Analysis of Road Accident 2019 & 2020 Data

Ahsan Mushtaq

P2688331

Abstract

An Analysis of Road accident data of 2019 & 2020 in cities of UK provided by the Department of Transport (UK Government). In this assignment I analyse the different factors and what elements causing the accidents in the UK. How much incidents will happen in the specific region and what are the causes behind them like how many causalities will happen, how many numbers of vehicle are involved in the incident. What is the road condition also road type whether it is Motorway or highway, whether condition and what are the carriageway hazards there? These insights are illustrating geographically as well using map visualization. In 2020, the coronavirus pandemic and its accompanying travel restrictions had an impact on road safety. This report looks into the impact on reported road fatalities and shows monthly trends.

Table of Contents

Introduction to the data

Data source and License

It is open government licensed data for public sector information. We are encouraged to use and re-use the Information to get useful insights from it that is available under this licence freely and flexibly, with only a few conditions. We can copy, publish, distribute and transmit the Information. We can adapt the Information and get useful insights. We can exploit the Info data commercially and non-commercially for instance, by combining it with other Information, or by including it in your own product, system or application.

The data used in this report can be accessed from (data.gov.uk/dataset/road-safety-data 2019 & 2020). This data can be used to get useful insights that which are the reasons for accidents and make better policies for better road safety regulations.

Data Format

Department of transport released yearly CSV and txt data files on his public Gov website which can be accessed. These CSV files cannot have any personal information depicting any accident happen in the UK yearly. This csv file contain column like accident, reference, date, number of causalities, number of vehicles, type of road etc. Its categorical data type which make a file with 32 classes.

Data Scope

The data in this report covers the covers the 2019 and 2020 data in the whole UK counties, district and roads. But this data only involves which is reported by police or transport authority. There should be small number of ambiguities.

The annual statistics on road fatalities are released twice a year. Provisional results for the first release of key statistics are released in June, and final complete numbers are provided in September. In November, in-year estimates, which provide provisional data for the first half of the year, are usually released. Estimates of drunk-driving accidents and fatalities are released separately, with preliminary statistics released in February and final figures released in August.

Methodology

Files Uploading and Data Cleansing

Two CSV files Accidents 2019 and 2020 are added to HDFS Web console. In jupyter new spark context is created and important libraries are imported. RDDs were created from the uploaded file using accident csv file.

Data cleansing

After creating rdd apply some methods like count(), first(), take() and then remove header. A new rdd is created and header were removed.

```
In [3]: #creating rdd and fetching four csv file of accidents causuality e-scooter and vehicles (2021)
        # data source: https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data
        rddRoad=sc.textFile("hdfs:///user/imat5322_819715/accident*.csv").map(lambda l:l.split(','))

In [4]: #appyling some methods like count take to check rdd
        #total number of data rows
        rddRoad.count()

Out[4]: 208737

In [5]: rddRoad.take(10)

In [6]: #checking first row entries
        rddRoad.first()

In [7]: #identify header for rddRoad
        header=rddRoad.first()
        header

In [8]: display(header)

In [9]: #assigning data without header to create a new data rdd
        filtered=rddRoad.filter(lambda line:line!=header)
        filtered.count()

Out[9]: 208735

In [10]: #check first 10 row entries
         filtered.take(10)
```

Figure 1: Data Cleansing

We can see from figure 1 that there 208737 rows first and after removing header there is 208735 as there is two files.

Data Frame

In figure 2, I am going to Mapping column names to row entries and create Data frame and print the schema.

```
In [11]: #Mapping column names to row entries

         AccidentVar=filtered.map(lambda x:Row(accident_index=x[0],accident_year=x[1],\
                 accident_reference=x[2],location_easting_osgr=x[3],location_northing_osgr=x[4],\
                 longitude=x[5],latitude= x[6],police_force=x[7],accident_severity=x[8],\
                 number_of_vehicles=x[9],number_of_casualties=x[10],date=x[11],day_of_week=x[12],\
                 time=x[13],local_authority_district=x[14],\
                 local_authority_ons_district= x[15],local_authority_highway=x[16],\
                 first_road_class=x[17],first_road_number=x[18],\
                 road_type=x[19],speed_limit= x[20],junction_detail=x[21],junction_control=x[22],\
                 second_road_class=x[23],second_road_number= x[24],pedestrian_crossing_human_control=x[25],pec
                 light_conditions=x[27],weather_conditions= x[28],road_surface_conditions=x[29],special_condit
                 carriageway_hazards=x[31],urban_or_rural_area= x[32],did_police_officer_attend_scene_of_accic
                 lsoa_of_accident_location=x[35]))

In [12]: #creating dataframe
         roadframe=sqlContext.createDataFrame(AccidentVar)

In [13]: #NOTE Here it shows the data type type of all headers is String I have change the type of some of the
         #Longitude and latitude to double in line IN[41]

         #checking dataframe for schema
         roadframe.printSchema()

root
 |-- accident_index: string (nullable = true)
 |-- accident_reference: string (nullable = true)
 |-- accident_severity: string (nullable = true)
 |-- accident_year: string (nullable = true)
 |-- carriageway_hazards: string (nullable = true)
 |-- date: string (nullable = true)
 |-- day_of_week: string (nullable = true)
 |-- did_police_officer_attend_scene_of_accident: string (nullable = true)
 |-- first_road_class: string (nullable = true)
 |-- first_road_number: string (nullable = true)
 |-- junction_control: string (nullable = true)
```

Figure 2: map column, creating data frame

Data Cleaning using Drop and Use of PANDAS

In the next step we drop some column which are not necessary for example, lsoa_of_accident_location and carriageway_hazards. Then shows schema using pandas for better data table representation.

```
In [16]: #data cleaning
         #drop column
         #here drop a coulums which are not useful
         filtered_rdd = roadframe.drop("lsoa_of_accident_location","carriageway_hazards")
```

```
In [17]: filtered_rdd.printSchema()
```

```
In [18]: # filtered_rdd.show()
         #Showing my data in Panda for better representation
         pdpyr=filtered_rdd.toPandas()
         pdpyr
```

Out[18]:

	accident_index	accident_reference	accident_severity	accident_year	date	day_of_week	did_police_officer_attend_
0	2019010128300	010128300	3	2019	18/02/2019	2	
1	2019010152270	010152270	3	2019	15/01/2019	3	
2	2019010155191	010155191	3	2019	01/01/2019	3	
3	2019010155192	010155192	2	2019	01/01/2019	3	
4	2019010155194	010155194	3	2019	01/01/2019	3	
...	...	...	...	...	...	...	
208730	2020991027064	991027064	2	2020	12/08/2020	4	
208731	2020991029573	991029573	3	2020	13/11/2020	6	
208732	2020991030297	991030297	2	2020	15/04/2020	4	
208733	2020991030900	991030900	3	2020	15/12/2020	3	
208734	2020991032575	991032575	3	2020	25/08/2020	3	

208735 rows × 34 columns

```
In [19]: display(filtered_rdd)
```

```
DataFrame[accident_index: string, accident_reference: string, accident_severity: string, accident_yea
r: string, date: string, day_of_week: string, did_police_officer_attend_scene_of_accident: string, fi
rst_road_class: string, first_road_number: string, junction_control: string, junction_detail: string,
latitude: string, light_conditions: string, local_authority_district: string, local_authority_highwa
y: string, local_authority_ons_district: string, location_easting_osgr: string, location_northing_osg
r: string, longitude: string, number_of_casualties: string, number_of_vehicles: string, pedestrian_cr
ossing_human_control: string, pedestrian_crossing_physical_facilities: string, police_force: string,
```

Figure 3: Using drop and Pandas

Changing Data Type

As the data type of all the header are string so we need to change it and in figure 3.1 I will change some of the required field data type for instance, for example Longitude and Latitude to double and number of casualties and number of vehicle to integer because this fields are required for analysis in pixiedust. NOTE: As there is 32 fields so I just changing data type of four to demonstrate the process.

```
In [30]: #Convert datatype of three coulmn from string to float & integer types
roadframe=roadframe.withColumn("longitude",roadframe["longitude"].cast(DoubleType()))
roadframe=roadframe.withColumn("latitude",roadframe["latitude"].cast(DoubleType()))
roadframe=roadframe.withColumn("number_of_casualties",roadframe["number_of_casualties"].cast(IntegerType()))
roadframe=roadframe.withColumn("number_of_vehicles",roadframe["number_of_vehicles"].cast(IntegerType()))
```

```
In [31]: roadframe.printSchema()
root
 |-- junction_detail: string (nullable = true)
 |-- latitude: double (nullable = true)
 |-- light_conditions: string (nullable = true)
 |-- local_authority_district: string (nullable = true)
 |-- local_authority_highway: string (nullable = true)
 |-- local_authority_ons_district: string (nullable = true)
 |-- location_easting_osgr: string (nullable = true)
 |-- location_northing_osgr: string (nullable = true)
 |-- longitude: double (nullable = true)
 |-- lsoa_of_accident_location: string (nullable = true)
 |-- number_of_casualties: integer (nullable = true)
 |-- number_of_vehicles: integer (nullable = true)
 |-- pedestrian_crossing_human_control: string (nullable = true)
```

Figure 4: Change data type

Data Filtering and Processing

In the following figure, we can see that how we filter the Filter rows containing Accident happen in specific District Leicester 'code for Leicester is E06000016' and then count it. There is 997 accident happened in Leicester in 2019 and 2020.

Also, we are checking the incident happened in Nottingham is 1405 and it is larger then Leicester. Therefore, we can compare any of two or more district accident ratios and their causes.

In[24], As there is a lot of column almost 32, So I selected specific column for better representation and explanation of data by using select method. Shown in the figure.

```
In [20]: # Create new Dataframe
         #Filter rows containing Accident happen in District Leicester 'code for leiceter is E06000016'
         district=roadframe.filter(roadframe["local_authority_ons_district"]=="E06000016")
         district.count()
Out[20]: 997

In [21]: district.show()
         ...

In [22]: # Create new Dataframe
         #Filter rows containing Accident happen in District Nottingham 'code for leiceter is E06000018'
         district1=roadframe.filter(roadframe["local_authority_ons_district"]=="E06000018")
         district1.count()
Out[22]: 1405

In [23]: filtered_rdd.show()
         ...

In [24]: #As there is a lot of column, So I selected specific cloumn for better representation and expalanation of data

         filtered_rdd1 = roadframe.select("date","local_authority_ons_district","number_of_casualties","weather_conditions","first_road_

In [25]: filtered_rdd1.show()
```

date	local_authority_ons_district	number_of_casualties	weather_conditions	first_road_class	number_of_vehicles
18/02/2019	E09000033	3	1	3	2
15/01/2019	E09000022	1	1	3	2
01/01/2019	E09000007	1	1	4	2
01/01/2019	E09000007	1	1	4	1
01/01/2019	E09000005	2	1	3	2
01/01/2019	E09000025	3	1	5	2
01/01/2019	E09000008	1	1	3	1
01/01/2019	E09000028	5	1	6	3
01/01/2019	E09000002	1	1	3	2
01/01/2019	E09000024	1	1	4	3
01/01/2019	E09000026	1	1	3	1
01/01/2019	E09000030	1	1	3	1
01/01/2019	E09000019	1	1	3	1
01/01/2019	E09000026	1	1	3	2

Figure 5: Data Filtering and Processing

In figure 5, we are filtering some more data for instance we can compare How many accidents happen in the year of 2019 and How many accidents happen in the year of 2020. There are 91199 accident happen in year 2020 and 117536 incident in 2019 and in the following line filter with Filter with Multiple Conditions filter the number of casualties in specific district that is Leicester on specific date.

```
In [26]: # How many accident happen in the year of 2020
         year=roadframe.filter(roadframe["accident_year"]=="2020")
         year.count()

Out[26]: 91199

In [27]: # How many accident happen in the year of 2019
         year1=roadframe.filter(roadframe["accident_year"]=="2019")
         year1.count()

Out[27]: 117536

In [28]: year.select("accident_year","date","local_authority_ons_district","number_of_casualties").show

Out[28]: <bound method DataFrame.show of DataFrame[accident_year: string, date: string, local_authority_ons_district: string, number_of_
         casualties: string]>

In [29]: #Filter with Multiple Conditions
         #fiter the number of causaluties in specific district that is leicester on specific date
         multiple=roadframe.filter((roadframe["local_authority_ons_district"]=="E06000016") & (roadframe["date"]=="01/01/2019") )
         multiple.select("date","local_authority_ons_district","number_of_casualties").show()

+----------+---------------------------+--------------------+
|      date|local_authority_ons_district|number_of_casualties|
+----------+---------------------------+--------------------+
|01/01/2019|                  E06000016|                   1|
+----------+---------------------------+--------------------+
```

Figure 6: Filter data

Making data easier to access and filter by using SQL context

We can filter and access data easily by using sql context. sql queries run against the data frame depending upon the analysis and required fields. E.g. return all fields or distinct fields. Also, we can group by, sort and count the data by using sql. As In [38] we counted accident on the basis of accident severity and whether condition.

```
In [35]: #SQL Context
         sqlContext.registerDataFrameAsTable(roadframe, "MyTable")

In [36]: sqlContext.sql("select * from MyTable")

In [37]: # Return distinct list of local_authority_highway
         sqlContext.sql("SELECT distinct local_authority_highway FROM Mytable").show()
```

```
+----------------------+
|local_authority_highway|
+----------------------+
|              S12000006|
|              E08000016|
|              E06000027|
|              S12000021|
|              E06000009|
|                     -1|
|              E08000005|
|              E06000041|
|              E06000018|
|              E06000023|
|              E06000019|
|              E08000014|
|              E06000036|
|              W06000012|
|              E06000012|
|              E10000034|
```

```
In [38]: Accident_type = roadframe.groupby("Accident_Severity","weather_conditions").count().sort("count")
         Accident_type.show()
```

```
+----------------+------------------+-----+
|Accident_Severity|weather_conditions|count|
+----------------+------------------+-----+
|               1|                -1|    1|
|               1|                 6|    1|
|               1|                 3|    7|
|               2|                 6|   21|
|               1|                 7|   37|
|               1|                 8|   42|
|               1|                 9|   46|
|               1|                 4|   65|
|               1|                 5|   72|
```

Figure 7: SQL Context

Analysis by using Data Visualization.

Graphical visualization

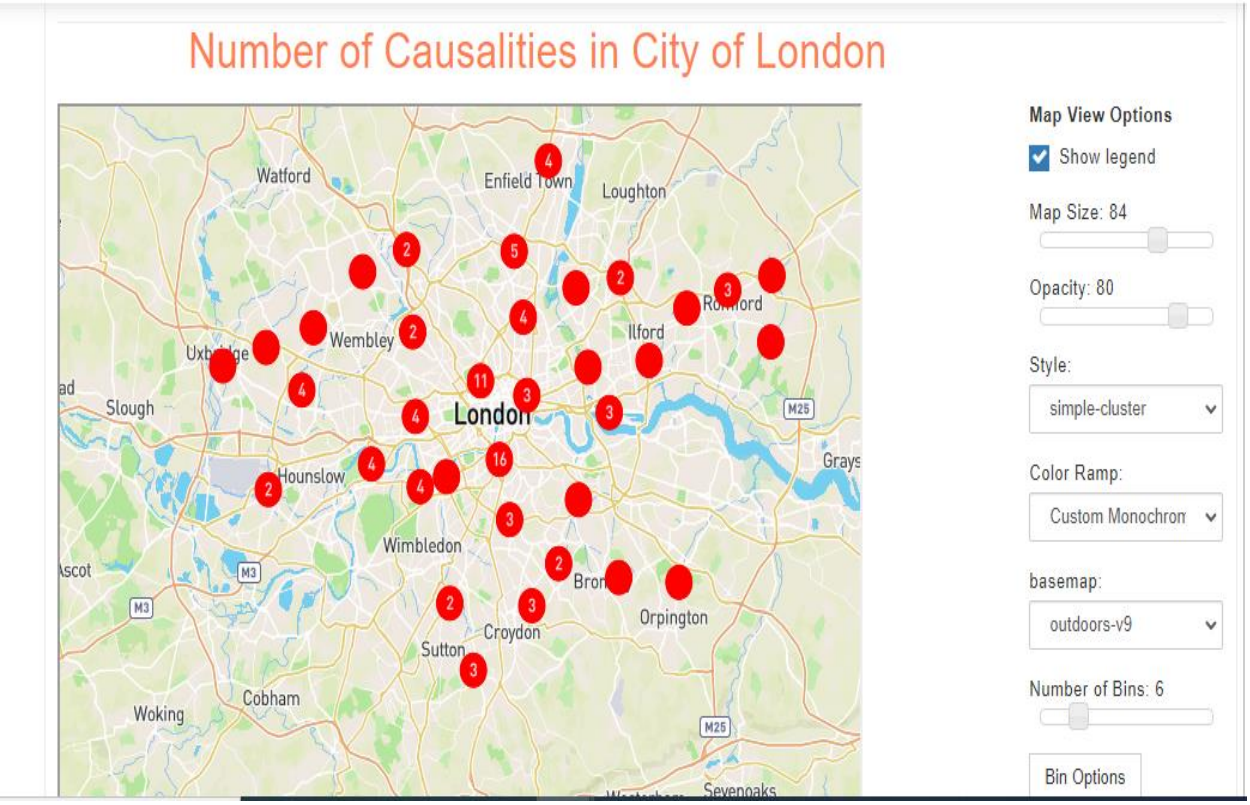

Figure 8: Number of causalities in London

In Figure 6, we can see the number of causalities by accidents in the city of London. The map clearly depicts that more incident happens resulting a greater number of causalities happen in the surroundings of city centre and as we move away from the city centre number of causalities decreases.

There were 25,341 reported collisions in London in 2019 and 2020, resulting in125 people being killed, 3,780 being seriously injured and 26,102 being slightly injured. (Transport for London – Casualties in Greater London during 2019 – Data Release).

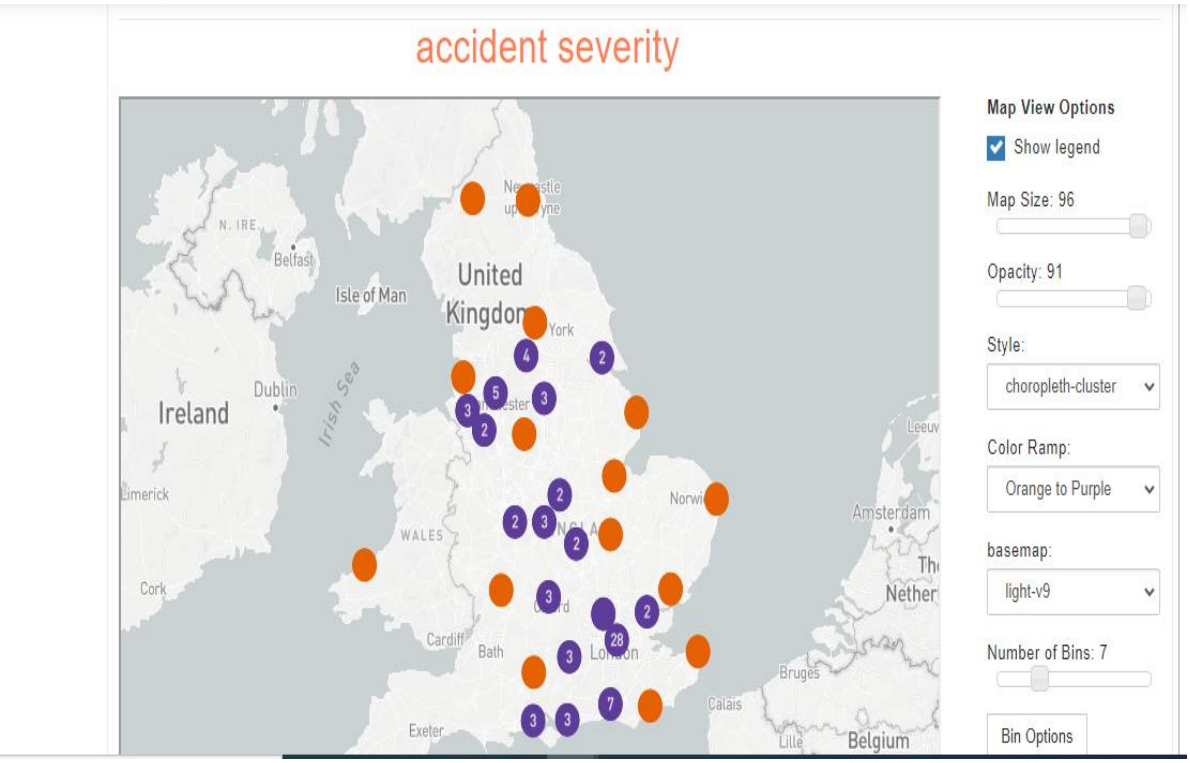

Figure 9: Accident severity

Figure 7 depicts the severity of accident in the UK. Number 1 point represents the fatal injury or accident, number 2 represents serious injury and number 3 point represents slight injury. As we can see from the map accident severity is greater in the city areas where big cities are present like London, Birmingham and Manchester etc. Also, in Greater London area the accident severity is greater than any other place. The best estimate, after adjusting for changes in reporting by police, is that there were 22,069 seriously injured and 92,054.58 slightly injured casualties, decreases of 22% and 25% since 2019, respectively (annual report).

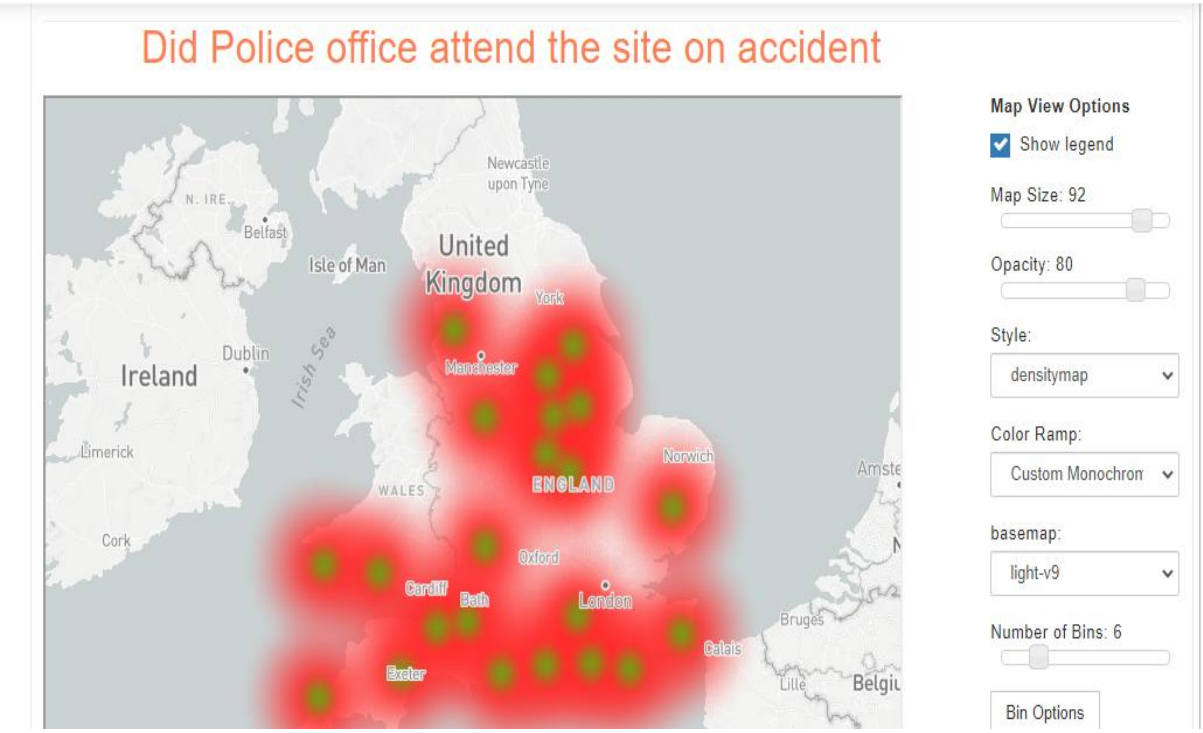

Figure 10: police attendance on the site of accident

In Figure 8 police attendance at incident site is describe. Police Service (MPS) introduced the Case Overview and Preparation Application (COPA) to report road traffic collisions (Transport for London – Casualties in Greater London during 2019 – Data Release). These systems employ a new approach of assessing the degree of injury incurred in crashes, as recommended by the Department for Transport, in which police officers record the type of injury instead of their assumptions about the severity of the injury. The recording system then assigns a number to the file. Injury severity varies depending on the type of injury. This is in contrast to the prior. Officers used a system to note whether an injury was 'slight' or 'serious' in their opinion. As a result of the implementation of these systems, more injuries have been labelled as serious rather than minor. Changes in the number of casualties are back-estimated, considering changes in the Police reporting of injury severity and self-reporting on the internet.

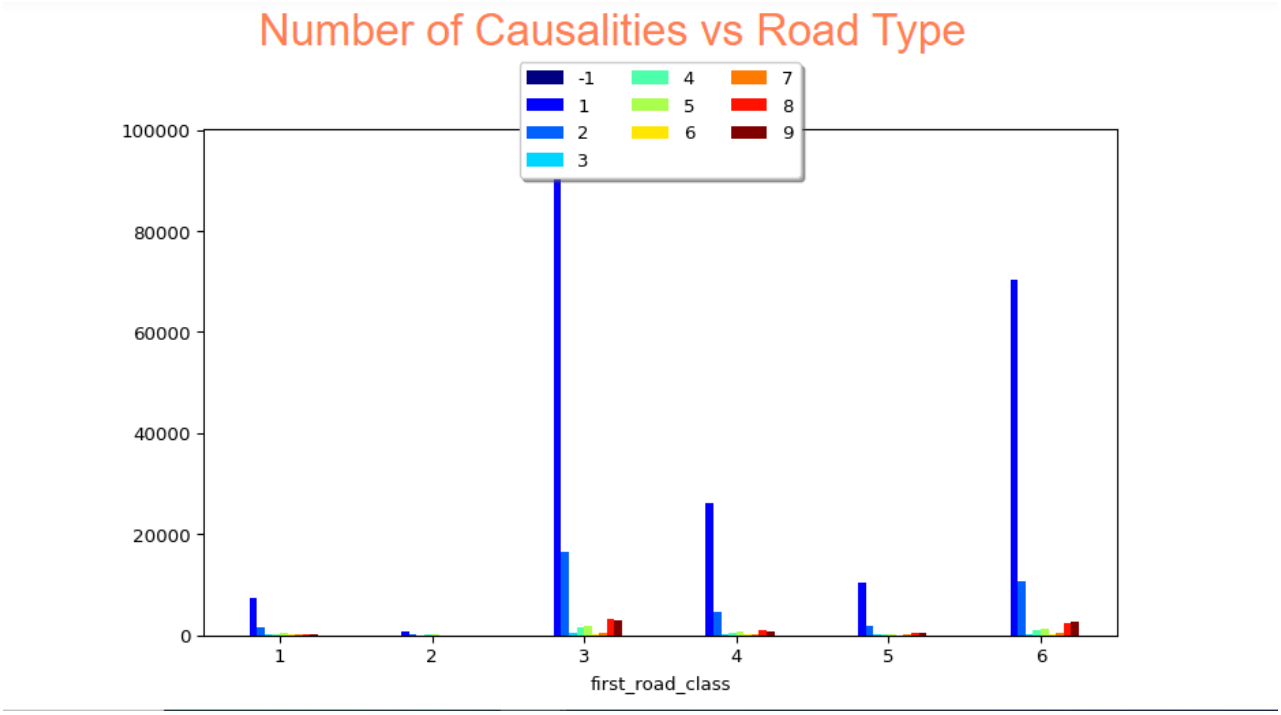

Figure 11: Number of causalities in various road type and whether

Representation of data

Weather condition

-1	Data missing or out of range
1	Fine no high winds
2	Raining no high winds
3	Snowing no high winds
4	Fine + high winds
5	Raining + high winds
6	Snowing + high winds
7	Fog or mist
8	Other
9	Unknown

Road type

1	Motorway
2	A(M)
3	A
4	B
5	C
6	Unclassified

The figure 9 illustrate that the road type 3 which is named A highway on this road most number accident happens when the weather condition is fine ad there is no high winds. And the least number of accidents happen in the A(M) first class road. Adverse weather continues to have a significant impact on the numerous accidents that occur across the UK every day of the week.

> While roads are at their most dangerous during the winter months, all types of weather conditions, including the sun, can have a significant impact on the cause of accidents. Whether it's raining or shining, the weather has the ability to cause or contribute to an automobile accident(Smith Jones -The Personal Injury Blog).

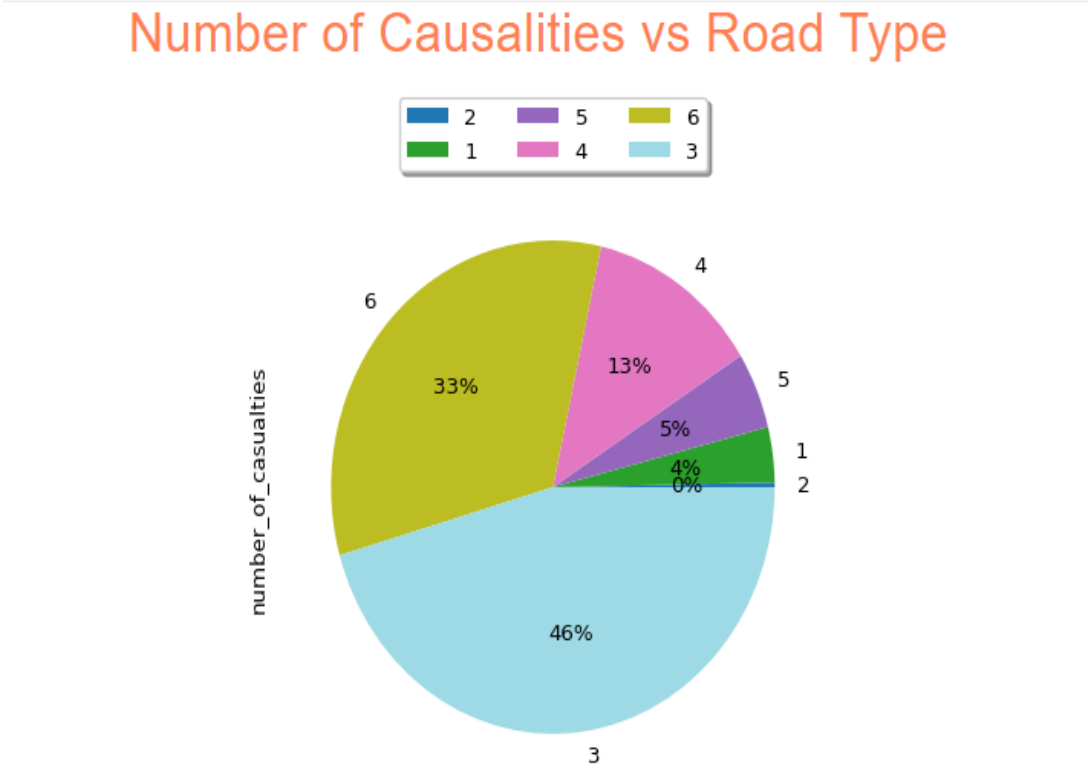

Figure 12: pie chart casualty vs road type

As you can see 46 % causalities happen at road A and 13% at road b and 33% at unclassified road so on. In 2020, the coronavirus pandemic and associated travel restrictions had an impact on road safety. This report looks into the impact on reported traffic accidents and shows monthly trends. In 2020, the coronavirus pandemic and associated travel restrictions had an impact on road safety. This report looks into the impact on reported traffic accidents and shows monthly trends.

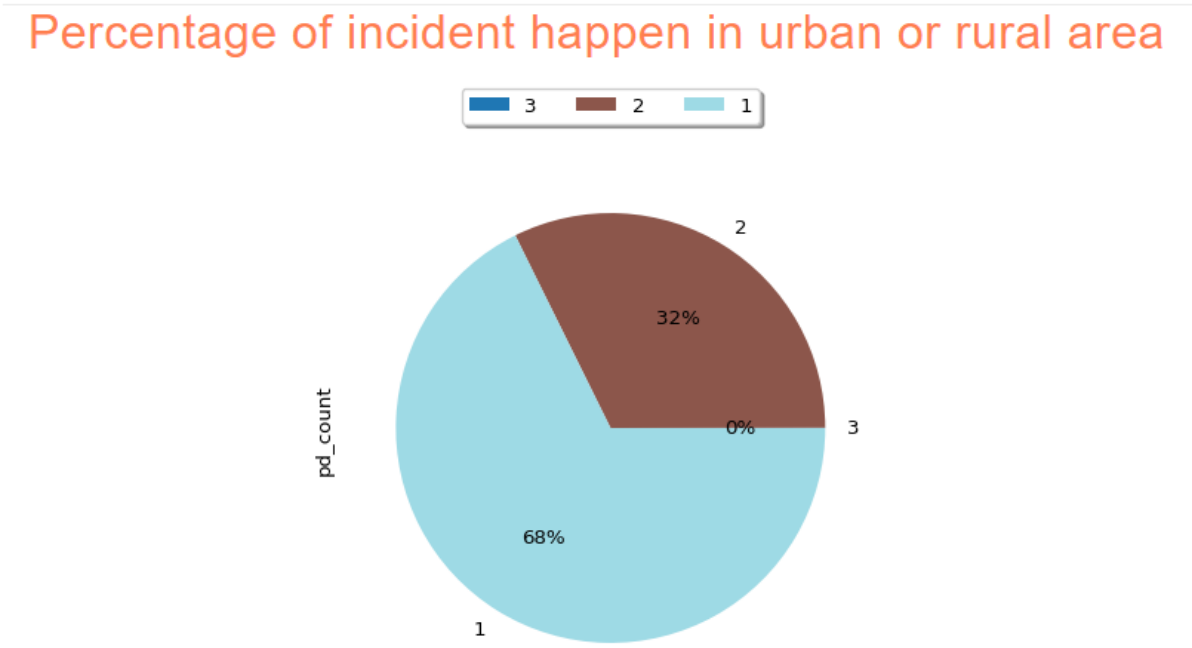

Figure 13: Percentage of incident happen in urban or rural area

Pie chart describe that 68% of accident happen in urban area 1 represent urban area and 32 % of accident happen in rural area.

Road accidents and traffic affected by lockdowns in 2020

In 2020, compared to 2019, the number of road traffic accidents in the United Kingdom decreased by 22%, from 117,536 to 91,199. This information only pertains to accidents that were reported to the police and resulted in an injury. Accidents were 65 percent lower in April, during the first nationwide lockdown, than they were in 2019. After climbing during the summer, numbers dropped during the November 2020 lockdown, but only by 30% compared to the previous year.

Conclusion:

Although the number of casualties of all types decreased in the year ending June 2020, these developments should be viewed with caution. This decrease is most likely due to a drop-in traffic volume and the coronavirus epidemic.

The figures 2020 show differences in casualty reductions by road user type (with smaller reductions for pedal cyclists and larger reductions for pedestrians) and severity (with

smaller reductions for fatalities and seriously injured casualties than slightly injured casualties) when compared to the same period in 2019.

REFRENCES:

Transport for London (2020). *Casualties in Greater London during 2019*. [online] Available at: https://content.tfl.gov.uk/casualties-in-greater-london-2019.pdf.

Anon, (2018). *How Does Adverse Weather Affect Car Accidents? | Smith Jones Solicitors*. [online] Available at: https://www.smithjonessolicitors.co.uk/blog/how-does-adverse-weather-affect-car-accidents/.

GOV.UK. (n.d.). *Reported road casualties Great Britain, annual report: 2020*. [online] Available at: https://www.gov.uk/government/statistics/reported-road-casualties-great-britain-annual-report-2020/reported-road-casualties-great-britain-annual-report-2020.

Transport, D. for (2021). *Road Safety Data*. [online] data.gov.uk. Available at: https://data.gov.uk/dataset/road-accidents-safety-data.

Baker, C. (2021). How were road accidents and traffic affected by lockdowns in 2020? *commonslibrary.parliament.uk*. [online] Available at: https://commonslibrary.parliament.uk/how-were-road-accidents-and-traffic-affected-by-lockdowns-in-2020/