COMP5721M: Programming for Data Science

Coursework 3: Data Analysis Project

A 5-year Analysis of UK Traffic Accidents from 2017-2021

Group 63 Members:

- Callum Scoby, gy19cs@leeds.ac.uk
- Fraser McKnespiey, gy22fgm@leeds.ac.uk
- Leon McGrail, gy22lpm@leeds.ac.uk
- Abdelrahman Ibrahim, gy22amai@leeds.ac.uk

1. Project Plan

1.1 The Data

Data was sourced from the UK Government Department of Transport. All data are police recorded traffic accidents.

Primary datasets

a. Road accidents from 2017 to 2021:

Each row in this dataset corresponds to one accident recorded during the time period. Each accident has a number of attributes including an accident index (a unique reference), day of the week, accident severity, number of vehicles and casualties, speed limit, local authority ONS district code, and driving conditions. Presented below are the data types of each column in this dataset.

Variable	Type	Notes
accident_index	Nominal	Unique identifier
accident_year	Discrete	
longitude	Continuous	
latitude	Continuous	
accident_severity	Ordinal	Ranking accident severity from 3 (slight) to 1 (fatal)
number_of_vehicles	Discrete	
number_of_casualties	Discrete	
date	Interval	
day_of_week	Nominal	A nominal representation of the days of week in values from 1 to 7
time	Continuous	
local_authority_ons_district	Nominal	Codes for UK districts
road_type	Nominal	
speed_limit	Ordinal	Speed limits of the road where the accident occured
light_conditions	Nominal	Categories of different lighting conditions in the time of the accident
weather_conditions	Nominal	Categories of different weather in the time of the accident
road_surface_conditions	Nominal	Categories of road surface condition in the time of the accident
urban_or_rural_area	Nominal	Categories for the type of the area (urban, rural, unidentified)

b. Vehicles involved in road accidents from 2017 to 2021

Each row in this dataset corresponds to one vehicle involved in a recorded accident. Thus,

a single accident may be represented by multiple rows. Each accident has attributes including accident index (suitable for joining with the accidents dataset), driver age bands, and driver sex. Presented below are the data types of each column in this dataset.

Variable	Type	Notes
accident_index	Nominal	Unique identifier and the common variable with the accident dataset
vehicle_type	Discrete	
sex_of_driver	Nominal	
age_of_driver	Discrete	
age_band_of_driver	Ordinal	Age bands of drivers from 1 (0 to 5 years) to 11 (75 plus years). Note that a driver can be younger than 5 because a bicycle is considered a vehicle
age_of_vehicle	Discrete	

Supporting Datasets

The Road Safety Data Guide published by the Department of Transport acts as the data dictionary for this analysis. It is also used to convert each Local Authority ONS District code into their respective name.

Data Accuracy

Accidents recorded in these datasets indicate a road accident where the police were notified, and at least one person was injured, or one vehicle was damaged. Thus, the data reflects only those accidents that were reported to or attended by police.

A small proportion of the data are missing values- in the case of latitude and longitude this proportion was around 1%. On the whole, however, the dataset is comprehensive and detailed. Sourcing from official Government publications further enhances confidence in the data accuracy.

1.2 Project Aim and Objectives (5 marks)

Our project aims to develop an understanding of how the occurrence and severity of traffic accidents in the UK are related to driving and demographic conditions through the analysis of UK Government data sourced over a 5-year period (2017-2021).

Our analysis seeks to examine how the occurrence of accidents varies spatially and temporally, and as a result of driving and demographic conditions. It also aims to understand how the odds of being in an accident of greater severity are associated with driving and demographic conditions.

To do so, we aim to initially visualise the spatial distribution of accident occurrences across the UK. We seek to do this in an interactive format that makes intuitive sense to a viewer. This will aid in the understanding of accident hotspots within the UK, while also providing locational context to the data. In tandem, we aim to understand the temporal

patterns of accident occurrences in the dataset. This may involve identifying the rate of change of accidents across the 5-year period and will add temporal context to the data.

Having gained an understanding of the distribution and pattern of accident occurrences, we seek to visualise trends in accident occurrences against contextual driving and demographic factors. Next, we aim to identify how factors affect the odds of being in a crash of greater severity. To do so, we aim to first statistically assess how accident severity odds vary as a function of contextual driving conditions such as light, weather, road conditions, speed limit, and road class. Finally, we seek to statistically assess how accident severity odds vary as a result of contextual demographic factors, including driver age and sex.

Specific Objective(s)

The four key objectives of this project are as follows:

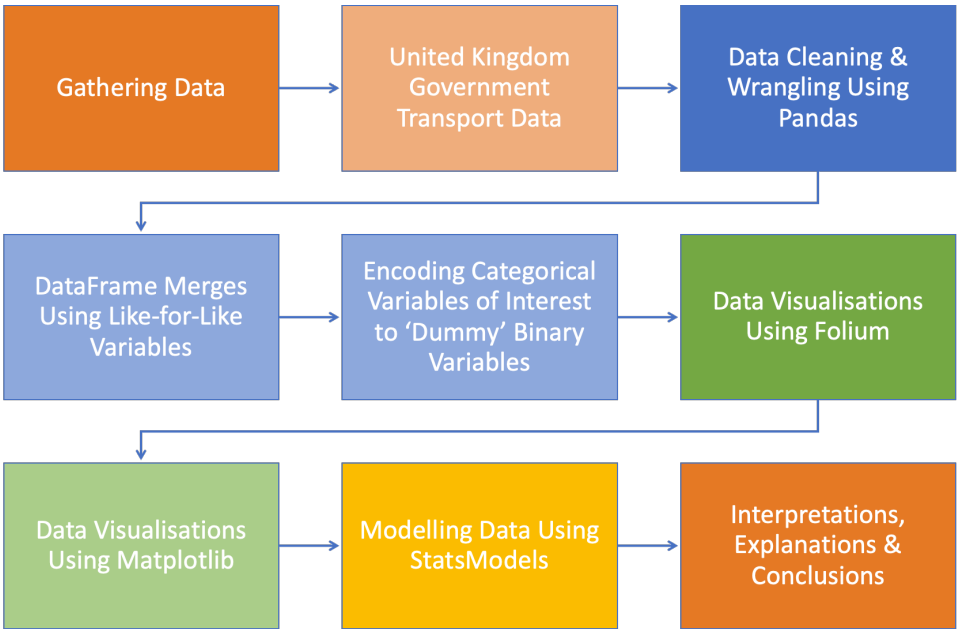
- **Objective 1:** *Exploring the spatial and temporal distribution of accidents within the UK using the Folium library*
- **Objective 2:** *Using the Matplotlib package, visualise how accident occurrences vary with driving and demographic conditions*
- **Objective 3:** *Statistically assess, using the statsmodels package, how driving conditions affect the odds of being in an accident of greater severity*
- **Objective 4:** *Statistically assess, using the statsmodels package, how demographic conditions affect the odds of being in an accident of greater severity*

1.3 System Design (5 marks)

Architecture

The following illustration highlights the pipeline taken during this Data Analysis Project.

1. Gathering Data
    - UK Government Transport Data:
        - After finding our data from the UK Government, the next stage was for cleaning and wrangling to prepare the data for analysis.
2. Data Cleaning & Wrangling Using Pandas
    - DataFrame Merges Using Like-for-Like Variables:
        - Extract demographic and vehicle data from the vehicles dataset and assign datapoints to their respective accident.
    - Encoding Categorical Variables of Interest to 'Dummy' Binary Variables:
        - We convert catagorical data (nominal and ordinal) to binary columns using numpy and list comprehension to prepare them for regression analysis
3. Data Visualisations
    - Spatial and Temporal Visualisations using Folium:
        - During the visualisation phase we aim to explore spatial and temporal patterns to find insightful information about the data.
    - Plotting visualisations using Matplotlib:
        - Using novel methods of Matplotlib to generate appropriate visualisations to further explore data.
    - Statistically assessing how driving conditons/demographic conditions affect accident severity using StatsModels:
        - Using Ordinal Linear Regression to assess how driving conditions and demographic factors affect the odds of being in an accident of greater severity
4. Interpretations, Explinations & Conclusions:
    - Finally, we call the functions to display outputs, and derive real world conclusions on the spatial and statistical facts about road accidents in the targeted 5 years.



Processing Modules and Algorithms

- **Data cleansing**
  - Once imported, each data frame (accidents, vehicles, district names) was subset to include only the variables of interest.
  - Missing data (represented by '-1' in the original data frames) was not removed because it had no bearing on our results. This allowed us to keep a comprehensive number of data points.
  - Accident severity levels (1 = fatal, 2 = serious, 3 = slight) were reversed (1 = slight, 2 = serious, 3 = fatal) for intuitive interpretation.
- **Merging data frames**
  - The cleansed data frames were merged to create a singular and uniform master data frame.
  - A column of 'date_time' was created using the datetime package to enable date and time manipulation in temporal analysis.
- **Converting variables to special representations**
  - The accident severity variable is an ordinal categorical variable (slight, serious, fatal) and so it was converted to a categorical and ordered data type, with the variable name 'accident_severity_OLR'
  - Dummy variables were created for all categorical variables of interest, producing binary columns for each value within a categorical variable. For example, the 'light conditions' variable contained values of '1', '4', '5', '6', '7', and '-1', with each representing a different light condition. Thus, each value was converted to its own binary variable.
- **Interactive spatial distribution maps were created using the Folium library**
  - Spatial distribution observed through using latitude and longitude for each accident which are then visualised using clusters and heat mapping.
- **Ordinal Logistic Regression (OLR) using statsmodels package**
  - OLR was undertaken to assess the statistical odds of being in an accident of greater severity as a result of the different binary variables. Odds ratios were created by calculating the inverse exponent of the regression coefficients.

2. Program Code

2.1 Data aquisition, cleaning and wrangling

In [1]:
```
import pandas as pd
import numpy as np
import datetime as dt
```

In [2]:
```
# Importing the two road safety datasets
accident_original = pd.read_csv('https://data.dft.gov.uk/road-accidents-safety-d
vehicle_original = pd.read_csv('https://data.dft.gov.uk/road-accidents-safety-da
district_name = pd.DataFrame(pd.read_excel('https://data.dft.gov.uk/road-acciden
```

In [3]:
```
# Subsetting the accidents_original df to the columns we want
accident_clean = accident_original[['accident_index', 'accident_year',
                                    'location_easting_osgr', 'location_northing_osgr'
                                     'longitude', 'latitude', 'accident_severity',
                                           'number_of_vehicles', 'number_of_casualtie
                                           'date', 'day_of_week', 'time',
                                           'local_authority_ons_district', 'first_roa
                                           'road_type', 'speed_limit', 'light_conditi
                                           'weather_conditions', 'road_surface_condit
                                           'urban_or_rural_area']]

# Subsetting the vehicle_original df to the columns we want
vehicle_clean = vehicle_original[['accident_index', 'vehicle_type',
                                   'sex_of_driver', 'age_of_driver',
                                   'age_band_of_driver', 'engine_capacity_cc',
                                   'driver_imd_decile', 'age_of_vehicle']]

# Subsetting the district_name df to the rows and columns we want
district_name = district_name.loc[district_name['field name'] == 'local_authorit
district_name = district_name.drop(['table', 'field name', 'note'], axis=1)
```

In [4]:
```
district_name = district_name.drop_duplicates()
```

In [5]:
```
#rename columns in district_name df to join with accident_clean df
district_name.rename(columns={'code/format': 'local_authority_ons_district', 'la
```

In [6]:
```
driver_age_bands = vehicle_clean.groupby('accident_index')['age_band_of_driver']

#produce a series with a set of all driver age bands involved in each accident
```

In [7]:
```
driver_age_bands.to_frame()
#convert the result to a new dataframe
```

Out[7]:

accident_index	age_band_of_driver
2017010001708	[5, 4]
2017010009342	[6, 7]
2017010009344	[-1, 6, 6]
2017010009348	[7, 6]
2017010009350	[11]
...	...
2021991196247	[6, 9]
2021991196607	[7, 9]
2021991197944	[8]
2021991200639	[5]
2021991201030	[6, 9]

562439 rows × 1 columns

In [8]:
```
accident_master = pd.merge(accident_clean, driver_age_bands, on="accident_index"
#merge age bands of drivers with accindents dataframe
```

In [9]:
```
#we repeat the process for sex of drivers
driver_sex = vehicle_clean.groupby('accident_index')['sex_of_driver'].apply(list
```

In [10]:
```
driver_sex.to_frame()
```

Out[10]:

accident_index	sex_of_driver
2017010001708	[1, 1]
2017010009342	[1, 1]
2017010009344	[3, 1, 2]
2017010009348	[2, 2]
2017010009350	[1]
...	...
2021991196247	[1, 1]
2021991196607	[1, 1]
2021991197944	[2]
2021991200639	[2]
2021991201030	[1, 1]

562439 rows × 1 columns

In [11]:
```
accident_master = pd.merge(accident_master, driver_sex, on="accident_index", how
#merge sex of drivers with accidents_drivers dataframe
```

In [12]:
```
age_0_5 = [1 if 1 in x else 0 for x in (accident_master['age_band_of_driver']) ]
accident_master["age_0_5"] = age_0_5
```

In [13]:
```
age_6_10 = [1 if 2 in x else 0 for x in (accident_master['age_band_of_driver'])]
accident_master["age_6_10"] = age_6_10
```

In [14]:
```
age_11_15 = [1 if 3 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_11_15"] = age_11_15
```

In [15]:
```
age_16_20 = [1 if 4 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_16_20"] = age_16_20
```

In [16]:
```
age_21_25 = [1 if 5 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_21_25"] = age_21_25
```

In [17]:
```
age_26_35 = [1 if 6 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_26_35"] = age_26_35
```

In [18]:
```
age_36_45 = [1 if 7 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_36_45"] = age_36_45
```

In [19]:
```
age_46_55 = [1 if 8 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_46_55"] = age_46_55
```

In [20]:
```
age_56_65 = [1 if 9 in x else 0 for x in (accident_master['age_band_of_driver'])
accident_master["age_56_65"] = age_56_65
```

In [21]:
```
age_66_75 = [1 if 10 in x else 0 for x in (accident_master['age_band_of_driver']
accident_master["age_66_75"] = age_66_75
```

In [22]:
```
age_75plus = [1 if 11 in x else 0 for x in (accident_master['age_band_of_driver'
accident_master["age_75plus"] = age_75plus
```

In [23]:
```
age_nodata = [1 if -1 in x else 0 for x in (accident_master['age_band_of_driver'
accident_master["age_nodata"] = age_nodata
```

In [24]:
```
sex_male = [1 if 1 in x else 0 for x in (accident_master['sex_of_driver'])]
accident_master["sex_male"] = sex_male
```

In [25]:
```
sex_female = [1 if 2 in x else 0 for x in (accident_master['sex_of_driver'])]
accident_master["sex_female"] = sex_female
```

In [26]:
```
sex_NA = [1 if 3 in x else 0 for x in (accident_master['sex_of_driver'])]
accident_master["sex_NA"] = sex_NA
```

In [27]:
```
sex_nodata = [1 if -1 in x else 0 for x in (accident_master['sex_of_driver'])]
accident_master["sex_nodata"] = sex_nodata
```

In [28]:
```
accident_master = accident_master.drop(['age_band_of_driver', 'sex_of_driver'],
```

In [29]:
```
accident_master = accident_master.replace({'accident_severity': 1}, 5)
accident_master = accident_master.replace({'accident_severity': 3}, 1)
accident_master = accident_master.replace({'accident_severity': 5}, 3)
```

In [30]:
```
accident_master['Monday'] = np.where((accident_master['day_of_week'] == 2), 1, 0
accident_master['Tuesday'] = np.where((accident_master['day_of_week'] == 3), 1,
accident_master['Wednesday'] = np.where((accident_master['day_of_week'] == 4), 1
accident_master['Thursday'] = np.where((accident_master['day_of_week'] == 5), 1,
accident_master['Friday'] = np.where((accident_master['day_of_week'] == 6), 1, 0
accident_master['Saturday'] = np.where((accident_master['day_of_week'] == 7), 1,
accident_master['Sunday'] = np.where((accident_master['day_of_week'] == 1), 1, 0
```

In [31]:
```
accident_master['Daylight'] = np.where((accident_master['light_conditions'] == 1
accident_master['Darkness_lights_lit'] = np.where((accident_master['light_condit
accident_master['Darkness_lights_unlit'] = np.where((accident_master['light_cond
accident_master['Darkness_no_lighting'] = np.where((accident_master['light_condi
accident_master['Darkness_lighting_unknown'] = np.where((accident_master['light_
```

In [32]:
```
accident_master['Fine_no_high_winds'] = np.where((accident_master['weather_condi
accident_master['Raining_no_high_winds'] = np.where((accident_master['weather_co
accident_master['Snowing_no_high_winds'] = np.where((accident_master['weather_co
accident_master['Fine_and_high_winds'] = np.where((accident_master['weather_cond
accident_master['Raining_and_high_winds'] = np.where((accident_master['weather_c
accident_master['Snowing_and_high_winds'] = np.where((accident_master['weather_c
accident_master['Fog_or_mist'] = np.where((accident_master['weather_conditions']
accident_master['Other'] = np.where((accident_master['weather_conditions'] == 8)
accident_master['Unknown'] = np.where((accident_master['weather_conditions'] ==
```

In [33]:
```
accident_master['Dry_road'] = np.where((accident_master['road_surface_conditions
accident_master['Wet_or_damp_road'] = np.where((accident_master['road_surface_co
accident_master['Snow_road'] = np.where((accident_master['road_surface_condition
accident_master['Frost_or_ice_road'] = np.where((accident_master['road_surface_c
accident_master['Flood_over_3cm_deep_road'] = np.where((accident_master['road_su
accident_master['Oil_or_diesel_road'] = np.where((accident_master['road_surface_
accident_master['Mud_road'] = np.where((accident_master['road_surface_conditions
accident_master['Unknown'] = np.where((accident_master['road_surface_conditions'
```

In [34]:
```
accident_master['Motorway_road_class'] = np.where((accident_master['first_road_c
accident_master['A_M_road_class'] = np.where((accident_master['first_road_class'
accident_master['A_road_road_class'] = np.where((accident_master['first_road_cla
accident_master['B_road_class'] = np.where((accident_master['first_road_class']
accident_master['C_road_class'] = np.where((accident_master['first_road_class']
accident_master['Unclassified_road_class'] = np.where((accident_master['first_ro
```

In [35]:
```
accident_master['20_mph'] = np.where((accident_master['speed_limit'] == 20), 1,
accident_master['30_mph'] = np.where((accident_master['speed_limit'] == 30), 1,
accident_master['40_mph'] = np.where((accident_master['speed_limit'] == 40), 1,
accident_master['50_mph'] = np.where((accident_master['speed_limit'] == 50), 1,
accident_master['60_mph'] = np.where((accident_master['speed_limit'] == 60), 1,
accident_master['70_mph'] = np.where((accident_master['speed_limit'] == 70), 1,
```

In [36]:
```
accident_master['Urban_area'] = np.where((accident_master['urban_or_rural_area']
accident_master['Rural_area'] = np.where((accident_master['urban_or_rural_area']
accident_master['Unallocated_area'] = np.where((accident_master['urban_or_rural_
```

In [37]:
```
#join district_name df with accident_clean df
accident_master = pd.merge(accident_master, district_name, 'left', on="local_aut
```

The next step is to create a new column called `date_time` which combines the `date` and `time` columns by using the Pandas function `pd.to_datetime()`. By doing this, we are able to easily computate temporal information when required

In [38]:
```
accident_master['date_time'] = accident_master['date'] +' '+ accident_master['ti
accident_master['date_time'] = pd.to_datetime(accident_master.date_time)
```

In [39]:
```
# Check the data
accident_master.head()
```

Out[39]:

	accident_index	accident_year	location_easting_osgr	location_northing_osgr	longitude	lat
0	2017010001708	2017	532920.0	196330.0	-0.080107	51.65
1	2017010009342	2017	526790.0	181970.0	-0.173845	51.52
2	2017010009344	2017	535200.0	181260.0	-0.052969	51.51
3	2017010009348	2017	534340.0	193560.0	-0.060658	51.62
4	2017010009350	2017	533680.0	187820.0	-0.072372	51.57

5 rows × 81 columns

In [40]:
```
accident_master.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 562439 entries, 0 to 562438
Data columns (total 81 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   accident_index                  562439 non-null  object
 1   accident_year                   562439 non-null  int64
 2   location_easting_osgr           562306 non-null  float64
 3   location_northing_osgr          562306 non-null  float64
 4   longitude                       562296 non-null  float64
 5   latitude                        562296 non-null  float64
 6   accident_severity               562439 non-null  int64
 7   number_of_vehicles              562439 non-null  int64
 8   number_of_casualties            562439 non-null  int64
 9   date                            562439 non-null  object
 10  day_of_week                     562439 non-null  int64
 11  time                            562439 non-null  object
 12  local_authority_ons_district    562439 non-null  object
 13  first_road_class                562439 non-null  int64
 14  road_type                       562439 non-null  int64
 15  speed_limit                     562439 non-null  int64
 16  light_conditions                562439 non-null  int64
 17  weather_conditions              562439 non-null  int64
 18  road_surface_conditions         562439 non-null  int64
 19  urban_or_rural_area             562439 non-null  int64
 20  age_0_5                         562439 non-null  int64
 21  age_6_10                        562439 non-null  int64
 22  age_11_15                       562439 non-null  int64
 23  age_16_20                       562439 non-null  int64
 24  age_21_25                       562439 non-null  int64
 25  age_26_35                       562439 non-null  int64
 26  age_36_45                       562439 non-null  int64
 27  age_46_55                       562439 non-null  int64
 28  age_56_65                       562439 non-null  int64
 29  age_66_75                       562439 non-null  int64
 30  age_75plus                      562439 non-null  int64
 31  age_nodata                      562439 non-null  int64
 32  sex_male                        562439 non-null  int64
 33  sex_female                      562439 non-null  int64
 34  sex_NA                          562439 non-null  int64
 35  sex_nodata                      562439 non-null  int64
 36  Monday                          562439 non-null  int32
 37  Tuesday                         562439 non-null  int32
 38  Wednesday                       562439 non-null  int32
 39  Thursday                        562439 non-null  int32
 40  Friday                          562439 non-null  int32
 41  Saturday                        562439 non-null  int32
 42  Sunday                          562439 non-null  int32
 43  Daylight                        562439 non-null  int32
 44  Darkness_lights_lit             562439 non-null  int32
 45  Darkness_lights_unlit           562439 non-null  int32
 46  Darkness_no_lighting            562439 non-null  int32
 47  Darkness_lighting_unknown       562439 non-null  int32
 48  Fine_no_high_winds              562439 non-null  int32
 49  Raining_no_high_winds           562439 non-null  int32
 50  Snowing_no_high_winds           562439 non-null  int32
 51  Fine_and_high_winds             562439 non-null  int32
 52  Raining_and_high_winds          562439 non-null  int32
 53  Snowing_and_high_winds          562439 non-null  int32
 54  Fog_or_mist                     562439 non-null  int32
```

```
 55  Other                          562439 non-null  int32
 56  Unknown                        562439 non-null  int32
 57  Dry_road                       562439 non-null  int32
 58  Wet_or_damp_road               562439 non-null  int32
 59  Snow_road                      562439 non-null  int32
 60  Frost_or_ice_road              562439 non-null  int32
 61  Flood_over_3cm_deep_road       562439 non-null  int32
 62  Oil_or_diesel_road             562439 non-null  int32
 63  Mud_road                       562439 non-null  int32
 64  Motorway_road_class            562439 non-null  int32
 65  A_M_road_class                 562439 non-null  int32
 66  A_road_road_class              562439 non-null  int32
 67  B_road_class                   562439 non-null  int32
 68  C_road_class                   562439 non-null  int32
 69  Unclassified_road_class        562439 non-null  int32
 70  20_mph                         562439 non-null  int32
 71  30_mph                         562439 non-null  int32
 72  40_mph                         562439 non-null  int32
 73  50_mph                         562439 non-null  int32
 74  60_mph                         562439 non-null  int32
 75  70_mph                         562439 non-null  int32
 76  Urban_area                     562439 non-null  int32
 77  Rural_area                     562439 non-null  int32
 78  Unallocated_area               562439 non-null  int32
 79  district_name                  562439 non-null  object
 80  date_time                      562439 non-null  datetime64[ns]
dtypes: datetime64[ns](1), float64(4), int32(43), int64(28), object(5)
memory usage: 259.6+ MB
```

2.2 Statistical modelling and visualization

In [41]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as stats
from statsmodels.miscmodels.ordinal_model import OrderedModel
import calendar
from ipyleaflet import Map, basemaps, basemap_to_tiles, Circle, Polyline
import folium as fl
from folium.plugins import FastMarkerCluster
from folium.plugins import HeatMapWithTime
import geopandas as gpd
```

In [42]:
```
# Converting accident_severity to an ordered categorical type variable, to ensur
# Ordinal Logistic Regression analysis
accident_master['accident_severity_OLR'] = pd.Categorical(accident_master['accid
accident_master['accident_severity_OLR'].dtype
```

Out[42]:
```
CategoricalDtype(categories=[1, 2, 3], ordered=True)
```

A note on Ordinal Logistic Regression

To assess the influence of variables upon accident severity, Ordinal Logistic Regression (OLR) was used. OLR takes categorical and continuous predictor variables and models the log odds outcomes of an ordinal categorical response variable. This technique was chosen because accident severity is an ordinal variable (slight, serious, fatal), and so must be analysed using an appropriate method.

OLR makes a number of **assumptions**:

1. The response variable is ordered.

Accident severity is an ordered variable, and was defined as such in *2.1 Data aquisition, cleaning and wrangling*

In [43]:
```
accident_master['accident_severity_OLR'].dtype
```

Out[43]:
```
CategoricalDtype(categories=[1, 2, 3], ordered=True)
```

2. One or more predictor variables are either continuous, categorical or ordinal- all variables of interest (driver age band, driver sex, day of week, light conditions, weather conditions, road conditions, road class, urban or rural area) are categorical in the original datasets, and were converted to dummy variables in the *2.1 Data aquisition, cleaning and wrangling*.

3. No multicollinearity is present- this will be assessed in turn for each analysis.

4. Proportional odds- the assumption that the effects of predictor variables are consistent or proportional across the thresholds of the ordinal response variable categories (in this case the threshold between a slight or serious accident, or between a serious and fatal accident).

OrderedModel is the statsmodel package that will be used to conduct OLR. OrderedModel is a relatively new addition to the statsmodels package and, as of yet, there are no official methods to validate Assumption 4. As such, we cannot be entirely sure that the assumption is satisfied. However, if this assumption is violated, it is likely to have little bearing on the accuracy of model results.

Where are the UK accident hotspots?

In [44]:
```
def accident_hotspots():
    print(accident_master['district_name'].value_counts().head())
```

In [45]:
```
accident_hotspots()
```

```
Birmingham       11900
Leeds             7110
Westminster       6898
Lambeth           6063
Tower Hamlets     5394
Name: district_name, dtype: int64
```

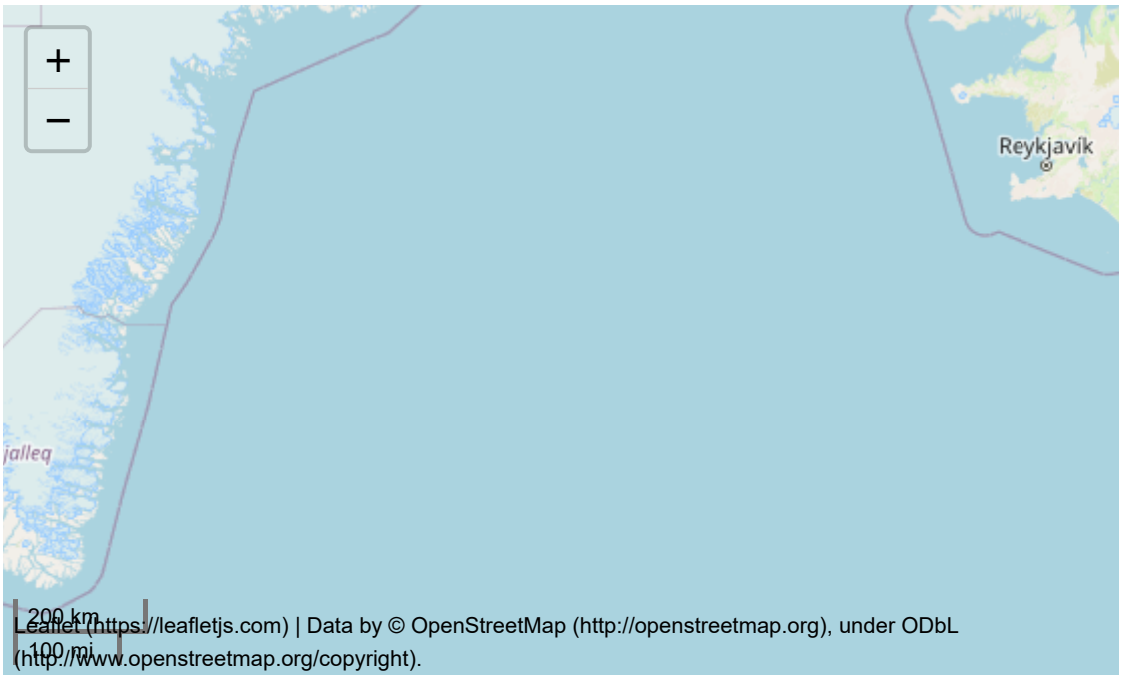
Geospatial distribution of accidents

In [46]:
```
accident_location = accident_master[['latitude', 'longitude', 'accident_severity
accident_location = accident_location.dropna(subset=['latitude','longitude'])
```

In [47]:
```
def accidents_interactive_map():
    map1 = fl.Map(location=[accident_location.latitude.mean(),\
                            accident_location.longitude.mean()],\
                  zoom_start=4.5, control_scale=True)
    map1.add_child(FastMarkerCluster(accident_location[['latitude', 'longitude']
    return map1
```

In [48]:
```
accidents_interactive_map() #Please check the interactive map in the HTML File
```

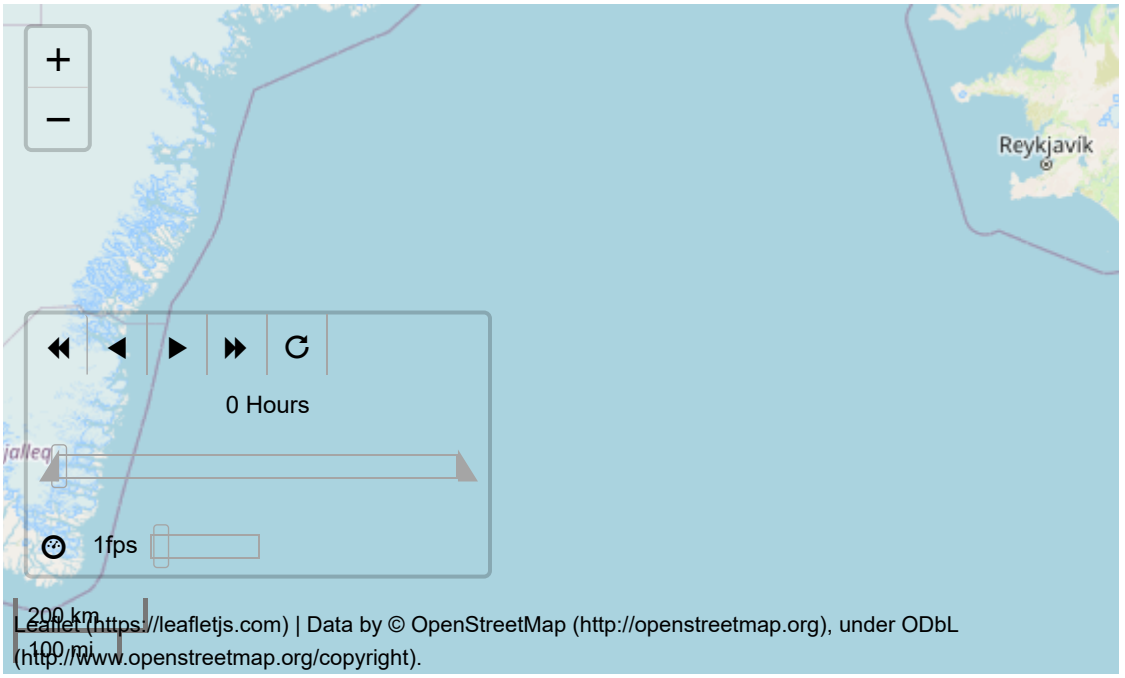
Out[48]:



In [49]:
```
def heat_interactive_map():
    map2 = fl.Map(location=[accident_location.latitude.mean(),\
                            accident_location.longitude.mean()],\
                  zoom_start=4.5, control_scale=True)
    hr_list = [[] for x in range(24)]
    for latitude,longitude,hour in zip(accident_location.latitude, accident_loca
        hr_list[hour].append([latitude, longitude])

    index = [str(i) + ' Hours' for i in range (24)]

    HeatMapWithTime(hr_list, index).add_to(map2)

    return map2
```

In [50]:
```
heat_interactive_map() #Please check the interactive map in the HTML File
```

Out[50]:



The odds of being in a greater severity accident by gender

Checking that the multicollinearity assumption is met

In [51]:
```
df_gender_multicollinearity_test = accident_master.iloc[:,32:34]
df_gender_multicollinearity_test_print = df_gender_multicollinearity_test.corr()
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_gender_multicollinearity_test_print)
```

	sex_male	sex_female
sex_male	1.000000	-0.400045
sex_female	-0.400045	1.000000

As the categories of male drivers and female drivers involved in accidents are effectively measuring the same thing (the proportion of drivers of each gender), complete multicollinearity is exhibited. As such, only 'sex_male' will be included in the OLR model. This means that 'sex_female' can be used as a benchmark to compare odds against.

OLR Model

In [52]:
```
def gender_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas

    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios (i.e. the proportional odds of being
    # severity (i.e. serious or fatal, rather than slight, as a result of the di
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

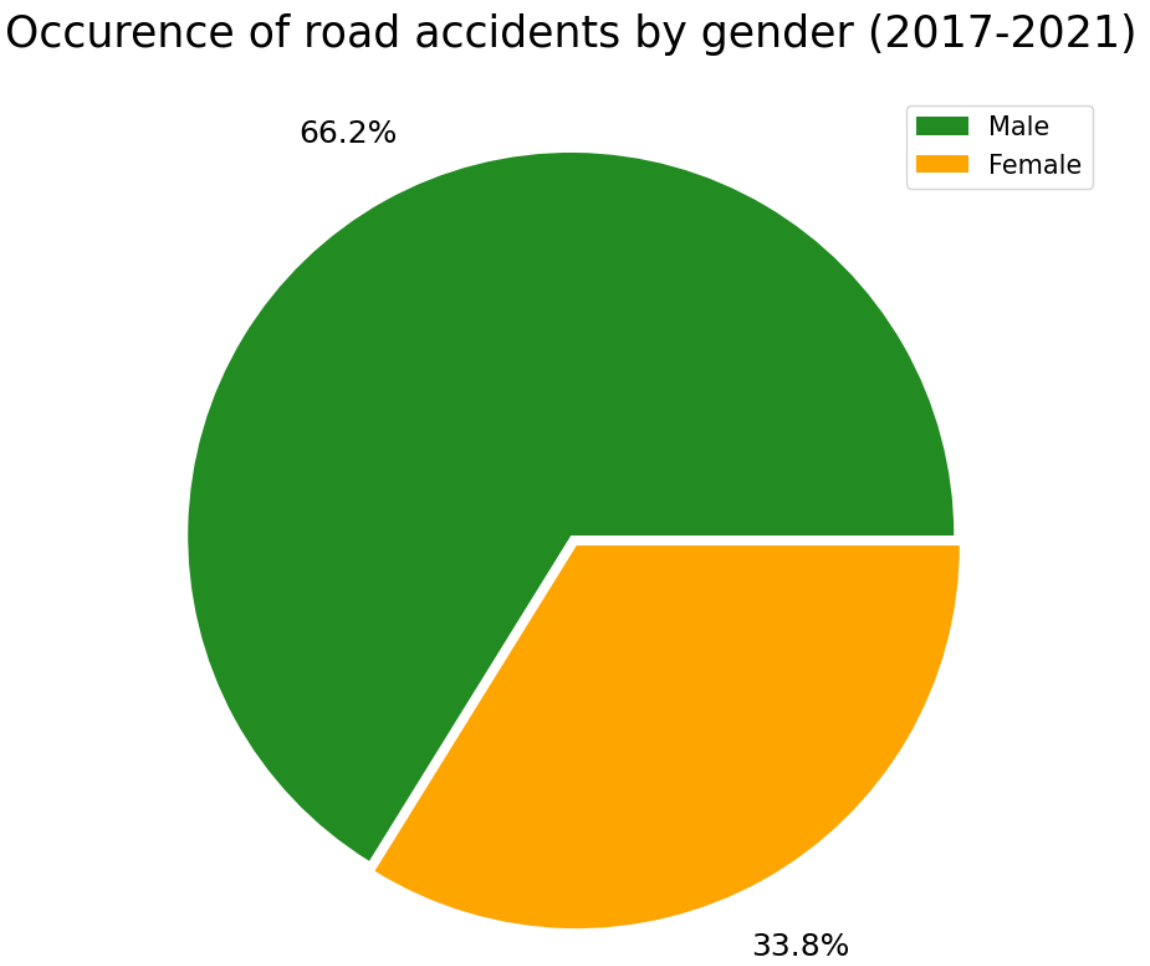
In [53]:
```
gender_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1415e+0
5
Model:                      OrderedModel   AIC:                         6.283e+0
5
Method:             Maximum Likelihood     BIC:                         6.283e+0
5
Date:                    Mon, 05 Dec 2022
Time:                            06:32:03
No. Observations:                562439
Df Residuals:                    562436
Df Model:                             3
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sex_male       0.3037      0.009     34.023      0.000       0.286       0.321
1/2            1.5998      0.008    195.877      0.000       1.584       1.616
2/3            1.0673      0.004    280.661      0.000       1.060       1.075
==============================================================================
                 OR  Lower CI  Upper CI
sex_male  1.354808  1.331315  1.378716
1/2       4.951825  4.873190  5.031728
2/3       2.907618  2.886026  2.929371
```

How does the occurence of road accidents vary with gender?

In [54]:
```
def gender_accident_occurences():
    df_accidents_by_gender = accident_master
    df_accidents_by_gender = df_accidents_by_gender.replace({'sex_female': 1}, 2
    gender = []
    gender.append((df_accidents_by_gender['sex_male'] == 1).sum())
    gender.append((df_accidents_by_gender['sex_female'] == 2).sum())
    gender_df = pd.DataFrame (gender, columns = ['gender'])

    # Series with number of mild injuries and serious injuries
    gender_pie_chart = gender_df['gender']

    # Pie plot with the percentage of victims with slight, serious and fatal inj
    explode = (0.00, 0.03)
    gender_pie_chart.plot(kind = 'pie',figsize = (10,10), explode = explode, col

    # Title and legend
    plt.legend(labels = ['Male', 'Female'],bbox_to_anchor = (0.85, 0.95), loc='u
    plt.title('Occurence of road accidents by gender (2017-2021)', fontsize = 25
    plt.ylabel('')
```

In [55]:
```
gender_accident_occurences()
```



Light conditions

The odds of being in a greater severity accident by light conditions

Checking that the multicollinearity assumption is met

In [56]:
```
df_lightcond_multicollinearity_test = accident_master.iloc[:,43:47]
df_lightcond_multicollinearity_test_print = df_lightcond_multicollinearity_test.
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_lightcond_multicollinearity_test_print)
```

	Daylight	Darkness_lights_lit	Darkness_lights_unlit	Darkness_no_lighting
Daylight	1.000000	-0.804548	-0.135613	-0.367540
Darkness_lights_lit	-0.804548	1.000000	-0.043276	-0.117287
Darkness_lights_unlit	-0.135613	-0.043276	1.000000	-0.019770
Darkness_no_lighting	-0.367540	-0.117287	-0.019770	1.000000

The 'daylight' category exhibits the highest levels of multicollinearity with the other variables. Correlation of -0.80 exists between daylight and darkness (lights lit), and -0.37 between daylight and darkness (no lighting). Additionally, daylight light conditions comprise a majority of the occurences of accidents. As such, daylight is not included in the OLR model, meaning it can be used as the reference variable.

Here, **Ordinal Logistic Regression** is used to understand whether light conditions affect the odds of being in an accident of greater severity (i.e. serious or fatal, rather than slight). This regression technique was chosen because accident severity is an ordinal variable (slight, serious, fatal), and so should not be analysed using linear regression techniques.

Daylight conditions were chosen as the reference variable, as they provide an intuitive benchmark against which to compare, and because they comprise the bulk of accidents. The dummy variables (variables that have been converted from a scale in a single column (e.g. 1-5 in this example) to multiple binary columns): 1) Darkness_lights_lit, 2) Darkness_lights_unlit, 3) Darkness_no_lighting are used, while Darkness_lighting_unknown is removed from the analysis as it does not provide useful information.

A **logit** ordinal regression method was employed rather than a probit method, because this allows for more intuitive analysis, and because the choice is negligible for a large sample size.

OLR Model

In [57]:
```
def light_conditions_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                        'Darkness_no_lighting']],
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

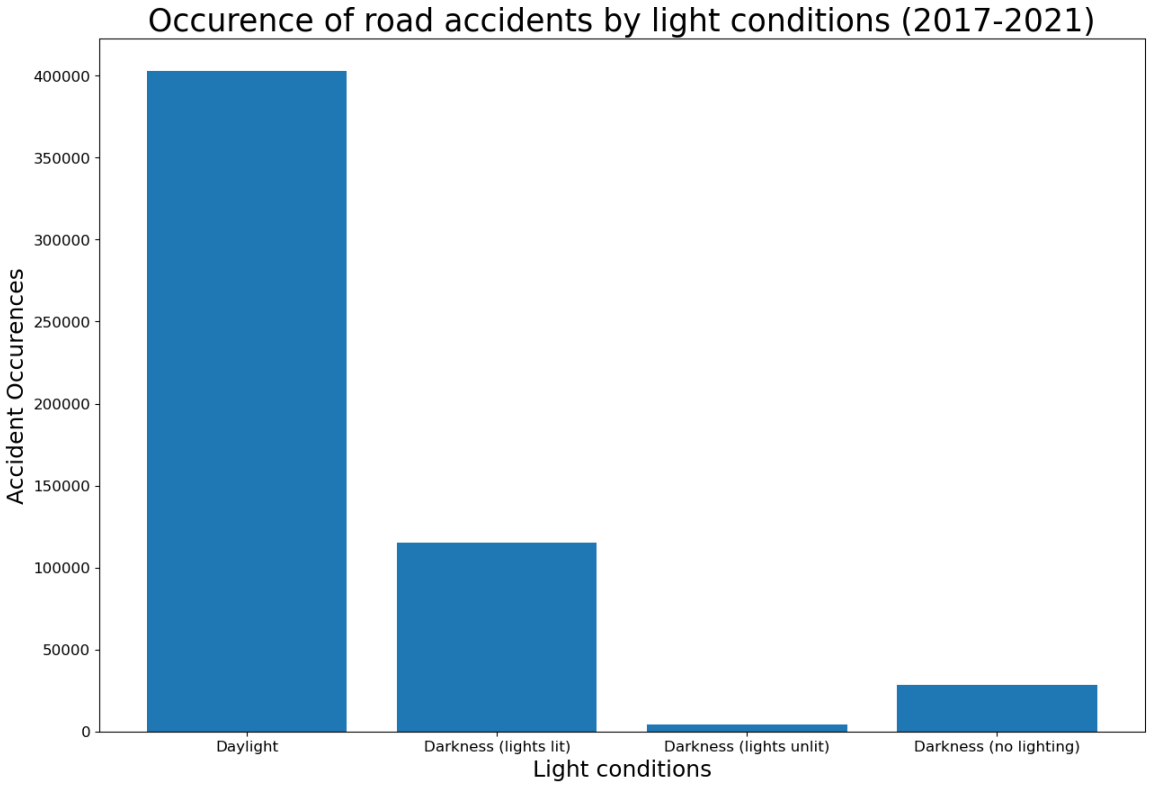
In [58]:
```
light_conditions_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1355e+0
Model:                      OrderedModel   AIC:                         6.271e+0
Method:             Maximum Likelihood     BIC:                         6.272e+0
Date:                   Mon, 05 Dec 2022
Time:                           06:32:59
No. Observations:                562439
Df Residuals:                    562434
Df Model:                             5
==============================================================================
                             coef    std err          z      P>|z|      [0.025
------------------------------------------------------------------------------
Darkness_lights_lit       0.1262      0.008     15.474      0.000       0.110
Darkness_lights_unlit     0.2645      0.037      7.151      0.000       0.192
Darkness_no_lighting      0.6648      0.013     49.803      0.000       0.639
1/2                       1.4168      0.004    362.126      0.000       1.409
2/3                       1.0686      0.004    281.249      0.000       1.061
==============================================================================
                              OR  Lower CI  Upper CI
Darkness_lights_lit     1.134477  1.116490  1.152753
Darkness_lights_unlit   1.302770  1.211677  1.400712
Darkness_no_lighting    1.944054  1.893854  1.995584
1/2                     4.123923  4.092421  4.155668
2/3                     2.911402  2.889801  2.933165
```

How does the occurence of road accidents vary with light conditions?

In [59]:
```
def light_conditions_accident_occurences():
    a = len(accident_master[accident_master.Daylight == 1])
    b = len(accident_master[accident_master.Darkness_lights_lit == 1])
    c = len(accident_master[accident_master.Darkness_lights_unlit == 1])
    d = len(accident_master[accident_master.Darkness_no_lighting == 1])


    light_conditions_df = pd.DataFrame({'light_conditions': ['Daylight', 'Darkne
                                        'Darkness_no_lighting'],\
                        'accident_count': [a, b, c, d]})

    x = light_conditions_df.light_conditions
    y = light_conditions_df.accident_count
    fig,ax = plt.subplots(figsize=(15, 10))
    plt.bar(x, y)
    labels = ['Daylight', 'Darkness (lights lit)','Darkness (lights unlit)', 'Da
    plt.xticks(x, labels, fontsize=12)
    plt.yticks(fontsize=12)
    plt.title('Occurence of road accidents by light conditions (2017-2021)', fon
    plt.xlabel('Light conditions', fontsize=18)
    plt.ylabel('Accident Occurences', fontsize=18)

    plt.show()
```

In [60]:
```
light_conditions_accident_occurences()
```



The odds of being in a greater severity accident by weather conditions

Checking that the multicollinearity assumption is met

In [61]:
```
df_weathercond_multicollinearity_test = accident_master.iloc[:,48:54]
df_weathercond_multicollinearity_test_print = df_weathercond_multicollinearity_t
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_weathercond_multicollinearity_test_print)
```

	Fine_no_high_winds	Raining_no_high_winds	Snowing_no_high_winds
Fine_no_high_winds	1.000000	-0.713908	-0.141944
Raining_no_high_winds	-0.713908	1.000000	-0.025706
Snowing_no_high_winds	-0.141944	-0.025706	1.000000
Fine_and_high_winds	-0.204097	-0.036961	-0.007349
Raining_and_high_winds	-0.214822	-0.038904	-0.007735
Snowing_and_high_winds	-0.070339	-0.012738	-0.002533

Fine (no high winds) weather conditions exhibit the greatest levels of multicollinearity with the other categories. Correlation of -0.71 exists between fine (no high winds) and raining (no high winds), and -0.20 against fine (with high winds). As such, the fine (no high winds) weather condition is not used in the OLR analysis, and instead acts as the reference variable.

Ordinal Logistic Regression is used to understand how weather conditions affect the odds of being in an accident of greater severity (i.e. serious or fatal, rather than slight).

Fine with no high winds conditions were chosen as the reference variable, as they provide an intuitive benchmark against which to compare, and because they comprise the bulk of accidents. Dummy variables: 1) Raining_no_high_winds, 2) Snowing_no_high_winds, 3) Fine_and_high_winds, 4) Raining_and_high_winds, 5) Snowing_and_high_winds, 6) Fog_or_mist are used.

OLR Model

In [62]:
```
def weather_conditions_severity_logistic_ordinal_regression():
# Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                'Fine_and_high_winds', 'R
                                                'Snowing_and_high_winds',
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

In [63]:
```
weather_conditions_severity_logistic_ordinal_regression()
```

OrderedModel Results

Dep. Variable:	accident_severity_OLR	Log-Likelihood:	-3.1464e+0
Model:	OrderedModel	AIC:	6.293e+0
Method:	Maximum Likelihood	BIC:	6.294e+0
Date:	Mon, 05 Dec 2022		
Time:	06:36:34		
No. Observations:	562439		
Df Residuals:	562431		
Df Model:	8		

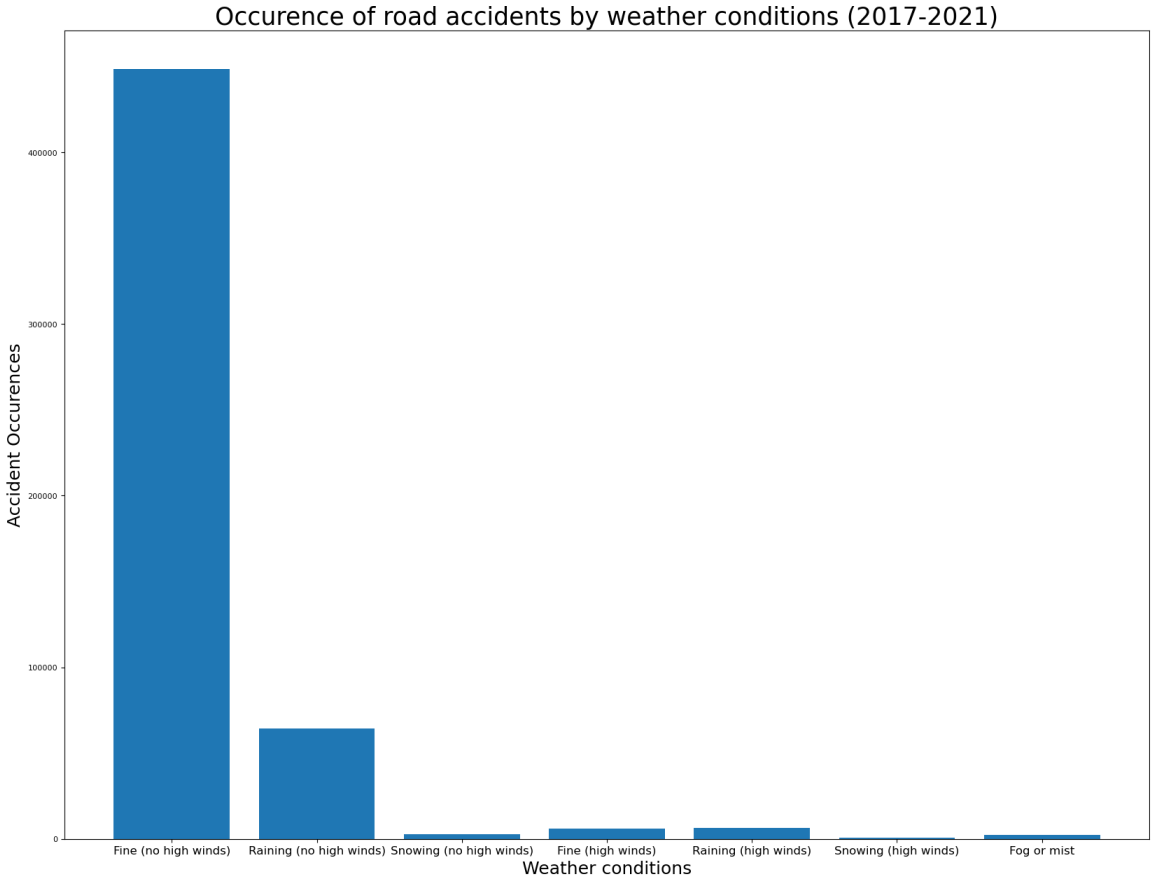
	coef	std err	z	P>|z|	[0.025
Raining_no_high_winds	-0.0739	0.011	-7.004	0.000	-0.095
Snowing_no_high_winds	-0.2293	0.050	-4.614	0.000	-0.327
Fine_and_high_winds	0.2313	0.030	7.598	0.000	0.172
Raining_and_high_winds	0.1859	0.029	6.338	0.000	0.128
Snowing_and_high_winds	-0.3333	0.103	-3.231	0.001	-0.535
Fog_or_mist	0.3012	0.047	6.420	0.000	0.209
1/2	1.3462	0.004	377.519	0.000	1.339
2/3	1.0667	0.004	280.368	0.000	1.059

	OR	Lower CI	Upper CI
Raining_no_high_winds	0.928754	0.909743	0.948163
Snowing_no_high_winds	0.795122	0.721332	0.876461
Fine_and_high_winds	1.260290	1.187279	1.337790
Raining_and_high_winds	1.204250	1.136988	1.275492
Snowing_and_high_winds	0.716568	0.585410	0.877111
Fog_or_mist	1.351520	1.232768	1.481710
1/2	3.842754	3.815991	3.869705
2/3	2.905877	2.884287	2.927627

How does the occurence of road accidents vary with weather conditions?

In [64]:
```
def weather_conditions_accident_occurences():
    h = len(accident_master[accident_master.Fine_no_high_winds == 1])
    i = len(accident_master[accident_master.Raining_no_high_winds == 1])
    j = len(accident_master[accident_master.Snowing_no_high_winds == 1])
    k = len(accident_master[accident_master.Fine_and_high_winds == 1])
    l = len(accident_master[accident_master.Raining_and_high_winds == 1])
    m = len(accident_master[accident_master.Snowing_and_high_winds == 1])
    n = len(accident_master[accident_master.Fog_or_mist == 1])


    weather_conditions_df = pd.DataFrame({'weather_conditions': ['Fine_no_high_w
                                          'Fine_and_high_winds', 'Raining_and_h
                                         'Fog_or_mist'],\
                        'accident_count': [h, i, j, k, l, m, n]})

    x = weather_conditions_df.weather_conditions
    y = weather_conditions_df.accident_count
    fig,ax = plt.subplots(figsize=(20, 15))
    plt.bar(x, y)
    labels = ['Fine (no high winds)', 'Raining (no high winds)','Snowing (no hig
                'Raining (high winds)', 'Snowing (high winds)', 'Fog or mist']
    plt.xticks(x, labels, fontsize=12)
    plt.yticks(fontsize=8)
    plt.title('Occurence of road accidents by weather conditions (2017-2021)', f
    plt.xlabel('Weather conditions', fontsize=18)
    plt.ylabel('Accident Occurences', fontsize=18)
    plt.show()
```

In [65]:
```
weather_conditions_accident_occurences()
```



The odds of being in a greater severity accident by road conditions

Checking that the multicollinearity assumption is met

In [66]:
```
df_roadcond_multicollinearity_test = accident_master.iloc[:,57:64]
df_roadcond_multicollinearity_test_print = df_roadcond_multicollinearity_test.co
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_roadcond_multicollinearity_test_print)
```

	Dry_road	Wet_or_damp_road	Snow_road	Frost_or_ice_road	Flood_
Dry_road	1.000000	-0.925059	-0.104222	-0.177414	
Wet_or_damp_road	-0.925059	1.000000	-0.038005	-0.064695	
Snow_road	-0.104222	-0.038005	1.000000	-0.007289	
Frost_or_ice_road	-0.177414	-0.064695	-0.007289	1.000000	
Flood_over_3cm_deep_road	-0.057693	-0.021038	-0.002370	-0.004035	
Oil_or_diesel_road	NaN	NaN	NaN	NaN	
Mud_road	NaN	NaN	NaN	NaN	

Dry road conditions exhibit the greatest levels of multicollinearity with the other categories. Correlation of -0.92 exists between dry road conditions and wet or damp roads, and -0.17 against frost or ice roads. As such, the dry road condition category is not used in the OLR analysis, and instead acts as the reference variable. Moreover, oil or diesel covered roads and mud covered roads are removed as the occurence of accidents in these conditions is zero.

Ordinal Logistic Regression is used to understand how weather conditions affect the odds of being in an accident of greater severity (i.e. serious or fatal, rather than slight).

Dry road conditions were chosen as the reference variable, as they provide an intuitive benchmark against which to compare, and because they comprise the bulk of accidents. Dummy variables: 1) Wet_or_damp_road, 2) Snow_road, 3) Frost_or_ice_road, 4) Flood_over_3cm_deep_road. Oil_or_diesel_road and Mud_road are removed as the occurences of accidents in these conditions is zero.

OLR Model

In [67]:
```
def road_conditions_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                        'Frost_or_ice_road', 'Flo
                                                        ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios (i.e. the odds of being in an accide
    # or fatal, rather than slight, as a result of the different road conditions
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

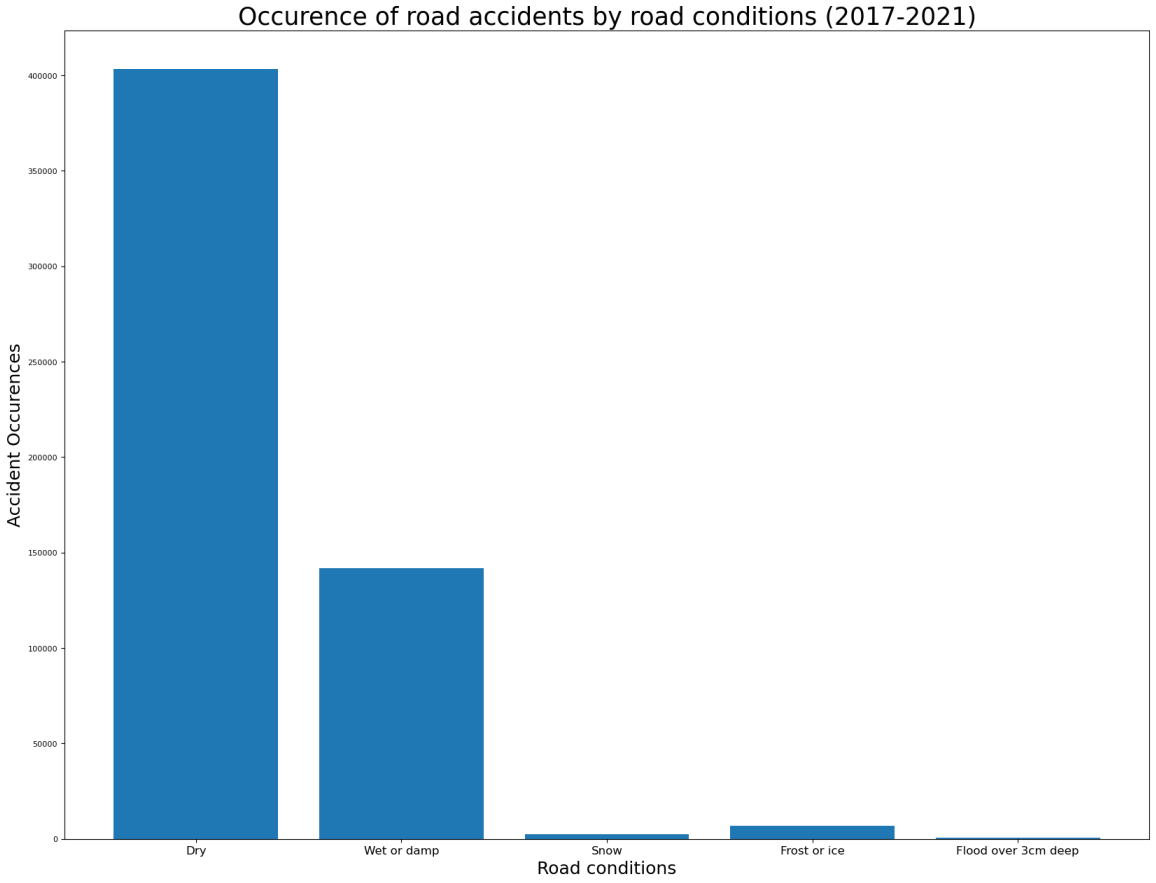
In [68]:
```
road_conditions_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1470e+0
Model:                      OrderedModel   AIC:                         6.294e+0
Method:             Maximum Likelihood     BIC:                         6.295e+0
Date:                   Mon, 05 Dec 2022
Time:                           06:38:24
No. Observations:                562439
Df Residuals:                    562433
Df Model:                             6
==============================================================================
                                coef    std err          z      P>|z|      [0.025
------------------------------------------------------------------------------
Wet_or_damp_road              0.0525      0.008      6.955      0.000       0.038
Snow_road                    -0.3428      0.057     -6.059      0.000      -0.454
Frost_or_ice_road            -0.0944      0.031     -3.053      0.002      -0.155
Flood_over_3cm_deep_road      0.1417      0.088      1.613      0.107      -0.030
1/2                           1.3609      0.004    351.677      0.000       1.353
2/3                           1.0666      0.004    280.319      0.000       1.059
==============================================================================
                                OR  Lower CI  Upper CI
Wet_or_damp_road          1.053878  1.038407  1.069579
Snow_road                 0.709753  0.635244  0.793002
Frost_or_ice_road         0.909880  0.856352  0.966755
Flood_over_3cm_deep_road  1.152238  0.969966  1.368760
1/2                       3.899644  3.870180  3.929334
2/3                       2.905530  2.883942  2.927280
```

How does the occurence of road accidents vary with road conditions?

In [69]:
```
def road_conditions_accident_occurences():
    o = len(accident_master[accident_master.Dry_road == 1])
    p = len(accident_master[accident_master.Wet_or_damp_road == 1])
    q = len(accident_master[accident_master.Snow_road == 1])
    r = len(accident_master[accident_master.Frost_or_ice_road == 1])
    s = len(accident_master[accident_master.Flood_over_3cm_deep_road == 1])



    road_conditions_df = pd.DataFrame({'road_surface_conditions': ['Dry_road', '
                                            'Frost_or_ice_road', 'Flood_over_3cm_
                                          ],\
                       'accident_count': [o, p, q, r, s]})

    x = road_conditions_df.road_surface_conditions
    y = road_conditions_df.accident_count
    fig,ax = plt.subplots(figsize=(20, 15))
    plt.bar(x, y)
    labels = ['Dry', 'Wet or damp','Snow', 'Frost or ice',
                 'Flood over 3cm deep']
    plt.xticks(x, labels, fontsize=12)
    plt.yticks(fontsize=8)
    plt.title('Occurence of road accidents by road conditions (2017-2021)', font
    plt.xlabel('Road conditions', fontsize=18)
    plt.ylabel('Accident Occurences', fontsize=18)

    plt.show()
```

In [70]:
```
road_conditions_accident_occurences()
```



The odds of being in a greater severity accident by day of week

Checking that the multicollinearity assumption is met

In [71]:
```
df_dayofweek_multicollinearity_test = accident_master.iloc[:,36:43]
df_dayofweek_multicollinearity_test_print = df_dayofweek_multicollinearity_test.
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_dayofweek_multicollinearity_test_print)
```

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Monday	1.000000	-0.168162	-0.169415	-0.171366	-0.179127	-0.158825	-0.144310
Tuesday	-0.168162	1.000000	-0.174170	-0.176176	-0.184154	-0.163282	-0.148361
Wednesday	-0.169415	-0.174170	1.000000	-0.177489	-0.185527	-0.164499	-0.149466
Thursday	-0.171366	-0.176176	-0.177489	1.000000	-0.187663	-0.166394	-0.151188
Friday	-0.179127	-0.184154	-0.185527	-0.187663	1.000000	-0.173929	-0.158034
Saturday	-0.158825	-0.163282	-0.164499	-0.166394	-0.173929	1.000000	-0.140123
Sunday	-0.144310	-0.148361	-0.149466	-0.151188	-0.158034	-0.140123	1.000000

Friday exhibits the greatest levels of multicollinearity with the other categories. Correlation of -0.19 exists between Friday and Thursday, -0.19 against Wednesday, and -0.18 against Tuesday. As such, the Friday category is not used in the OLR analysis, and instead acts as the reference variable.

Ordinal Logistic Regression is used to understand how day of the week affects the odds of being in an accident of greater severity (i.e. serious or fatal, rather than slight).

Monday was chosen as the reference variable for an intuitive comparison baseline.

Again, a **logit** ordinal regression method was employed rather than a probit method, because this allows for more intuitive analysis, and because the choice is negligible for a large sample size.

OLR Model

In [72]:
```
def day_of_week_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                        'Thursday', 'Friday' ,'Sa
                                                        ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios (i.e. the odds of being in an accide
    # or fatal, rather than slight, as a result of the day of week))
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

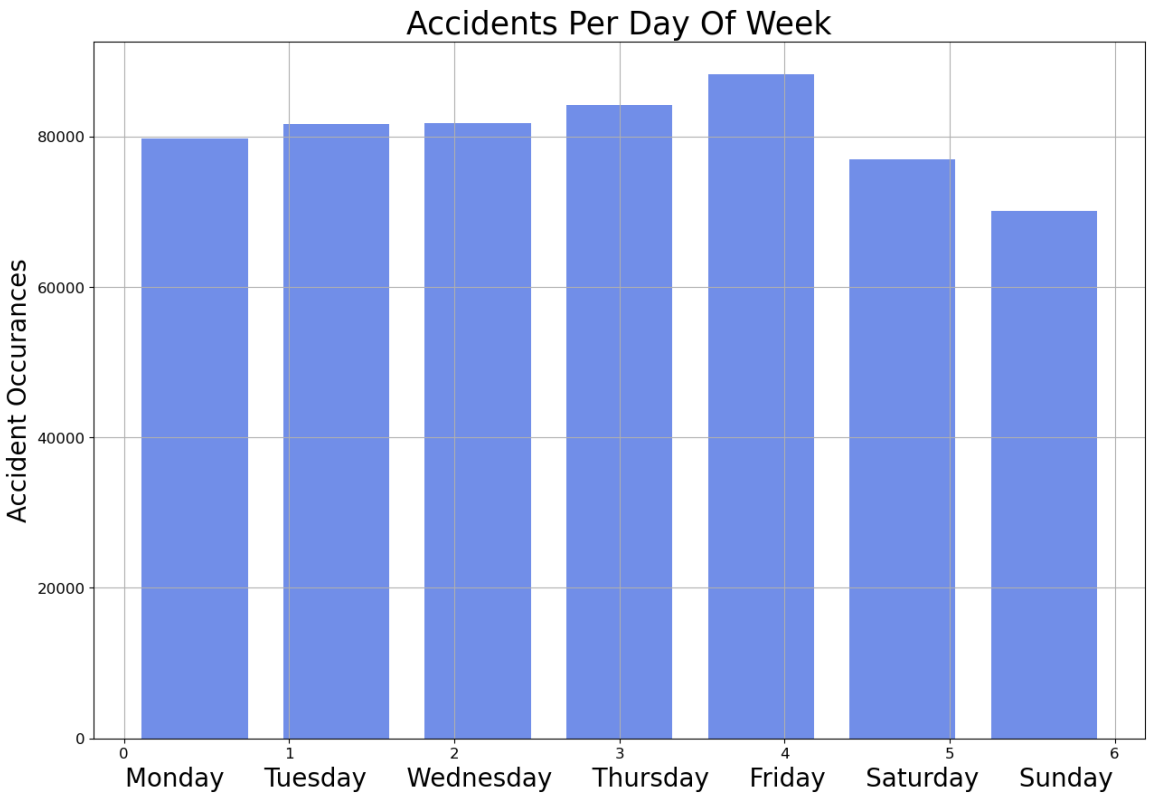
In [73]:
```
day_of_week_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:           -3.1445e+0
5
Model:                      OrderedModel   AIC:                        6.289e+0
5
Method:             Maximum Likelihood   BIC:                        6.290e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           06:40:20
No. Observations:              562439
Df Residuals:                  562431
Df Model:                           8
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Tuesday       -0.0156      0.012     -1.255      0.210      -0.040       0.009
Wednesday     -0.0067      0.012     -0.540      0.589      -0.031       0.018
Thursday      -0.0065      0.012     -0.523      0.601      -0.031       0.018
Friday         0.0097      0.012      0.803      0.422      -0.014       0.033
Saturday       0.1392      0.012     11.152      0.000       0.115       0.164
Sunday         0.2187      0.013     16.964      0.000       0.193       0.244
1/2            1.3921      0.009    156.535      0.000       1.375       1.410
2/3            1.0670      0.004    280.495      0.000       1.060       1.074
==============================================================================
                 OR  Lower CI  Upper CI
Tuesday    0.984492  0.960751  1.008819
Wednesday  0.993326  0.969474  1.017765
Thursday   0.993570  0.969823  1.017899
Friday     1.009758  0.986095  1.033988
Saturday   1.149391  1.121607  1.177864
Sunday     1.244402  1.213359  1.276238
1/2        4.023483  3.953957  4.094231
2/3        2.906617  2.885027  2.928368
```

Distribution of Accidents Per Day of Week

In [74]:
```
def accident_by_day():
    accident_master['date_time'] = pd.to_datetime(accident_master.date_time)

    plt.figure(figsize = (15, 10))
    accident_master.date_time.dt.dayofweek.hist(bins = 7, rwidth = 0.75, alpha =
    plt.title("Accidents Per Day Of Week", fontsize = 25)
    plt.ylabel('Accident Occurances', fontsize = 20)
    plt.xlabel('Monday     Tuesday     Wednesday     Thursday     Friday     Sat
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)
```

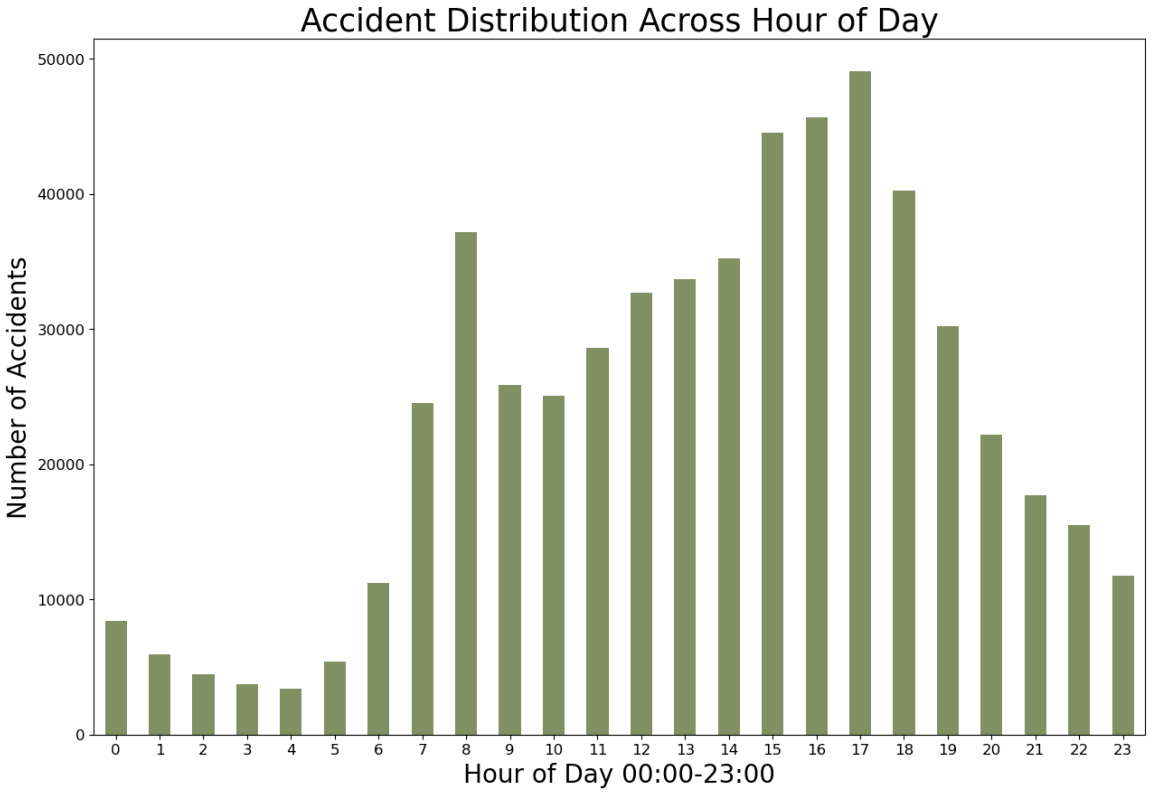
In [75]:
```
accident_by_day()
```



Distribution of Accidents Per Day of hour

In [76]:
```
def accidents_per_hr():
    accidents_hr = accident_master.groupby(accident_master['date_time'].dt.hour)
    accidents_hr.plot(kind = 'bar', figsize = (15, 10), color = 'darkolivegreen'

    plt.title("Accident Distribution Across Hour of Day", fontsize = 25)
    plt.ylabel('Number of Accidents', fontsize = 20)
    plt.xlabel('Hour of Day 00:00-23:00', fontsize = 20)
    plt.xticks(fontsize = 12, rotation = 0)
    plt.yticks(fontsize = 12)
```

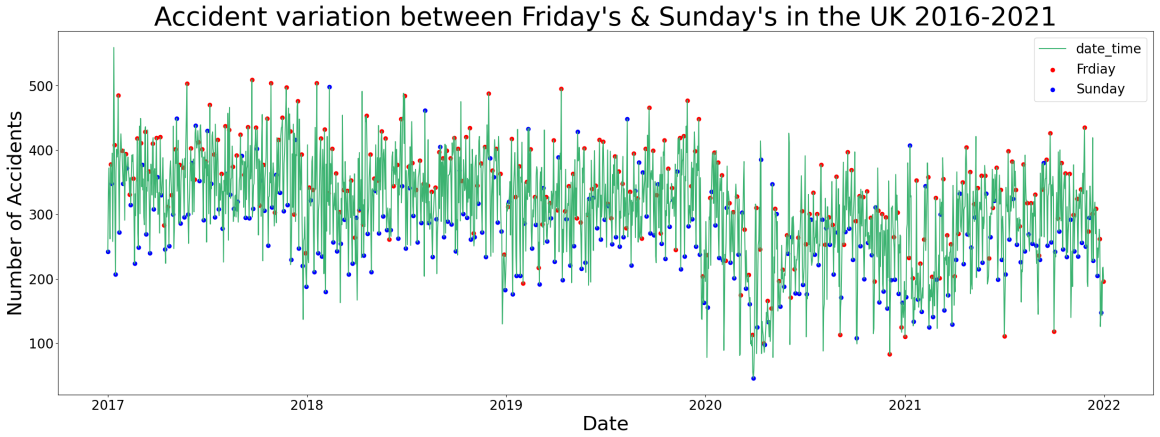
In [77]:
```
accidents_per_hr()
```



Accident variation between Fridays (max) and Sundays (min)

In [78]:
```
def friday_sunday_accidents():
    accidents = accident_master.groupby(accident_master['date_time'].dt.date).co
    accidents.plot(figsize = (30, 10), color = 'mediumseagreen')

    friday = accident_master.groupby(accident_master[accident_master['date_time'
    plt.scatter(friday.index, friday, color = 'red', label = 'Frdiay')

    sunday = accident_master.groupby(accident_master[accident_master['date_time'
    plt.scatter(sunday.index, sunday, color = 'blue', label = 'Sunday')

    plt.title("Accident variation between Friday's & Sunday's in the UK 2016-202
    plt.ylabel('Number of Accidents', fontsize = 30)
    plt.xlabel('Date', fontsize = 30)
    plt.xticks(fontsize = 20)
    plt.yticks(fontsize = 20)
    plt.legend(loc = 1, prop = {'size':20})
```

In [79]:
```
friday_sunday_accidents()
```



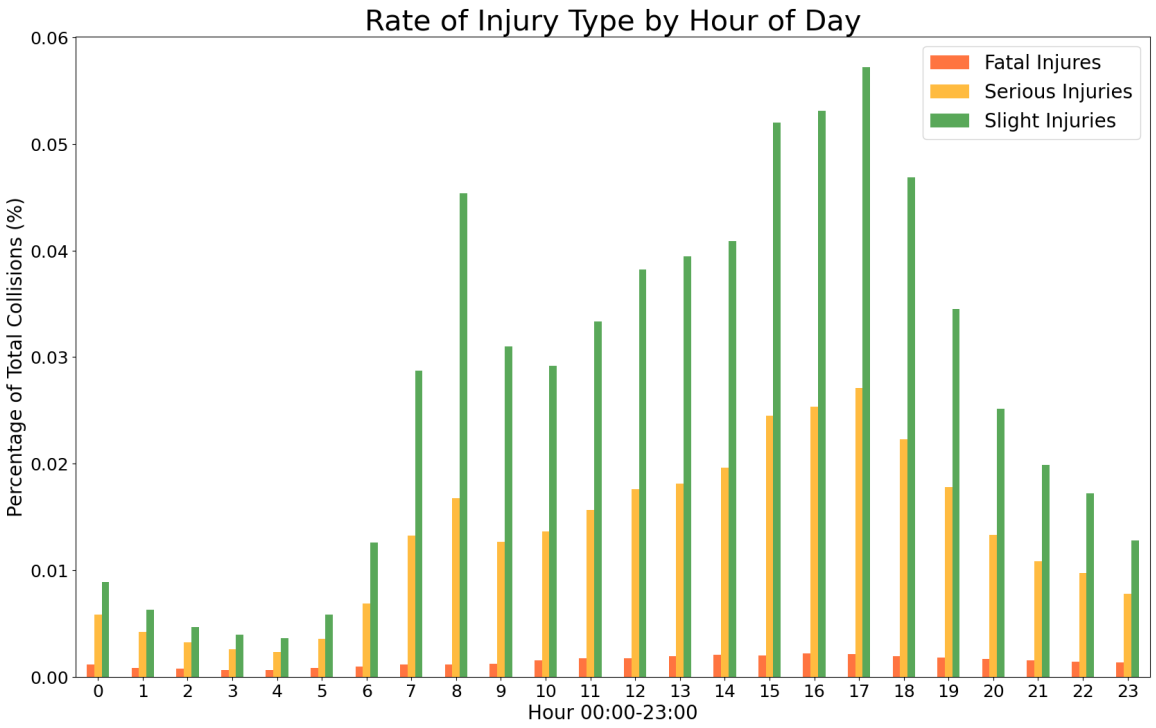
Slight - Serious - Fatal Injuries

In [81]:
```
def injury_by_type():
    # Series with number of mild injuries and serious injuries
    injury = accident_master[['accident_severity']].value_counts()

    # Pie plot with the percentage of victims with slight, serious and fatal inj
    explode = (0.03, 0.03, 0)
    injury.plot(kind = 'pie',figsize = (10,10), colors = ['forestgreen','orange'

    # Title and legend
    plt.legend(title = "Injury Level", labels = ['Slight', 'Serious', 'Fatal'],b
    plt.title('Injury Type from UK Road Accidents', fontsize = 25)
    plt.ylabel('')
```

Rate of Injuries per Hour of Day

In [82]:
```
def rate_of_injury_per_hour():
    accident_severity = accident_master['accident_severity']
    # Number of fatal injuries per day of the week
    accidents_fatal = accident_master[accident_master['accident_severity'] == 3]
    # Percentage of fatal injuries per day of the week
    rate_fatal = accidents_fatal/accident_severity.sum()

    # Number of serious injuries per day of the week
    accidents_serious = accident_master[accident_master['accident_severity'] ==
    # Percentage of seriois injuries per day of the week
    rate_serious = accidents_serious/accident_severity.sum()

    # Number of slight injuries per day of the week
    accidents_slight = accident_master[accident_master['accident_severity'] == 1
    # Percentage of slight injuries per day of the week
    rate_slight = accidents_slight/accident_severity.sum()

    # Combine both series as a dataframe in order to plot them as a side by side
    rates = pd.DataFrame({'Fatal Injures':rate_fatal,'Serious Injuries':rate_ser
    rates.plot(kind = 'bar', figsize = (20,12), width = 0.5, color = ['orangered

    # Title and labels
    plt.title('Rate of Injury Type by Hour of Day',fontsize = 30)
    plt.xlabel('Hour 00:00-23:00',fontsize = 20)
    plt.ylabel('Percentage of Total Collisions (%)',fontsize = 20)
    plt.xticks(fontsize = 18, rotation = 0)
    plt.yticks(fontsize = 18)
    plt.legend(fontsize = 20)
```

In [83]:
```
rate_of_injury_per_hour()
```



The odds of being in a greater severity accident by speed limit

Checking that the multicollinearity assumption is met

In [85]:
```
df_speedlimit_multicollinearity_test = accident_master.iloc[:,70:76]
df_speedlimit_multicollinearity_test_print = df_speedlimit_multicollinearity_tes
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_speedlimit_multicollinearity_test_print)
```

	20_mph	30_mph	40_mph	50_mph	60_mph	70_mph
20_mph	1.000000	-0.404773	-0.102786	-0.070134	-0.127833	-0.085269
30_mph	-0.404773	1.000000	-0.361848	-0.246901	-0.450024	-0.300181
40_mph	-0.102786	-0.361848	1.000000	-0.062697	-0.114277	-0.076226
50_mph	-0.070134	-0.246901	-0.062697	1.000000	-0.077975	-0.052012
60_mph	-0.127833	-0.450024	-0.114277	-0.077975	1.000000	-0.094802
70_mph	-0.085269	-0.300181	-0.076226	-0.052012	-0.094802	1.000000

30 mph exhibits the greatest levels of multicollinearity with the other categories. Correlation of -0.40 exists between 30 mph and 20 mph, -0.36 against 40 mph, -0.25 against 50 mph and -0.45 against 60 mph. As such, the 30mph category is not used in the OLR analysis, and instead acts as the reference variable.

OLR Model

In [86]:
```
def speedlimit_severity_logistic_ordinal_regression():
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                        '40_mph', '50_mph' ,'60_m
                                                        ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

In [87]:
```
speedlimit_severity_logistic_ordinal_regression()
```

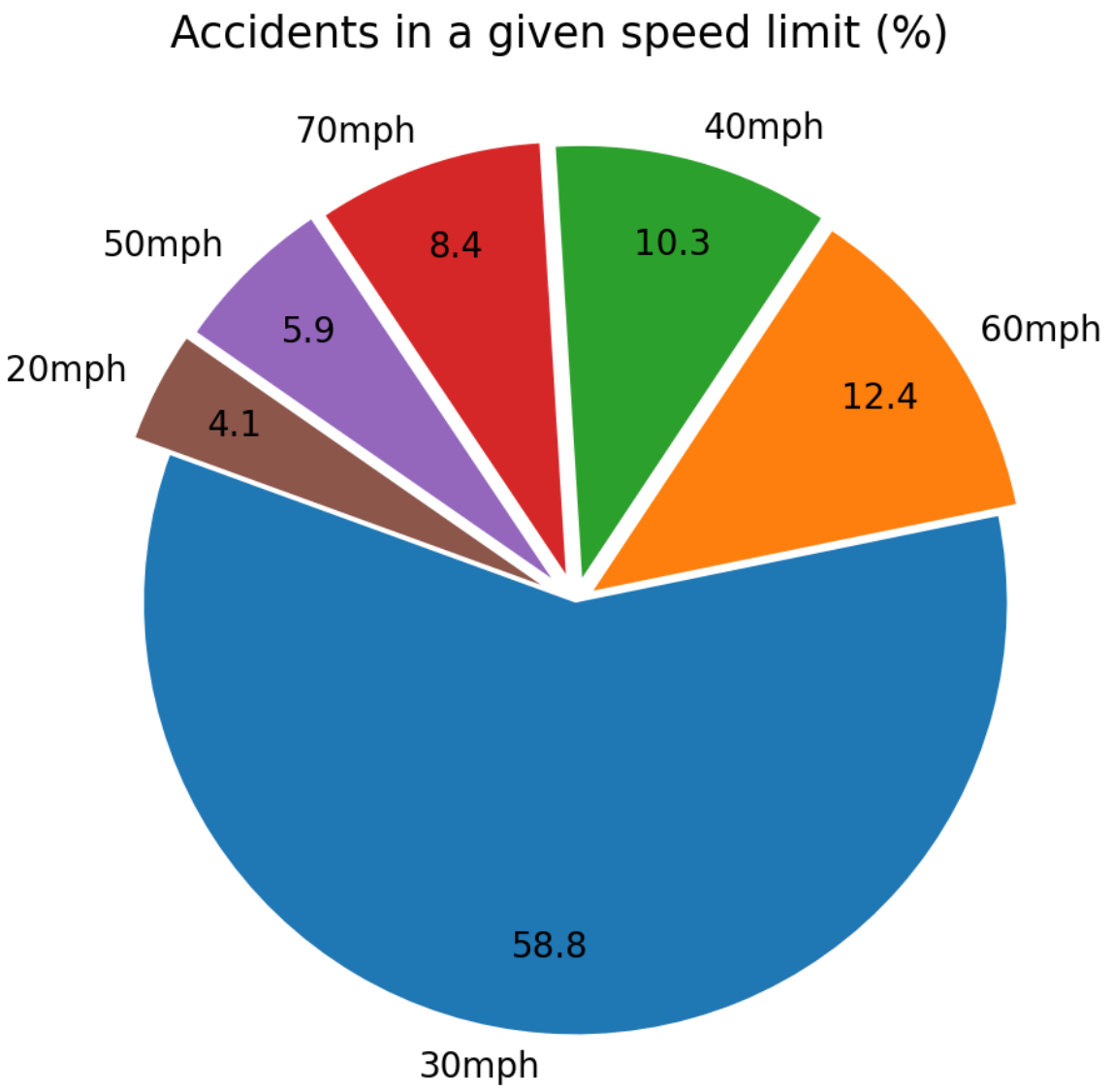
```
                             OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1188e+0
5
Model:                      OrderedModel   AIC:                         6.238e+0
5
Method:               Maximum Likelihood   BIC:                         6.238e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           06:42:20
No. Observations:                562439
Df Residuals:                    562432
Df Model:                             7
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
30_mph         0.1717      0.012     14.243      0.000       0.148       0.195
40_mph         0.3797      0.016     24.116      0.000       0.349       0.411
50_mph         0.4667      0.019     24.356      0.000       0.429       0.504
60_mph         0.8375      0.014     60.329      0.000       0.810       0.865
70_mph         0.3239      0.018     18.500      0.000       0.290       0.358
1/2            1.6398      0.011    146.360      0.000       1.618       1.662
2/3            1.0713      0.004    282.493      0.000       1.064       1.079
==============================================================================
              OR  Lower CI  Upper CI
30_mph  1.187265  1.159549  1.215643
40_mph  1.461880  1.417453  1.507698
50_mph  1.594751  1.535966  1.655785
60_mph  2.310639  2.248616  2.374373
70_mph  1.382531  1.335891  1.430800
1/2     5.153911  5.041972  5.268335
2/3     2.919256  2.897637  2.941035
```

Distribution of accidents within Speed Limit Zones

In [88]:
```
accident_speed_limit= accident_master[accident_master['speed_limit'] != -1] #dis
```

In [89]:
```
def accident_in_speed_limit():
    acc_speed_limit = accident_speed_limit.speed_limit.value_counts()
    labels = ['30mph', '60mph', '40mph', '70mph', '50mph', '20mph']

    explode = (0, 0.05, 0.06, 0.07, 0.08, 0.09)
    fig = plt.figure(figsize = (10,10))
    plt.pie(acc_speed_limit.values, labels=labels, autopct = '%.1f', pctdistance
    plt.axis('equal')
    plt.figtext(0.5, 0.93, 'Accidents in a given speed limit (%)', fontsize = 25
    plt.show()
```

In [90]:
```
accident_in_speed_limit()
```



The odds of being in a greater severity accident by road class

Checking that the multicollinearity assumption is met

In [91]:
```
df_roadclass_multicollinearity_test = accident_master.iloc[:,64:69]
df_roadclass_multicollinearity_test_print = df_roadclass_multicollinearity_test.
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_roadclass_multicollinearity_test_print)
```

	Motorway_road_class	A_M_road_class	A_road_road_class	B_road_class	C_
Motorway_road_class	1.000000	-0.009384	-0.161197	-0.067216	
A_M_road_class	-0.009384	1.000000	-0.045903	-0.019140	
A_road_road_class	-0.161197	-0.045903	1.000000	-0.328777	
B_road_class	-0.067216	-0.019140	-0.328777	1.000000	
C_road_class	-0.043323	-0.012337	-0.211909	-0.088361	

The A-road class category exhibits the greatest levels of multicollinearity with the other categories. Correlation of -0.16 exists against the motorway category, -0.05 against A (M) roads, -0.33 against B roads, and -0.21 against C roads. As such, the A-road class category is not used in the OLR analysis, and instead acts as the reference variable.

Ordinal Logistic Regression is used to understand how road class affects the odds of being in an accident of greater severity (i.e. serious or fatal, rather than slight).

A-roads were chosen as the reference variable for an intuitive comparison baseline, and unclassified roads were removed as they do not infer any meaningful information for this analysis.

Again, a **logit** ordinal regression method was employed rather than a probit method, because this allows for more intuitive analysis, and because the choice is negligible for a large sample size.

OLR Model

In [92]:
```
def road_class_severity_logistic_ordinal_regression():
# Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                'B_road_class', 'C_road_class
                                                ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

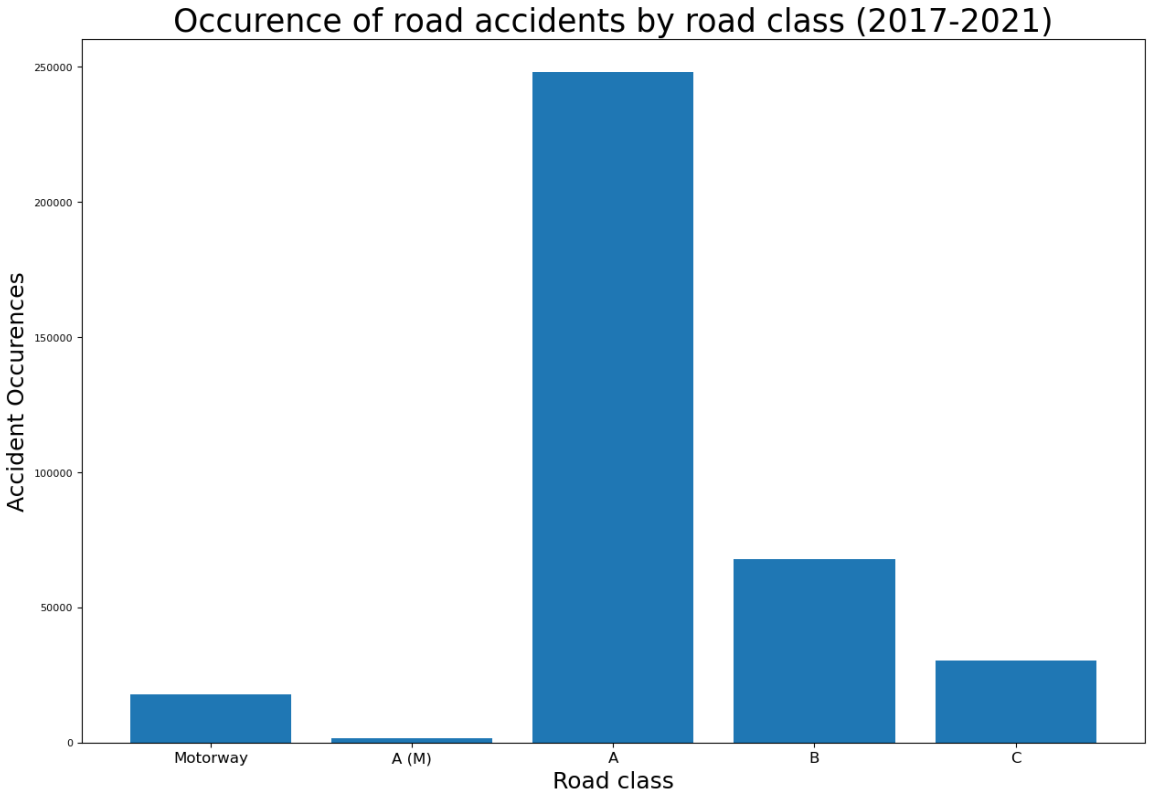
In [93]:
```
road_class_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1459e+0
Model:                      OrderedModel   AIC:                         6.292e+0
Method:             Maximum Likelihood     BIC:                         6.293e+0
Date:                Mon, 05 Dec 2022
Time:                        06:43:50
No. Observations:             562439
Df Residuals:                 562433
Df Model:                          6
==============================================================================
                        coef    std err          z      P>|z|      [0.025      0
------------------------------------------------------------------------------
A_road_road_class     0.0494      0.007      6.767      0.000       0.035       
A_M_road_class       -0.0414      0.066     -0.628      0.530      -0.170       
B_road_class          0.1579      0.011     14.871      0.000       0.137       
C_road_class         -0.1059      0.016     -6.724      0.000      -0.137      -
1/2                   1.3857      0.005    257.682      0.000       1.375       
2/3                   1.0669      0.004    280.412      0.000       1.059       
==============================================================================
                         OR  Lower CI  Upper CI
A_road_road_class  1.050668  1.035734  1.065817
A_M_road_class     0.959481  0.843366  1.091582
B_road_class       1.170997  1.146886  1.195614
C_road_class       0.899534  0.872198  0.927727
1/2                3.997593  3.955681  4.039950
2/3                2.906216  2.884625  2.927968
```

How does the occurence of road accidents vary with road class?

In [94]:
```
def road_class_accident_occurences():
    t = len(accident_master[accident_master.Motorway_road_class == 1])
    u = len(accident_master[accident_master.A_M_road_class == 1])
    v = len(accident_master[accident_master.A_road_road_class == 1])
    w = len(accident_master[accident_master.B_road_class == 1])
    y = len(accident_master[accident_master.C_road_class == 1])

    road_types_df = pd.DataFrame({'road_type': ['Motorway_road_class', 'A_M_road
                                                'B_road_class', 'C_road_class'
                                               ],\
                          'accident_count': [t, u, v, w, y]})

    x = road_types_df.road_type
    y = road_types_df.accident_count
    fig,ax = plt.subplots(figsize=(15, 10))
    plt.bar(x, y)
    labels = ['Motorway', 'A (M)','A', 'B',
              'C']
    plt.xticks(x, labels, fontsize=12)
    plt.yticks(fontsize=8)
    plt.title('Occurence of road accidents by road class (2017-2021)', fontsize=
    plt.xlabel('Road class', fontsize=18)
    plt.ylabel('Accident Occurences', fontsize=18)

    plt.show()
```

In [95]:
```
road_class_accident_occurences()
```



The odds of being in a greater severity accident by age

Checking that the multicollinearity assumption is met

In [96]:
```
df_age_multicollinearity_test = accident_master.iloc[:,20:31]
df_age_multicollinearity_test_print = df_age_multicollinearity_test.corr()
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_age_multicollinearity_test_print)
```

	age_0_5	age_6_10	age_11_15	age_16_20	age_21_25	age_26_35	age_36_45	age
age_0_5	1.000000	-0.001056	-0.001484	-0.006154	-0.002168	-0.007962	-0.002653	-0.
age_6_10	-0.001056	1.000000	-0.004110	-0.012451	-0.012733	-0.014789	-0.014932	-0.
age_11_15	-0.001484	-0.004110	1.000000	-0.028676	-0.034463	-0.043470	-0.032258	-0.
age_16_20	-0.006154	-0.012451	-0.028676	1.000000	-0.082581	-0.120568	-0.095359	-0.
age_21_25	-0.002168	-0.012733	-0.034463	-0.082581	1.000000	-0.125260	-0.105383	-0.
age_26_35	-0.007962	-0.014789	-0.043470	-0.120568	-0.125260	1.000000	-0.147041	-0.
age_36_45	-0.002653	-0.014932	-0.032258	-0.095359	-0.105383	-0.147041	1.000000	-0.
age_46_55	-0.004544	-0.010468	-0.030171	-0.088341	-0.097436	-0.143493	-0.119745	1.
age_56_65	-0.002126	-0.009995	-0.021726	-0.067737	-0.076389	-0.115097	-0.096623	-0.
age_66_75	-0.000475	-0.007357	-0.014364	-0.047338	-0.059339	-0.088660	-0.070563	-0.
age_75plus	-0.002514	-0.008178	-0.014408	-0.046496	-0.057181	-0.082704	-0.061671	-0.

The age (26-35) category exhibits the greatest levels of multicollinearity with the other categories. Correlation of -0.12 exists against the age (16-20) category, -0.13 against the age (21-25) cateogry, -0.15 against age (36-45), and -0.14 against age (46-55). As such, the age (26-35) category is not used in the OLR analysis, and instead acts as the reference variable.

OLR Model

In [97]:
```
def age_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                        'age_36_45', 'age_46_55',
                                                        ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

In [98]:
```
age_severity_logistic_ordinal_regression()
```

```
                                OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1358e+0
5
Model:                      OrderedModel   AIC:                         6.272e+0
5
Method:             Maximum Likelihood     BIC:                         6.273e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           06:46:44
No. Observations:                562439
Df Residuals:                    562430
Df Model:                             9
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
age_16_20      0.0350      0.011      3.330      0.001       0.014       0.056
age_26_35     -0.1653      0.008    -21.943      0.000      -0.180      -0.151
age_36_45     -0.1441      0.008    -18.126      0.000      -0.160      -0.128
age_46_55      0.0020      0.008      0.247      0.805      -0.014       0.018
age_56_65      0.1436      0.009     15.942      0.000       0.126       0.161
age_66_75      0.2168      0.012     18.400      0.000       0.194       0.240
age_75plus     0.3295      0.014     23.303      0.000       0.302       0.357
1/2            1.3207      0.007    191.591      0.000       1.307       1.334
2/3            1.0683      0.004    281.103      0.000       1.061       1.076
==============================================================================
                   OR  Lower CI  Upper CI
age_16_20    1.035670  1.014525  1.057255
age_26_35    0.847642  0.835219  0.860250
age_36_45    0.865832  0.852449  0.879425
age_46_55    1.001973  0.986450  1.017740
age_56_65    1.154398  1.134199  1.174956
age_66_75    1.242038  1.213690  1.271048
age_75plus   1.390220  1.352226  1.429281
1/2          3.746070  3.695798  3.797026
2/3          2.910397  2.888800  2.932157
```

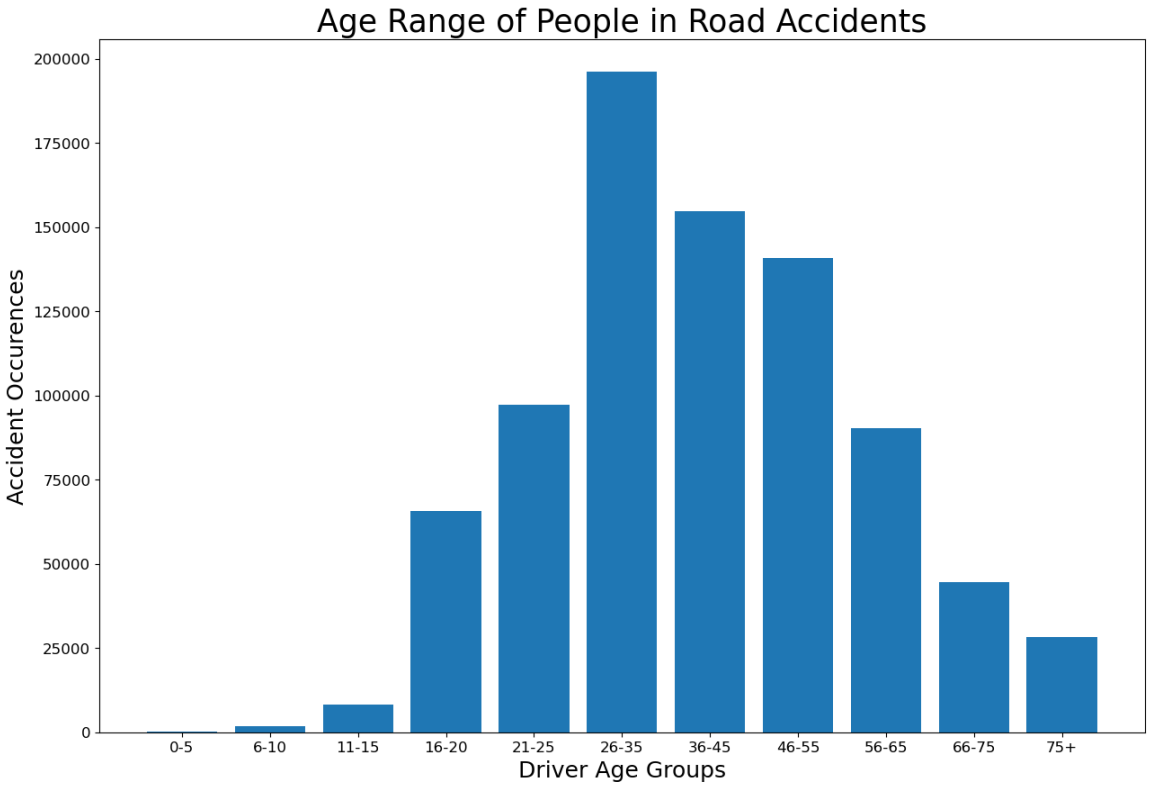
Age of People Involved in accidents

In [99]:

```
a = len(accident_master[accident_master.age_0_5 == 1])
b = len(accident_master[accident_master.age_6_10 == 1])
c = len(accident_master[accident_master.age_11_15 == 1])
d = len(accident_master[accident_master.age_16_20 == 1])
e = len(accident_master[accident_master.age_21_25 == 1])
f = len(accident_master[accident_master.age_26_35 == 1])
g = len(accident_master[accident_master.age_36_45 == 1])
h = len(accident_master[accident_master.age_46_55 == 1])
i = len(accident_master[accident_master.age_56_65 == 1])
j = len(accident_master[accident_master.age_66_75 == 1])
k = len(accident_master[accident_master.age_75plus == 1])

age_bands_df = pd.DataFrame({'driver_age_band': ['0-5', '6-10', '11-15',\
                                        '16-20', '21-25', '26-35', '36-45',\
                                        '46-55', '56-65', '66-75', '75+'],\
                   'accident_cont': [a, b, c, d, e, f, g, h, i, j, k]})
```

In [100…]
```
def accident_age_range():
    x = age_bands_df.driver_age_band
    y = age_bands_df.accident_cont
    fig,ax = plt.subplots(figsize=(15, 10))
    plt.bar(x, y)
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)
    plt.title('Age Range of People in Road Accidents', fontsize=25)
    plt.xlabel('Driver Age Groups', fontsize=18)
    plt.ylabel('Accident Occurences', fontsize=18)

    plt.show()
```

In [101…]
```
accident_age_range()
```



Urban or rural area

The odds of being in a greater severity accident by area type

Checking that the multicollinearity assumption is met

In [102…]
```
df_urban_rural_multicollinearity_test = accident_master.iloc[:,76:78]
df_urban_rural_multicollinearity_test_print = df_urban_rural_multicollinearity_t
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_urban_rural_multicollinearity_test_print)
```

	Urban_area	Rural_area
Urban_area	1.000000	-0.999493
Rural_area	-0.999493	1.000000

As the categories of urban and rural areas involved in accidents are effectively measuring the same thing (the proportion of accidents in an urban or rural area), complete multicollinearity is exhibited. As such, only 'Urban_area' will be included in the OLR model. This means that 'Rural_area' can be used as the reference variable to compare odds against.

OLR Model

In [103…

```
def urban_or_rural_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
                                                            ]], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the proportional odds ratios (i.e. the odds of being in an accide
    # or fatal, rather than slight, as a result of the road type))
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

In [104…

```
urban_or_rural_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:           -3.1255e+0
5
Model:                      OrderedModel   AIC:                        6.251e+0
5
Method:             Maximum Likelihood   BIC:                        6.251e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           06:47:04
No. Observations:              562439
Df Residuals:                  562436
Df Model:                           3
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Urban_area    -0.4556      0.007    -66.935      0.000      -0.469      -0.442
1/2            1.0539      0.005    196.543      0.000       1.043       1.064
2/3            1.0699      0.004    281.856      0.000       1.063       1.077
==============================================================================
                  OR  Lower CI  Upper CI
Urban_area  0.634070  0.625668  0.642586
1/2         2.868947  2.838951  2.899259
2/3         2.915215  2.893606  2.936985
```

The odds of being in a greater severity accident by overall driving conditions

Checking that the multicollinearity assumption is met

In [105…

```
df_driving_conditions_multicollinearity_test = accident_master.iloc[:,43:78]
df_driving_conditions_multicollinearity_test_print = df_driving_conditions_multi
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_driving_conditions_multicollinearity_test_print)
```

	Daylight	Darkness_lights_lit	Darkness_lights_unlit	Darkness_no_lig
Daylight	1.000000	-0.804548	-0.135613	-0.36
Darkness_lights_lit	-0.804548	1.000000	-0.043276	-0.1
Darkness_lights_unlit	-0.135613	-0.043276	1.000000	-0.0
Darkness_no_lighting	-0.367540	-0.117287	-0.019770	1.0
Darkness_lighting_unknown	-0.235817	-0.075253	-0.012684	-0.0
Fine_no_high_winds	0.153839	-0.103933	-0.026769	-0.0
Raining_no_high_winds	-0.126858	0.106166	0.022833	0.0
Snowing_no_high_winds	-0.023879	0.011205	0.002444	0.0
Fine_and_high_winds	-0.008482	0.004099	0.002352	0.0
Raining_and_high_winds	-0.059128	0.034368	0.010955	0.0
Snowing_and_high_winds	-0.011565	0.000745	0.003493	0.0
Fog_or_mist	-0.049516	0.010402	0.012083	0.0
Other	-0.036141	0.026340	0.006082	0.0
Unknown	-0.019212	0.002297	0.003340	-0.0
Dry_road	0.221685	-0.157084	-0.033218	-0.1
Wet_or_damp_road	-0.210331	0.160838	0.031656	0.1
Snow_road	-0.016458	0.003188	0.002779	0.0
Frost_or_ice_road	-0.043735	0.007888	0.003446	0.0
Flood_over_3cm_deep_road	-0.018605	-0.006164	0.006181	0.0
Oil_or_diesel_road	NaN	NaN	NaN	
Mud_road	NaN	NaN	NaN	
Motorway_road_class	-0.015368	-0.021073	0.010626	0.0
A_M_road_class	-0.003999	-0.010883	-0.001161	0.0
A_road_road_class	-0.024377	0.032949	-0.003575	-0.0
B_road_class	0.002842	-0.015805	-0.001292	0.0
C_road_class	-0.002466	0.004447	-0.003294	-0.0
Unclassified_road_class	0.030694	-0.016671	0.002373	-0.0
20_mph	-0.003789	0.034700	-0.005448	-0.0
30_mph	0.002996	0.111208	-0.001380	-0.2
40_mph	-0.002091	0.002917	0.002058	0.0
50_mph	0.001946	-0.026876	0.003479	0.0
60_mph	0.014866	-0.154576	-0.006215	0.2
70_mph	-0.021290	-0.041146	0.013278	0.1
Urban_area	-0.027976	0.179534	-0.003964	-0.2
Rural_area	0.027969	-0.179523	0.004005	0.2

Multicollinearity amongst the driving conditions variables is low and so all factors remain in the analysis.

OLR Model

In [106…

```
def driving_conditions_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
    'Raining_no_high_winds','Snowing_no_high_winds','Fine_and_high_winds','Raini
    'Snowing_and_high_winds','Fog_or_mist', 'Dry_road','Wet_or_damp_road','Snow_
    'Motorway_road_class','A_M_road_class','A_road_road_class','B_road_class','C
    '20_mph','30_mph','40_mph','50_mph','60_mph', '70_mph']], distr='logit')
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the odds ratios
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

The odds of being in a greater severity accident by overall demographics

Checking that the multicollinearity assumption is met

In [107…

```
df_demographics_multicollinearity_test = accident_master.iloc[:,20:34]
df_demographics_multicollinearity_test_print = df_demographics_multicollinearity
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(df_demographics_multicollinearity_test_print)
```

	age_0_5	age_6_10	age_11_15	age_16_20	age_21_25	age_26_35	age_36_45	ag
age_0_5	1.000000	-0.001056	-0.001484	-0.006154	-0.002168	-0.007962	-0.002653	-0
age_6_10	-0.001056	1.000000	-0.004110	-0.012451	-0.012733	-0.014789	-0.014932	-0
age_11_15	-0.001484	-0.004110	1.000000	-0.028676	-0.034463	-0.043470	-0.032258	-0
age_16_20	-0.006154	-0.012451	-0.028676	1.000000	-0.082581	-0.120568	-0.095359	-0
age_21_25	-0.002168	-0.012733	-0.034463	-0.082581	1.000000	-0.125260	-0.105383	-0
age_26_35	-0.007962	-0.014789	-0.043470	-0.120568	-0.125260	1.000000	-0.147041	-0
age_36_45	-0.002653	-0.014932	-0.032258	-0.095359	-0.105383	-0.147041	1.000000	-0
age_46_55	-0.004544	-0.010468	-0.030171	-0.088341	-0.097436	-0.143493	-0.119745	1
age_56_65	-0.002126	-0.009995	-0.021726	-0.067737	-0.076389	-0.115097	-0.096623	-0
age_66_75	-0.000475	-0.007357	-0.014364	-0.047338	-0.059339	-0.088660	-0.070563	-0
age_75plus	-0.002514	-0.008178	-0.014408	-0.046496	-0.057181	-0.082704	-0.061671	-0
age_nodata	-0.003230	-0.008173	0.002533	-0.077773	-0.086083	-0.119090	-0.114533	-0
sex_male	0.003854	0.010928	0.042380	0.050820	0.053969	0.085371	0.080536	0
sex_female	0.002260	0.011485	-0.008457	0.017672	0.056541	0.073641	0.071944	0

Multicollinearity amongst the demographic variables is low and so all factors remain in the analysis.

OLR Model

In [108…

```
def demographics_severity_logistic_ordinal_regression():
    # Creating the model
    mod_log = OrderedModel(accident_master['accident_severity_OLR'],accident_mas
    'age_46_55','age_56_65','age_66_75','age_75plus','sex_male']], distr='logit'
    # Fitting the model and printing the summary
    res_log = mod_log.fit(method='bfgs', disp=False)
    print(res_log.summary())

    # Creating the odds ratios (i.e. the odds of being in an accident of greater
    odds_ratios = pd.DataFrame(
        {"OR": res_log.params, "Lower CI": res_log.conf_int()[0], "Upper CI": re
    odds_ratios = np.exp(odds_ratios)
    print(odds_ratios)
```

3. Project Outcome

3.1 Overview of Results

The high level spatial distribution of the data shows that the majority of accidents occurred in urban areas between 2016 and 2021. The three ONS districts with the highest number of accidents were Birmingham (11,900), Leeds (7,110) and Westminster (6,898).

People aged 26 to 35 were the most involved in accidents between 2016 and 2021. 66% of the drivers involved were males and 39% were females.

Ordinal Logistic Regression indicates that men have higher odds than women of being in an accident of greater severity (serious or fatal, rather than slight). Results also suggest that drivers aged 75 or older have the highest odds of an accident of greater severity amongst age groups.

Ordinal Logistic Regression indicates that dry road conditions produce the greatest odds of being in an accident of greater severity, whilst foggy conditions are the most impactful weather condition in increasing the odds of being in an accident of greater severity. Results further indicate that driving on roads with a 60 mph speed limit produces the greatest odds amongst all speed limit categories. Finally, the odds of being in an accident of greater severity are highest on B- class roads when compared to all other road classes.

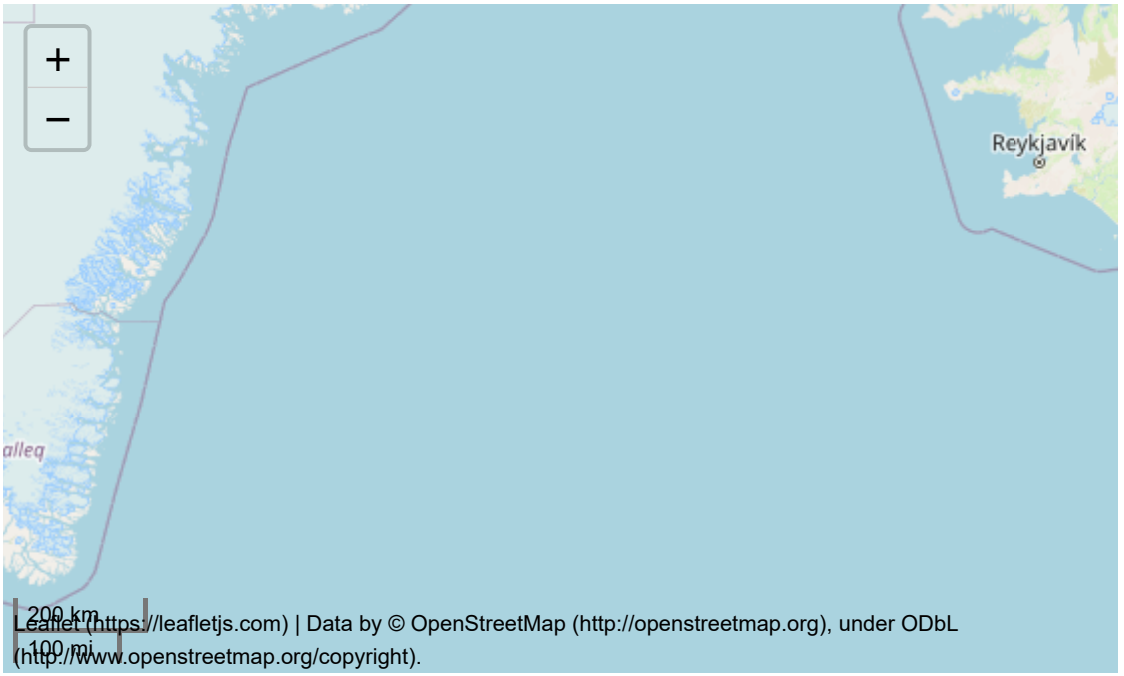
3.2 Objective 1: Exploring the spatial and temporal distribution of accidents within the UK using the Folium library

The below interactive map has been created to group accident locations into clusters using the Marker Cluster Object from the Folium library. It depicts vehicle accidents that experienced injuries (slight, serious or severe), where accidents are spatially grouped into clusters. The primary purpose of the map is to inform exploratory geospatial understanding of accident occurrence and guide the formulation of research questions. For example. Urban areas including London, Birmingham, and Manchester experience significantly higher rates of accidents than rural areas.

In [109…

```
accidents_interactive_map() #Please check the interactive map in the HTML File
```

Out[109]:



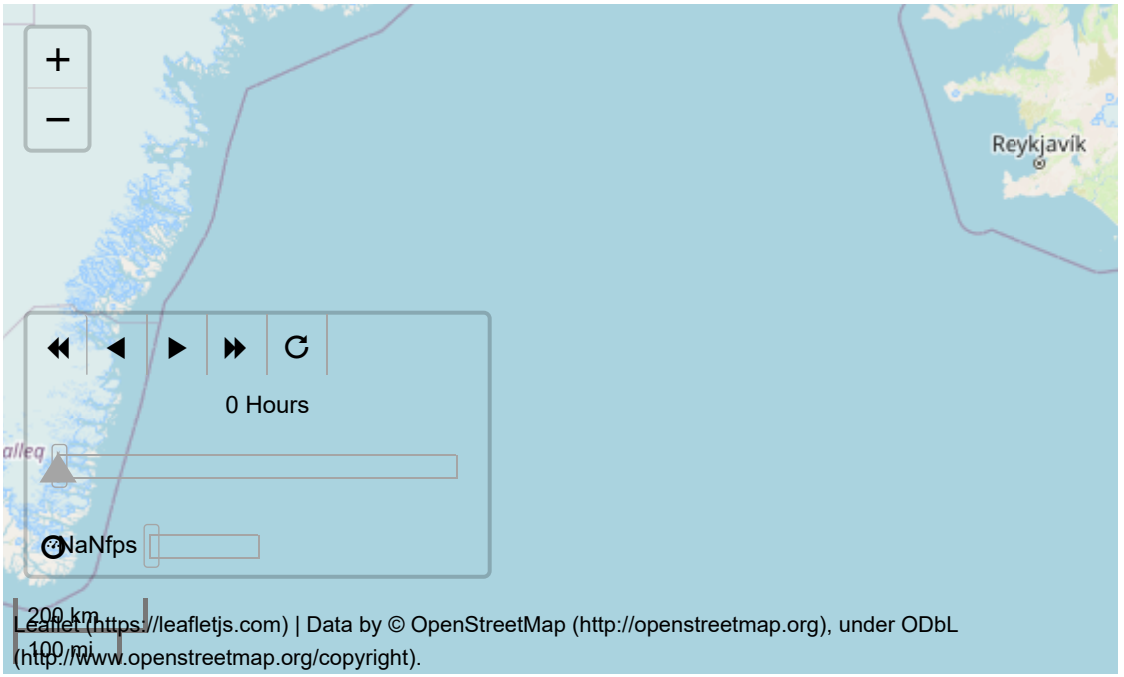
The following animated heatmap shows all accident locations in each area for a specific hour. From exploring the visualization and timeline zoomed in on Leeds by specifying the zoom level we can observe how the number of accidents increases from 07:00-08:00 hours and again from 14:00-18:00 hours. Before, in between and after these observed intervals, the heat map highlights a lower frequency of accident occurrences within the Leeds region.

The map also shows that during the night and early hours of the morning, accidents mostly occur on connecting roads between urban agglomerations. However, starting from 7:00 to 21:00, accidents occur more often within urban agglomerations.

In [110…

```
heat_interactive_map() #Please check the interactive map in the HTML File
```

Out[110]:

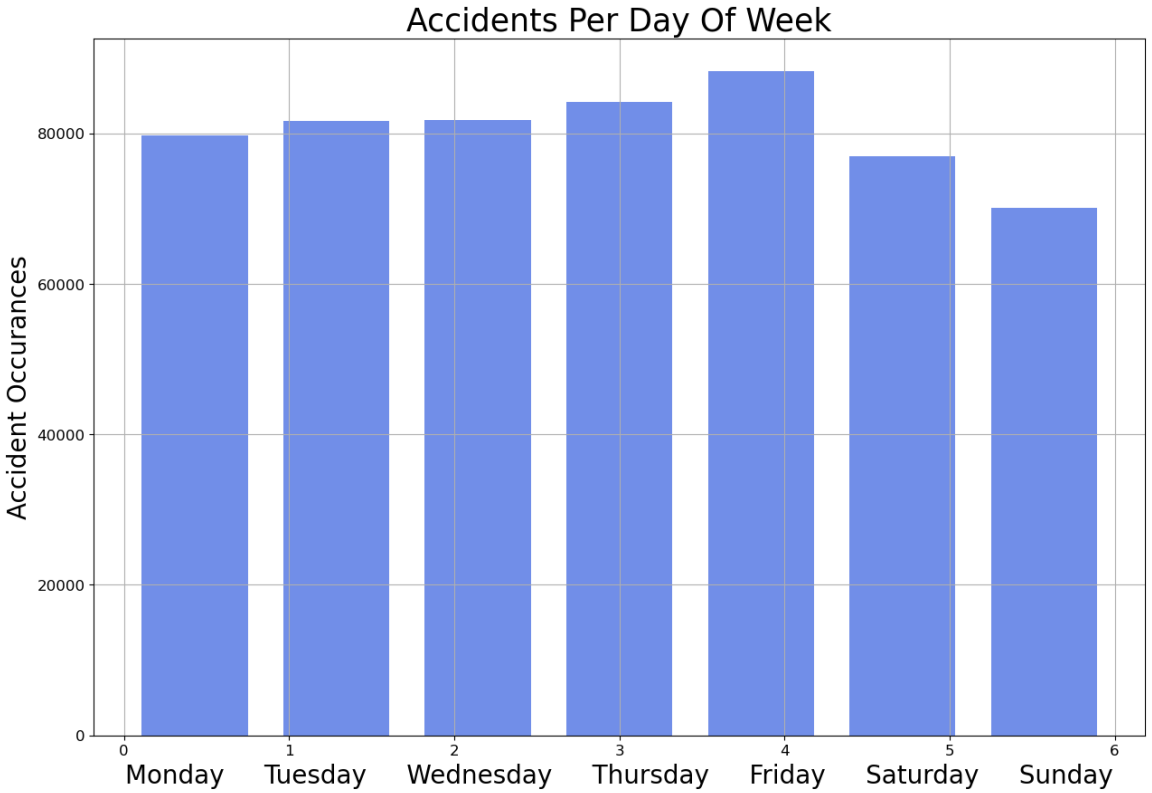


As shown in the below plot, the number of vehicle accidents in the UK decrease at the weekend. As weekdays progress, the number of observed collisions increase, with Friday experiencing the highest number of total accidents (~90,000). Conversly, accident rates at weekends are significantly reduced.

An explination for this trend is likely due to vehicle useage trends, where people use their vehicles less frequently on the weekends as workers are not required to commute to work. Conversely, Friday's may experience such high rates of accidents due to tiredness, rushing to get home, and a lack of attention to surroundings after a busy work week.

In [111…

```
accident_by_day()
```



As previously observed, Friday's typically experience the highest frequency of incidents, while Sunday's experience the fewest. The following graph demonstrates the varation of accidents occurence between Fridays and Sudnays from 2016 to 2021. As a rule, we can observe total accidents on Friday's with the red markers, while Sunday's are represented by blue markers.

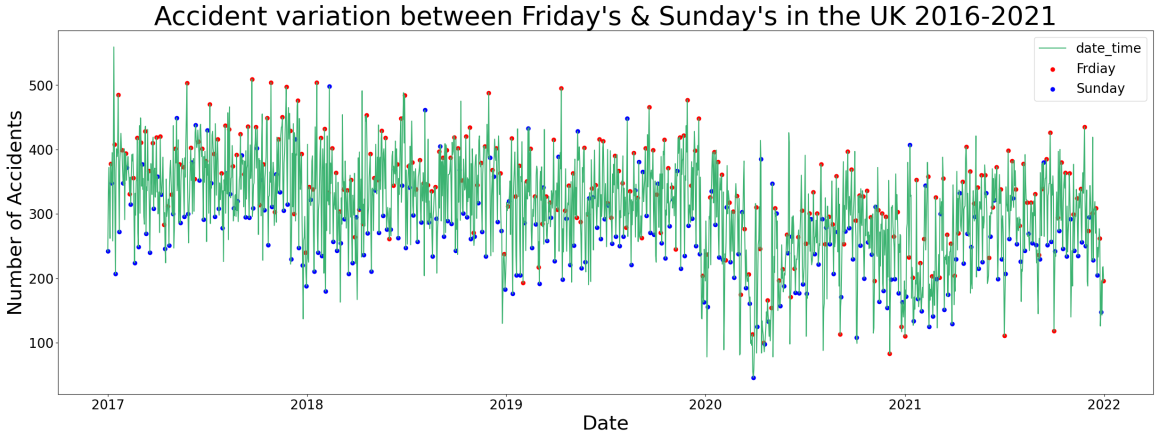
Using these two days to aid with this visualisation, we can see that there are between 200 and 500 accidents per day from 2017 to early 2020, whereas this range decreases to between 100 and 400 accidents per day from early 2020.

This visualisation is particularly interesting as we can see clear fluctuations between 2020 and 2022, which is most likely due to the encorcement of COVID-19 lockdowns within the UK,:

- 1st lockdown = March 2020-June 2020
- 2nd lockdown = November 2020
- 3rd lockdown = January 2021-March 2021

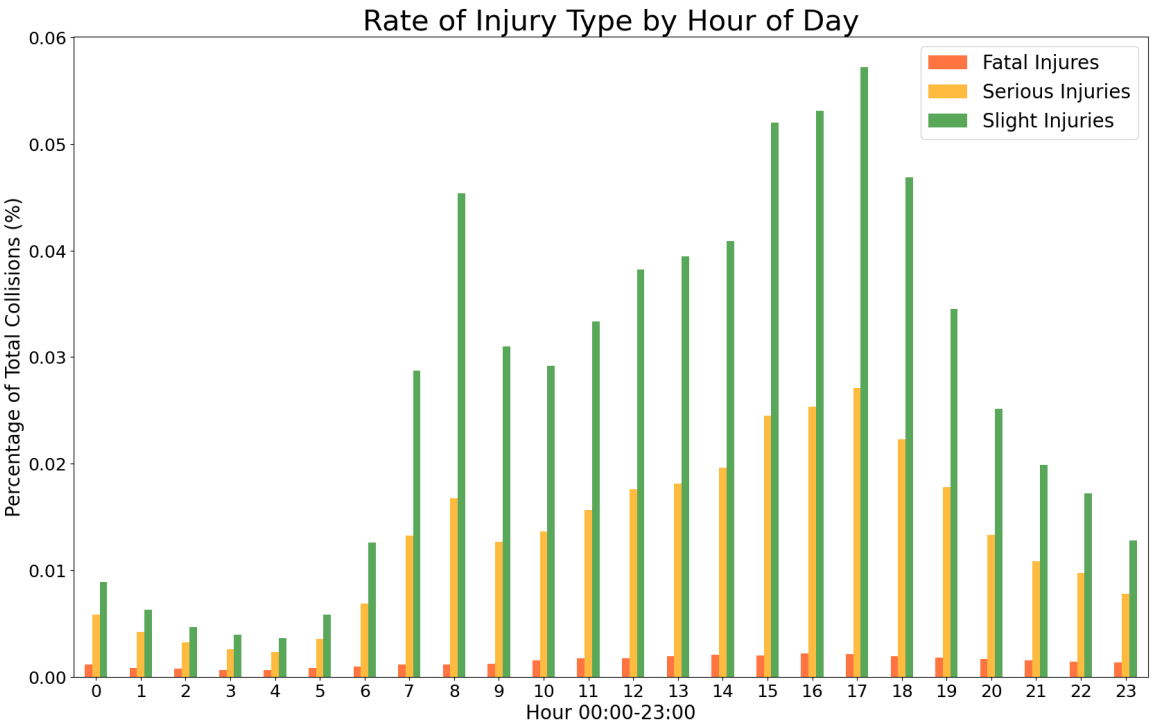
As a consequence of these lockdowns, road networks experienced significantly reduced useage as people opted to shop online, work from home and isolate from the virus.

In [112…
```
friday_sunday_accidents()
```



The following graph shows the percentage of injuries according the hour which highlights that accidents tend to be more severe in late-evenings and night. It clearly shows that accidents tend to be more severe during night and late-evening.

In [113…
```
rate_of_injury_per_hour()
```

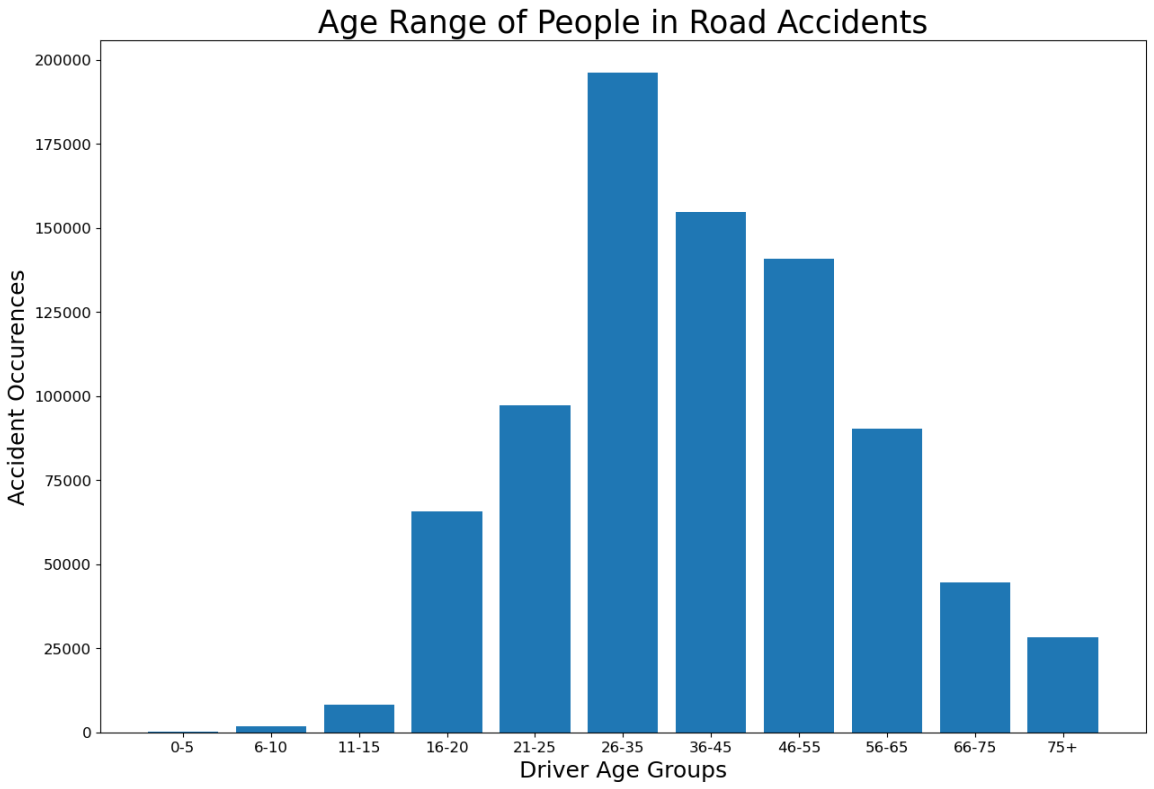


3.3 Objective 2: Using the Matplotlib package, visualise how accident occurrences vary with driving and demographic conditions

The graph below demonstrates the level of involvment of each age group in road accidents. It is clear that people aged between 26 and 35 are the most invloved in road accidents between 2016 and 2021.

In [114…

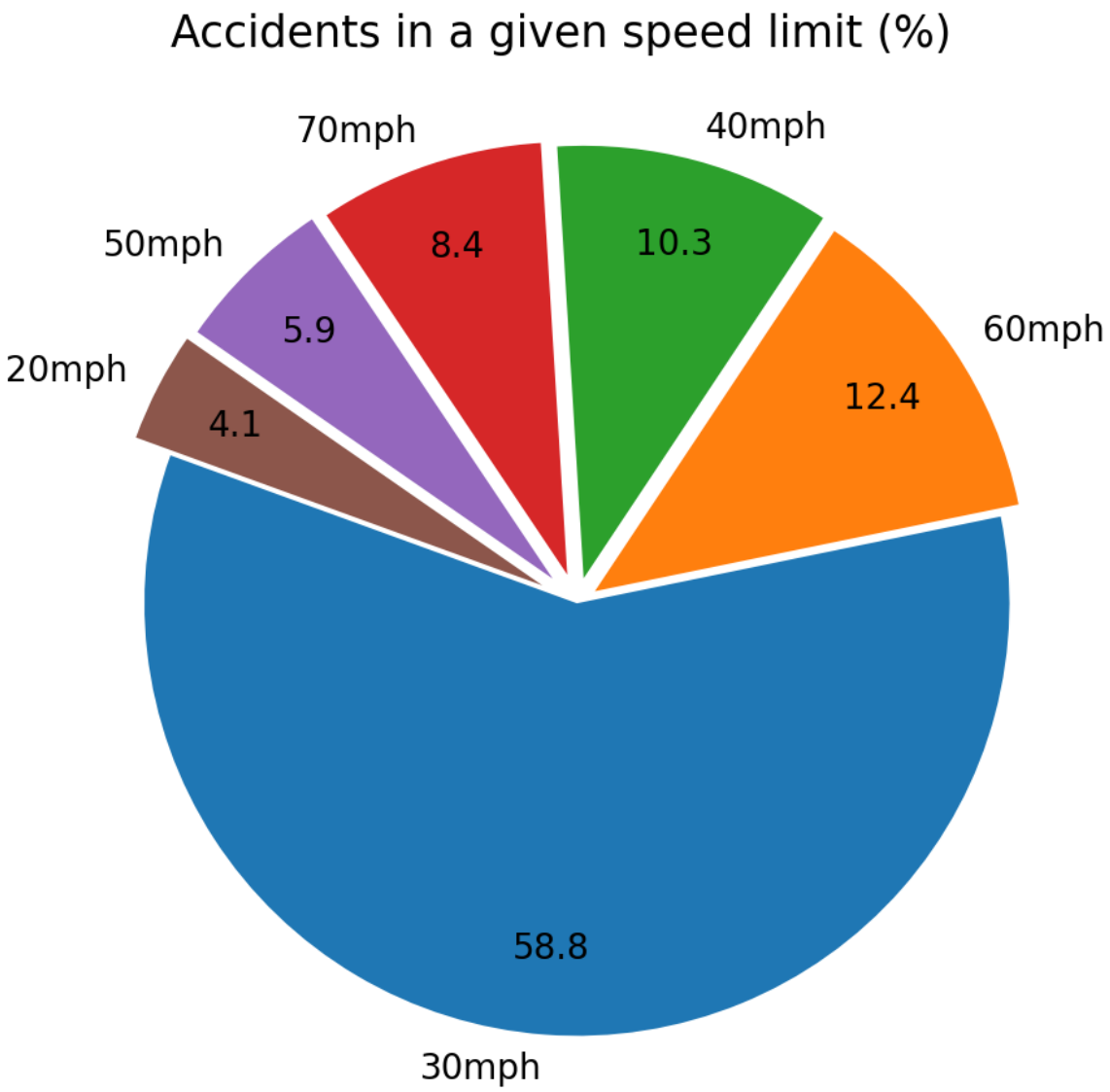
```
accident_age_range()
```



The following visualisation highlights some interesting information. As you can see, more than half of all collisions were on roads where the speed limit was 30 miles per hour (58.8%). We were expecting more collision to be on Motorways (70mph) or A-Roads (60mph).

However, this pattern does seem to make sense as zones with lower speed limits typically experience increased congestion, stop signs, changing lanes, traffic lights etc.

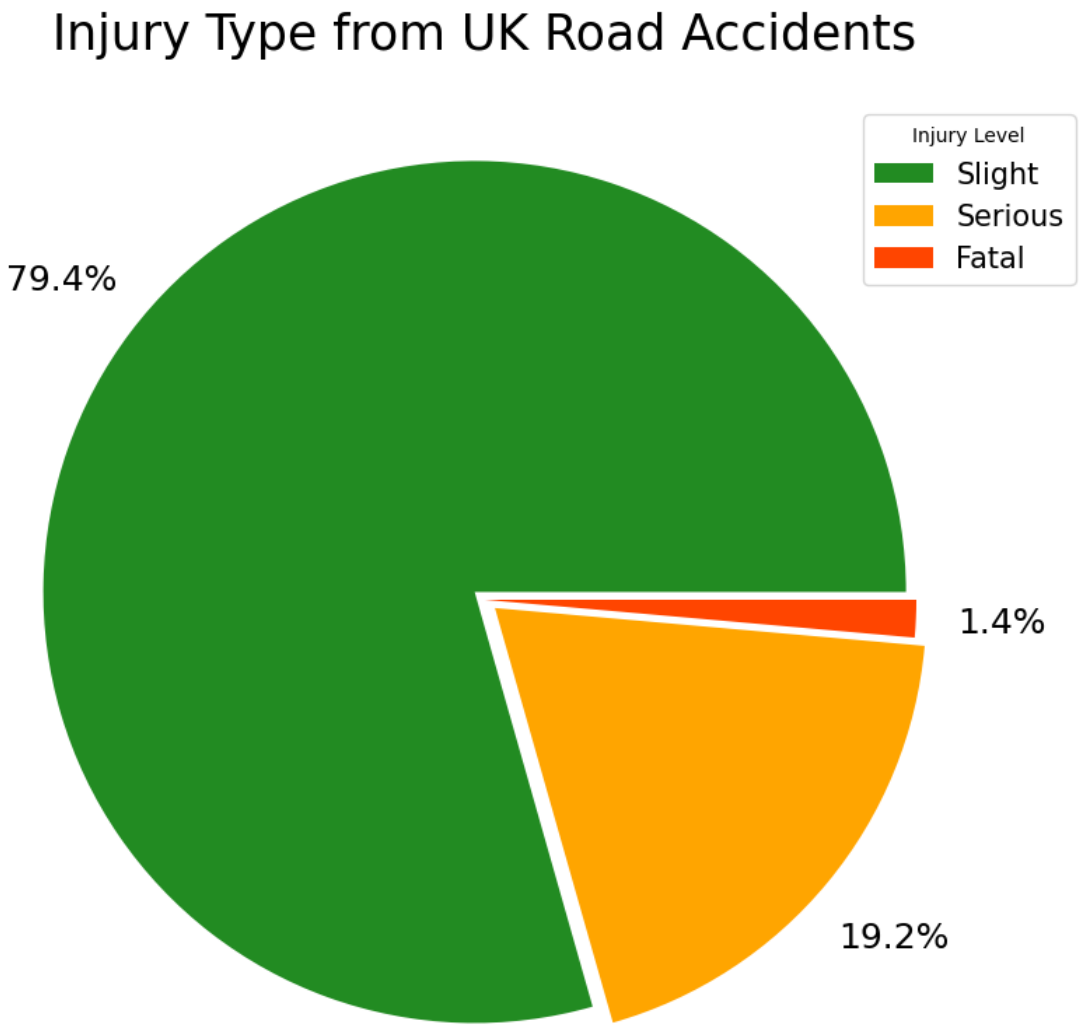
In [115…
```
accident_in_speed_limit()
```



In terms of injury severity, only 1.4% of injuries recored between 2016 and 2021 were fatal. However, the plot does highlight that 19.2% of injuries recored were serious.

Although the majority (79.4%) of injuries were slight, it would be interesting to analyse under what circumstances (date, time, location) serious injuries are more frequent. The following pie chart demostrates the percentages of each injury type.

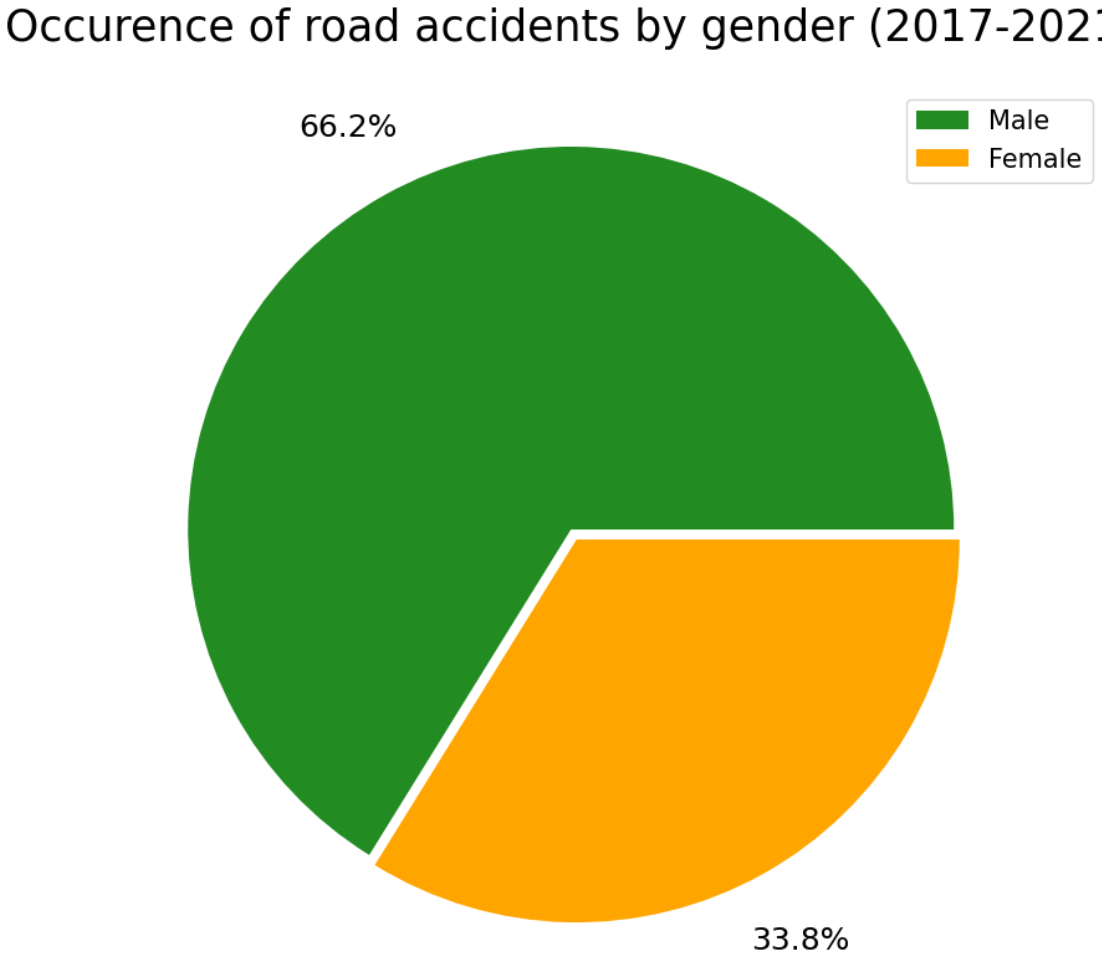
In [116…
```
injury_by_type()
```



The following graph shows the percentage of males and females involved in road accidents between 2016 and 2021. We examine the odds of being in an accident of greater severity by gender later under Objective 4.

In [117…

```
gender_accident_occurences()
```


Occurence of road accidents by gender (2017-2021)
66.2%
33.8%
Male
Female

3.4 Objective 3: Statistically assess, using the statsmodels package, how driving conditions affect the odds of being in an accident of greater severity

Ordinal Logistic Regression was used to understand how various driving conditions (light, weather, road class, speed limit) are associated with the odds of being in an accident of greater severity (serious or fatal, rather than slight). Odds are variable among speed limits, with the odds of being in an accident of greater severity at 50 mph 1.34 times higher than if travelling at 30 mph, and 1.95 times higher at 60 mph. Interestingly, the odds at 70 mph are lower than those at 60 mph, with only 1.16 times higher odds compared to 30 mph. Moreover, results indicate that the odds of a greater severity accident on a B- class road are only 1.13 times higher than an A-road, whilst motorways have 0.83 times higher odds.

Results also indicate that complete darkness increases the odds of being in an accident of greater severity by 1.94 times when compared to daylight, whilst when lights are lit, these odds fall to 1.13 times higher. Moreover, fog conditions have the largest impact of all weather conditions, making the odds an accident of greater severity 1.35 times higher than in fine and non-windy conditions.

Having ascertained how each categorical value within a driving condition category affects the odds of an accident of greater severity, OLR was further implemented to analyse the proportional impacts of road type, speed limit, weather conditions and light conditions on the odds of an accident of greater severity, whilst holding all other variables constant. The results indicate that the two most important factors are dry roads and wet roads- increasing the odds, respectively, of an accident of greater severity by 1.82 and 1.81. One potential reason for these unexpected and conflicting results may be that in dry conditions, drivers perhaps become complacent. These findings are significant at the 95% confidence level, indicating that they are statistically associated with the odds of being in an accident of greater severity.

In contrast, the lowest odds of being in an accident of greater severity are seen when travelling on a motorway (0.63), a finding that echoes the results of the road class OLR. Overall, this analysis illustrates how the odds of being in an accident of greater severity are variable both within a driving condition category, and when compared to other influential driving condition factors.

The odds of being in a greater severity accident by weather conditions

In [118…

```
weather_conditions_severity_logistic_ordinal_regression()
```

```
                                OrderedModel Results
==============================================================================
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1464e+0
Model:                      OrderedModel   AIC:                         6.293e+0
Method:             Maximum Likelihood     BIC:                         6.294e+0
Date:                   Mon, 05 Dec 2022
Time:                           06:50:17
No. Observations:                562439
Df Residuals:                    562431
Df Model:                             8
==============================================================================
                             coef    std err          z      P>|z|      [0.025
------------------------------------------------------------------------------
Raining_no_high_winds     -0.0739      0.011     -7.004      0.000      -0.095
Snowing_no_high_winds     -0.2293      0.050     -4.614      0.000      -0.327
Fine_and_high_winds        0.2313      0.030      7.598      0.000       0.172
Raining_and_high_winds     0.1859      0.029      6.338      0.000       0.128
Snowing_and_high_winds    -0.3333      0.103     -3.231      0.001      -0.535
Fog_or_mist                0.3012      0.047      6.420      0.000       0.209
1/2                        1.3462      0.004    377.519      0.000       1.339
2/3                        1.0667      0.004    280.368      0.000       1.059
==============================================================================
                               OR  Lower CI  Upper CI
Raining_no_high_winds    0.928754  0.909743  0.948163
Snowing_no_high_winds    0.795122  0.721332  0.876461
Fine_and_high_winds      1.260290  1.187279  1.337790
Raining_and_high_winds   1.204250  1.136988  1.275492
Snowing_and_high_winds   0.716568  0.585410  0.877111
Fog_or_mist              1.351520  1.232768  1.481710
1/2                      3.842754  3.815991  3.869705
2/3                      2.905877  2.884287  2.927627
```

The odds of being in a greater severity accident by speed limit

In [119…

```
speedlimit_severity_logistic_ordinal_regression()
```

```
                            OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1188e+0
5
Model:                      OrderedModel   AIC:                         6.238e+0
5
Method:             Maximum Likelihood   BIC:                         6.238e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           06:52:01
No. Observations:                562439
Df Residuals:                    562432
Df Model:                             7
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
30_mph         0.1717      0.012     14.243      0.000       0.148       0.195
40_mph         0.3797      0.016     24.116      0.000       0.349       0.411
50_mph         0.4667      0.019     24.356      0.000       0.429       0.504
60_mph         0.8375      0.014     60.329      0.000       0.810       0.865
70_mph         0.3239      0.018     18.500      0.000       0.290       0.358
1/2            1.6398      0.011    146.360      0.000       1.618       1.662
2/3            1.0713      0.004    282.493      0.000       1.064       1.079
==============================================================================
              OR  Lower CI  Upper CI
30_mph  1.187265  1.159549  1.215643
40_mph  1.461880  1.417453  1.507698
50_mph  1.594751  1.535966  1.655785
60_mph  2.310639  2.248616  2.374373
70_mph  1.382531  1.335891  1.430800
1/2     5.153911  5.041972  5.268335
2/3     2.919256  2.897637  2.941035
```

The odds of being in a greater severity accident by road class

In [120…

```
road_class_severity_logistic_ordinal_regression()
```

OrderedModel Results

Dep. Variable:	accident_severity_OLR
Model:	OrderedModel
Method:	Maximum Likelihood
Date:	Mon, 05 Dec 2022
Time:	06:53:30
No. Observations:	562439
Df Residuals:	562433
Df Model:	6
Log-Likelihood:	-3.1459e+0
AIC:	6.292e+0
BIC:	6.293e+0

	coef	std err	z	P>|z|	[0.025	0
A_road_road_class	0.0494	0.007	6.767	0.000	0.035	(
A_M_road_class	-0.0414	0.066	-0.628	0.530	-0.170	(
B_road_class	0.1579	0.011	14.871	0.000	0.137	(
C_road_class	-0.1059	0.016	-6.724	0.000	-0.137	-(
1/2	1.3857	0.005	257.682	0.000	1.375	1
2/3	1.0669	0.004	280.412	0.000	1.059	1

	OR	Lower CI	Upper CI
A_road_road_class	1.050668	1.035734	1.065817
A_M_road_class	0.959481	0.843366	1.091582
B_road_class	1.170997	1.146886	1.195614
C_road_class	0.899534	0.872198	0.927727
1/2	3.997593	3.955681	4.039950
2/3	2.906216	2.884625	2.927968

The odds of being in a greater severity accident by light conditions

In [121…

```
light_conditions_severity_logistic_ordinal_regression()
```

OrderedModel Results

Dep. Variable:	accident_severity_OLR
Model:	OrderedModel
Method:	Maximum Likelihood
Date:	Mon, 05 Dec 2022
Time:	06:54:24
No. Observations:	562439
Df Residuals:	562434
Df Model:	5
Log-Likelihood:	-3.1355e+0
AIC:	6.271e+0
BIC:	6.272e+0

	coef	std err	z	P>|z|	[0.025
Darkness_lights_lit	0.1262	0.008	15.474	0.000	0.110
Darkness_lights_unlit	0.2645	0.037	7.151	0.000	0.192
Darkness_no_lighting	0.6648	0.013	49.803	0.000	0.639
1/2	1.4168	0.004	362.126	0.000	1.409
2/3	1.0686	0.004	281.249	0.000	1.061

	OR	Lower CI	Upper CI
Darkness_lights_lit	1.134477	1.116490	1.152753
Darkness_lights_unlit	1.302770	1.211677	1.400712
Darkness_no_lighting	1.944054	1.893854	1.995584
1/2	4.123923	4.092421	4.155668
2/3	2.911402	2.889801	2.933165

The odds of being in a greater severity accident by overall driving conditions

In [122…

```
driving_conditions_severity_logistic_ordinal_regression()
```

OrderedModel Results

Dep. Variable:	accident_severity_OLR	Log-Likelihood:	-3.1001e+0
Model:	OrderedModel	AIC:	6.201e+0
Method:	Maximum Likelihood	BIC:	6.204e+0
Date:	Mon, 05 Dec 2022		
Time:	07:27:07		
No. Observations:	562439		
Df Residuals:	562411		
Df Model:	28		

	coef	std err	z	P>|z|	[0.025
Darkness_lights_lit	0.2515	0.009	29.500	0.000	0.235
Darkness_lights_unlit	0.3009	0.037	8.059	0.000	0.228
Darkness_no_lighting	0.3684	0.014	25.502	0.000	0.340
Fine_no_high_winds	0.3323	0.017	19.097	0.000	0.298
Raining_no_high_winds	0.1741	0.021	8.472	0.000	0.134
Snowing_no_high_winds	0.2402	0.063	3.788	0.000	0.116
Fine_and_high_winds	0.4752	0.035	13.613	0.000	0.407
Raining_and_high_winds	0.3724	0.034	10.831	0.000	0.305
Snowing_and_high_winds	0.0783	0.115	0.682	0.496	-0.147
Fog_or_mist	0.3012	0.051	5.942	0.000	0.202
Dry_road	0.6004	0.040	15.027	0.000	0.522
Wet_or_damp_road	0.5941	0.040	14.760	0.000	0.515
Snow_road	0.1287	0.084	1.535	0.125	-0.036
Frost_or_ice_road	0.2784	0.050	5.532	0.000	0.180
Motorway_road_class	-0.4595	0.026	-17.682	0.000	-0.510
A_M_road_class	-0.3513	0.068	-5.180	0.000	-0.484
A_road_road_class	-0.0608	0.008	-7.680	0.000	-0.076
B_road_class	0.0386	0.011	3.532	0.000	0.017
C_road_class	-0.1266	0.016	-7.921	0.000	-0.158
Urban_area	-0.2332	0.010	-23.824	0.000	-0.252
20_mph	-0.3486	0.257	-1.357	0.175	-0.852
30_mph	-0.2128	0.257	-0.829	0.407	-0.716
40_mph	-0.0811	0.257	-0.316	0.752	-0.585
50_mph	-0.0341	0.257	-0.133	0.894	-0.538
60_mph	0.2515	0.257	0.979	0.328	-0.252
70_mph	-0.0404	0.257	-0.157	0.875	-0.545
1/2	1.9875	0.259	7.659	0.000	1.479
2/3	1.0741	0.004	283.791	0.000	1.067

	OR	Lower CI	Upper CI
Darkness_lights_lit	1.285979	1.264667	1.307650
Darkness_lights_unlit	1.351110	1.255756	1.453704
Darkness_no_lighting	1.445456	1.405102	1.486970
Fine_no_high_winds	1.394186	1.347437	1.442557
Raining_no_high_winds	1.190165	1.143184	1.239076
Snowing_no_high_winds	1.271512	1.122922	1.439764
Fine_and_high_winds	1.608405	1.502034	1.722309
Raining_and_high_winds	1.451233	1.356653	1.552406
Snowing_and_high_winds	1.081395	0.863492	1.354285
Fog_or_mist	1.351416	1.223623	1.492556
Dry_road	1.822763	1.685479	1.971228
Wet_or_damp_road	1.811373	1.673967	1.960059
Snow_road	1.137346	0.964946	1.340547
Frost_or_ice_road	1.320972	1.196913	1.457891
Motorway_road_class	0.631594	0.600230	0.664597
A_M_road_class	0.703760	0.616165	0.803807
A_road_road_class	0.941009	0.926520	0.955724

B_road_class	1.039352	1.017330	1.061850
C_road_class	0.881093	0.853921	0.909129
Urban_area	0.791972	0.776921	0.807314
20_mph	0.705672	0.426547	1.167451
30_mph	0.808325	0.488786	1.336758
40_mph	0.922070	0.557264	1.525693
50_mph	0.966447	0.583710	1.600146
60_mph	1.285936	0.777196	2.127689
70_mph	0.960450	0.579942	1.590614
1/2	7.297085	4.388113	12.134477
2/3	2.927430	2.905794	2.949228

3.5 Objective 4: Statistically assess, using the statsmodels package, how demographic conditions affect the odds of being in an accident of greater severity

Ordinal Logistic Regression was also used to understand how demographic factors are associated with the odds of being in an accident of greater severity (serious or fatal, rather than slight). Results indicate that, for males, the odds of being in an accident of greater severity are 1.35 times higher than those for females. This suggests that men are more likely to be in an accident of greater severity. Comparisons of the odds of accidents of a greater severity amongst age groups yield less clear- cut patterns. When compared to drivers aged 26-35, results indicate that drivers aged 16-20 have 1.16 times higher odds of being in an accident of greater severity. However, drivers aged 21-25 have 0.97 times higher odds when compared to those aged 26-35, and drivers aged 36-45 have 0.90 times higher odds. Such decreasing odds as age bands progress may be a result of average driving experience and length accumulating, making drivers more road safe and thus decreasing the odds of being in an accident of greater severity.

Interestingly, in the next age band, 46-55, the odds of being in an accident of greater severity are 1.04 times higher than for drivers aged 26-35, while for drivers aged 56-65 these odds increase to 1.19 times higher. As age band increases, so to do the odds, with drivers aged 66-75 having odds of being in an accident of greater severity 1.29 times higher than those aged 26-35, and drivers aged 75 or over experiencing odds 1.44 times higher than drivers aged 26-35.

Thus, from the OLR results, it is clear that for new drivers (assuming aged 17- the age that one can achieve their driving licence) the odds of being in an accident of greater severity are initially high, but these reduce as age progresses until around 45 years old, before increasing again to reach a peak in the 75 or older age band. These findings contrast with the number of accident occurrences amongst age groups, as shown in …., where drivers aged 26-35 experience the greatest number of accident occurrences. As such, the results of this OLR analysis illustrate how the number of accidents does not accurately reflect the severity of accidents experienced.

OLR was further utilised to analyse the proportional impacts of driver age and sex upon the odds of an accident of greater severity, whilst holding all other variables constant. Results indicate that being a male increases the odds of being in an accident of a greater severity to the largest extent (1.42), whilst holding all other variables constant. Drivers aged 75 or over experience the second largest increase (1.30), whilst drivers aged 66-75 experience 1.16 times higher odds of being in an accident of greater severity. All results were significant at the 95% confidence interval. In summary, this analysis illustrates how the odds of being in an accident of greater severity are variable both within a demographic category, and when compared to other influential demographic factors.

The odds of being in a greater severity accident by gender

In [123…
```
gender_severity_logistic_ordinal_regression()
```

```
                             OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1415e+0
5
Model:                      OrderedModel   AIC:                         6.283e+0
5
Method:               Maximum Likelihood   BIC:                         6.283e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           07:27:24
No. Observations:                562439
Df Residuals:                    562436
Df Model:                             3
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sex_male       0.3037      0.009     34.023      0.000       0.286       0.321
1/2            1.5998      0.008    195.877      0.000       1.584       1.616
2/3            1.0673      0.004    280.661      0.000       1.060       1.075
==============================================================================
                OR  Lower CI  Upper CI
sex_male  1.354808  1.331315  1.378716
1/2       4.951825  4.873190  5.031728
2/3       2.907618  2.886026  2.929371
```

The odds of being in a greater severity accident by age

In [124…
```
age_severity_logistic_ordinal_regression()
```

```
                                OrderedModel Results
==============================================================================
=
Dep. Variable:     accident_severity_OLR   Log-Likelihood:            -3.1358e+0
5
Model:                      OrderedModel   AIC:                         6.272e+0
5
Method:             Maximum Likelihood   BIC:                         6.273e+0
5
Date:                   Mon, 05 Dec 2022
Time:                           07:29:42
No. Observations:                562439
Df Residuals:                    562430
Df Model:                             9
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
age_16_20      0.0350      0.011      3.330      0.001       0.014       0.056
age_26_35     -0.1653      0.008    -21.943      0.000      -0.180      -0.151
age_36_45     -0.1441      0.008    -18.126      0.000      -0.160      -0.128
age_46_55      0.0020      0.008      0.247      0.805      -0.014       0.018
age_56_65      0.1436      0.009     15.942      0.000       0.126       0.161
age_66_75      0.2168      0.012     18.400      0.000       0.194       0.240
age_75plus     0.3295      0.014     23.303      0.000       0.302       0.357
1/2            1.3207      0.007    191.591      0.000       1.307       1.334
2/3            1.0683      0.004    281.103      0.000       1.061       1.076
==============================================================================
                  OR  Lower CI  Upper CI
age_16_20   1.035670  1.014525  1.057255
age_26_35   0.847642  0.835219  0.860250
age_36_45   0.865832  0.852449  0.879425
age_46_55   1.001973  0.986450  1.017740
age_56_65   1.154398  1.134199  1.174956
age_66_75   1.242038  1.213690  1.271048
age_75plus  1.390220  1.352226  1.429281
1/2         3.746070  3.695798  3.797026
2/3         2.910397  2.888800  2.932157
```

The odds of being in a greater severity accident by overall demographics

In [125…

```
demographics_severity_logistic_ordinal_regression()
```

OrderedModel Results

Dep. Variable:	accident_severity_OLR	Log-Likelihood:	-3.1280e+05
Model:	OrderedModel	AIC:	6.256e+05
Method:	Maximum Likelihood	BIC:	6.257e+05
Date:	Mon, 05 Dec 2022		
Time:	07:33:08		
No. Observations:	562439		
Df Residuals:	562428		
Df Model:	11		

	coef	std err	z	P>|z|	[0.025	0.975]
age_16_20	-0.0419	0.011	-3.870	0.000	-0.063	-0.021
age_21_25	-0.1371	0.009	-14.432	0.000	-0.156	-0.118
age_26_35	-0.2412	0.008	-30.401	0.000	-0.257	-0.226
age_36_45	-0.2165	0.008	-26.121	0.000	-0.233	-0.200
age_46_55	-0.0725	0.008	-8.734	0.000	-0.089	-0.056
age_56_65	0.0708	0.009	7.618	0.000	0.053	0.089
age_66_75	0.1518	0.012	12.667	0.000	0.128	0.175
age_75plus	0.2650	0.014	18.488	0.000	0.237	0.293
sex_male	0.3516	0.009	37.729	0.000	0.333	0.370
1/2	1.4930	0.010	153.153	0.000	1.474	1.512
2/3	1.0694	0.004	281.603	0.000	1.062	1.077

	OR	Lower CI	Upper CI
age_16_20	0.958949	0.938808	0.979523
age_21_25	0.871898	0.855816	0.888282
age_26_35	0.785723	0.773601	0.798034
age_36_45	0.805304	0.792325	0.818495
age_46_55	0.930050	0.915038	0.945308
age_56_65	1.073326	1.053963	1.093046
age_66_75	1.163893	1.136879	1.191549
age_75plus	1.303435	1.267325	1.340573
sex_male	1.421352	1.395626	1.447552
1/2	4.450384	4.366160	4.536232
2/3	2.913509	2.891905	2.935275

4. Conclusion (5 marks)

4.1 Achievements

In conclusion, this project has visualised the spatial and temporal distribution of recorded accidents between 2017-2021. It has also identified trends in accident occurrence as driving conditions and demographic factors vary. Importantly, the project has also demonstrated how the odds of being in an accident of a greater severity are variable with different driving conditions and demographic considerations, and found that dry roads, complete darkness, and B- road type are the most influential factors in increasing the odds of a greater severity accident. Thus, the project successfully met the initial objectives.

4.2 Limitations

This project has various limitations, the most notable being an inability to predict the goodness-of-fit of the Ordinal Logistic Regression models produced. As OrderedModel is a relatively new addition to the statsmodel package, it does not yet have a method to conduct goodness-of-fits tests. One appropriate technique would be using a Brant test, but this could not be conducted in Python. As such, we cannot be certain as to how well the regression models perform, and so their results must be treated with a degree of caution. Moreover, OrderedModel does not have a method for assessing proportional odds (Assumption 4), meaning that confidence in the reliability of the model results is limited.

Additionally, missing data in the original datasets meant the full scope of accidents recorded could not be analysed, potentially reducing the applicability and reproducibility of the results. Importantly, the data contained the non-typical period of the COVID-19 lockdowns, the impact of which upon mobility, traffic, and accident occurrences potentially skewed our analyses.

4.3 Future Work

Future work would likely explore additional potentially interesting variables in the original dataset, such as engine capacity or deprivation index of driver, over a more substantive time period, and would incorporate road usage data to understand whether accident occurrence and severity are statistically associated with the number of vehicles on the road network. Statistical tests such as the Mann-Whitney U test would provide an insight into whether differences in median accident occurrence between each value of categorical variable are statistically significant.

Modelling techniques such as random forests may improve the accuracy and reliability of this project, while taking a classification- based approach would allow this work to be used to forecast future trends in road traffic accident occurrence and severity, potentially allowing for better traffic management and resource distribution across road networks

allowing for better traffic management and resource distribution across road networks, minimising the risk to those using the road network.