Setup Dataset:

1.) Open This Link and Download the Dataset(CSV file): "
https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents "

2.) Name the Dataset file to "US_Accidents_Dec21_updated.csv"

3.) Now, move this csv file into folder "eda\src\usaccidents\US_Accidents_Dec21_updated.csv", so csv file is inside usaccidents folder

Installation of Required libraries:

Install the following libraries: (Pandas), (Numpy), (Folium), (Seaborn), (Matplotlib)

pip install pandas
pip install numpy
pip install folium
pip install seaborn

(Run below two in order to install matplotlib)
python -m pip install -U pip
python -m pip install -U matplotlib

How to run the program:

1.) Open the "eda\src" folder inside the terminal

2.) Now run the following command in terminal: "python manage.py runserver"

3.) Click (Ctrl+Click) on the link "http://127.0.0.1:8000/" from inside the terminal

4.) Type and append "/mainpage" to the link in the browser. (eg., "http://127.0.0.1:8000/mainpage")

5.) It will take approx 3 minutes and the Data Analysis will be loaded on the webpage.

How does Program work:

1.) Below is the urls.py file which when /mainpage is requested it calls the “main_page” function from the views.py file inside the application “usaccidents”

```
from django.contrib import admin
from django.urls import path


from usaccidents.views import main_page

urlpatterns = [

    path('mainpage/',main_page,name='mainpage'),
    path('admin/', admin.site.urls),
]
```

2.) Below is the main_page() function called from urls.py request :

```
def main_page(request, *args, **kwargs):
    df,map_html=main()
    my_context={"dict":df, "map_html":map_html}
    return render(request, "main_page.html",my_context)
```

3.) Below is the main function which executes entire data analysis and generate output to view it in “main_page.html” file

```
def main():
    datafile = 'usaccidents\\US_Accidents_Dec21_updated.csv'
    my_list=[]

    # Read CSV 0
    df = pd.read_csv(datafile)
    my_list.append(df)

    # Read Columns 1
    my_list.append(df.columns)

    # Read  Info 2
    # print(df.info())
    output1 = StringIO()
    df.info(buf=output1)
    my_list.append(output1.getvalue())
```

```
    #DF Describe Std  Mean  Devaiation 3
    # pd.set_option('display.max_columns',None)
    my_list.append(df.describe())
    # pd.reset_option('display.max_columns')

    # Missing values per columns
    missing_percentages = df.isna().sum().sort_values(ascending=False) /
len(df)
    ax = missing_percentages[missing_percentages != 0].plot(kind='barh',
figsize=(8,4.8))
    plt.subplots_adjust(left=0.2)
    plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\missing.jpg')

    #Remove columns that you don't want to use but not good practice

    #Exploratory Analysis & Visualisation

    #Cities
    my_list.append(df.City) #[4]
    my_list.append(len(df.City.unique())) #[5]

    #Cities and their values of accidents 6
    cities_by_accidents = df.City.value_counts()
    my_list.append(cities_by_accidents)

    #Top 20 cities by accidents 7
    my_list.append(cities_by_accidents[:20])

    ax = cities_by_accidents[:20].plot(kind='barh', figsize=(8,4.8))
    plt.subplots_adjust(left=0.2)
    plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\top20city.jpg')



    sns.histplot(cities_by_accidents, log_scale=True)
    plt.subplots_adjust(left=0.2, bottom=0.2)
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\cityvsacci.jpg')

    # cities with only 1 accident count 8
    my_list.append(cities_by_accidents[cities_by_accidents == 1])


    #Timing Analaysis
    my_list.append(df.Start_Time) #[9]
```

```
    df['Start_Time'] = df['Start_Time'].astype(str)
    df_filtered=df[df['Start_Time'].str.match(r'^\d{4}-\d{2}-\d{2}
\d{2}:\d{2}:\d{2}$') == True]

    # df_filtered['Start_Time'] =
pd.to_datetime(df_filtered['Start_Time'],format='%Y-%m-%d %H:%M:%S')
    # sns.histplot(df_filtered['Start_Time'].dt.hour, bins=24, kde=False,
stat='density')
    # plt.subplots_adjust(left=0.2, bottom=0.2)
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\starttime.jpg')

#Weekdays vs their density of records present
    # sns.histplot(df_filtered['Start_Time'].dt.dayofweek, bins=7, kde=False,
stat='density')
    # plt.subplots_adjust(left=0.2, bottom=0.2)
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\weekdata.jpg')

#Sunday density of accidents on hours
    # sundays_start_time =
df_filtered['Start_Time'][df_filtered['Start_Time'].dt.dayofweek == 6]
    # sns.histplot(sundays_start_time.dt.hour, bins=24, kde=False,
stat='density')
    # plt.subplots_adjust(left=0.2, bottom=0.2)
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\sundaydata.jpg')

#Monday Density of accidents on hours
    # monday_start_time =
df_filtered['Start_Time'][df_filtered['Start_Time'].dt.dayofweek == 0]
    # sns.histplot(monday_start_time.dt.hour, bins=24, kde=False,
stat='density')
    # plt.subplots_adjust(left=0.2, bottom=0.2)
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\mondaydata.jpg')


#Visualization and Positional Analysis with density of accidents
    my_list.append(df.Start_Lat) #[10]
    my_list.append(df.Start_Lng) #[11]


#Plot in Map
    # sample_df = df.sample(int(0.1 * len(df)))
    # sns.scatterplot(x=sample_df.Start_Lng, y=sample_df.Start_Lat,
size=0.001)
    # plt.subplots_adjust(left=0.2, bottom=0.2)
```

```
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\accipos.jpg')


#Interactive map with density of accidents in USA
    zip(list(df.Start_Lat), list(df.Start_Lng))
    sample_df = df.sample(int(0.001 * len(df)))
    lat_lon_pairs = list(zip(list(sample_df.Start_Lat),
list(sample_df.Start_Lng)))
    map = folium.Map()
    HeatMap(lat_lon_pairs).add_to(map)
    map_html = map.get_root().render()


#Temperature Analysis
    my_list.append(df['Temperature(F)']) #[12]

    df['Temperature(C)']=(df['Temperature(F)']-32)*(5/9)
    df['Temperature(C)'].hist(bins=range(-20,45),density=True)
    plt.xlabel('Temperature(C)')
    plt.ylabel('Density of accidents')
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\accivstemp.jpg')


#Weather Conditions during accidents
    my_list.append(df['Weather_Condition'].unique()) #[13]
    weather = df['Weather_Condition'].value_counts()
    percent = weather/weather.sum()*100
    percent = percent.apply(lambda x: x if x>=0.7 else None)
    percent['Others']=percent.isnull().sum()
    percent=percent.dropna()
    weather.plot.pie(autopct='%.1f%%')
    # plt.savefig('E:\\Win K\\4th
SEM\\PSC\\direct1\\src\\static\\images\\weather.jpg')



    return my_list,map_html
```

Below is the HTML file which renders from the views.py after request:

```
<html >
<head>
    <title>Document</title>
    <style>
        @import
url('https://fonts.googleapis.com/css2?family=Poppins&family=Roboto&display=sw
ap');
        *{
            margin:0;
            padding:0;
            box-sizing: border-box;
            font-family: 'Poppins';
        }

        body{
            font-size: x-large;
            font-family: 'Poppins';
        }
        ul li{
            font-size:large;
        }
        h1{
            text-align: center;
            text-decoration: underline;
            text-underline-offset: 5px;
        }
        .maintitle{
            font-size: 30px;
            margin-top: 30px;
            border: 2px solid #333;
            /* border-radius: 3px; */
            background-color: darkblue;
            color:rgb(225,225,225);
            text-align: center;
            padding:5px 0;
        }
        .title1{
            font-size:x-large;
            margin-top: 17px;
            border-top: 2px solid #333;
            border-left: 6px solid #333;
            border-right: 6px solid #333;
```

```
            padding: 6px;
            font-weight: bold;
            text-indent: 10px;
            text-align: center;
            text-decoration: underline;
            text-underline-offset: 6px;
        }
        .prestyle{
            /* border: 3px solid #333; */
            font-size: 14px;
            font-family: 'Cascadia Code';
            color:black;
            letter-spacing: 0.1ch;
        }
        div{
            font-size: large;
            font-weight:600;
        }
        .datas{
            font-size:14px;
            padding: 6px;
            margin: 5px 5px;
            border:1.5px solid #333;
            background-color: aliceblue;
            border-radius: 4px;
            display: inline-block;
        }
        .datas:hover{
            scale: 1.02;
            transition:0.4s;
        }
        .imgs img{
            display: block;
            margin:0 auto;
        }
    </style>
</head>
<body>
    <h1>US Accidents(2016-2021) Exploratory Data Analysis</h1>

    <h2>DataSet Information</h2>
    <ul>
        <li>Source     : Kaggle</li>
        <li>Contents   : Information about Accidents</li>
        <li>How is it Useful : Useful to take preventive measures to decrease
accidents</li>
        <li>Note : Data does not contain info about New York (Top Populated
City)</li>
```

```
    </ul>

    <p class="maintitle">Data Cleaning and Preparations</p>

    <p class="title1">Preview of dataset(Read File)</p>
        <pre class="prestyle">
            {{ dict.0 }}
        </pre>
    <p class="title1">The List of Columns</p>
        <pre class="prestyle">{{ dict.1 }}</pre>

    <p class="title1">DataSet Info</p>
        <pre class="prestyle">{{ dict.2 }}</pre>

    <p class="title1">DataSet Numeric Value Analysis</p>
        <pre class="prestyle">{{ dict.3 }}</pre>

    <p class="title1">Missing Values per Columns</p>
        <p class="imgs">
            {% load static %}
            <img src="{% static 'images/missing.jpg' %}"
alt="Missing_value_Percentages">
        </p>
            <div>
                  >>> Here we can remove 'Number' Column due to
more no of missing Values
                but it is not a good practice so we stay with our current data
                and not use these columns
            </div>


    <p class="maintitle">Exploratory analysis & visualization on: Cities</p>

    <p class="title1">City Dataset Information</p>
        <pre class="prestyle">{{ dict.4 }}</pre>

    <p class="title1">No of Unique cities present in Dataset:</p>
    <div>
           >>> There are {{ dict.5 }} unique cities where
accidents occurred
    </div>

    <p class="title1">Cities with their no of accidents</p>
    <pre class="prestyle">{{ dict.6 }}</pre>

    <p class="title1">Top 20 cities with their accidents count and plot</p>
        <pre class="prestyle">{{ dict.7 }}</pre>

        <p class="imgs">
```

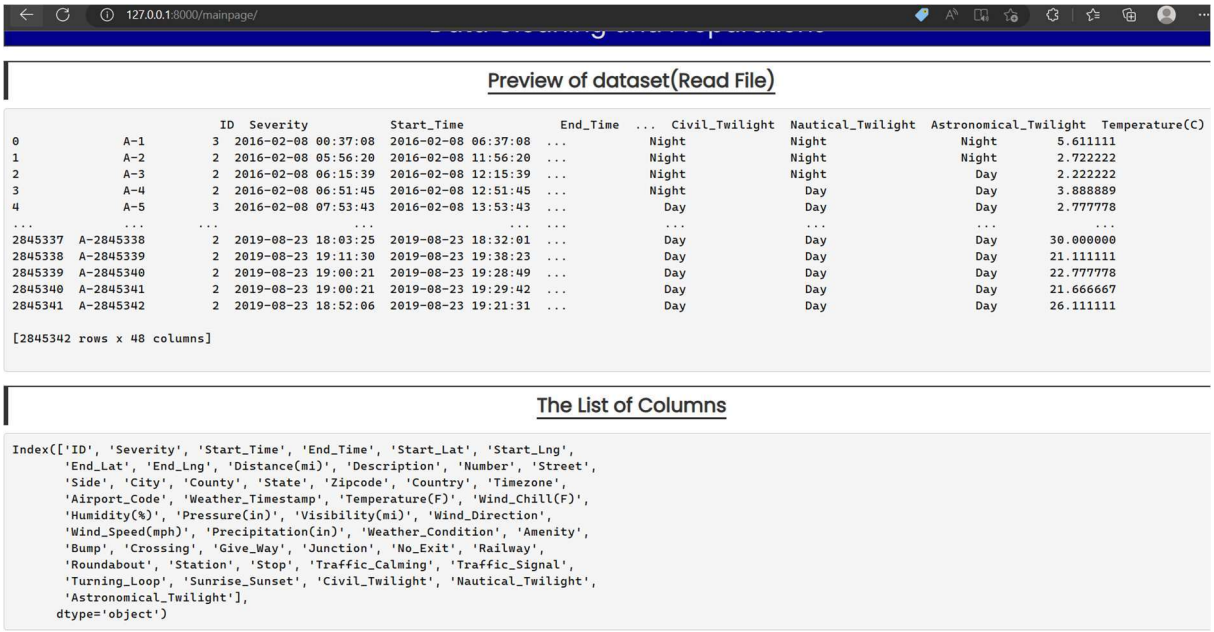
```
            {% load static %}
            <img src="{% static 'images/top20city.jpg' %}" alt="">
        </p>

    <p class="title1">Cities vs Accidents</p>
        <p class="imgs">
            {% load static %}
            <img src="{% static 'images/cityvsacci.jpg' %}" alt="">
        </p>

    <p class="title1">Cities with only 1 accident count</p>
        <pre class="prestyle">{{ dict.8 }}</pre>


    <p class="maintitle">Exploratory analysis & visualization on: Accident
Timings</p>

    <p class="title1">Accidents Timing list</p>
    <pre class="prestyle">{{ dict.9 }}</pre>

    <p class="title1">Plot Analysis of Accident Timing</p>
        <p class="imgs">
            {% load static %}
            <img src="{% static 'images/starttime.jpg' %}" alt="">
            <div>
                  >>> We can conclude that most of the
accidents occurr between 3:00 PM to 5:00 PM.
                And 7:00 AM to 9:00 AM.
            </div>
        </p>

    <p class="title1">Week days (0-Monday, 6-Sunday) vs density of records
present</p>
    <p class="imgs">
        {% load static %}
        <img src="{% static 'images/weekdata.jpg' %}" alt="">
        <div>
             >>> Weekend Days data has less density compared
to Workdays
        </div>
    </p>

    <p class="title1">Density of Accidents on Sunday</p>
    <p class="imgs">
        {% load static %}
        <img src="{% static 'images/sundaydata.jpg' %}" alt="">
        <div>
             >>> There's a peak in accidents at 00:00 AM
night.
```

```
            Most of the Accidents happen between 12:00 PM to 5:00 PM
        </div>
    </p>

    <p class="title1">Density of Accidents on Monday</p>
    <p class="imgs">
        {% load static %}
        <img src="{% static 'images/mondaydata.jpg' %}" alt="">
        <div>
              >>> Major Accidents occur in Morning 7:00AM to
9:00AM and between 2:00 PM to 4:30 PM. Most probably due to people going
            to work in morning and returning back at noon till evening.
        </div>
    </p>

    <p class="maintitle">Visualization and Positional Analysis: Accident
Density</p>

    <p class="title1">Latitude data</p>
        <pre class="prestyle">{{ dict.10 }}</pre>

    <p class="title1">Longitude data</p>
        <pre class="prestyle">{{ dict.11 }}</pre>

    <p class="title1">Map of Latitude and Longitude points</p>
    <p class="imgs">
        {% load static %}
        <img src="{% static 'images/accipos.jpg' %}" alt="">
        <div>
              >>> Density of accidents are significantly more
on Eastern and Western side of USA
        </div>
    </p>

    <p class="title1">Interactive Map of Accident Desity</p>
    <div style="width:1100px; height:520px; margin:0 auto;">
        {{ map_html|safe }}
    </div>

    <p class="maintitle">Exploratory analysis & visualization on:
Temperature</p>

    <p class="title1">Temperature Readings</p>
        <pre class="prestyle">{{ dict.12 }}</pre>

        <div>
               >>> Temperature Data is in Fahrenheit
        </div>
```

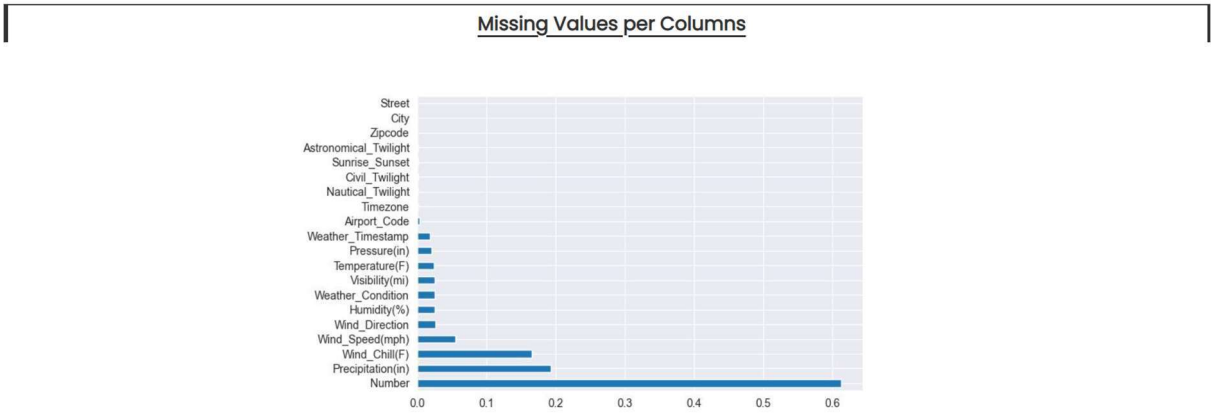
```
    <p class="title1">Density of Accidents vs Temperature</p>
        <p class="imgs">
            {% load static %}
            <img src="{% static 'images/accivstemp.jpg' %}" alt="">
            <div>
                  >>> There's no significant relation of
occurrence of accidents with Temperature.
                However, below 10°C accidents are less likely to occur.
            </div>
        </p>


    <p class="maintitle">Exploratory analysis & visualization on: Weather
Conditions</p>

    <p class="title1">Unique Weather Conditions During Accidents</p>
        <p>
            <!-- {{ dict.13 }} -->
            {% for x in dict.13 %}
            <div class="datas"> {{ x }} </div>
            {% endfor %}
        </p>
        <p class="imgs">
            {% load static %}
            <img src="{% static 'images/weather.jpg' %}" alt="">
            <div>
                  >>> It can be concluded that during difficult
weather conditions like Rain, Fog or ThunderStorm less accidents occur.
                And Most of the Accidents occur in Normal Weather Conditions.
            </div>
        </p>

</body>
</html>
```

Below are some insights of the analysis done on the data:

Preview of dataset:

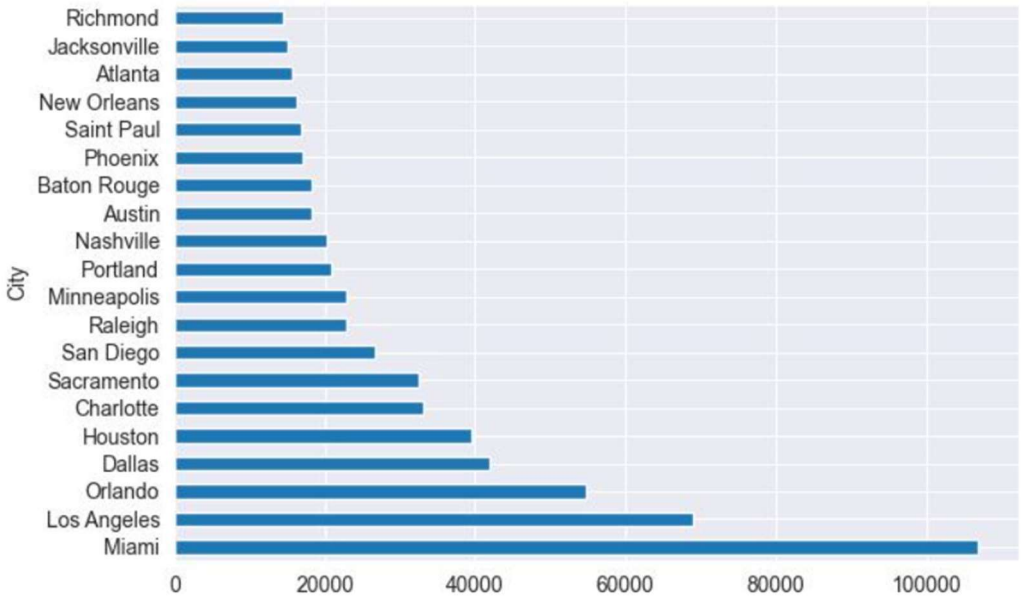


Data Cleaning (Missing Values) of dataset:



>>> Here we can remove 'Number' Column due to more no of missing Values but it is not a good practice so we stay with our current data and not use these columns

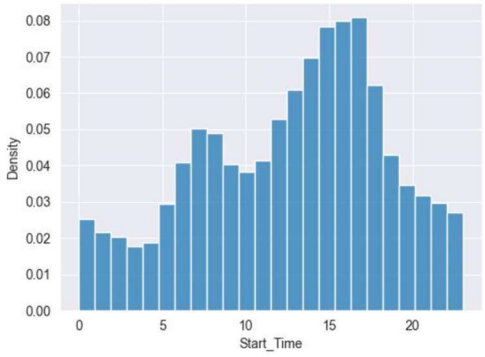
Top cities with accidents of dataset:



Cities vs Accidents

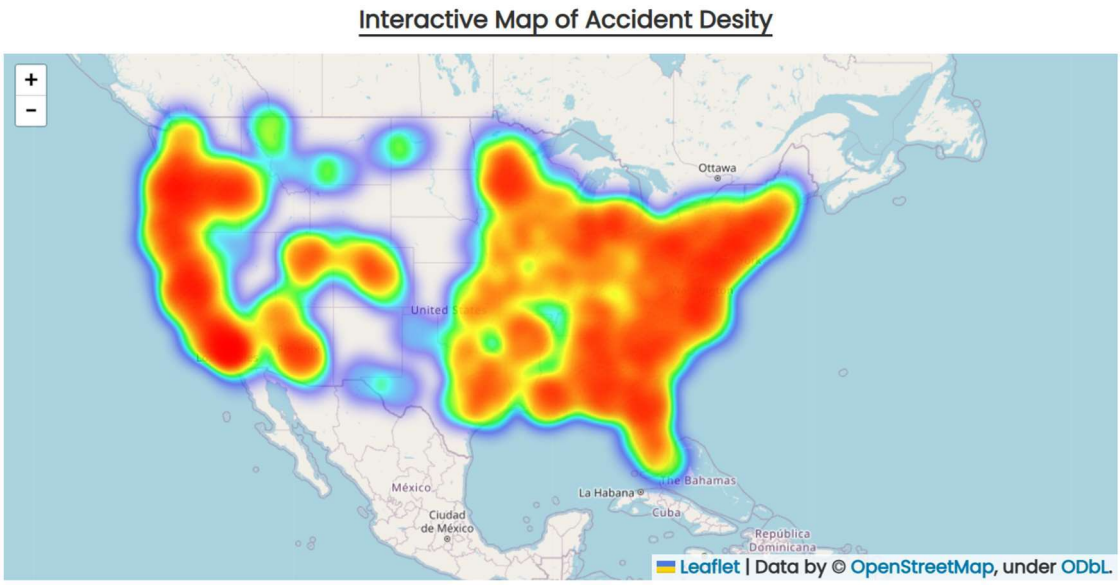
Accidents Timing of dataset:

Plot Analysis of Accident Timing



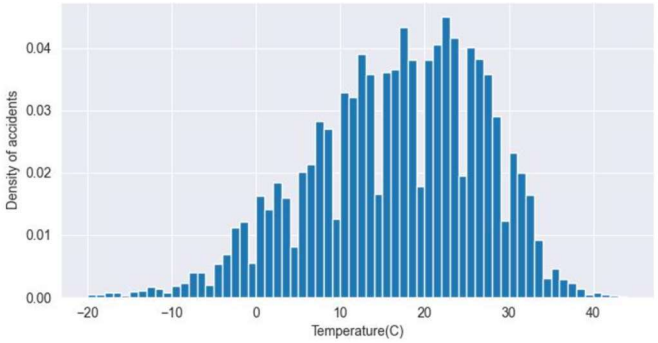
>>> We can conclude that most of the accidents occurr between 3:00 PM to 5:00 PM. And 7:00 AM to 9:00 AM.

Interactive Heatmap of accidents location from dataset:



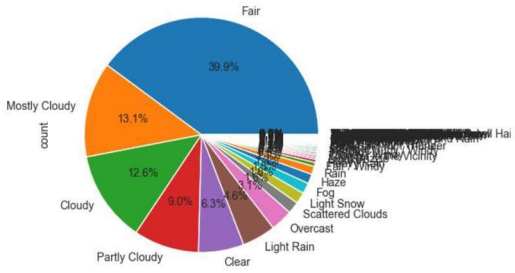
Temperature and Weather Conditions:

>>> Temperature Data is in Fahrenheit



>>> There's no significant relation of occurrence of accidents with Temperature. However, below 10°C accidents are less likely to occur.

Light Freezing Rain / Windy | Duststorm | Light Snow and Sleet / Windy | Heavy Rain Shower / Windy | Sand / Dust Whirlwinds / Windy | Light Rain Shower / Windy | Thunder and Hail | Freezing Rain | Heavy Sleet | Sleet | Freezing Drizzle | Snow and Sleet / Windy | Heavy Freezing Drizzle | Heavy Freezing Rain | Blowing Sand | Thunder / Wintry Mix / Windy | Mist / Windy | Sleet / Windy | Patches of Fog / Windy | Sand / Dust Whirls Nearby | Heavy Rain Shower | Drifting Snow | Heavy Blowing Snow | Low Drifting Snow | Light Blowing Snow | Heavy Rain Showers | Light Haze | Heavy Thunderstorms with Small Hail | Heavy Snow with Thunder | Thunder and Hail / Windy



>>> It can be concluded that during difficult weather conditions like Rain, Fog or ThunderStorm less accidents occur. And Most of the Accidents occur in Normal Weather Conditions.