```
!pip install folium
```

```
Requirement already satisfied: folium in c:\users\shiva\anaconda3\lib\
site-packages (0.17.0)
Requirement already satisfied: branca>=0.6.0 in c:\users\shiva\
anaconda3\lib\site-packages (from folium) (0.7.2)
Requirement already satisfied: jinja2>=2.9 in c:\users\shiva\
anaconda3\lib\site-packages (from folium) (3.1.4)
Requirement already satisfied: numpy in c:\users\shiva\anaconda3\lib\
site-packages (from folium) (1.26.4)
Requirement already satisfied: requests in c:\users\shiva\anaconda3\
lib\site-packages (from folium) (2.32.2)
Requirement already satisfied: xyzservices in c:\users\shiva\
anaconda3\lib\site-packages (from folium) (2022.9.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\shiva\
anaconda3\lib\site-packages (from jinja2>=2.9->folium) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
shiva\anaconda3\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\shiva\
anaconda3\lib\site-packages (from requests->folium) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\shiva\
anaconda3\lib\site-packages (from requests->folium) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\shiva\
anaconda3\lib\site-packages (from requests->folium) (2024.7.4)
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import folium
from folium.plugins import HeatMap
import numpy as np
```

```
# Load the dataset
file_path = 'RTA Dataset.csv'
data = pd.read_csv(file_path)


# Display the first few rows of the dataset
data.head()
```

```
       Time Day_of_week Age_band_of_driver Sex_of_driver
Educational_level  \
0  17:02:00      Monday              18-30          Male   Above high
school
1  17:02:00      Monday              31-50          Male  Junior high
school
2  17:02:00      Monday              18-30          Male  Junior high
school
3   1:06:00      Sunday              18-30          Male  Junior high
school
```

```
4   1:06:00      Sunday              18-30          Male  Junior high
school

  Vehicle_driver_relation Driving_experience      Type_of_vehicle  \
0                Employee             1-2yr           Automobile
1                Employee         Above 10yr  Public (> 45 seats)
2                Employee             1-2yr      Lorry (41?100Q)
3                Employee            5-10yr  Public (> 45 seats)
4                Employee             2-5yr                  NaN

  Owner_of_vehicle Service_year_of_vehicle  ... Vehicle_movement  \
0            Owner               Above 10yr  ...   Going straight
1            Owner                  5-10yrs  ...   Going straight
2            Owner                      NaN  ...   Going straight
3    Governmental                       NaN  ...   Going straight
4            Owner                  5-10yrs  ...   Going straight

    Casualty_class Sex_of_casualty Age_band_of_casualty
Casualty_severity  \
0               na              na                   na
na
1               na              na                   na
na
2  Driver or rider            Male                31-50
3
3       Pedestrian          Female                18-30
3
4               na              na                   na
na

  Work_of_casuality Fitness_of_casuality Pedestrian_movement  \
0              NaN                  NaN    Not a Pedestrian
1              NaN                  NaN    Not a Pedestrian
2           Driver                  NaN    Not a Pedestrian
3           Driver               Normal    Not a Pedestrian
4              NaN                  NaN    Not a Pedestrian

            Cause_of_accident Accident_severity
0             Moving Backward     Slight Injury
1                  Overtaking     Slight Injury
2   Changing lane to the left    Serious Injury
3  Changing lane to the right     Slight Injury
4                  Overtaking     Slight Injury

[5 rows x 32 columns]

data.tail()

                     Time Day_of_week Age_band_of_driver Sex_of_driver
\
```

```
12311 2024-08-03 16:15:00   Wednesday              31-50          Male

12312 2024-08-03 18:00:00      Sunday            Unknown          Male

12313 2024-08-03 13:55:00      Sunday            Over 51          Male

12314 2024-08-03 13:55:00      Sunday              18-30        Female

12315 2024-08-03 13:55:00      Sunday              18-30          Male


        Educational_level Vehicle_driver_relation
Driving_experience  \
12311                 NaN               Employee              2-5yr

12312   Elementary school               Employee             5-10yr

12313  Junior high school               Employee             5-10yr

12314  Junior high school               Employee         Above 10yr

12315  Junior high school               Employee             5-10yr


       Type_of_vehicle Owner_of_vehicle
Service_year_of_vehicle  ...  \
12311   Lorry (11?40Q)            Owner                     NaN  ...

12312       Automobile            Owner                     NaN  ...

12313            Bajaj            Owner                   2-5yrs  ...

12314  Lorry (41?100Q)            Owner                   2-5yrs  ...

12315            Other            Owner                   2-5yrs  ...


        Casualty_class Sex_of_casualty Age_band_of_casualty
Casualty_severity  \
12311              na              na                   na
na
12312              na              na                   na
na
12313  Driver or rider            Male                31-50
3
12314              na              na                   na
na
12315       Pedestrian          Female                    5
3

      Work_of_casuality Fitness_of_casuality  \
```

```
12311            Driver               Normal
12312            Driver               Normal
12313            Driver               Normal
12314            Driver               Normal
12315            Driver               Normal

                                   Pedestrian_movement  \
12311                                 Not a Pedestrian
12312                                 Not a Pedestrian
12313                                 Not a Pedestrian
12314                                 Not a Pedestrian
12315  Crossing from nearside - masked by parked or s...

                         Cause_of_accident Accident_severity hour
12311                        No distancing     Slight Injury   16
12312                        No distancing     Slight Injury   18
12313            Changing lane to the right    Serious Injury   13
12314  Driving under the influence of drugs     Slight Injury   13
12315            Changing lane to the right     Slight Injury   13

[5 rows x 33 columns]
```

```
# Data overview
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 32 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Time                         12316 non-null  object
 1   Day_of_week                  12316 non-null  object
 2   Age_band_of_driver           12316 non-null  object
 3   Sex_of_driver                12316 non-null  object
 4   Educational_level            11575 non-null  object
 5   Vehicle_driver_relation      11737 non-null  object
 6   Driving_experience           11487 non-null  object
 7   Type_of_vehicle              11366 non-null  object
 8   Owner_of_vehicle             11834 non-null  object
 9   Service_year_of_vehicle      8388 non-null   object
 10  Defect_of_vehicle            7889 non-null   object
 11  Area_accident_occured        12077 non-null  object
 12  Lanes_or_Medians             11931 non-null  object
 13  Road_allignment              12174 non-null  object
 14  Types_of_Junction            11429 non-null  object
 15  Road_surface_type            12144 non-null  object
 16  Road_surface_conditions      12316 non-null  object
 17  Light_conditions             12316 non-null  object
 18  Weather_conditions           12316 non-null  object
 19  Type_of_collision            12161 non-null  object
```

```
 20  Number_of_vehicles_involved  12316 non-null  int64
 21  Number_of_casualties         12316 non-null  int64
 22  Vehicle_movement             12008 non-null  object
 23  Casualty_class               12316 non-null  object
 24  Sex_of_casualty              12316 non-null  object
 25  Age_band_of_casualty         12316 non-null  object
 26  Casualty_severity            12316 non-null  object
 27  Work_of_casuality            9118 non-null   object
 28  Fitness_of_casuality         9681 non-null   object
 29  Pedestrian_movement          12316 non-null  object
 30  Cause_of_accident            12316 non-null  object
 31  Accident_severity            12316 non-null  object
dtypes: int64(2), object(30)
memory usage: 3.0+ MB
```

```
data.describe()
```

	Number_of_vehicles_involved	Number_of_casualties
count	12316.000000	12316.000000
mean	2.040679	1.548149
std	0.688790	1.007179
min	1.000000	1.000000
25%	2.000000	1.000000
50%	2.000000	1.000000
75%	2.000000	2.000000
max	7.000000	8.000000

```
# Checking the duplicate values
data.duplicated().sum()
```

```
0
```

```
# Checking the null values
data.isnull().sum()
```

```
Time                           0
Day_of_week                    0
Age_band_of_driver             0
Sex_of_driver                  0
Educational_level            741
Vehicle_driver_relation      579
Driving_experience           829
Type_of_vehicle              950
Owner_of_vehicle             482
Service_year_of_vehicle     3928
Defect_of_vehicle           4427
Area_accident_occured        239
Lanes_or_Medians             385
Road_allignment              142
Types_of_Junction            887
Road_surface_type            172
```

```
Road_surface_conditions           0
Light_conditions                  0
Weather_conditions                0
Type_of_collision               155
Number_of_vehicles_involved       0
Number_of_casualties              0
Vehicle_movement                308
Casualty_class                    0
Sex_of_casualty                   0
Age_band_of_casualty              0
Casualty_severity                 0
Work_of_casuality              3198
Fitness_of_casuality           2635
Pedestrian_movement               0
Cause_of_accident                 0
Accident_severity                 0
hour                              0
dtype: int64
```

```
# Filling the null values
data['Educational_level'].fillna(data['Educational_level'].mode()[0],
inplace=True)
data['Vehicle_driver_relation'].fillna(data['Vehicle_driver_relation']
.mode()[0], inplace=True)
data['Driving_experience'].fillna(data['Driving_experience'].mode()
[0], inplace=True)
data['Type_of_vehicle'].fillna(data['Type_of_vehicle'].mode()[0],
inplace=True)
data['Owner_of_vehicle'].fillna(data['Owner_of_vehicle'].mode()[0],
inplace=True)
data['Service_year_of_vehicle'].fillna(data['Service_year_of_vehicle']
.mode()[0], inplace=True)
data['Defect_of_vehicle'].fillna(data['Defect_of_vehicle'].mode()[0],
inplace=True)
data['Area_accident_occured'].fillna(data['Area_accident_occured'].mod
e()[0], inplace=True)
data['Lanes_or_Medians'].fillna(data['Lanes_or_Medians'].mode()[0],
inplace=True)
data['Road_allignment'].fillna(data['Road_allignment'].mode()[0],
inplace=True)
data['Types_of_Junction'].fillna(data['Types_of_Junction'].mode()[0],
inplace=True)
data['Road_surface_type'].fillna(data['Road_surface_type'].mode()[0],
inplace=True)
data['Type_of_collision'].fillna(data['Type_of_collision'].mode()[0],
inplace=True)
data['Vehicle_movement'].fillna(data['Vehicle_movement'].mode()[0],
inplace=True)
data['Work_of_casuality'].fillna(data['Work_of_casuality'].mode()[0],
inplace=True)
```

```
data['Fitness_of_casuality'].fillna(data['Fitness_of_casuality'].mode(
)[0], inplace=True)
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:2:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
  data['Educational_level'].fillna(data['Educational_level'].mode()
[0], inplace=True)
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:3:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
data['Vehicle_driver_relation'].fillna(data['Vehicle_driver_relation']
.mode()[0], inplace=True)
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:4:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
  data['Driving_experience'].fillna(data['Driving_experience'].mode()
[0], inplace=True)
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:5:
```

```
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Type_of_vehicle'].fillna(data['Type_of_vehicle'].mode()[0],
inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:6:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Owner_of_vehicle'].fillna(data['Owner_of_vehicle'].mode()[0],
inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:7:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



data['Service_year_of_vehicle'].fillna(data['Service_year_of_vehicle']
.mode()[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:8:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
```

```
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Defect_of_vehicle'].fillna(data['Defect_of_vehicle'].mode()
[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:9:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



data['Area_accident_occured'].fillna(data['Area_accident_occured'].mod
e()[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:10:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Lanes_or_Medians'].fillna(data['Lanes_or_Medians'].mode()[0],
inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:11:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
```

```
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Road_allignment'].fillna(data['Road_allignment'].mode()[0],
inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:12:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Types_of_Junction'].fillna(data['Types_of_Junction'].mode()
[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:13:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Road_surface_type'].fillna(data['Road_surface_type'].mode()
[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:14:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Type_of_collision'].fillna(data['Type_of_collision'].mode()
```

```
[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:15:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Vehicle_movement'].fillna(data['Vehicle_movement'].mode()[0],
inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:16:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Work_of_casuality'].fillna(data['Work_of_casuality'].mode()
[0], inplace=True)
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1330502609.py:17:
FutureWarning: A value is trying to be set on a copy of a DataFrame or
Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



data['Fitness_of_casuality'].fillna(data['Fitness_of_casuality'].mode(
)[0], inplace=True)
```

```
data.isnull().sum()
```

```
Time                            0
Day_of_week                     0
Age_band_of_driver              0
Sex_of_driver                   0
Educational_level               0
Vehicle_driver_relation         0
Driving_experience              0
Type_of_vehicle                 0
Owner_of_vehicle                0
Service_year_of_vehicle         0
Defect_of_vehicle               0
Area_accident_occured           0
Lanes_or_Medians                0
Road_allignment                 0
Types_of_Junction               0
Road_surface_type               0
Road_surface_conditions         0
Light_conditions                0
Weather_conditions              0
Type_of_collision               0
Number_of_vehicles_involved     0
Number_of_casualties            0
Vehicle_movement                0
Casualty_class                  0
Sex_of_casualty                 0
Age_band_of_casualty            0
Casualty_severity               0
Work_of_casuality               0
Fitness_of_casuality            0
Pedestrian_movement             0
Cause_of_accident               0
Accident_severity               0
hour                            0
dtype: int64
```

```
# Statistical description of the data
data.describe().T
```

```
                               count                           mean  \
Time                           12316  2024-08-03 14:17:50.768106496
Number_of_vehicles_involved  12316.0                       2.040679
Number_of_casualties         12316.0                       1.548149
hour                         12316.0                      13.835823

                                          min                  25%
\
Time                         2024-08-03 00:01:00  2024-08-03 10:31:00

Number_of_vehicles_involved                  1.0                  2.0
```

```
Number_of_casualties                          1.0                   1.0

hour                                          0.0                  10.0


                                              50%                   75%
\
Time                          2024-08-03 15:10:00  2024-08-03 18:10:00

Number_of_vehicles_involved                   2.0                   2.0

Number_of_casualties                          1.0                   2.0

hour                                         15.0                  18.0


                                              max       std
Time                          2024-08-03 23:59:00       NaN
Number_of_vehicles_involved                   7.0   0.68879
Number_of_casualties                          8.0  1.007179
hour                                         23.0  5.202923
```

```python
# Value counts of accident severity
data.groupby('Accident_severity').size()
```

```
Accident_severity
Fatal injury        158
Serious Injury     1743
Slight Injury     10415
dtype: int64
```

```python
# Value counts of driver's age
data['Age_band_of_driver'].value_counts()
```

```
Age_band_of_driver
18-30       4271
31-50       4087
Over 51     1585
Unknown     1548
Under 18     825
Name: count, dtype: int64
```

```python
# Value counts of weather conditions
data['Weather_conditions'].value_counts()
```

```
Weather_conditions
Normal               10063
Raining               1331
Other                  296
Unknown                292
Cloudy                 125
```

```
Windy                    98
Snow                     61
Raining and Windy        40
Fog or mist              10
Name: count, dtype: int64
```

```
# Value counts of types of collision
data['Type_of_collision'].value_counts()
```

```
Type_of_collision
Vehicle with vehicle collision             8929
Collision with roadside objects            1786
Collision with pedestrians                  896
Rollover                                    397
Collision with animals                      171
Collision with roadside-parked vehicles      54
Fall from vehicles                           34
Other                                        26
Unknown                                      14
With Train                                    9
Name: count, dtype: int64
```

```
# Convert the 'time' column to datetime format if it's not already
data['Time'] = pd.to_datetime(data['Time'])
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\613933959.py:2:
UserWarning: Could not infer format, so each element will be parsed
individually, falling back to `dateutil`. To ensure parsing is
consistent and as-expected, please specify a format.
  data['Time'] = pd.to_datetime(data['Time'])
```

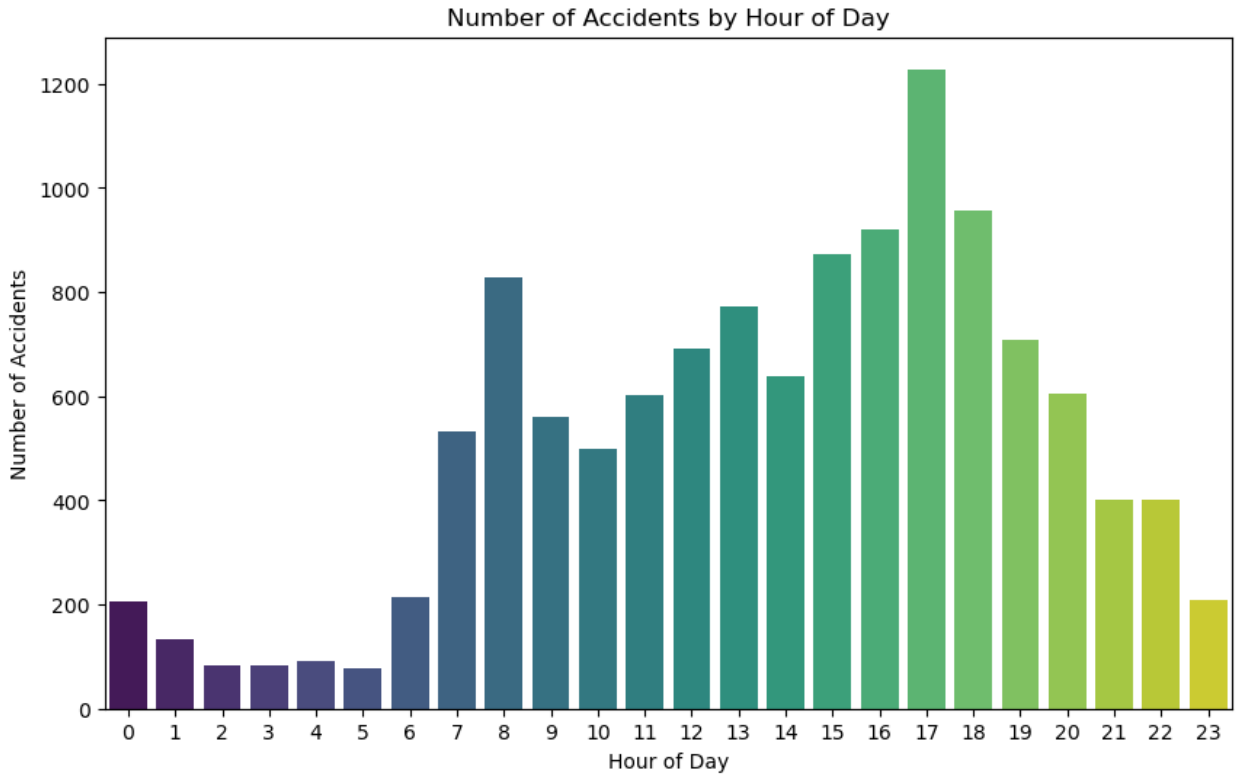
```
# Extract hour from the 'time' column
data['hour'] = data['Time'].dt.hour
```

```
# Count the number of accidents by hour
accidents_by_hour = data['hour'].value_counts().sort_index()
```

```
# Plot the number of accidents by hour
plt.figure(figsize=(10, 6))
sns.barplot(x=accidents_by_hour.index, y=accidents_by_hour.values,
palette="viridis")
plt.xlabel('Hour of Day')
plt.ylabel('Number of Accidents')
plt.title('Number of Accidents by Hour of Day')
plt.xticks(range(0, 24))
plt.show()
```

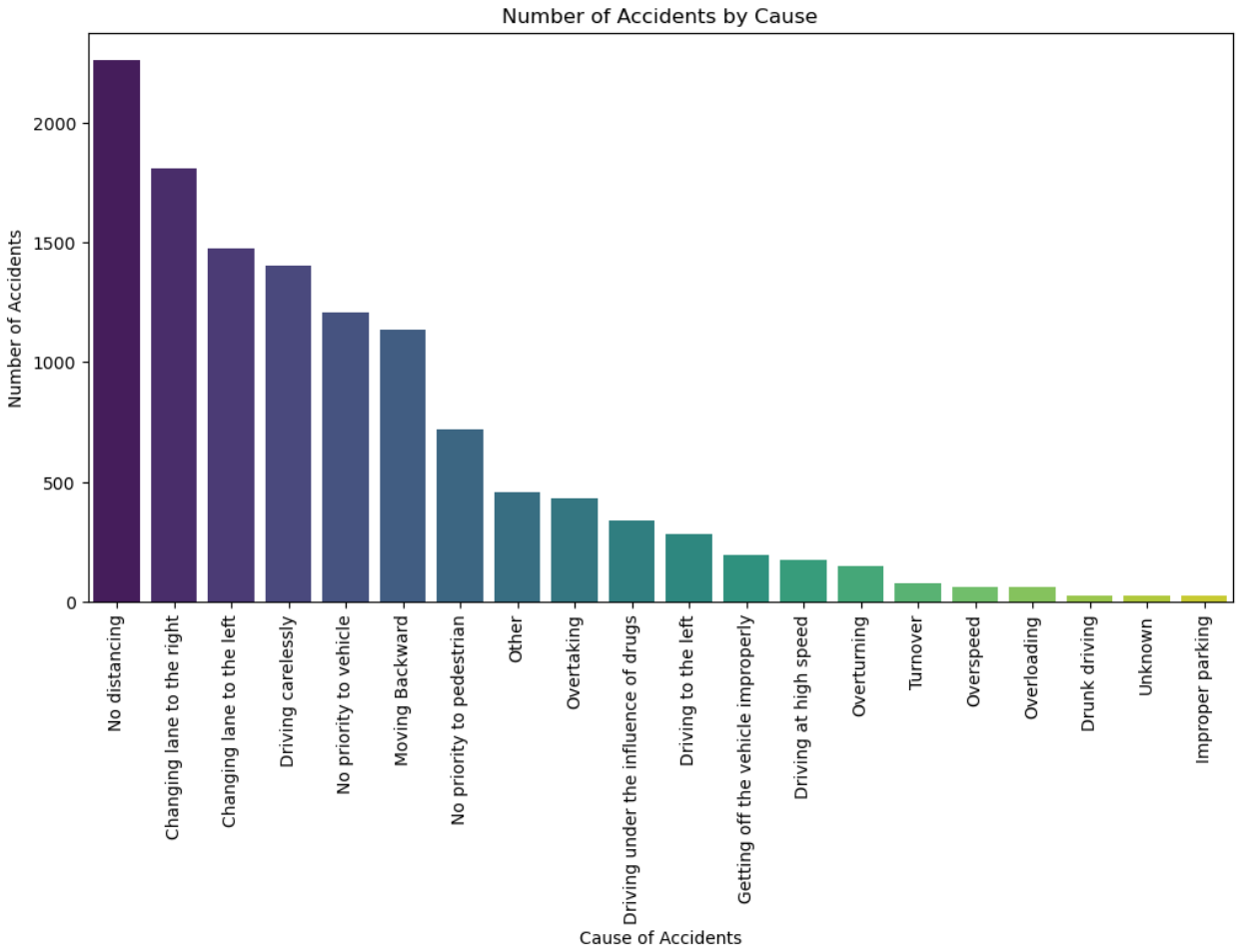
```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\3656318898.py:3:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
```

```
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x=accidents_by_hour.index, y=accidents_by_hour.values,
palette="viridis")
```

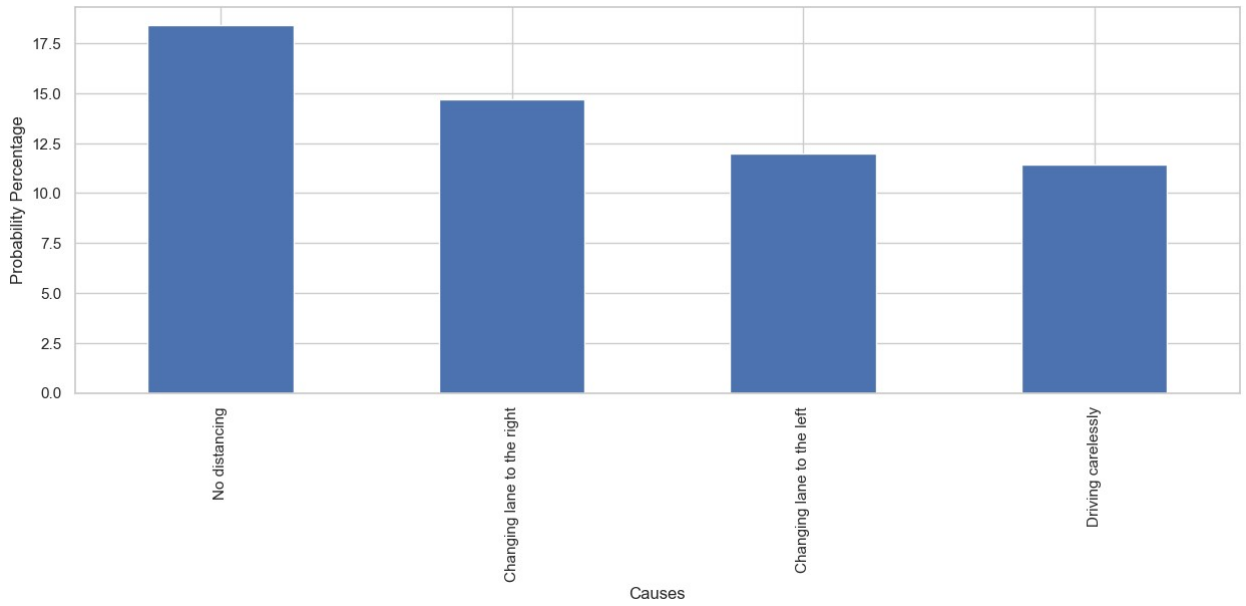



```
# Count the number of accidents by weather condition
accidents_by_cause = data['Cause_of_accident'].value_counts()

# Plot the number of accidents by weather condition
plt.figure(figsize=(12, 6))
sns.barplot(x=accidents_by_cause.index, y=accidents_by_cause.values,
palette="viridis")
plt.xlabel('Cause of Accidents')
plt.ylabel('Number of Accidents')
plt.title('Number of Accidents by Cause')
plt.xticks(rotation=90)
plt.show()

C:\Users\shiva\AppData\Local\Temp\ipykernel_736\4097903947.py:3:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.
```

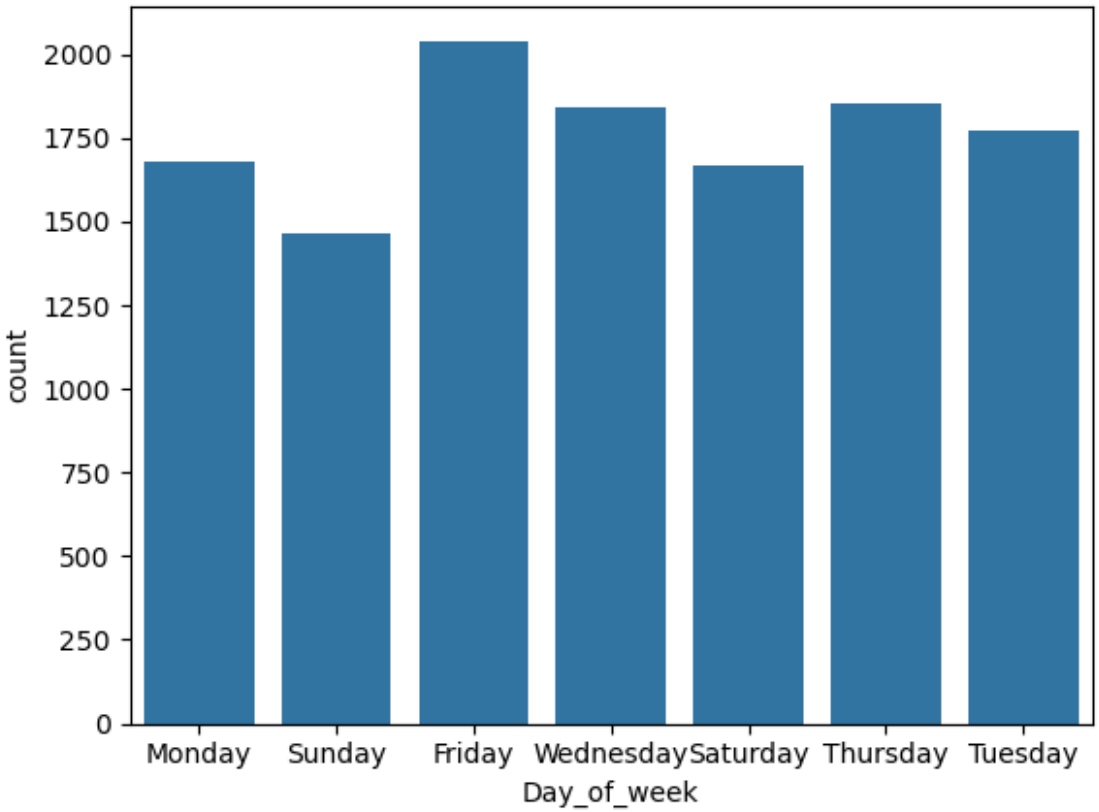
```
  sns.barplot(x=accidents_by_cause.index, y=accidents_by_cause.values,
palette="viridis")
```




```
# Top 4 causes by probability percentage
plt.figure(figsize =(15,5))
a= data.loc[data['Cause_of_accident'] !="Unknown"]
bar_plot=((a.groupby('Cause_of_accident')
['Cause_of_accident'].count().sort_values(ascending=False)/
a['Cause_of_accident'].count())*100).head(4).plot.bar()
bar_plot.set_ylabel("Probability Percentage")
bar_plot.set_xlabel("Causes")

Text(0.5, 0, 'Causes')
```




```
# Count distribution of Days
sns.countplot(x="Day_of_week",data=data)
plt.xticks(rotation='horizontal')
plt.show()
```

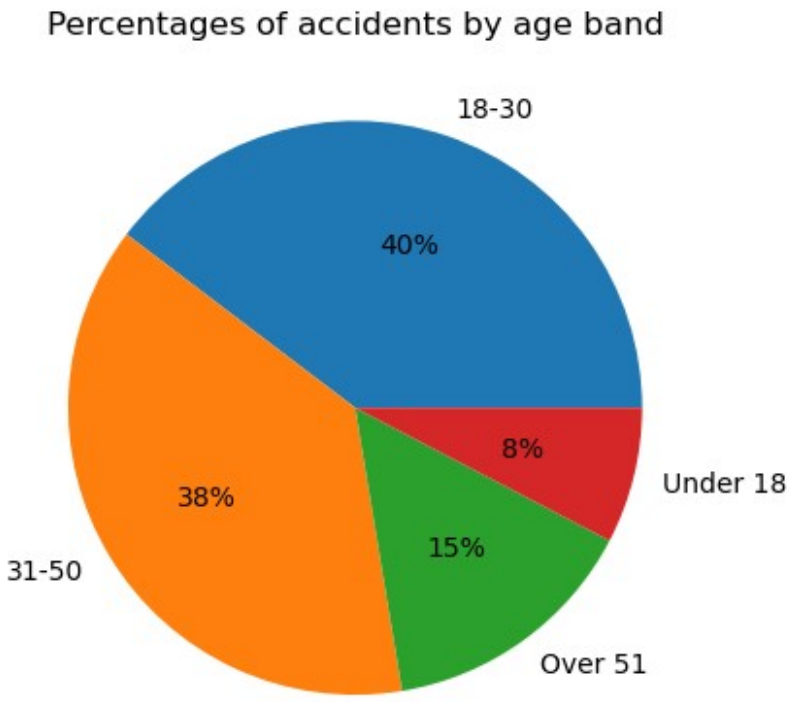


```
# Percentage distribution of driver's age
b=data.loc[data['Age_band_of_driver'] !="Unknown",
['Age_band_of_driver', 'Accident_severity']]
b=((b.groupby(['Age_band_of_driver']).size()/data["Age_band_of_driver"
].count())*100)
b
```

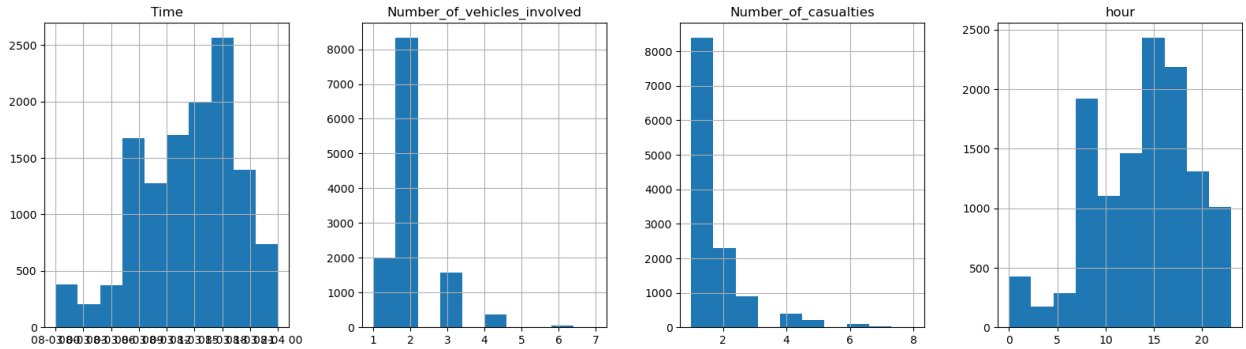
```
Age_band_of_driver
18-30       34.678467
31-50       33.184475
Over 51     12.869438
Under 18     6.698603
dtype: float64
```

```
# Pie chart dsitribution of age groups
b.plot.pie(autopct='%1.0f%%', title='Percentages of accidents by age
band')
```

```
<Axes: title={'center': 'Percentages of accidents by age band'}>
```



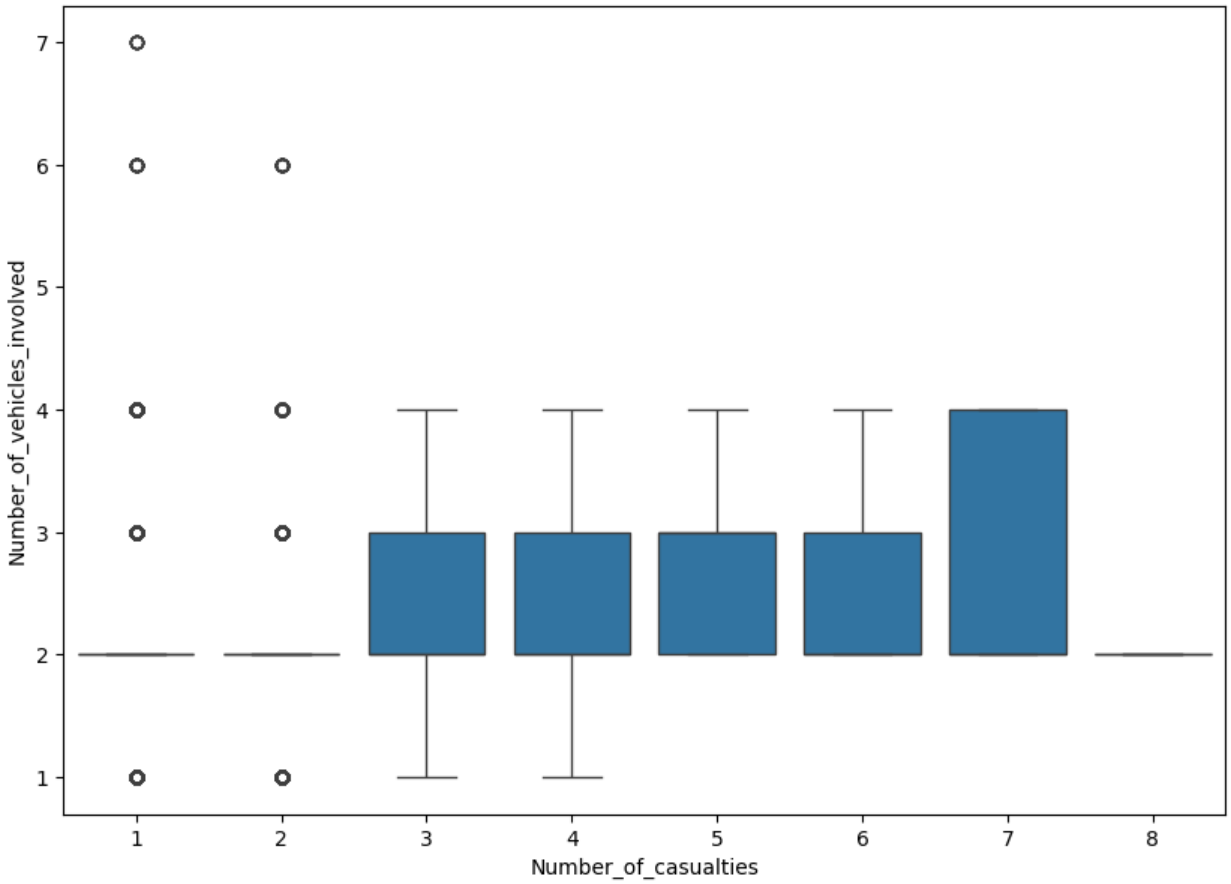

```
# Histogram distribution of numerical features
data.hist(layout=(1,6), figsize=(30,5))
plt.show()
```



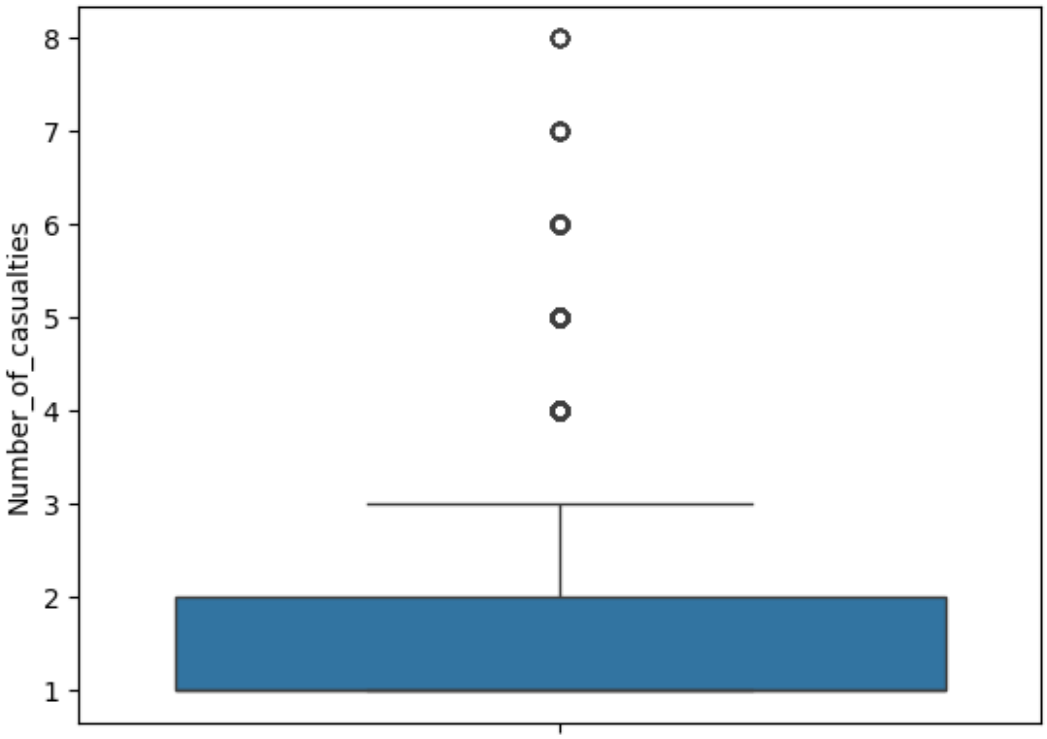

```
# Value counts of casualties
data['Number_of_casualties'].value_counts()
```

```
Number_of_casualties
1    8397
2    2290
3     909
4     394
5     207
6      89
7      22
8       8
Name: count, dtype: int64
```

```
# Box plot distribution
plt.figure(figsize=(10,7))
sns.boxplot(data=data, y='Number_of_vehicles_involved',
x='Number_of_casualties')
plt.show()
```



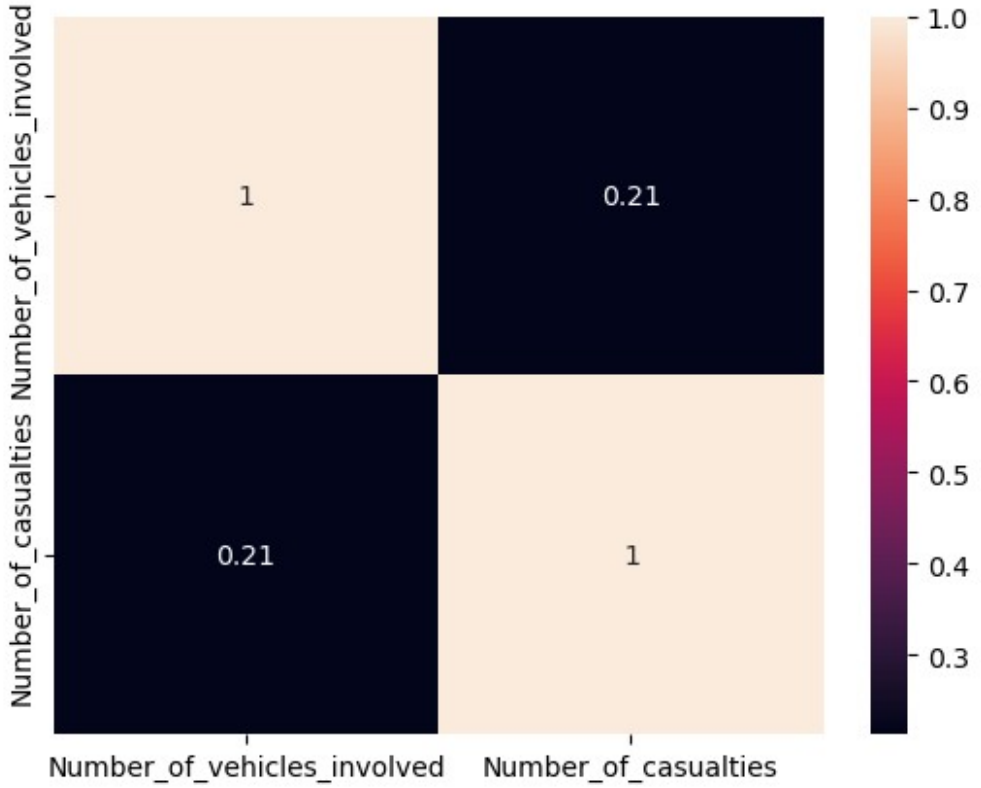

```
# Box plot distribution of casualties
sns.boxplot(data=data, y='Number_of_casualties')
plt.show()
```



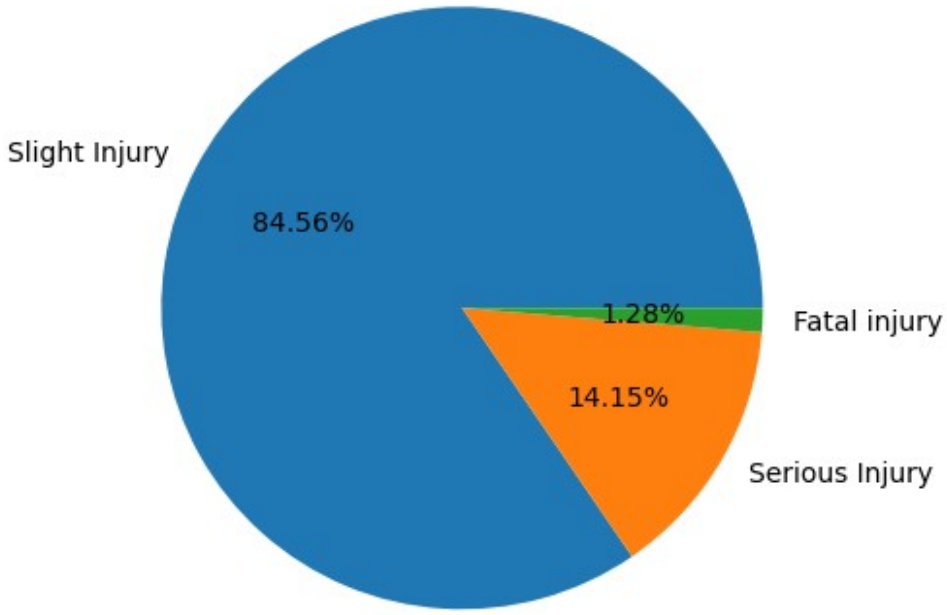

```
# Value counts of vehicles involved in accidents
data['Number_of_vehicles_involved'].value_counts()
```

```
Number_of_vehicles_involved
2    8340
1    1996
3    1568
4     363
6      42
7       7
Name: count, dtype: int64
```

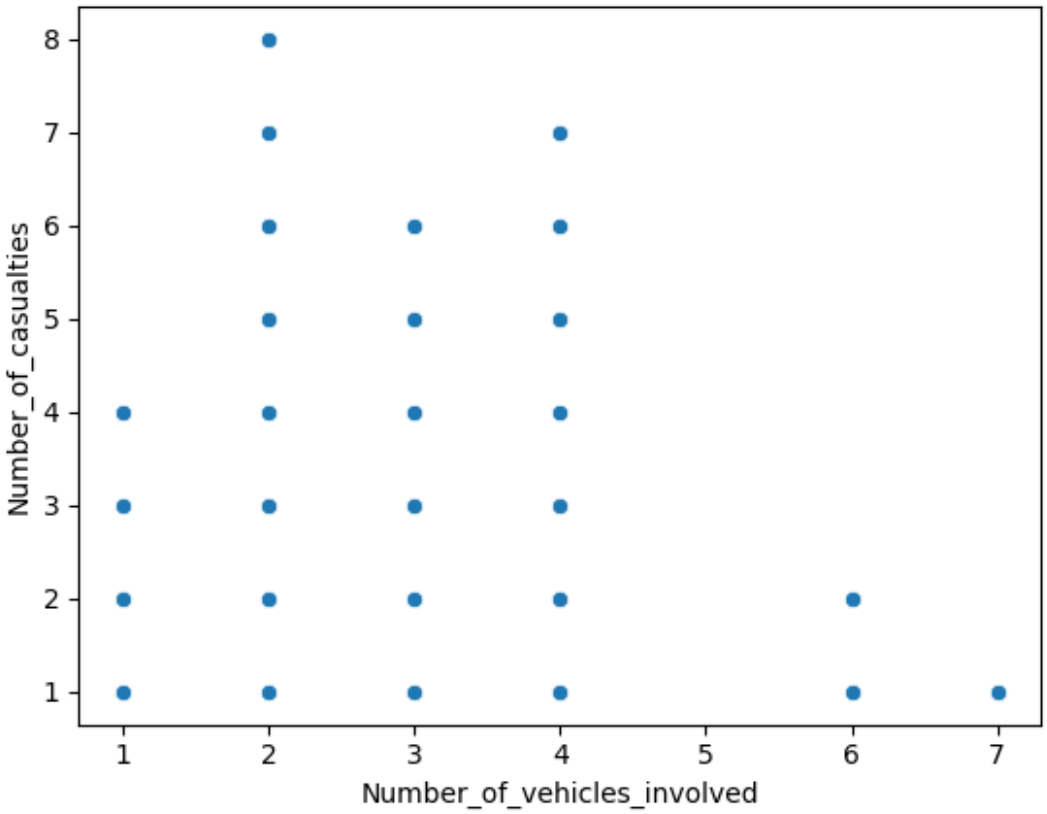
```
# Correlation description
correlation_matrix =
data[['Number_of_vehicles_involved','Number_of_casualties']].corr()
sns.heatmap(correlation_matrix, annot=True)
plt.show()
```



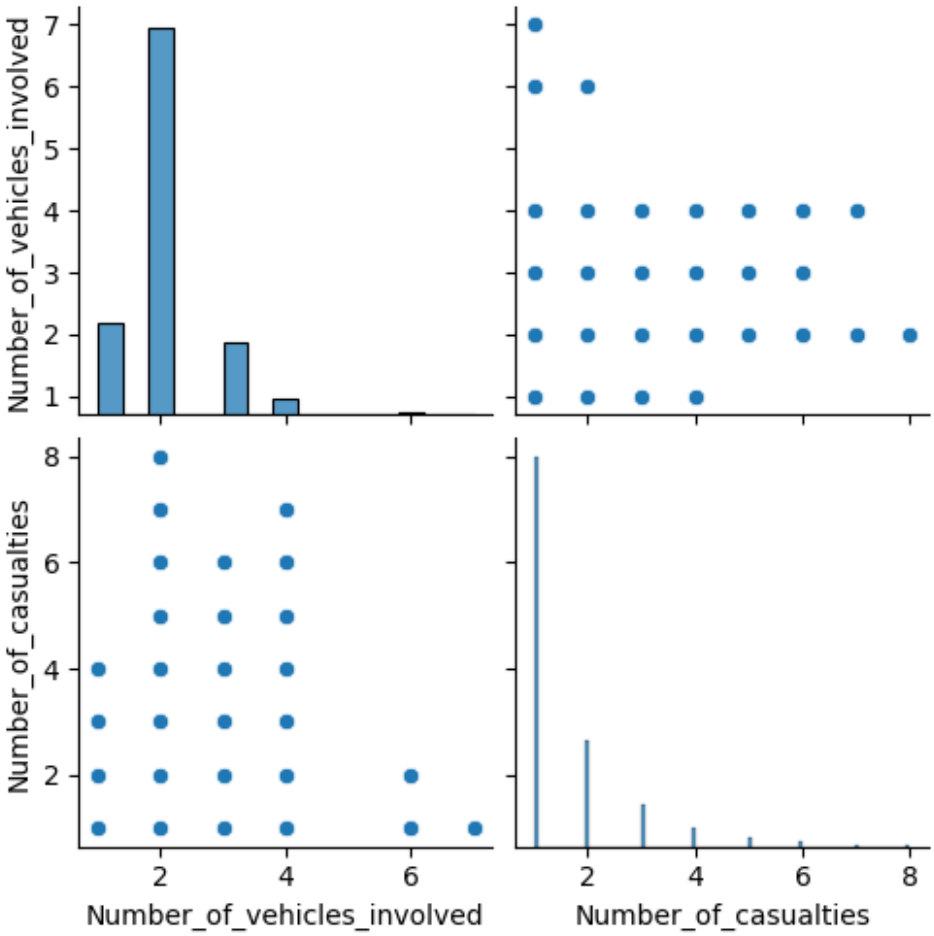

```
# Pie chart distribution of accident severity
plt.figure(figsize=(5,7))
plt.pie(x=data['Accident_severity'].value_counts().values,
        labels=data['Accident_severity'].value_counts().index,
        autopct='%2.2f%%')
plt.show()
```




```
# Scatter plot relation
sns.scatterplot(x=data['Number_of_vehicles_involved'],
y=data['Number_of_casualties'])
plt.show()
```



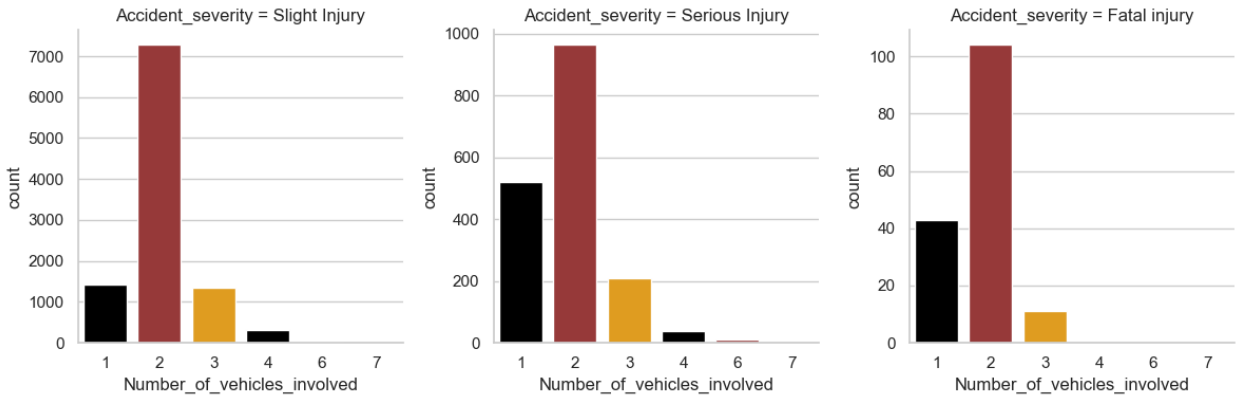

```
# Pair plot distribution
sns.pairplot(data[['Number_of_vehicles_involved','Number_of_casualties
']])
plt.show()
```




```
# creating agrid with columns as survived=0 and survived=1
grid = sns.FacetGrid(data=data, col='Accident_severity', height=4,
aspect=1, sharey=False)
# mapping bar plot and the data on to the grid
grid.map(sns.countplot, 'Number_of_vehicles_involved',
palette=['black', 'brown', 'orange'])
plt.show()
```

```
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:718:
UserWarning: Using the countplot function without specifying `order`
is likely to produce an incorrect plot.
  warnings.warn(warning)
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:854:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  func(*plot_args, **plot_kwargs)
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:854:
```

```
UserWarning:
The palette list has fewer values (3) than needed (6) and will cycle,
which may produce an uninterpretable plot.
  func(*plot_args, **plot_kwargs)
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:854:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  func(*plot_args, **plot_kwargs)
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:854:
UserWarning:
The palette list has fewer values (3) than needed (5) and will cycle,
which may produce an uninterpretable plot.
  func(*plot_args, **plot_kwargs)
C:\Users\shiva\anaconda3\Lib\site-packages\seaborn\axisgrid.py:854:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  func(*plot_args, **plot_kwargs)
```

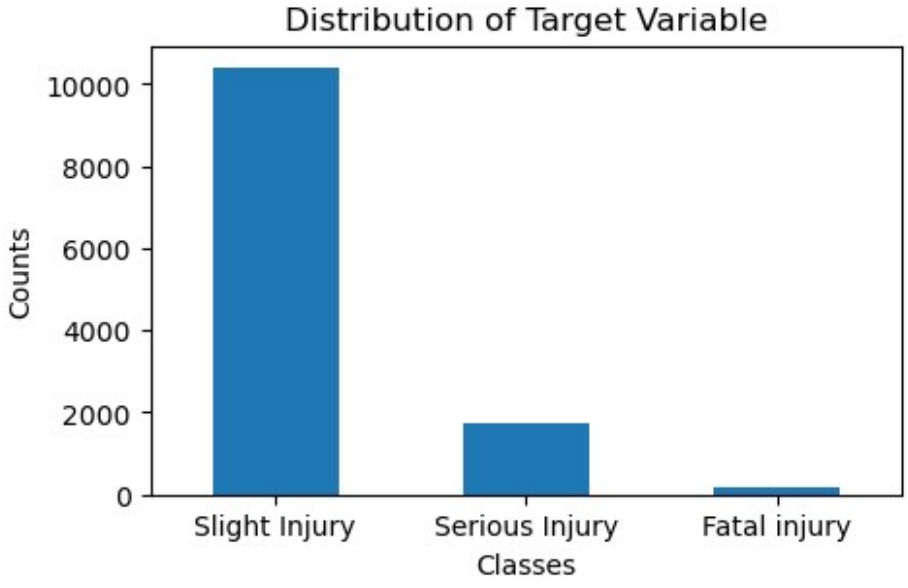



```python
# Accident severity count distribution
target_count = data['Accident_severity'].value_counts()
print('Class 0:', target_count[0])
print('Class 1:', target_count[1])
print('Proportion:', round(target_count[0] / target_count[1], 2), ':
1')

# Create a bar plot
plt.figure(figsize=(5, 3))
target_count.plot(kind='bar')
plt.title('Distribution of Target Variable')
```

```
plt.xlabel('Classes')
plt.ylabel('Counts')
plt.xticks(rotation='horizontal')
plt.show()
```

```
Class 0: 10415
Class 1: 1743
Proportion: 5.98 : 1
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\4021125373.py:3:
FutureWarning: Series.__getitem__ treating keys as positions is
deprecated. In a future version, integer keys will always be treated
as labels (consistent with DataFrame behavior). To access a value by
position, use `ser.iloc[pos]`
  print('Class 0:', target_count[0])
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\4021125373.py:4:
FutureWarning: Series.__getitem__ treating keys as positions is
deprecated. In a future version, integer keys will always be treated
as labels (consistent with DataFrame behavior). To access a value by
position, use `ser.iloc[pos]`
  print('Class 1:', target_count[1])
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\4021125373.py:5:
FutureWarning: Series.__getitem__ treating keys as positions is
deprecated. In a future version, integer keys will always be treated
as labels (consistent with DataFrame behavior). To access a value by
position, use `ser.iloc[pos]`
  print('Proportion:', round(target_count[0] / target_count[1], 2), ':
1')
```



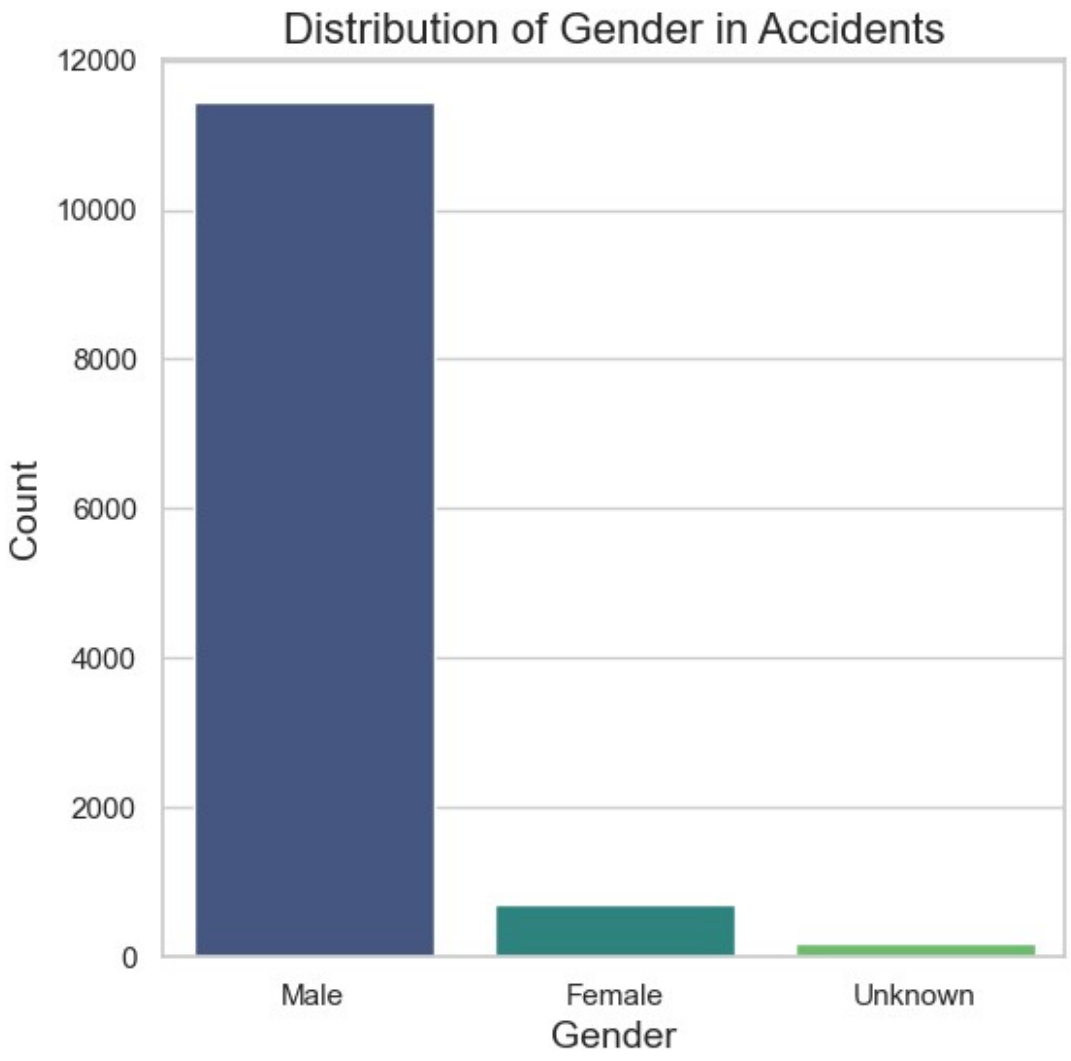

```
# Percentage calculation of Gender of the driver
((data.groupby(['Sex_of_driver']).size() /
```

```
data["Sex_of_driver"].count()) * 100).add_prefix('Accidents Caused by
')
```

```
Sex_of_driver
Accidents Caused by Female      5.691783
Accidents Caused by Male       92.862943
Accidents Caused by Unknown     1.445274
dtype: float64
```

```
# Set the style for seaborn
sns.set(style="whitegrid")

# Distribution of Gender using a bar chart
plt.figure(figsize=(6,6))
sns.countplot(x='Sex_of_driver', data=data, palette='viridis')
plt.title('Distribution of Gender in Accidents', fontsize=16)
plt.xlabel('Gender', fontsize=14)
plt.ylabel('Count', fontsize=14)
plt.show()
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\1875402105.py:6:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.countplot(x='Sex_of_driver', data=data, palette='viridis')
```




```
# Storing categorical features into a variable
categorical=[i for i in data.columns if data[i].dtype=='O']
print('The categorical variables are',categorical)
```

```
The categorical variables are ['Day_of_week', 'Age_band_of_driver',
'Sex_of_driver', 'Educational_level', 'Vehicle_driver_relation',
'Driving_experience', 'Type_of_vehicle', 'Owner_of_vehicle',
'Service_year_of_vehicle', 'Defect_of_vehicle',
'Area_accident_occured', 'Lanes_or_Medians', 'Road_allignment',
'Types_of_Junction', 'Road_surface_type', 'Road_surface_conditions',
'Light_conditions', 'Weather_conditions', 'Type_of_collision',
'Vehicle_movement', 'Casualty_class', 'Sex_of_casualty',
'Age_band_of_casualty', 'Casualty_severity', 'Work_of_casuality',
'Fitness_of_casuality', 'Pedestrian_movement', 'Cause_of_accident',
'Accident_severity']
```

```
# Storing numerical features into a variable
numerical=[i for i in data.columns if data[i].dtype!='O']
print('The numerica variables are',numerical)
```

```
The numerica variables are ['Time', 'Number_of_vehicles_involved',
'Number_of_casualties', 'hour']
```

```
# Overview of the data
ff = pd.crosstab(index=data['Cause_of_accident'],
columns=data['Accident_severity'], margins=True)
pd.crosstab(index=data['Cause_of_accident'],
columns=data['Accident_severity'], margins=True)
```

```
print("P(Cause = Speed | Severity = Fatal) = " +
str(((ff.iloc[2,2]+ff.iloc[15,2])/(data.where(data["Accident_severity"
] == 2)['Accident_severity'].count()))))
ff
```

```
P(Cause = Speed | Severity = Fatal) = inf
```

```
C:\Users\shiva\AppData\Local\Temp\ipykernel_736\2289211363.py:5:
RuntimeWarning: divide by zero encountered in scalar divide
  print("P(Cause = Speed | Severity = Fatal) = " +
str(((ff.iloc[2,2]+ff.iloc[15,2])/(data.where(data["Accident_severity"
] == 2)['Accident_severity'].count()))))
```

Accident_severity	Fatal injury	Serious Injury	\
Cause_of_accident			
Changing lane to the left	16	206	
Changing lane to the right	23	260	
Driving at high speed	2	31	
Driving carelessly	22	209	
Driving to the left	4	53	
Driving under the influence of drugs	5	46	
Drunk driving	0	3	
Getting off the vehicle improperly	3	29	
Improper parking	1	2	
Moving Backward	26	162	
No distancing	20	303	
No priority to pedestrian	5	95	
No priority to vehicle	13	149	
Other	7	64	
Overloading	2	10	
Overspeed	1	15	
Overtaking	4	75	
Overturning	2	23	
Turnover	2	6	
Unknown	0	2	
All	158	1743	

Accident_severity	Slight Injury	All

Cause_of_accident		
Changing lane to the left	1251	1473
Changing lane to the right	1525	1808
Driving at high speed	141	174
Driving carelessly	1171	1402
Driving to the left	227	284
Driving under the influence of drugs	289	340
Drunk driving	24	27
Getting off the vehicle improperly	165	197
Improper parking	22	25
Moving Backward	949	1137
No distancing	1940	2263
No priority to pedestrian	621	721
No priority to vehicle	1045	1207
Other	385	456
Overloading	47	59
Overspeed	45	61
Overtaking	351	430
Overturning	124	149
Turnover	70	78
Unknown	23	25
All	10415	12316