The aim of this project is based on the business case scenario - We are working with the Government body of UK and we need to improve road saftey by predicting the Accident Severity based on different set of conditions as follows:

Road type

light conditions

Speed limit

Road Surface Conditions

Special Conditions at Site

Carriageway Hazards

Urban or Rural Area

Number of Vehicles

Vehicle Manoeuvre

1st Point of Impact

Day of the week

This problem would be solved by classification model as we need to predict accident severity having categorical values as fatal, serious and slight.

In [1]:
```
#importing numpy and pandas library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
#collecting data
import os
os.chdir("E:/term 2 business analytics/big data for decision making/big data coursework")
print(os.getcwd())
```

```
E:\term 2 business analytics\big data for decision making\big data coursework
```

In [3]:
```
#reading data from excel file
df=pd.read_csv(r"Road Safety Data - Accidents 2019.csv")

#output data to the screen
df
```

```
C:\Users\DELL\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3071: DtypeWarning: Columns (0,31) have mixed types.Specify dtype option on import or set low_memory=Fals
e.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Out[3]:

	Accident_Index	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	Latitude	Police_Force	Accident_Severity	Number_of_Vehicles	Number_of_Casualties	Date	...	Pedestrian_Crossing-Human_Control	Pedestrian_Physica
0	2019010128300	528218.0	180407.0	-0.153842	51.508057	1	3	2	3	18/02/2019	...	0	
1	2019010152270	530219.0	172463.0	-0.127949	51.436208	1	3	2	1	15/01/2019	...	-1	
2	2019010155191	530222.0	182543.0	-0.124193	51.526795	1	3	2	1	01/01/2019	...	0	
3	2019010155192	525531.0	184605.0	-0.191044	51.546387	1	2	1	1	01/01/2019	...	0	
4	2019010155194	524920.0	184004.0	-0.200064	51.541121	1	3	2	2	01/01/2019	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
117531	2019984106919	312635.0	573392.0	-3.368899	55.047323	98	3	1	1	18/05/2019	...	0	
117532	2019984107019	337522.0	591682.0	-2.983499	55.215407	98	3	4	1	30/05/2019	...	0	
117533	2019984107219	318544.0	567087.0	-3.274645	54.991685	98	3	2	1	21/06/2019	...	0	
117534	2019984107419	336525.0	584226.0	-2.997491	55.148292	98	3	1	1	29/06/2019	...	0	
117535	201998QC01004	291367.0	608364.0	-3.715064	55.357237	98	2	1	1	21/04/2019	...	0	

117536 rows × 32 columns

In [4]:
```
df.info()
```

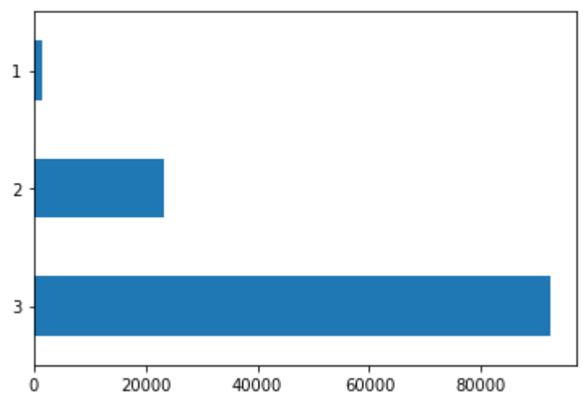
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117536 entries, 0 to 117535
Data columns (total 32 columns):
 #   Column                                      Non-Null Count   Dtype
---  ------                                      --------------   -----
 0   Accident_Index                              117536 non-null  object
 1   Location_Easting_OSGR                       117508 non-null  float64
 2   Location_Northing_OSGR                      117508 non-null  float64
 3   Longitude                                   117508 non-null  float64
 4   Latitude                                    117508 non-null  float64
 5   Police_Force                                117536 non-null  int64
 6   Accident_Severity                           117536 non-null  int64
 7   Number_of_Vehicles                          117536 non-null  int64
 8   Number_of_Casualties                        117536 non-null  int64
 9   Date                                        117536 non-null  object
 10  Day_of_Week                                 117536 non-null  int64
 11  Time                                        117473 non-null  object
 12  Local_Authority_(District)                  117536 non-null  int64
 13  Local_Authority_(Highway)                   117536 non-null  object
 14  1st_Road_Class                              117536 non-null  int64
 15  1st_Road_Number                             117536 non-null  int64
 16  Road_Type                                   117536 non-null  int64
 17  Speed_limit                                 117536 non-null  int64
 18  Junction_Detail                             117536 non-null  int64
 19  Junction_Control                            117536 non-null  int64
 20  2nd_Road_Class                              117536 non-null  int64
 21  2nd_Road_Number                             117536 non-null  int64
 22  Pedestrian_Crossing-Human_Control           117536 non-null  int64
 23  Pedestrian_Crossing-Physical_Facilities     117536 non-null  int64
 24  Light_Conditions                            117536 non-null  int64
 25  Weather_Conditions                          117536 non-null  int64
 26  Road_Surface_Conditions                     117536 non-null  int64
 27  Special_Conditions_at_Site                  117536 non-null  int64
 28  Carriageway_Hazards                         117536 non-null  int64
 29  Urban_or_Rural_Area                         117536 non-null  int64
 30  Did_Police_Officer_Attend_Scene_of_Accident  117536 non-null  int64
 31  LSOA_of_Accident_Location                   111822 non-null  object
dtypes: float64(4), int64(23), object(5)
memory usage: 28.7+ MB
```

Test Train split

In [5]:
```
#use stratisfied sampling in order to get similar distribution of accident severity type in our test as well as training data
```

In [6]:
```
df['Accident_Severity'].value_counts().plot(kind='barh') #display frequency for each type of accident severity
```

Out[6]: <AxesSubplot:>

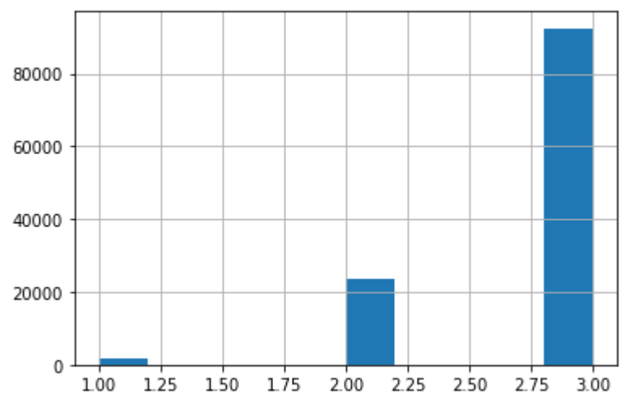


In [7]:
```
df['Accident_Severity'].value_counts() #showing number of records of each category of accident severity
```

Out[7]:
```
3    92456
2    23422
1     1658
Name: Accident_Severity, dtype: int64
```

In [8]:
```
df["Accident_Severity"].hist()
```

Out[8]: <AxesSubplot:>



So we go with the one with least number of records i.e 1658. This means we need to perform stratisfied sampling for equal proportions of all the three categories.

In [9]:
```
#creating categorical variable holding accident severity types

df["Accident_Severity_cat"]=pd.cut(df["Accident_Severity"], bins=[0,1,2,3], labels=[1,2,3])
```

In [10]:
```
df["Accident_Severity_cat"].dtype
```

Out[10]: CategoricalDtype(categories=[1, 2, 3], ordered=True)

In [11]:
```
from sklearn.model_selection import StratifiedShuffleSplit
Stratisfied_splitter= StratifiedShuffleSplit(n_splits=1, test_size=0.3, random_state=123)

train_index,test_index = list(Stratisfied_splitter.split(df, df["Accident_Severity_cat"]))[0]

strat_train_set = df.loc[train_index]
strat_test_set = df.loc[test_index]
```

In [12]:
```
def accident_Severity_cat_proportions(data):
    return data["Accident_Severity_cat"].value_counts() / len(data)

from sklearn.model_selection import train_test_split

# create a random split
rand_train_set, rand_test_set = train_test_split(df, test_size=0.3, random_state=123)

# create a temporary dataframe for easy visualization
df_tmp = pd.DataFrame({
    "Overall": accident_Severity_cat_proportions(df),
    "Random test set": accident_Severity_cat_proportions(rand_test_set),
    "Stratified test set": accident_Severity_cat_proportions(strat_test_set),
    "Stratified train set": accident_Severity_cat_proportions(strat_train_set),

}).sort_index()

# add two columns for the percent of the difference to the overall proportion

df_tmp["Rand. %error"] = (df_tmp["Random test set"] - df_tmp["Overall"])/ df_tmp["Overall"] * 100

df_tmp["Strat. %error"] = (df_tmp["Stratified test set"] - df_tmp["Overall"])/ df_tmp["Overall"] * 100

#df_tmp["Strat. %errortrain"] = (df_tmp["Stratified train set"] - df_tmp["Overall"])/ df_tmp["Overall"] * 100


df_tmp
```

Out[12]:

	Overall	Random test set	Stratified test set	Stratified train set	Rand. %error	Strat. %error
1	0.014106	0.013386	0.014095	0.014111	-5.107092	-0.080985
2	0.199275	0.201412	0.199285	0.199271	1.072492	0.005125
3	0.786619	0.785202	0.786620	0.786618	-0.180111	0.000154

In [13]:
```
del strat_train_set["Accident_Severity_cat"]
del strat_test_set["Accident_Severity_cat"]
```

In [14]:
```
trainset = strat_train_set  #renaming to shorter variables
testset = strat_test_set
```

In [15]:
```
trainset
```

Out[15]:

	Accident_Index	Location_Easting_OSGR	Location_Northing_OSGR	Longitude	Latitude	Police_Force	Accident_Severity	Number_of_Vehicles	Number_of_Casualties	Date	...	Pedestrian_Crossing-Human_Control	Pedestrian_Physical
103938	2019522001185	358929.0	178418.0	-2.593113	51.503193	52	3	2	1	09/11/2019	...	0	
111014	201963A101619	259840.0	244590.0	-4.046854	52.081882	63	2	2	2	29/10/2019	...	0	
17696	2019010207283	532646.0	166141.0	-0.095416	51.378831	1	3	2	3	23/09/2019	...	0	
90915	2019460853693	577019.0	167436.0	0.542224	51.378375	46	3	1	1	03/06/2019	...	0	
73451	2019400859184	494295.0	224478.0	-0.630561	51.910873	40	3	2	2	13/07/2019	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	361002.0	387140.0	-2.587707	53.379678	7	1	2	1	08/01/2019	...	0	
83134	2019440064734	460620.0	101588.0	-1.140964	50.810670	44	2	1	1	22/02/2019	...	0	
102425	2019521902482	361991.0	170704.0	-2.548167	51.434050	52	2	2	1	08/03/2019	...	0	
6719	2019010175779	533726.0	182319.0	-0.073796	51.523963	1	2	2	1	18/04/2019	...	0	
6996	2019010176615	526196.0	182692.0	-0.182144	51.529047	1	3	2	1	23/04/2019	...	0	

82275 rows × 32 columns

Explorartory Data Analysis

We will perform EDA on the training part of the data only after the split as using information from EDA to decide which model to use, to tweak parameters, and so forth is part of the training process and hence should not be allowed access to test data

In [16]:
```
trainset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82275 entries, 103938 to 6996
Data columns (total 32 columns):
 #   Column                                       Non-Null Count  Dtype
---  ------                                       --------------  -----
 0   Accident_Index                               82275 non-null  object
 1   Location_Easting_OSGR                        82255 non-null  float64
 2   Location_Northing_OSGR                       82255 non-null  float64
 3   Longitude                                    82255 non-null  float64
 4   Latitude                                     82255 non-null  float64
 5   Police_Force                                 82275 non-null  int64
 6   Accident_Severity                            82275 non-null  int64
 7   Number_of_Vehicles                           82275 non-null  int64
 8   Number_of_Casualties                         82275 non-null  int64
 9   Date                                         82275 non-null  object
 10  Day_of_Week                                  82275 non-null  int64
 11  Time                                         82232 non-null  object
 12  Local_Authority_(District)                   82275 non-null  int64
 13  Local_Authority_(Highway)                    82275 non-null  object
 14  1st_Road_Class                               82275 non-null  int64
 15  1st_Road_Number                              82275 non-null  int64
 16  Road_Type                                    82275 non-null  int64
 17  Speed_limit                                  82275 non-null  int64
 18  Junction_Detail                              82275 non-null  int64
 19  Junction_Control                             82275 non-null  int64
 20  2nd_Road_Class                               82275 non-null  int64
 21  2nd_Road_Number                              82275 non-null  int64
 22  Pedestrian_Crossing-Human_Control            82275 non-null  int64
 23  Pedestrian_Crossing-Physical_Facilities      82275 non-null  int64
 24  Light_Conditions                             82275 non-null  int64
 25  Weather_Conditions                           82275 non-null  int64
 26  Road_Surface_Conditions                      82275 non-null  int64
 27  Special_Conditions_at_Site                   82275 non-null  int64
 28  Carriageway_Hazards                          82275 non-null  int64
 29  Urban_or_Rural_Area                          82275 non-null  int64
 30  Did_Police_Officer_Attend_Scene_of_Accident  82275 non-null  int64
 31  LSOA_of_Accident_Location                    78264 non-null  object
dtypes: float64(4), int64(23), object(5)
memory usage: 20.7+ MB
```

In [17]:
```
#specifying the columns of interest according to our business case scenario into the column variable
columns = ['Accident_Index', 'Accident_Severity','Day_of_Week','Road_Type','Light_Conditions','Speed_limit','Road_Surface_Conditions','Special_Conditions_at_Site','Carriageway Hazar
df_train = pd.DataFrame(trainset, columns=columns)

df_train
```

Out[17]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles
103938	2019522001185	3	7	6	4	30	2	0	NaN	1	2
111014	201963A101619	2	3	6	1	60	2	0	NaN	2	2
17696	2019010207283	3	2	6	1	20	1	0	NaN	1	2
90915	2019460853693	3	2	6	1	20	1	0	NaN	1	1
73451	2019400859184	3	7	6	1	30	1	0	NaN	2	2
...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	1	3	6	1	30	1	0	NaN	1	2
83134	2019440064734	2	6	6	1	30	1	0	NaN	1	1
102425	2019521902482	2	6	3	1	20	2	0	NaN	1	2
6719	2019010175779	2	5	6	1	20	1	0	NaN	1	2
6996	2019010176615	3	3	9	1	20	5	0	NaN	1	2

82275 rows × 11 columns

Replace columns to contain actual nominal values from left to right

Replacing the values for accident severity

In [18]:
```
replacement_accident_severity = {
    1: "Fatal",
    2: "Serious",
    3: "Slight",
}
df_train['Accident_Severity'] = df_train['Accident_Severity'].map(replacement_accident_severity)

df_train
```

Out[18]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles
103938	2019522001185	Slight	7	6	4	30	2	0	NaN	1	2
111014	201963A101619	Serious	3	6	1	60	2	0	NaN	2	2
17696	2019010207283	Slight	2	6	1	20	1	0	NaN	1	2
90915	2019460853693	Slight	2	6	1	20	1	0	NaN	1	1
73451	2019400859184	Slight	7	6	1	30	1	0	NaN	2	2
...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	3	6	1	30	1	0	NaN	1	2
83134	2019440064734	Serious	6	6	1	30	1	0	NaN	1	1
102425	2019521902482	Serious	6	3	1	20	2	0	NaN	1	2
6719	2019010175779	Serious	5	6	1	20	1	0	NaN	1	2
6996	2019010176615	Slight	3	9	1	20	5	0	NaN	1	2

82275 rows × 11 columns

In [19]:
```
#### Replacing the values for days of the week
```

In [20]:
```
replacement_Day_of_Week = {
  1: "Sunday",
2: "Monday",
3:"Tuesday",
4:"Wednesday",
5:"Thursday",
6:"Friday",
7:"Saturday"
}

df_train['Day_of_Week'] = df_train['Day_of_Week'].map(replacement_Day_of_Week)

df_train
```

Out[20]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles
103938	2019522001185	Slight	Saturday	6	4	30	2	0	NaN	1	2
111014	201963A101619	Serious	Tuesday	6	1	60	2	0	NaN	2	2
17696	2019010207283	Slight	Monday	6	1	20	1	0	NaN	1	2
90915	2019460853693	Slight	Monday	6	1	20	1	0	NaN	1	1
73451	2019400859184	Slight	Saturday	6	1	30	1	0	NaN	2	2
...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	6	1	30	1	0	NaN	1	2
83134	2019440064734	Serious	Friday	6	1	30	1	0	NaN	1	1
102425	2019521902482	Serious	Friday	3	1	20	2	0	NaN	1	2
6719	2019010175779	Serious	Thursday	6	1	20	1	0	NaN	1	2
6996	2019010176615	Slight	Tuesday	9	1	20	5	0	NaN	1	2

82275 rows × 11 columns

In [21]:
```
#### Replacing the values for road type
```

In [22]:
```
replacement_Road_Type = {
1:"Roundabout",
2:"One way street",
3:"Dual carriageway",
6:"Single carriageway",
7:"Slip road",
9:"Unknown",
12:"One way street/Slip road",
-1:"Data missing or out of range"
}

df_train['Road_Type'] = df_train['Road_Type'].map(replacement_Road_Type)
df_train['Road_Type'] = df_train['Road_Type'].replace("Data missing or out of range", np.NaN)
df_train
```

Out[22]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles
103938	2019522001185	Slight	Saturday	Single carriageway	4	30	2	0	NaN	1	2
111014	201963A101619	Serious	Tuesday	Single carriageway	1	60	2	0	NaN	2	2
17696	2019010207283	Slight	Monday	Single carriageway	1	20	1	0	NaN	1	2
90915	2019460853693	Slight	Monday	Single carriageway	1	20	1	0	NaN	1	1
73451	2019400859184	Slight	Saturday	Single carriageway	1	30	1	0	NaN	2	2
...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	Single carriageway	1	30	1	0	NaN	1	2
83134	2019440064734	Serious	Friday	Single carriageway	1	30	1	0	NaN	1	1
102425	2019521902482	Serious	Friday	Dual carriageway	1	20	2	0	NaN	1	2
6719	2019010175779	Serious	Thursday	Single carriageway	1	20	1	0	NaN	1	2
6996	2019010176615	Slight	Tuesday	Unknown	1	20	5	0	NaN	1	2

82275 rows × 11 columns

In [23]:
```
#### Replacing the values for light conditions
```

In [24]:
```
replacement_Light_Conditions = {
1:"Daylight",
4:"Darkness - lights lit",
5:"Darkness - lights unlit",
6:"Darkness - no lighting",
7:"Darkness - lighting unknown",
-1:"Data missing or out of range"
}

df_train['Light_Conditions'] = df_train['Light_Conditions'].map(replacement_Light_Conditions)

df_train
```

Out[24]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles
103938	2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	2	0	NaN	1	2
111014	201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	2	0	NaN	2	2
17696	2019010207283	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	2
90915	2019460853693	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	1
73451	2019400859184	Slight	Saturday	Single carriageway	Daylight	30	1	0	NaN	2	2
...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	1	0	NaN	1	2
83134	2019440064734	Serious	Friday	Single carriageway	Daylight	30	1	0	NaN	1	1
102425	2019521902482	Serious	Friday	Dual carriageway	Daylight	20	2	0	NaN	1	2
6719	2019010175779	Serious	Thursday	Single carriageway	Daylight	20	1	0	NaN	1	2
6996	2019010176615	Slight	Tuesday	Unknown	Daylight	20	5	0	NaN	1	2

82275 rows × 11 columns

In [25]:
```
#df_train['Light_Conditions'].unique()
```

In [26]:
```
#create a column daylight which holds true for daylight and false otherwise
#this means it will hold false for the 'Data missing or out of range' as well
def func(row):
    if ("Darkness" not in row['Light_Conditions'] and
        "Daylight" in row['Light_Conditions']):
        return True
    else:
        return False

df_train["Daylight"] = df_train.apply(func, axis=1)

df_train
```

Out[26]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight
103938	2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	2	0	NaN	1	2	False
111014	201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	2	0	NaN	2	2	True
17696	2019010207283	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	2	True
90915	2019460853693	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	1	True
73451	2019400859184	Slight	Saturday	Single carriageway	Daylight	30	1	0	NaN	2	2	True
...	...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	1	0	NaN	1	2	True
83134	2019440064734	Serious	Friday	Single carriageway	Daylight	30	1	0	NaN	1	1	True
102425	2019521902482	Serious	Friday	Dual carriageway	Daylight	20	2	0	NaN	1	2	True
6719	2019010175779	Serious	Thursday	Single carriageway	Daylight	20	1	0	NaN	1	2	True
6996	2019010176615	Slight	Tuesday	Unknown	Daylight	20	5	0	NaN	1	2	True

82275 rows × 12 columns

In [27]:
```
def light_status(row):
    if ("lit" in row['Light_Conditions'] and "unlit" not in row['Light_Conditions']):
        return "lit"
    elif ("unlit" in row['Light_Conditions']):
        return "unlit"
    elif ("no lighting" in row['Light_Conditions']):
        return "no lighting"
    else:
        return(np.NaN)

df_train["light_status"] = df_train.apply(light_status, axis=1)

df_train
```

Out[27]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	lig
103938	2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	2	0	NaN	1	2	False	
111014	201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	2	0	NaN	2	2	True	
17696	2019010207283	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	2	True	
90915	2019460853693	Slight	Monday	Single carriageway	Daylight	20	1	0	NaN	1	1	True	
73451	2019400859184	Slight	Saturday	Single carriageway	Daylight	30	1	0	NaN	2	2	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	1	0	NaN	1	2	True	
83134	2019440064734	Serious	Friday	Single carriageway	Daylight	30	1	0	NaN	1	1	True	
102425	2019521902482	Serious	Friday	Dual carriageway	Daylight	20	2	0	NaN	1	2	True	
6719	2019010175779	Serious	Thursday	Single carriageway	Daylight	20	1	0	NaN	1	2	True	
6996	2019010176615	Slight	Tuesday	Unknown	Daylight	20	5	0	NaN	1	2	True	

82275 rows × 13 columns

This is done to check for value as false for daylight and NaN for light_status in two conditions Darkness-light unknown and Data missing out of range - we can consider removing these values during Data cleaning and transformation step

In [28]:
```
df_train.loc[(df_train['Daylight'] == False) & df_train['light_status'].isnull()]
```

Out[28]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	lig
94807	2019470879232	Slight	Monday	Single carriageway	Darkness - lighting unknown	20	1	0	NaN	1	2	False	
24535	2019010227069	Slight	Thursday	Dual carriageway	Darkness - lighting unknown	40	2	-1	NaN	1	2	False	
20393	2019010215190	Slight	Thursday	Roundabout	Darkness - lighting unknown	30	1	0	NaN	1	2	False	
74322	2019410814635	Slight	Friday	Single carriageway	Darkness - lighting unknown	30	1	0	NaN	1	1	False	
38465	2019100899204	Slight	Wednesday	Dual carriageway	Darkness - lighting unknown	40	1	0	NaN	1	1	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
55756	2019210841634	Slight	Thursday	Single carriageway	Darkness - lighting unknown	50	1	0	NaN	2	2	False	
102218	2019521901041	Slight	Saturday	Single carriageway	Darkness - lighting unknown	20	1	0	NaN	1	2	False	
39516	2019110896945	Serious	Friday	Single carriageway	Darkness - lighting unknown	30	2	0	NaN	1	1	False	
16422	2019010203843	Slight	Saturday	Single carriageway	Darkness - lighting unknown	20	-1	0	NaN	1	2	False	
24422	2019010226740	Slight	Saturday	Single carriageway	Darkness - lighting unknown	20	-1	-1	NaN	1	1	False	

1862 rows × 13 columns

In [29]:
```
#df_train.loc[(df_train['Accident_Index'] == 2019010226740)]
```

In [30]:
```
df_train.shape
```

Out[30]: (82275, 13)

In [31]:
```
#### Replacing the values for road surface conditions
```

In [32]:
```
replacement_Road_Surface_Conditions = {
1:"Dry",
2:"Wet or damp",
3:"Snow",
4:"Frost or ice",
5:"Flood",
6:"Oil or diesel",
7:"Mud",
-1:"Data missing or out of range"
}


df_train['Road_Surface_Conditions'] = df_train['Road_Surface_Conditions'].map(replacement_Road_Surface_Conditions)
df_train
```

Out[32]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	lig
103938	2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	Wet or damp	0	NaN	1	2	False	
111014	201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	Wet or damp	0	NaN	2	2	True	
17696	2019010207283	Slight	Monday	Single carriageway	Daylight	20	Dry	0	NaN	1	2	True	
90915	2019460853693	Slight	Monday	Single carriageway	Daylight	20	Dry	0	NaN	1	1	True	
73451	2019400859184	Slight	Saturday	Single carriageway	Daylight	30	Dry	0	NaN	2	2	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34838	2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	Dry	0	NaN	1	2	True	
83134	2019440064734	Serious	Friday	Single carriageway	Daylight	30	Dry	0	NaN	1	1	True	
102425	2019521902482	Serious	Friday	Dual carriageway	Daylight	20	Wet or damp	0	NaN	1	2	True	
6719	2019010175779	Serious	Thursday	Single carriageway	Daylight	20	Dry	0	NaN	1	2	True	
6996	2019010176615	Slight	Tuesday	Unknown	Daylight	20	Flood	0	NaN	1	2	True	

82275 rows × 13 columns

In [33]:
```
df_train['Road_Surface_Conditions'] = df_train['Road_Surface_Conditions'].replace("Data missing or out of range", np.NaN)
```

In [34]:
```
df_train
```

Out[34]:

Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	light_status
2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	Wet or damp	0	NaN	1	2	False	lit
201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	Wet or damp	0	NaN	2	2	True	NaN
2019010207283	Slight	Monday	Single carriageway	Daylight	20	Dry	0	NaN	1	2	True	NaN
2019460853693	Slight	Monday	Single carriageway	Daylight	20	Dry	0	NaN	1	1	True	NaN
2019400859184	Slight	Saturday	Single carriageway	Daylight	30	Dry	0	NaN	2	2	True	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	Dry	0	NaN	1	2	True	NaN
2019440064734	Serious	Friday	Single carriageway	Daylight	30	Dry	0	NaN	1	1	True	NaN
2019521902482	Serious	Friday	Dual carriageway	Daylight	20	Wet or damp	0	NaN	1	2	True	NaN
2019010175779	Serious	Thursday	Single carriageway	Daylight	20	Dry	0	NaN	1	2	True	NaN
2019010176615	Slight	Tuesday	Unknown	Daylight	20	Flood	0	NaN	1	2	True	NaN

ws × 13 columns

In [36]:
```
replacement_Special_Conditions_site = {
0:"None",
1:"Auto traffic signal - out",
2:"Auto signal part defective",
3:"Road sign or marking defective or obscured",
4:"Roadworks",
5:"Road surface defective",
6:"Oil or diesel",
7:"Mud",
-1:"Data missing or out of range"
}


df_train['Special_Conditions_at_Site'] = df_train['Special_Conditions_at_Site'].map(replacement_Special_Conditions_site)
df_train['Special_Conditions_at_Site'] = df_train['Special_Conditions_at_Site'].replace("Data missing or out of range", np.NaN)
df_train
```

Out[36]:

Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	light_status
2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	Wet or damp	None	NaN	1	2	False	lit
201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	Wet or damp	None	NaN	2	2	True	NaN
2019010207283	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	1	2	True	NaN
2019460853693	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	1	1	True	NaN
2019400859184	Slight	Saturday	Single carriageway	Daylight	30	Dry	None	NaN	2	2	True	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	Dry	None	NaN	1	2	True	NaN
2019440064734	Serious	Friday	Single carriageway	Daylight	30	Dry	None	NaN	1	1	True	NaN
2019521902482	Serious	Friday	Dual carriageway	Daylight	20	Wet or damp	None	NaN	1	2	True	NaN
2019010175779	Serious	Thursday	Single carriageway	Daylight	20	Dry	None	NaN	1	2	True	NaN
2019010176615	Slight	Tuesday	Unknown	Daylight	20	Flood	None	NaN	1	2	True	NaN

ws × 13 columns

In [40]:
```
replacement_Carriageway_Hazards = {
0:"None",
1:"Vehicle load on road",
2:"Other object on road",
3:"Previous accident",
4:"Dog on road",
5:"Other animal on road",
6:"Pedestrian in carriageway - not injured",
7:"Any animal in carriageway (except ridden horse)",
-1:"Data missing or out of range"

}


df_train['Carriageway Hazards'] = df_train['Carriageway Hazards'].map(replacement_Carriageway_Hazards)
df_train['Carriageway Hazards'] = df_train['Carriageway Hazards'].replace("Data missing or out of range", np.NaN)
df_train
```

Out[40]:

Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	light_status
2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	Wet or damp	None	NaN	1	2	False	lit
201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	Wet or damp	None	NaN	2	2	True	NaN
2019010207283	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	1	2	True	NaN
2019460853693	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	1	1	True	NaN
2019400859184	Slight	Saturday	Single carriageway	Daylight	30	Dry	None	NaN	2	2	True	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	Dry	None	NaN	1	2	True	NaN
2019440064734	Serious	Friday	Single carriageway	Daylight	30	Dry	None	NaN	1	1	True	NaN
2019521902482	Serious	Friday	Dual carriageway	Daylight	20	Wet or damp	None	NaN	1	2	True	NaN
2019010175779	Serious	Thursday	Single carriageway	Daylight	20	Dry	None	NaN	1	2	True	NaN
2019010176615	Slight	Tuesday	Unknown	Daylight	20	Flood	None	NaN	1	2	True	NaN

ws × 13 columns

In [41]:
```
replacement_Urban_or_Rural_Area = {
1:"Urban",
2:"Rural",
3:"Unallocated"
}


df_train['Urban_or_Rural_Area'] = df_train['Urban_or_Rural_Area'].map(replacement_Urban_or_Rural_Area)

df_train
```

Out[41]:

	Accident_Index	Accident_Severity	Day_of_Week	Road_Type	Light_Conditions	Speed_limit	Road_Surface_Conditions	Special_Conditions_at_Site	Carriageway Hazards	Urban_or_Rural_Area	Number_of_Vehicles	Daylight	lig
103938	2019522001185	Slight	Saturday	Single carriageway	Darkness - lights lit	30	Wet or damp	None	NaN	Urban	2	False	
111014	201963A101619	Serious	Tuesday	Single carriageway	Daylight	60	Wet or damp	None	NaN	Rural	2	True	
17696	2019010207283	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	Urban	2	True	
90915	2019460853693	Slight	Monday	Single carriageway	Daylight	20	Dry	None	NaN	Urban	1	True	
73451	2019400859184	Slight	Saturday	Single carriageway	Daylight	30	Dry	None	NaN	Rural	2	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	
34838	2019070007679	Fatal	Tuesday	Single carriageway	Daylight	30	Dry	None	NaN	Urban	2	True	
83134	2019440064734	Serious	Friday	Single carriageway	Daylight	30	Dry	None	NaN	Urban	1	True	
102425	2019521902482	Serious	Friday	Dual carriageway	Daylight	20	Wet or damp	None	NaN	Urban	2	True	
6719	2019010175779	Serious	Thursday	Single carriageway	Daylight	20	Dry	None	NaN	Urban	2	True	
6996	2019010176615	Slight	Tuesday	Unknown	Daylight	20	Flood	None	NaN	Urban	2	True	

82275 rows × 13 columns

In [ ]: