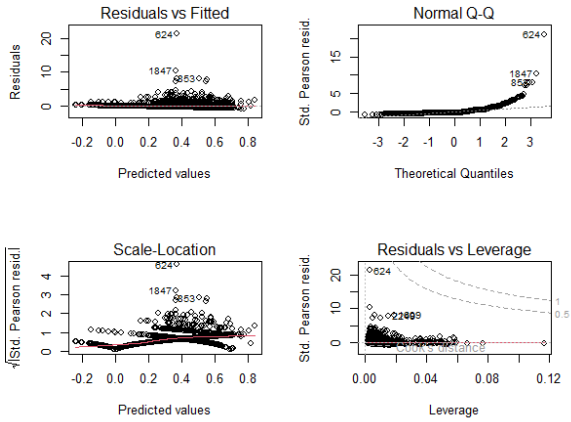
In the stepwise AIC result, we get the variables – speed limit, weather conditions , carriageway hazards, trunk road flag, pedestrian location, and sex of driver, that seem to have strong relationship with the dependent variable.

After applying regression on the updated set of variables, we observe the summary.

```
> summary(step_model)

Call:
glm(formula = number_of_accidents ~ speed_limit + weather_conditions +
    carriageway_hazards + trunk_road_flag + pedestrian_location +
```

```
   sex_of_driver, family = poisson, data = train_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8964  -0.4329  -0.3400   0.1898  10.4574

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)         0.394729   0.170042   2.321  0.02027 *
speed_limit         0.004142   0.001391   2.978  0.00290 **
weather_conditions -0.021634   0.009898  -2.186  0.02884 *
carriageway_hazards -0.028809   0.017470  -1.649  0.09915 .
trunk_road_flag     -0.128780   0.066102  -1.948  0.05139 .
pedestrian_location -0.043733   0.010298  -4.247 2.17e-05 ***
sex_of_driver        0.082002   0.027396   2.993  0.00276 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

We can see that the intercept term has an estimated coefficient of 0.394729, indicating the expected log count of accidents when all predictor variables are zero. The variables speed limit and sex of the driver have positive coefficients while the remaining variables have negative coefficients. We also see that all the variables are significant to the regression model.



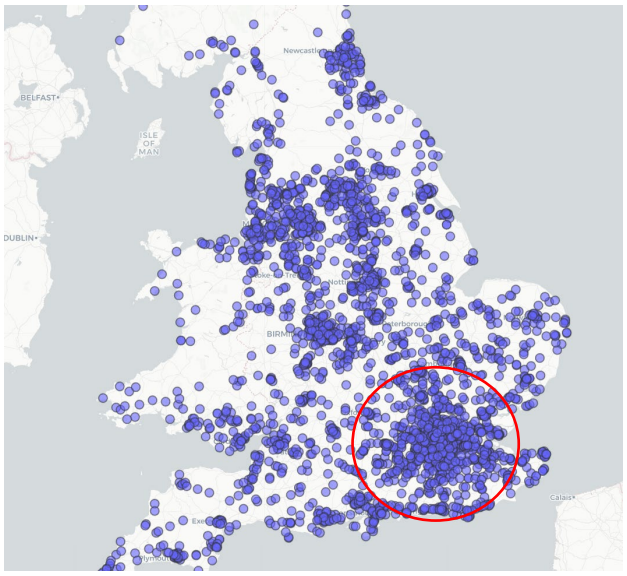
Consider the Residuals vs Fitted plot, we can see that there is no pattern in the scatter and the data points are scattered around the reference red line.. We are also able to spot some outliers here. In the Normal Q-Q plot, the data is mostly aligned to the reference quantile line. In the case of Scale-location plot, we can see that the variance is not constant in the data. For Residuals vs Leverage plot, it is evident that all the data points lie in the 1 and 0.5 contours.

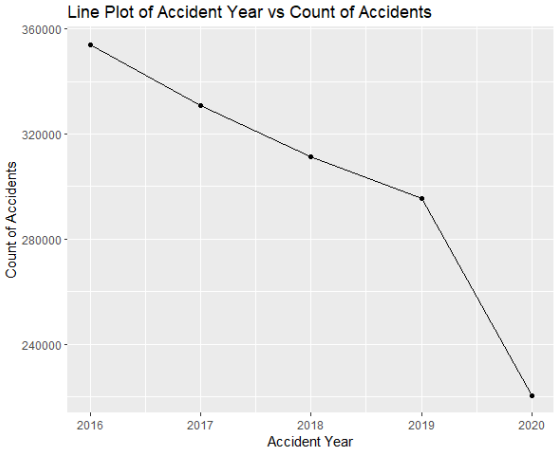
```
# Calculate RMSE
> predicted_counts <- predict(model, newdata = test_data, type = "response")
> rmse <- sqrt(mean((test_data$number_of_accidents - predicted_counts)^2))
> print(rmse)
[1] 1.028794
```

To calculate the accuracy of the model, we use Root Mean Squared Error. It is a measure of the average difference between the observed and predicted values in a model. In this case, that the Poisson regression model has a reasonably good fit to the data, with the predicted counts deviating by an average of around 1.03 units from the observed counts.

Now, we consider the second exploratory idea in which we find the year in which maximum number of accidents took place. After plotting the graph of count of accidents against the accident year, we see that maximum number of accidents occurred in 2016.

In the third exploratory idea, we wanted to see if which location has highest number of accidents in UK. For that we have plotted the locations of accident using mapview. In this plot, we can clearly see that London has highest number of accidents.





Discussion

The accident data contains a large number of observations which can provide a robust basis for analysis and modelling. The availability of detailed accident information allows for a comprehensive analysis of different aspects of accidents, such as location, time, and causal factors. The use of geographic coordinates (longitude and latitude) allows for spatial analysis and visualization, enabling insights into patterns and trends across different locations.

The disadvantage of this dataset is that it contains values in coded form. So, it is necessary to understand the codes for each column along with the column names. Another problem with the dataset is that it does not contain number of accident values. We have to aggregate the data as per the columns that are considered in the model. The data may not capture accidents that were not reported or did not involve any injuries, potentially leading to an underrepresentation of certain types of accidents.

This model can be used to assess the risk of accidents on various factors such as road type, speed limit, light conditions, pedestrian location, and vehicle manoeuvre. We are also able to identify high-risk areas or situations where interventions are needed to reduce the occurrence of accidents. The insights from the model can inform policy decisions related to road infrastructure, traffic management, and safety regulations. Understanding the factors that contribute to accidents can assist in allocating resources effectively. By identifying the predictors with the most significant impact, resources can be prioritized towards interventions that address those factors, leading to more efficient use of resources and potentially reducing the number of accidents.

Appendix

Data Pre-processing Code

```
dataset_accident <- read_csv("C:/Users/tejas/OneDrive/Desktop/project - class 765/dft-road-casualty-statistics-accident-last-5-years.csv",
  col_types = cols(accident_index = col_character(),
                   accident_year = col_integer(),  location_easting_osgr = col_integer(),
                   location_northing_osgr = col_integer(),
                   police_force = col_integer(),accident_severity = col_character(),
                   number_of_vehicles = col_integer(), number_of_casualties = col_integer(),
                   date = col_date(format = "%d/%m/%Y"), day_of_week = col_integer(),
                   time = col_time(format = "%H:%M"), first_road_class = col_integer(),
                   first_road_number = col_character(), road_type = col_integer(),
                   speed_limit = col_integer(), junction_detail = col_integer(),
                   junction_control = col_integer(), second_road_class = col_integer(),
                   second_road_number = col_integer(),pedestrian_crossing_human_control = col_integer(),
                   pedestrian_crossing_physical_facilities = col_integer(), light_conditions = col_integer(),
                   weather_conditions = col_integer(),road_surface_conditions = col_integer(),
                   special_conditions_at_site = col_integer(), carriageway_hazards = col_integer(),
                   urban_or_rural_area = col_integer(),
                   did_police_officer_attend_scene_of_accident = col_integer(),
                   trunk_road_flag = col_integer()))

#removing irrelevant columns
dataset_accident <- dataset_accident %>% select(-c(accident_reference,
                                        location_easting_osgr,  location_northing_osgr,
                                        police_force,  local_authority_district ,local_authority_ons_district,
                                        local_authority_highway,
                                        did_police_officer_attend_scene_of_accident,
                                        lsoa_of_accident_location))


#Replace null values with median for longitude and latitude

dataset_accident <- dataset_accident %>%
  mutate(longitude = ifelse(is.na(longitude), median(longitude, na.rm = TRUE), longitude),
         latitude = ifelse(is.na(latitude), median(latitude, na.rm = TRUE), latitude))

#replacing -1 values

cols_to_replace <- c("road_type","speed_limit","junction_detail",
                     "junction_detail","second_road_number","pedestrian_crossing_human_control",
                     "pedestrian_crossing_physical_facilities","light_conditions",
                     "weather_conditions","road_surface_conditions","special_conditions_at_site",
                     "carriageway_hazards","urban_or_rural_area","trunk_road_flag")

#replace null values for speed limit
mean_speed <- mean(dataset_accident$speed_limit, na.rm = TRUE)
dataset_accident$speed_limit <- ifelse(is.na(dataset_accident$speed_limit), mean_speed,
dataset_accident$speed_limit)

for (col in cols_to_replace) {
  na_vals <- dataset_accident[[col]] == -1
  if (sum(!is.na(na_vals) & na_vals) > 0) {
    col_mean <- mean(dataset_accident[[col]][!na_vals])
    dataset_accident[[col]][na_vals] <- col_mean
  }
}
```

```
dataset_casualty <- read_csv("C:/Users/tejas/OneDrive/Desktop/project - class 765/dft-road-casualty-statistics-casualty-last-5-years.csv",
                  col_types = cols(accident_index = col_character(), accident_year = col_integer(),
                  accident_reference = col_character(), vehicle_reference = col_integer(),
                  casualty_reference = col_integer(),casualty_class = col_integer(),
                  sex_of_casualty = col_integer(),age_of_casualty = col_integer(),
                  age_band_of_casualty = col_integer(), casualty_severity = col_integer(),
                  pedestrian_location = col_integer(), pedestrian_movement = col_integer(),
                  car_passenger = col_integer(),  bus_or_coach_passenger = col_integer(),
                  pedestrian_road_maintenance_worker = col_integer(), casualty_type = col_integer(),
                  casualty_home_area_type = col_integer(), casualty_imd_decile = col_integer()))

#remove irrelevant columns
dataset_casualty <- dataset_casualty %>% select(-c(accident_reference,accident_year))

#replacing -1 values
cols_to_replace <- c("sex_of_casualty","age_of_casualty","age_band_of_casualty",
                   "pedestrian_location","pedestrian_movement","car_passenger",
                   "bus_or_coach_passenger","pedestrian_road_maintenance_worker",
                   "casualty_imd_decile","casualty_home_area_type")

for (col in cols_to_replace) {
  na_vals <- dataset_casualty[[col]] == -1
  if (sum(!is.na(na_vals) & na_vals) > 0) {
    col_mean <- mean(dataset_casualty[[col]][!na_vals])
    dataset_casualty[[col]][na_vals] <- col_mean
  }
}

dataset_vehicle <- read_csv("C:/Users/tejas/OneDrive/Desktop/project - class 765/dft-road-casualty-statistics-vehicle-last-5-years.csv",
                  col_types = cols(accident_index = col_character(),accident_year = col_integer(),
                  accident_reference = col_character(),vehicle_reference = col_integer(),
                  vehicle_type = col_integer(),  towing_and_articulation = col_integer(),
                  vehicle_manoeuvre = col_integer(),vehicle_direction_from = col_integer(),
                  vehicle_direction_to = col_integer(), vehicle_location_restricted_lane = col_integer(),
                  junction_location = col_integer(),  skidding_and_overturning = col_integer(),
                  hit_object_in_carriageway = col_integer(),vehicle_leaving_carriageway = col_integer(),
                  hit_object_off_carriageway = col_integer(), first_point_of_impact = col_integer(),
                  vehicle_left_hand_drive = col_integer(), journey_purpose_of_driver = col_integer(),
                  sex_of_driver = col_integer(), age_of_driver = col_integer(),
                  age_band_of_driver = col_integer(),engine_capacity_cc = col_integer(),
                  propulsion_code = col_integer(), age_of_vehicle = col_integer(),
                  generic_make_model = col_character(),driver_imd_decile = col_integer(),
                  driver_home_area_type = col_integer()))

#remove irrelevant columns
dataset_vehicle <- dataset_vehicle %>% select(-
c(towing_and_articulation,accident_year,vehicle_reference))

#replacing -1 values

cols_to_replace <- c("vehicle_type","vehicle_manoeuvre","vehicle_location_restricted_lane",
                   "junction_location","skidding_and_overturning","hit_object_in_carriageway",
                   "vehicle_leaving_carriageway","hit_object_off_carriageway",
                   "first_point_of_impact","vehicle_left_hand_drive","journey_purpose_of_driver",
                   "sex_of_driver","age_of_driver","age_band_of_driver","engine_capacity_cc",

"propulsion_code","generic_make_model","driver_imd_decile","driver_home_area_type")
```

```
for (col in cols_to_replace) {
  na_vals <- dataset_casualty[[col]] == -1
  if (sum(!is.na(na_vals) & na_vals) > 0) {
    col_mean <- median(dataset_casualty[[col]][!na_vals])
    dataset_casualty[[col]][na_vals] <- col_mean
  }
}

#merging data
data <- dataset_accident %>%
  left_join(dataset_casualty, by = "accident_index") %>%
  left_join(dataset_vehicle, by = "accident_index")
# Plot code 1
value_counts <- table(dataset_accident_before$second_road_number)
count_minus_one <- value_counts["-1"]
count_remaining <- sum(value_counts) - count_minus_one
pie(c(count_remaining, count_minus_one), labels = c("Values other than -1", "-1"), col = c("blue",
"red"), main = "Proportion of -1 Values in Second Road Number")
#Plot code 2
ggplot(dataset_casualty_before, aes(x = casualty_imd_decile)) +
  geom_histogram(binwidth = 1, fill = "grey", color = "black") +
  geom_vline(xintercept = -1, linetype = "dashed", color = "red") +
  labs(x = "(IMD) Index of Multiple Deprivation", y = "Frequency", title = "Impact of -1 Values") +
  theme_bw()
```

Analytical Plan Code

```
#Plot 1
data_new <- data

data_new$road_surface_conditions <- ifelse(data_new$road_surface_conditions == 1, "Dry",
                                    ifelse(data_new$road_surface_conditions == 2, "Wet or damp",
                                        ifelse(data_new$road_surface_conditions == 3, "Snow",
                                            ifelse(data_new$road_surface_conditions == 4, "Frost or Ice",
                                                ifelse(data_new$road_surface_conditions == 5, "Flood over
3cm. deep",
                                                    ifelse(data_new$road_surface_conditions == 6, "Oil or
diesel",
                                                        ifelse(data_new$road_surface_conditions == 7,
"Mud", "Unknown")))))))

category_counts <- table(data_new$road_surface_conditions)

# Define custom colors
custom_colors <- c("skyblue", "lightgreen", "lightpink", "orange", "yellow", "purple", "brown")

# Plot a pie chart with category labels on the side
pie(category_counts, labels = "", main = "Road Surface Conditions", col = custom_colors)
legend("right", legend = names(category_counts), fill = custom_colors, cex = 0.6)
#Plot 2
data_new$weather_conditions <- ifelse(data_new$weather_conditions == 1, "Fine no high winds",
                                ifelse(data_new$weather_conditions == 2, "Raining no high winds",
                                    ifelse(data_new$weather_conditions == 3, "Snowing no high winds",
                                        ifelse(data_new$weather_conditions == 4, "Fine + high winds",
                                            ifelse(data_new$weather_conditions == 5, "Raining + high
winds",
```

```
                                                    ifelse(data_new$weather_conditions == 6, "Snowing + high winds",
                                                           ifelse(data_new$weather_conditions == 7, "Fog or mist",
                                                                  ifelse(data_new$weather_conditions == 8, "Other","Unknown")))))))

category_counts <- table(data_new$weather_conditions)

# Plot a bar graph
barplot(category_counts, main = "Weather Conditions", xlab = "Conditions", ylab = "Frequency", col = "skyblue")
```

Results Code

```
selected_cols <- c("accident_year",
                   "road_type"  ,  "speed_limit","second_road_class", "second_road_number" ,
                   "light_conditions","weather_conditions","road_surface_conditions",
                   "special_conditions_at_site","carriageway_hazards", "urban_or_rural_area",
                   "trunk_road_flag" , "pedestrian_location", "vehicle_type",
                   "vehicle_manoeuvre" ,"skidding_and_overturning","age_of_driver", "sex_of_driver",
                   "age_of_vehicle","driver_imd_decile")

df_selected <- subset(data, select = selected_cols)

grouped_data <- df_selected %>%
  group_by(accident_year,
           road_type  , speed_limit, second_road_class,  second_road_number , light_conditions,
           weather_conditions, road_surface_conditions, special_conditions_at_site,
           carriageway_hazards, urban_or_rural_area, trunk_road_flag ,
          pedestrian_location, vehicle_type, vehicle_manoeuvre , skidding_and_overturning,
           age_of_driver, sex_of_driver, age_of_vehicle, driver_imd_decile) %>%
  summarise(number_of_accidents = n())

#sampling
# Define the sample size
sample_size <- 3500
# Calculate the sampling interval
sampling_interval <- ceiling(nrow(grouped_data) / sample_size)
# Randomly select a starting point
starting_point <- sample(1:sampling_interval, 1)
# Perform systematic sampling
sample_indices <- seq(starting_point, nrow(grouped_data), by = sampling_interval)
# Extract the sample from the dataset
sample_data <- grouped_data[sample_indices, ]
#Train and Test dadta
set.seed(123)  # Set seed for reproducibility
indices <- sample(1:nrow(sample_data), nrow(sample_data)*0.7)
train_data <- sample_data[indices, ]
test_data <- sample_data[-indices, ]

#Possion Reg
model <- glm(number_of_accidents ~ ., data = train_data, family = poisson)
# Perform stepwise selection using AIC
step_model <- stepAIC(model, direction = "both")
# Print the summary of the selected model
```

```
summary(step_model)
# Predict the values using the test data
predictions <- predict(step_model, newdata = test_data, type = "response")
#Plot
old.par = par(mfrow = c(2, 2))
plot(model)
par(old.par)
#second task plot
# Calculate the count of accidents for each year
  accident_counts <- data %>%
  group_by(accident_year) %>%
  summarise(count = n())
# Create the line plot
ggplot(accident_counts, aes(x = accident_year, y = count, group = 1)) +
  geom_line() +
  geom_point() +
  labs(x = "Accident Year", y = "Count of Accidents", title = "Line Plot of Accident Year vs Count of
Accidents")
#task 3
library(mapview)
library(sp)

sp_points <- SpatialPointsDataFrame(coords = locations[, c("longitude", "latitude")],
                                    data = locations,
                                    proj4string = CRS("+proj=longlat +datum=WGS84"))

# Plot the points using mapview
mapview(sp_points)
```

EOF