CSE 4/587 Fall 2023
Project Phase #1

Team
Name: Sanjukta Baruah
UBIT Name: sbaruah
Name: Sujay Shrivastava
UBIT Name: sujayshr

1. **Problem Statement: Form a title and problem statement that clearly state the problem and questions you are trying to answer.**
   Additionally:
   a. **[5 marks] Discuss the background of the problem leading to your objectives. Why is it a significant problem?**
   b. **[5 marks] Explain the potential of your project to contribute to your problem domain. Discuss why this contribution is crucial?**

Title: Analyzing Car Accidents in the USA (2016-2021): Identifying Contributing Factors and Predicting Severity

Problem Statement: Our analysis seeks to identify primary factors contributing to car accidents and predict their severity from 2016 to 2021 in the USA.

a. We're tackling the issue of enhancing road safety by delving into car accidents. This is a crucial concern because accidents on the road can result in injuries and damage. We're driven to make our roads safer, but the challenge is that the available data is often incomplete, leaving out key details needed to find effective solutions. Our motivation for selecting this dataset was influenced by our own experiences living in the South Campus area, where we noticed a high frequency of accidents near a particular three-road junction. One observable issue contributing to these accidents is the confusion between pedestrian walk signals and red traffic lights during turns.

b. This project carries significant potential for enhancing road safety, as we investigate the number of accidents based on various influential factors, including weather conditions, time, visibility, amenities, intersections, road irregularities, state, city, and other key features. This contribution is highly significant because a staggering 3,700 individuals lose their lives every day in fatal accidents worldwide. If our model proves successful, even with an 85% accuracy rate, we can predict when and where accidents are most likely to occur, along with the prevailing weather conditions. This knowledge can enable us to proactively prevent accidents by taking additional safety measures, such as optimizing traffic signals and improving their synchronization. Furthermore, it opens the door to potential modifications in road and bump materials to make them safer. Ultimately, this project has the potential to save lives and reduce the toll of accidents on our roadways.

We'll address specific questions to better understand and tackle our problem:

1. When do accidents usually happen?
a. Daylight compared to nighttime
b. Weekdays versus weekends
c. Peak times during the day

2. Where do accidents tend to occur?
a. By county
b. Within city boundaries
c. Specific zip code areas
d. Along particular street segments
e. Precise accident locations

3. Which weather conditions are most closely linked to accidents?
a. Identifying the most hazardous weather conditions for accidents
b. Pinpointing the top three weather conditions contributing to accidents
c. Evaluating the severity of weather circumstances during each accident event

---

2. **Data Sources [5 marks]: Collect your data. Your data can come from multiple sources. For example, Medical, Bank, sports, health, Kaggle, Amazon reviews, Twitter, Youtube, Reddit, etc. This data has to be large enough for the data analysis to yield significance. At least 2000 rows/records. Although there is no requirement for the number of columns, as this will vary based on project, your data should contain enough columns to successfully complete the deliverables for Phase 1. If your data is already clean, or does not have a sufficient number of columns to complete Phase 1, you must find different/additional data sources. You must cite and link your data sources in the report.**

   → We have sourced our dataset from Kaggle, focusing on US Accidents spanning from 2016 to 2021. You can access the dataset through the following link: https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents

   Description: This dataset offers a comprehensive view of car accidents across 49 states in the USA. It encompasses accident information gathered from February 2016 to March 2023, sourced from multiple APIs that continuously provide real-time traffic incident data. These APIs aggregate data from a variety of sources, including state and federal transportation departments, law enforcement agencies, traffic cameras, and road network sensors. At present, the dataset comprises around 7.7 million accident records, offering a valuable resource for analyzing and understanding traffic incidents in the United States.

Information about the data set

There are 47 columns, 2845342 rows and data types are as follows:

- Bool (13 columns)
- Object (20)
- float64 (13 columns)
- Int 64 (1 column)

```
----------------------------------------------------------------------
No. of Rows: 7728394        No. of Columns: 46
----------------------------------------------------------------------
                       DataType  Missing Values  Missing %  Unique Values
ID                       object               0       0.00        7728394
Source                   object               0       0.00              3
Severity                  int64               0       0.00              4
Start_Time               object               0       0.00        6131796
End_Time                 object               0       0.00        6705355
Start_Lat               float64               0       0.00        2428358
Start_Lng               float64               0       0.00        2482533
End_Lat                 float64         3402762      44.03        1568172
End_Lng                 float64         3402762      44.03        1605789
Distance(mi)            float64               0       0.00          22382
Description              object               5       0.00        3761578
Street                   object           10869       0.14         336306
City                     object             253       0.00          13678
County                   object               0       0.00           1871
State                    object               0       0.00             49
Zipcode                  object            1915       0.02         825094
Country                  object               0       0.00              1
Timezone                 object            7808       0.10              4
Airport_Code             object           22635       0.29           2045
Weather_Timestamp        object          120228       1.56         941331
Temperature(F)          float64          163853       2.12            860
Wind_Chill(F)           float64         1999019      25.87           1001
Humidity(%)             float64          174144       2.25            100
Pressure(in)            float64          140679       1.82           1144
Visibility(mi)          float64          177098       2.29             92
Wind_Direction           object          175206       2.27             24
Wind_Speed(mph)         float64          571233       7.39            184
Precipitation(in)       float64         2203586      28.51            299
Weather_Condition        object          173459       2.24            144
Amenity                    bool               0       0.00              2
Bump                       bool               0       0.00              2
Crossing                   bool               0       0.00              2
Give_Way                   bool               0       0.00              2
Junction                   bool               0       0.00              2
No_Exit                    bool               0       0.00              2
Railway                    bool               0       0.00              2
Roundabout                 bool               0       0.00              2
Station                    bool               0       0.00              2
Stop                       bool               0       0.00              2
Traffic_Calming            bool               0       0.00              2
Traffic_Signal             bool               0       0.00              2
Turning_Loop               bool               0       0.00              1
Sunrise_Sunset           object           23246       0.30              2
Civil_Twilight           object           23246       0.30              2
Nautical_Twilight        object           23246       0.30              2
Astronomical_Twilight    object           23246       0.30              2
----------------------------------------------------------------------
```

---

3. Data Cleaning/Processing [10 Marks]: Your dataset has to be cleaned and properly processed. Please submit a report where you explain each processing/cleaning step properly. We expect to see comments and markup for this step. In order to get full marks you must clearly document 7 (10 for 587 students) distinct processing/cleaning operations.

Data Cleaning

1. **<u>Data Fields Removal</u>**

   To streamline our analysis, we'll remove redundant or irrelevant fields from the dataset. Justification is given below:

S.No	**Irrelevant Fields/Features**	**Reason for removing**
1.	'Id' and 'Source'	These fields have minimal impact on our analysis.

2.	'End_Lat' and 'End_Lng'	Since we already have the initial coordinates, these fields are redundant.
3.	'Airport_Code'	Given that all the data relates to the USA, specifying the nearest airport code is unnecessary.
4.	'Country'	Since all the data is related to the USA, this field doesn't add extra value.
5.	'Weather_Timestamp'	We have more relevant weather-related fields.
6.	'Civil_Twilight', 'Nautical_Twilight', and 'Astronomical_Twilight'	These fields might not be directly relevant to our analysis.
7.	'Timezone'	We can obtain this information from other relevant fields.

```
columns_to_drop = ['ID', 'Country', 'Source', 'End_Lat', 'End_Lng', 'Civil_Twilight', 'Nautical_Twilight', 'Astronomical_Twilight','Street']
df = df.drop(columns=columns_to_drop, axis=1)
```

2. **<u>Mode Imputation:</u>**

   Impute empty value with most occurring value (or mode)

```
# Calculate the mode (most frequent city) for each 'State' group
mode_cities = df.groupby('State')['City'].transform(lambda x: x.mode()[0])

# Fill missing 'City' values with the mode for their respective 'State' group
df['City'].fillna(mode_cities, inplace=True)
```

   In mode imputation, the missing values in a feature column are replaced with the mode value, which represents the most frequently occurring value in that column. This approach is particularly suitable when dealing with skewed data. For instance, when addressing missing data in fields like 'city,' utilizing mode values can be a practical choice. It's worth noting that mode imputation can be applied to both numerical and categorical data, making it a versatile method for handling missing data.

3. **<u>Mean Imputation:</u>**
   - When mean imputation is employed, it ensures that the mean of the observed data remains intact. This means that even in scenarios where data is randomly missing, the resulting mean estimate remains unbiased and fair. This approach offers the added benefit of preserving the sample size, keeping it consistent with the full dataset.
   - Given that filling these columns with zero would lack meaningful context, we chose to populate them with their respective average values.

```
#filling missing values with interpolate method
df['Temperature(F)']=df['Temperature(F)'].fillna(df['Temperature(F)'].mean())
df['Wind_Chill(F)']=df['Wind_Chill(F)'].fillna(df['Wind_Chill(F)'].mean())
df['Humidity(%)']=df['Humidity(%)'].fillna(df['Humidity(%)'].mean())
df['Pressure(in)']=df['Pressure(in)'].fillna(df['Pressure(in)'].mean())
df['Wind_Speed(mph)']=df['Wind_Speed(mph)'].fillna(df['Wind_Speed(mph)'].mean())
df['Precipitation(in)']=df['Precipitation(in)'].fillna(df['Precipitation(in)'].mean())
```

4. We have decided to remove the columns 'Visibility(mi)', 'Wind_Direction', 'Weather_Condition', 'Timezone', and 'Sunrise_Sunset' from the dataset. This decision was made because these columns collectively account for approximately 3% of the entire dataset's null values. Despite the removal of rows containing null values in these columns, we are left with a substantial 7,475,297 observations in our dataset, given its large size.

```
columns_to_check = ['Description','Visibility(mi)', 'Wind_Direction','Weather_Condition','Timezone','Sunrise_Sunset']
df.dropna(subset=columns_to_check, inplace=True)
```

5. **<u>Data Deduplication: Ensuring Uniqueness</u>**

```
print("Num of duplicates", df.duplicated().sum())
df = df.drop_duplicates()
```

```
Num of duplicates 102358
```

We conducted a review of our dataset and identified the presence of 500 duplicate values. To maintain data integrity and ensure our analysis is based on unique and distinct records, we made the decision to remove these duplicates. This step not only streamlines the dataset but also helps us avoid any unintentional biases that duplicate entries might introduce.

6. **<u>Handling Near-Duplicates:</u>**
Given the dataset's substantial size and the relatively insignificant nature of these columns, we made the decision to eliminate these column's ('Visibility(mi),' 'Wind_Direction,' 'Weather_Condition,' 'Timezone,' and 'Sunrise_Sunset)those rows that have null/missing values. It's important to note that these rows collectively accounted for only 3% of the dataset. Even after their removal, we retained a substantial 7,475,297 rows within the dataset, ensuring that it remains both manageable and meaningful for our analysis.

```
No. of Weather Conditions: 143

List of unique weather conditions: ['Light Rain', 'Overcast', 'Mostly Cloudy', 'Rain', 'Light Snow', 'Haze', 'Scattered Clouds', 'Partly Cloudy', 'Clear', 'Snow', 'Light Freezing Drizzl
e', 'Light Drizzle', 'Fog', 'Shallow Fog', 'Heavy Rain', 'Light Freezing Rain', 'Cloudy', 'Drizzle', 'Light Rain Showers', 'Mist', 'Smoke', 'Patches of Fog', 'Light Freezing Fog', 'Light
Haze', 'Light Thunderstorms and Rain', 'Thunderstorms and Rain', 'Fair', 'Volcanic Ash', 'Blowing Sand', 'Blowing Dust / Windy', 'Widespread Dust', 'Fair / Windy', 'Rain Showers', 'Light
Rain / Windy', 'Hail', 'Heavy Drizzle', 'Showers in the Vicinity', 'Light Rain Shower', 'Light Rain with Thunder', 'Mostly Cloudy / Windy', 'Partly Cloudy / Windy', 'Thunder in the Vicin
ity', 'T-Storm', 'Heavy Thunderstorms and Rain', 'Thunderstorm', 'Thunder', 'Heavy T-Storm', 'Funnel Cloud', 'Heavy T-Storm / Windy', 'Blowing Snow', 'Light Thunderstorms and Snow', 'Hea
vy Snow', 'Low Drifting Snow', 'Light Ice Pellets', 'Ice Pellets', 'Squalls', 'N/A Precipitation', 'Cloudy / Windy', 'Light Fog', 'Snow Grains', 'Snow Showers', 'Heavy Thunderstorms and
Snow', 'Rain / Windy', 'Heavy Rain / Windy', 'Heavy Ice Pellets', 'Light Snow / Windy', 'Heavy Freezing Rain', 'Small Hail', 'Heavy Rain Showers', 'Thunder / Windy', 'Drizzle and Fog',
'T-Storm / Windy', 'Blowing Dust', 'Smoke / Windy', 'Haze / Windy', 'Tornado', 'Light Drizzle / Windy', 'Widespread Dust / Windy', 'Wintry Mix', 'Wintry Mix / Windy', 'Light Snow with Th
under', 'Fog / Windy', 'Light Snow Shower', 'Sleet', 'Light Snow and Sleet', 'Rain Shower', 'Snow and Sleet', 'Snow / Windy', 'Light Sleet', 'Heavy Snow / Windy', 'Freezing Drizzle', 'Li
ght Freezing Rain / Windy', 'Thunder / Wintry Mix', 'Blowing Snow / Windy', 'Freezing Rain', 'Light Snow and Sleet / Windy', 'Snow and Sleet / Windy', 'Sleet / Windy', 'Heavy Freezing Ra
in / Windy', 'Squalls / Windy', 'Light Rain Shower / Windy', 'Snow and Thunder / Windy', 'Light Sleet / Windy', 'Sand / Dust Whirlwinds', 'Drizzle / Windy', 'Sand / Dust Whirls Nearby',
'Thunder and Hail', 'Heavy Sleet', 'Freezing Rain / Windy', 'Light Snow Shower / Windy', 'Partial Fog', 'Thunder / Wintry Mix / Windy', 'Patches of Fog / Windy', 'Rain and Sleet', 'Light
Snow Grains', 'Partial Fog / Windy', 'Sand / Dust Whirlwinds / Windy', 'Snow and Thunder', 'Heavy Snow with Thunder', 'Light Snow Showers', 'Heavy Blowing Snow', 'Light Hail', 'Heavy Smo
ke', 'Sand', 'Heavy Thunderstorms with Small Hail', 'Light Thunderstorm', 'Heavy Freezing Drizzle', 'Light Blowing Snow', 'Thunderstorms and Snow', 'Dust Whirls', 'Mist / Windy', 'Rain S
hower / Windy', 'Sleet and Thunder', 'Heavy Sleet and Thunder', 'Drifting Snow / Windy', 'Shallow Fog / Windy', 'Thunder and Hail / Windy', 'Heavy Sleet / Windy', 'Sand / Windy', 'Heavy
Rain Shower / Windy', 'Blowing Snow Nearby', 'Heavy Rain Shower', 'Drifting Snow']
```

7. **<u>One-hot Encoding</u>**

Conversion of Boolean Values for Model Compatibility:

- We transformed Boolean values into 0s and 1s, a conversion that simplifies the data for better compatibility with the models we will build in phase 2.

Enhancing Predictions through One-Hot Encoding:

- One-hot encoding represents a critical step in improving the predictive power and classification accuracy of a model.
- This essential process involves converting categorical data variables into a format suitable for input into machine learning and deep learning algorithms.

No. of Rows: 7475293 No. of Columns: 37

	DataType	Missing Values	Missing %	Unique Values
Severity	int64	0	0.0	4
Start_Time	object	0	0.0	5959099
End_Time	object	0	0.0	6508383
Start_Lat	float64	0	0.0	2359788
Start_Lng	float64	0	0.0	2409544
Distance(mi)	float64	0	0.0	21871
Description	object	0	0.0	3652561
City	object	0	0.0	12246
County	object	0	0.0	1813
State	object	0	0.0	49
Zipcode	object	0	0.0	802726
Timezone	object	0	0.0	4
Airport_Code	object	0	0.0	1996
Weather_Timestamp	object	0	0.0	933052
Temperature(F)	float64	0	0.0	857
Wind_Chill(F)	float64	0	0.0	1001
Humidity(%)	float64	0	0.0	101
Pressure(in)	float64	0	0.0	1138
Visibility(mi)	float64	0	0.0	91
Wind_Direction	object	0	0.0	24
Wind_Speed(mph)	float64	0	0.0	185
Precipitation(in)	float64	0	0.0	293
Weather_Condition	object	0	0.0	143
Amenity	int64	0	0.0	2
Bump	int64	0	0.0	2
Crossing	int64	0	0.0	2
Give_Way	int64	0	0.0	2
Junction	int64	0	0.0	2
No_Exit	int64	0	0.0	2
Railway	int64	0	0.0	2
Roundabout	int64	0	0.0	2
Station	int64	0	0.0	2
Stop	int64	0	0.0	2
Traffic_Calming	int64	0	0.0	2
Traffic_Signal	int64	0	0.0	2
Turning_Loop	int64	0	0.0	1
Sunrise_Sunset	object	0	0.0	2

8. **<u>Label Encoding – Feature Engineering</u>**

We used Label Encoding on specific categorical columns in our dataset, such as 'State,' 'County,' 'City,' 'Wind_Direction,' 'Weather_Condition,' and 'Sunrise_Sunset.' The purpose of this transformation was to make these columns more machine-friendly.

Label Encoding helps turn words into numbers, making it easier for machine learning models to work with this data. By doing this, we make sure that our models can understand and use the information in these categorical columns effectively. This step is important for improving the dataset's suitability for machine learning, as it helps the models include these categorical factors in their predictions. Label Encoding is a critical

part of feature engineering, enhancing the models' ability to make meaningful predictions.

```
state_encoder = LabelEncoder()
county_encoder = LabelEncoder()
city_encoder = LabelEncoder()
wind_direction_encoder = LabelEncoder()
weather_condition_encoder = LabelEncoder()
sunrise_sunset_encoder = LabelEncoder()

# Encode each column and add encoded columns to the DataFrame
object_columns_df["State_LabelEncoded"] = state_encoder.fit_transform(object_columns_df["State"])
object_columns_df["County_LabelEncoded"] = county_encoder.fit_transform(object_columns_df["County"])
object_columns_df["City_LabelEncoded"] = city_encoder.fit_transform(object_columns_df["City"])
object_columns_df["Wind_Direction_LabelEncoded"] = wind_direction_encoder.fit_transform(object_columns_df["Wind_Direction"])
object_columns_df["Weather_Condition_LabelEncoded"] = weather_condition_encoder.fit_transform(object_columns_df["Weather_Condition"])
object_columns_df["Sunrise_Sunset_LabelEncoded"] = sunrise_sunset_encoder.fit_transform(object_columns_df["Sunrise_Sunset"])
```

9. **Datetime Consistency**

We conducted a collective UTC transformation on specific datetime columns within the dataset, namely "End_Time" and "Start_Time". This transformation serves the unified purpose of ensuring that these temporal attributes are consistently represented in Coordinated Universal Time (UTC).

The motivation behind this collective transformation is to harmonize temporal data across various timezones and datetime formats present in the dataset. This approach simplifies subsequent data analysis, correlations, and model building by providing a common temporal basis, fostering standardization and compatibility across the dataset's temporal information.

```
# Convert "End_Time" to datetime objects with error handling
object_columns_df["End_Time"] = pd.to_datetime(object_columns_df["End_Time"], errors='coerce')

# Define a dictionary mapping timezones to UTC offsets (hours)
timezone_offsets = {
    'US/Eastern': -5,   # Eastern Time (ET) UTC offset
    'US/Pacific': -8,   # Pacific Time (PT) UTC offset
    'US/Central': -6,   # Central Time (CT) UTC offset
    'US/Mountain': -7,  # Mountain Time (MT) UTC offset
    # Add more timezone offsets as needed
}

# Use NumPy broadcasting to add the corresponding UTC offset to "End_Time"
object_columns_df["End_Time_UTC"] = object_columns_df["End_Time"] + pd.to_timedelta(object_columns_df["Timezone"].map(timezone_offsets), unit='h')

# Convert "Start_Time" to datetime objects
object_columns_df["Start_Time"] = pd.to_datetime(object_columns_df["Start_Time"], errors='coerce')

# Use NumPy broadcasting to add the corresponding UTC offset to "Start_Time"
object_columns_df["Start_Time_UTC"] = object_columns_df["Start_Time"] + pd.to_timedelta(object_columns_df["Timezone"].map(timezone_offsets), unit='h')
```

10. **Feature Transformation**

In our continuous efforts to enrich the dataset's temporal information, we executed a function that extracted various datetime components from the "End_Time_UTC" and "Start_Time_UTC" columns. This process resulted in the creation of new features, including year, month, day, hour, minute, and second, for each of these datetime columns.

The decision to retain the "minute" and "second" components was a thoughtful one, based on their potential significance for machine learning models. While "minute" and

"second" components were not part of the original dataset, their inclusion is strategically aimed at providing the models with additional temporal granularity. This finer granularity might offer valuable insights and time-related patterns that could contribute to the predictive power of the models.

Conversely, the extraction of other datetime components like year, month, day, and hour aligns with the standard practice of feature engineering, enhancing the dataset's temporal information while ensuring model compatibility and interpretability. This process is instrumental in empowering machine learning models to make more informed and precise predictions, drawing from the richness of temporal data.

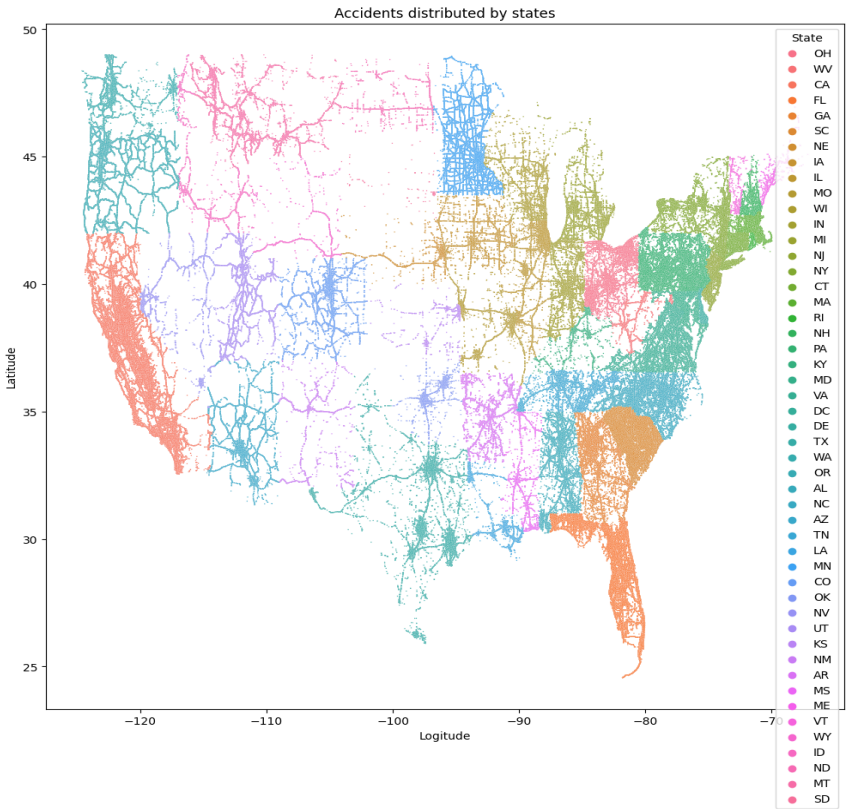
```
def extract_datetime_components(df, column_name):

    # Extract datetime components
    df[f'{column_name}_year'] = df[column_name].dt.year.astype(float)
    df[f'{column_name}_month'] = df[column_name].dt.month.astype(float)
    df[f'{column_name}_day'] = df[column_name].dt.day.astype(float)
    df[f'{column_name}_hour'] = df[column_name].dt.hour.astype(float)
    df[f'{column_name}_minute'] = df[column_name].dt.minute.astype(float)
    df[f'{column_name}_second'] = df[column_name].dt.second.astype(float)

    return df
```

```
object_columns_df = extract_datetime_components(object_columns_df, 'End_Time_UTC')
object_columns_df = extract_datetime_components(object_columns_df, 'Start_Time_UTC')
```

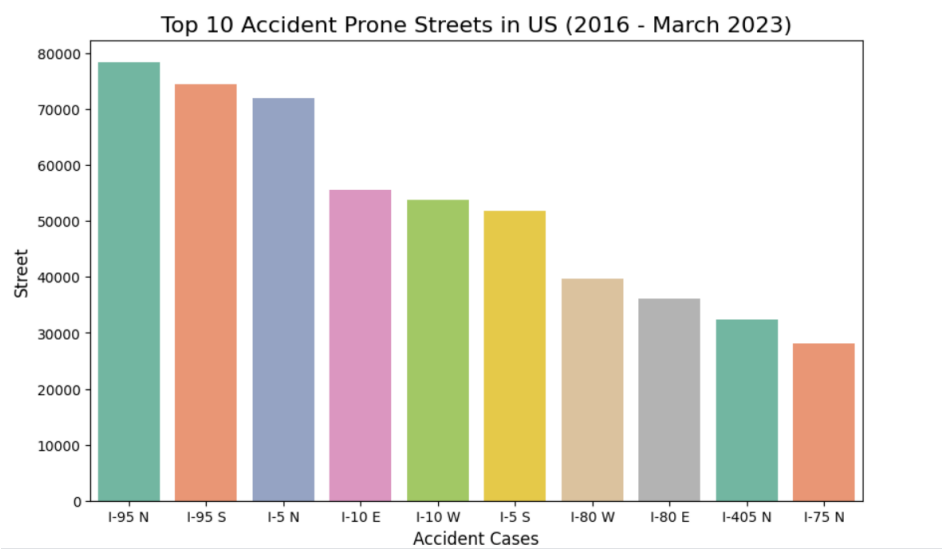
---

4. **Exploratory Data Analysis (EDA) [25 Marks]: Perform exploratory data analysis as defined in the NIST publication [2] and as originally described by John Tukey [3]. Record CSE 4/587 Fall 2023 the outcomes and what you learned and how you will use this information. For example, in choosing features (columns) and dropping columns, and in short feature engineering. You need to demonstrate 7 (10 for 587 students) different, significant and relevant EDA operations and describe how you used these to process the data sets further to provision them for downstream modeling and analytics. Figures and tables should be included where relevant.**
   These are the EDA steps that we undertook in our project:

1. The scatter plot below illustrate the distribution of accidents by states. This visualization is generated exclusively using the latitude and longitude values provided in the dataset.

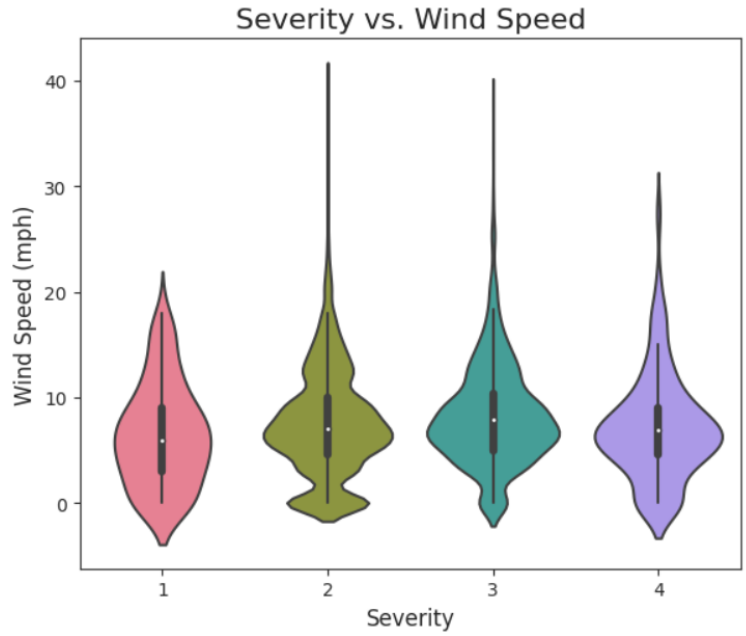


2. An analysis of the top 10 accident-prone streets in the United States from 2016 to March 2023.

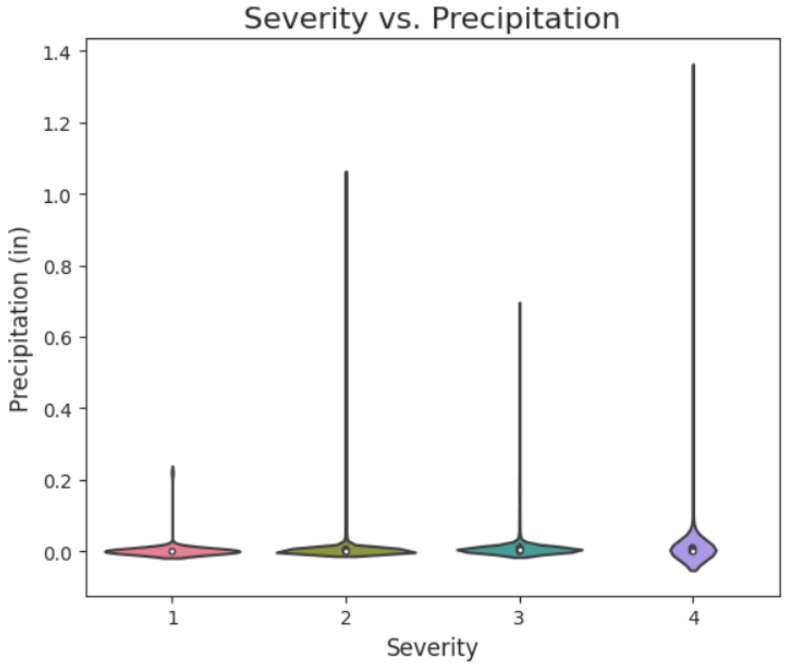


3. This figure illustrates the relationship between accident severity and wind speed. It helps to identify whether higher wind speeds correlate with more severe accidents, which is

essential for safety measures.

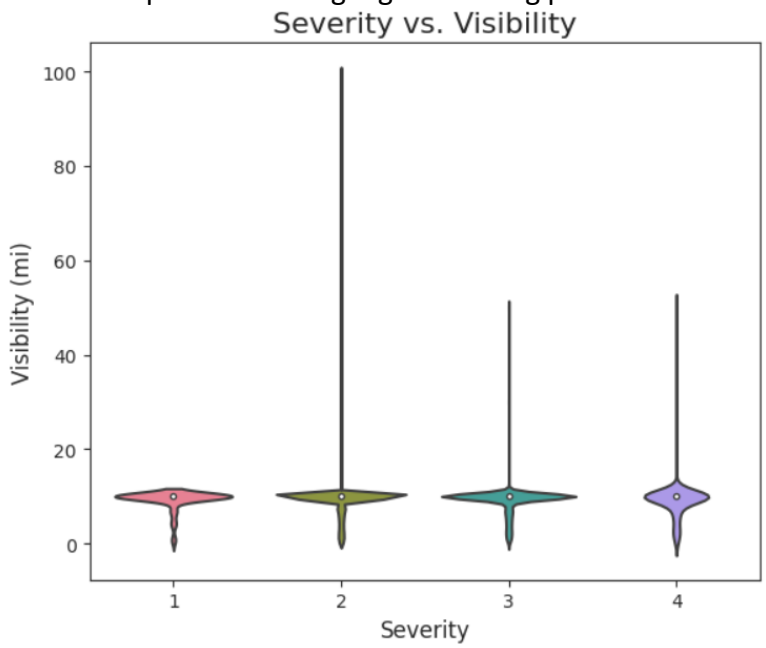


5. The chart demonstrates how accident severity is influenced by precipitation levels. It's valuable for understanding the impact of weather conditions on road safety.

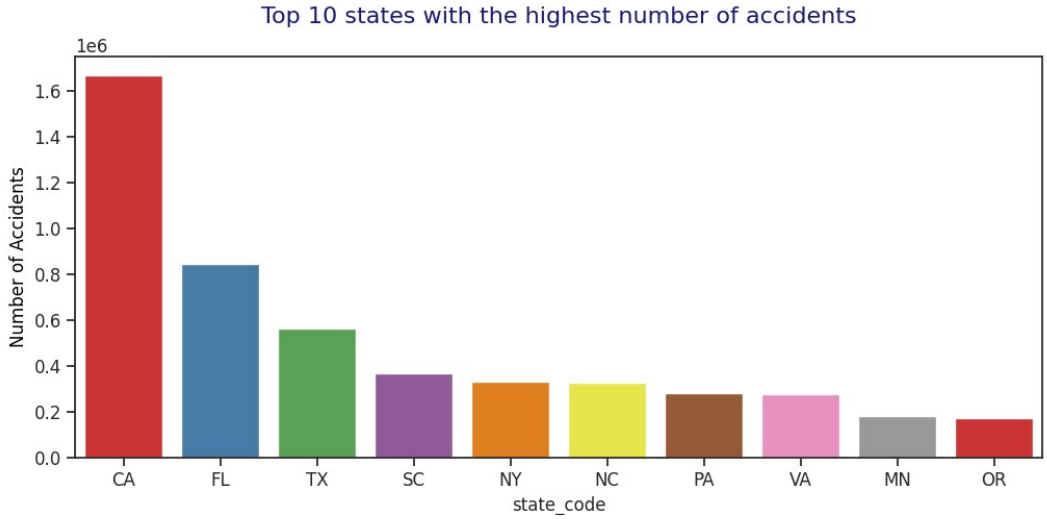


6. This plot reveals the connection between visibility and accident severity. It aids in recognizing how poor visibility contributes to more severe accidents and suggests the

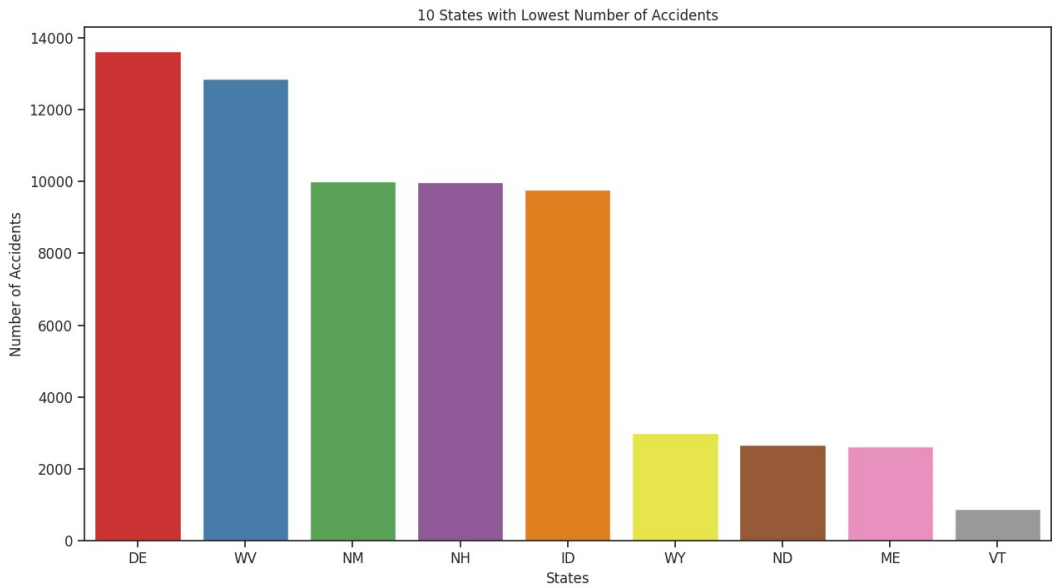
need for improved road signage or driving precautions.



7. Observe the fact that 7 out of the 10 states in the top 10 list are located in the southern or southeastern United States.
   There are a number of possible explanations for this trend. One possibility is that the warmer weather in these states leads to more people being on the roads, which increases the risk of accidents. Additionally, the roads in some of these states may be older and less well-maintained, which can also contribute to accidents.

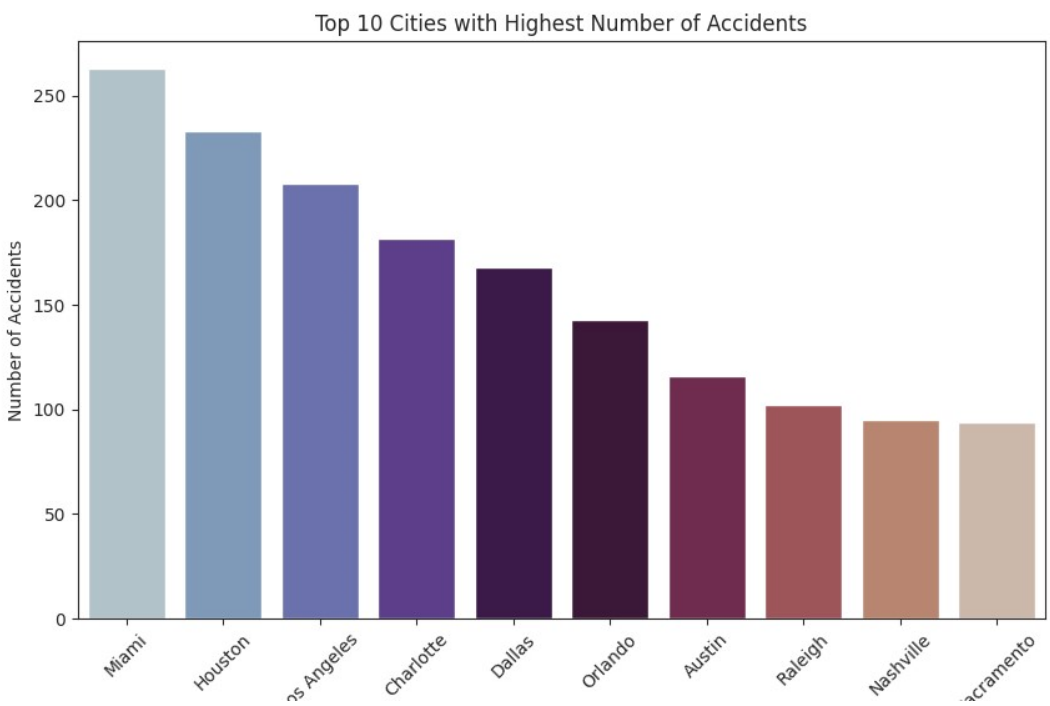


8. Conversely, this chart shows that the states with the lowest number of accidents are generally less populous and have lower rates of vehicle ownership.

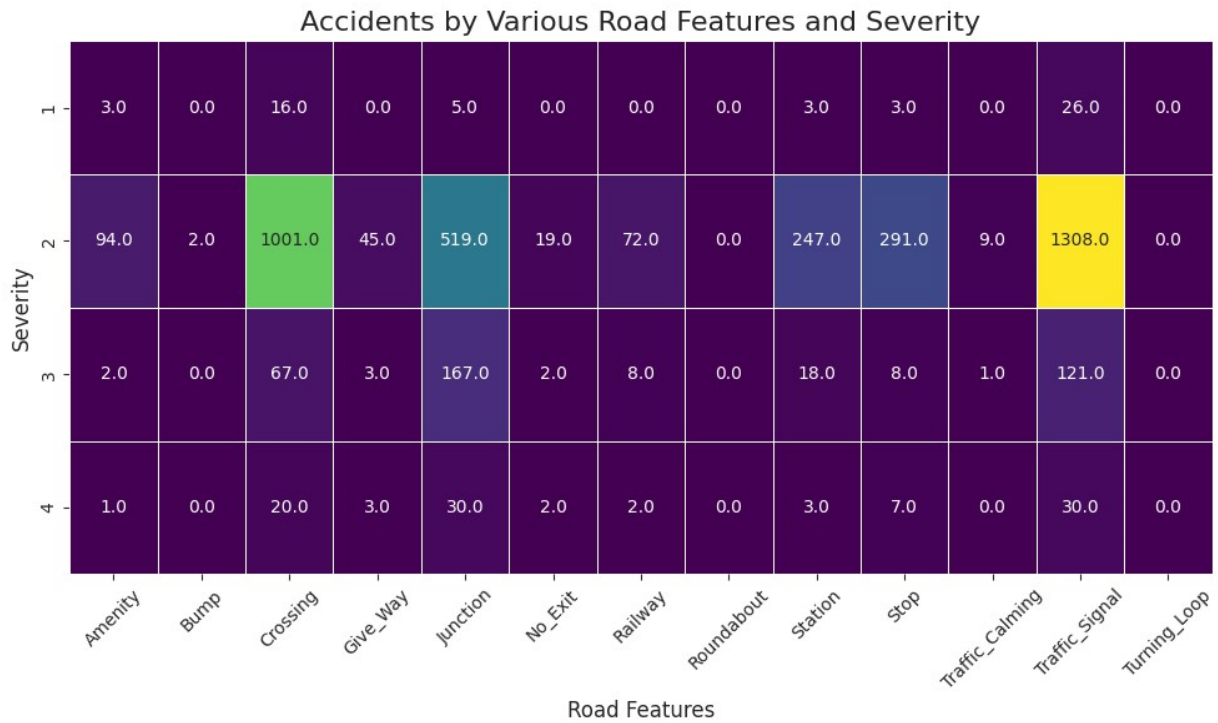


9. The top 10 cities with the highest number of accidents are major metropolitan areas with high traffic volume.

   This is supported by the fact that all of the cities in the top 10 list are large metropolitan areas with a lot of traffic. These cities typically have more congested roads, more pedestrians and cyclists, and more complex road networks, all of which can contribute to an increased risk of accidents.



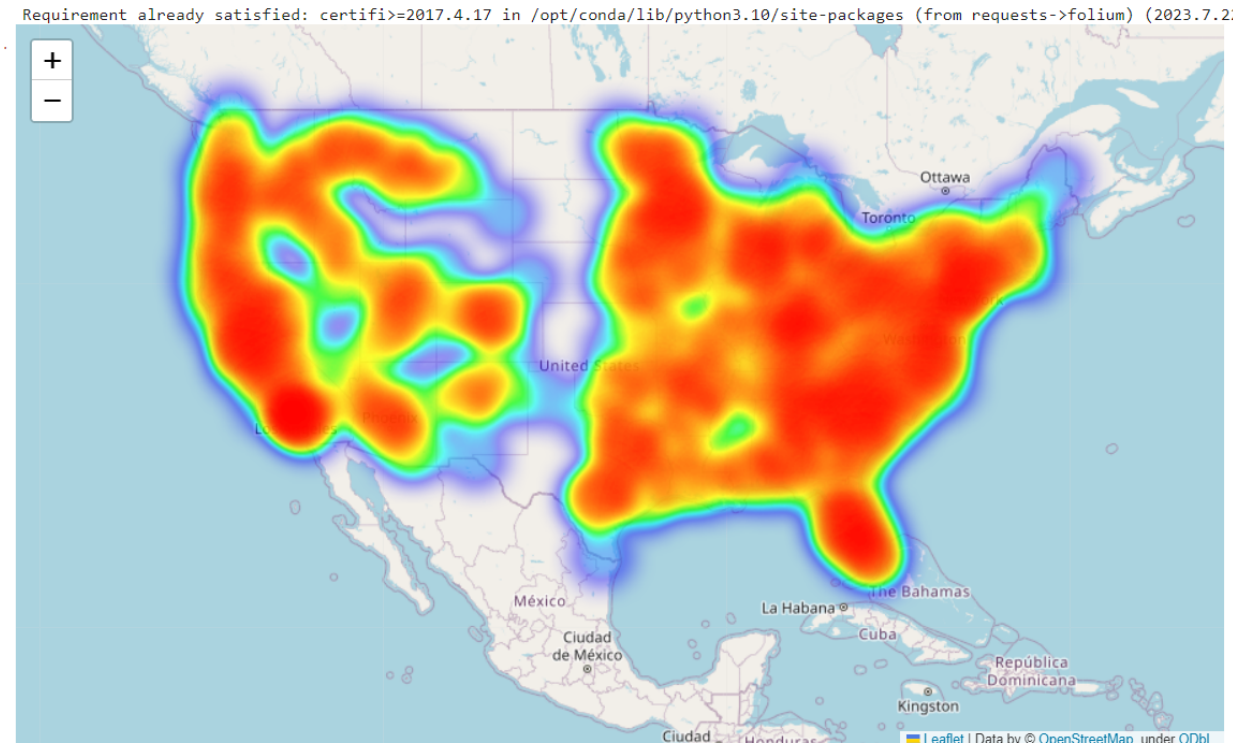
10. This heatmap explores the number of accidents across different road features, The majority of accidents occur at intersections.

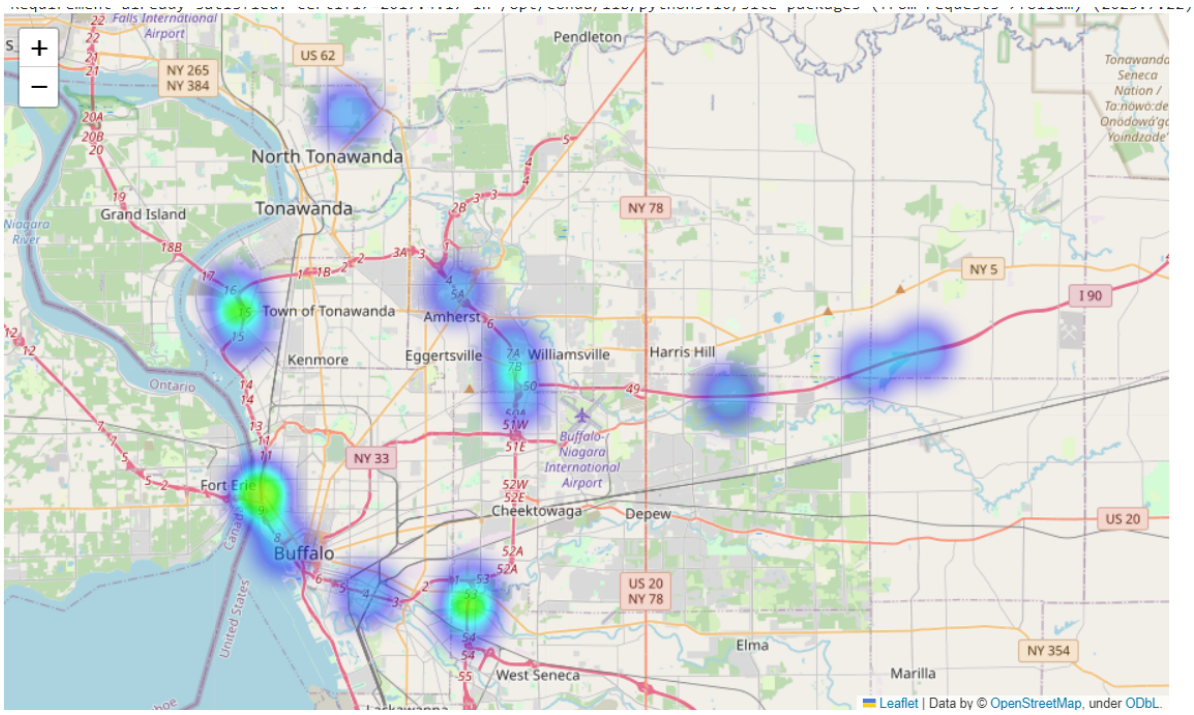


11. Heatmap of Accidents

We can see more accidents have occurred on the coastlines where more people live, so there might be correlation. We have added accident prone area near our University At Buffalo as well.

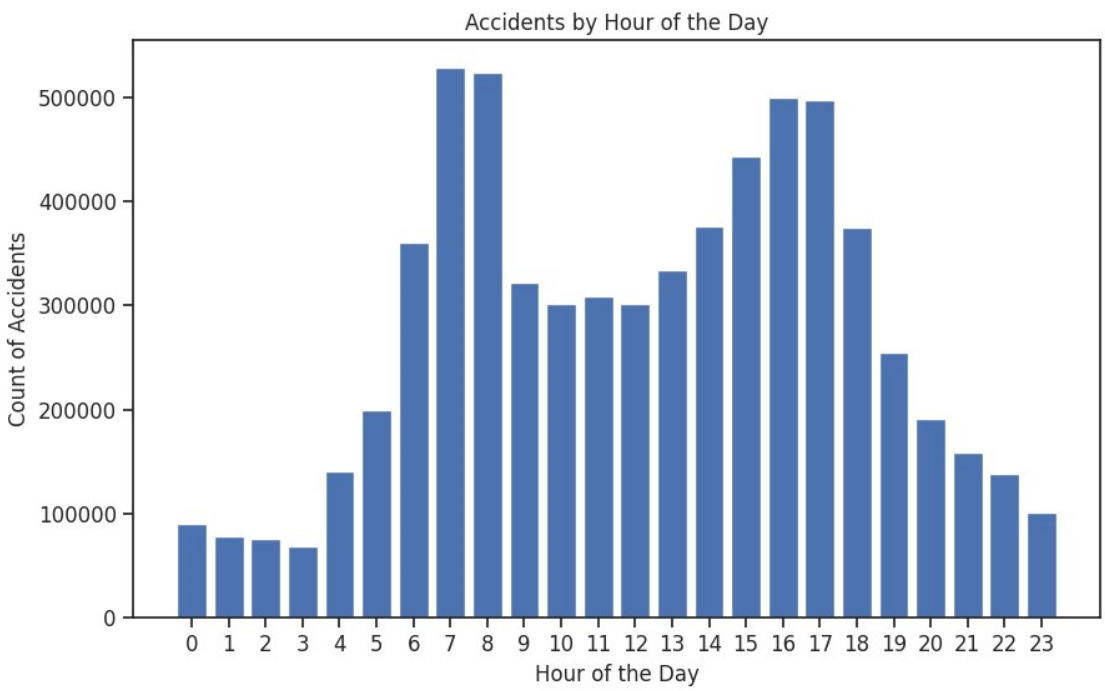
```
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests->folium) (2023.7.22)
```



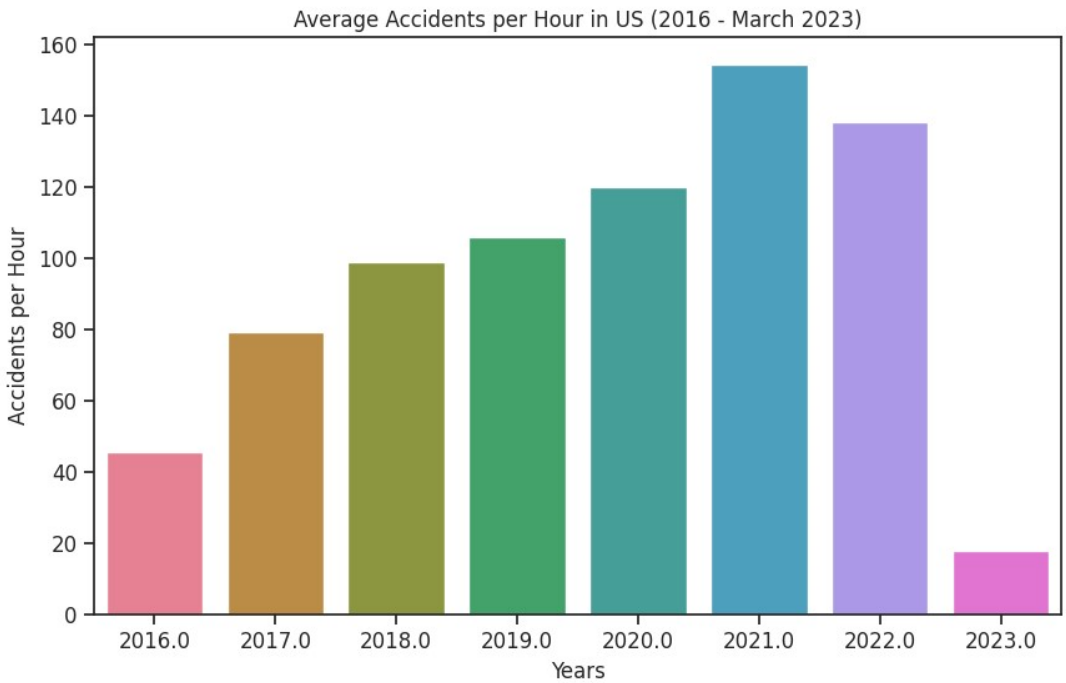


12. Accidents by Hour

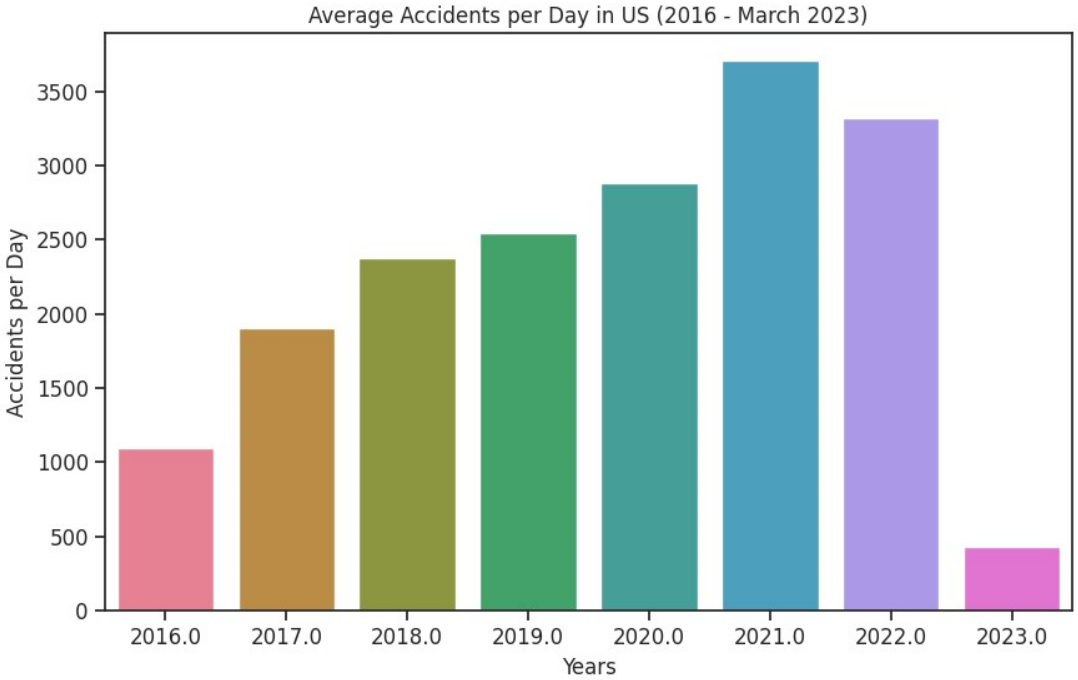
Most accidents happen during rush hour, so avoiding driving during those times could reduce your risk of being involved in an accident.



13. The average number of accidents per hour in the US has increased over time except for 2022, maybe because of Pandemic.

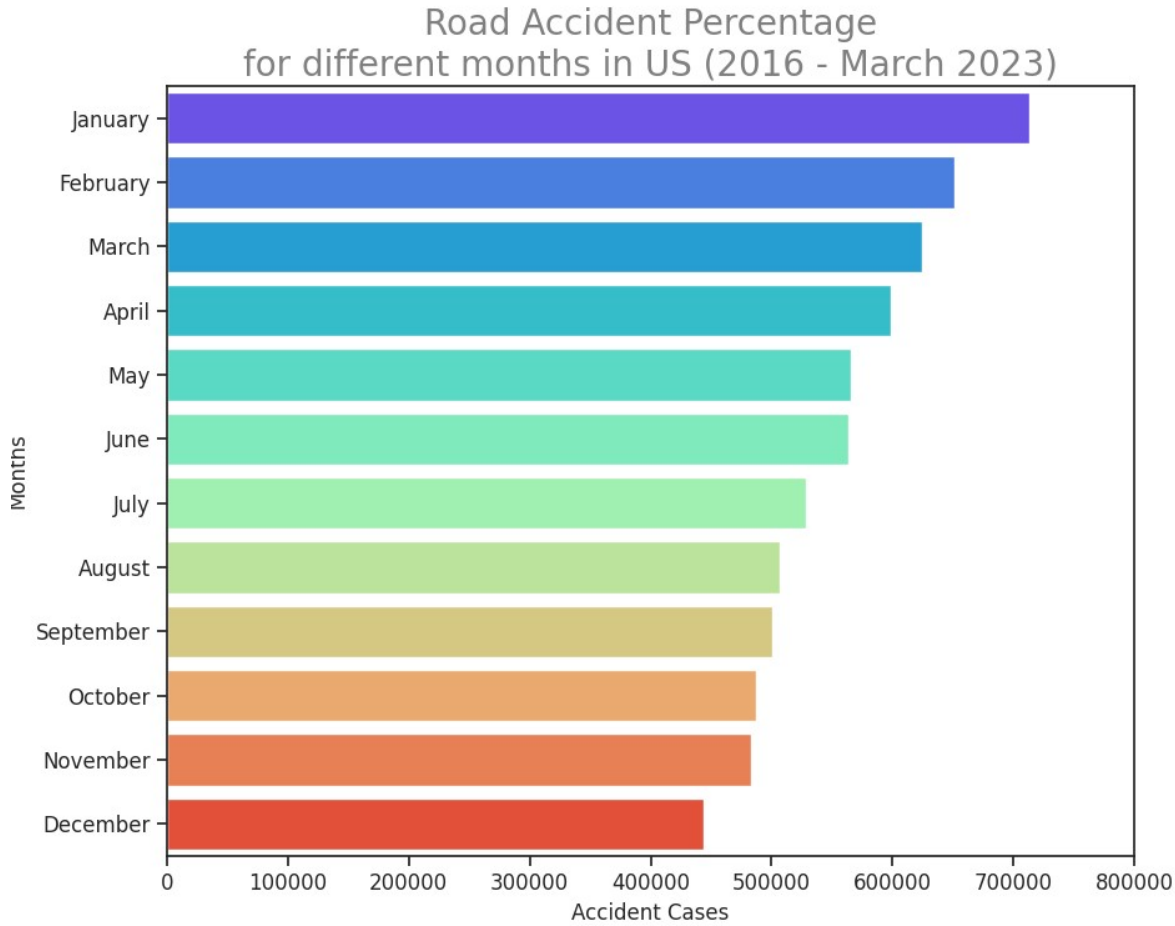


14. The average number of accidents per day in the US has increased over time except for 2022, maybe because of Pandemic.



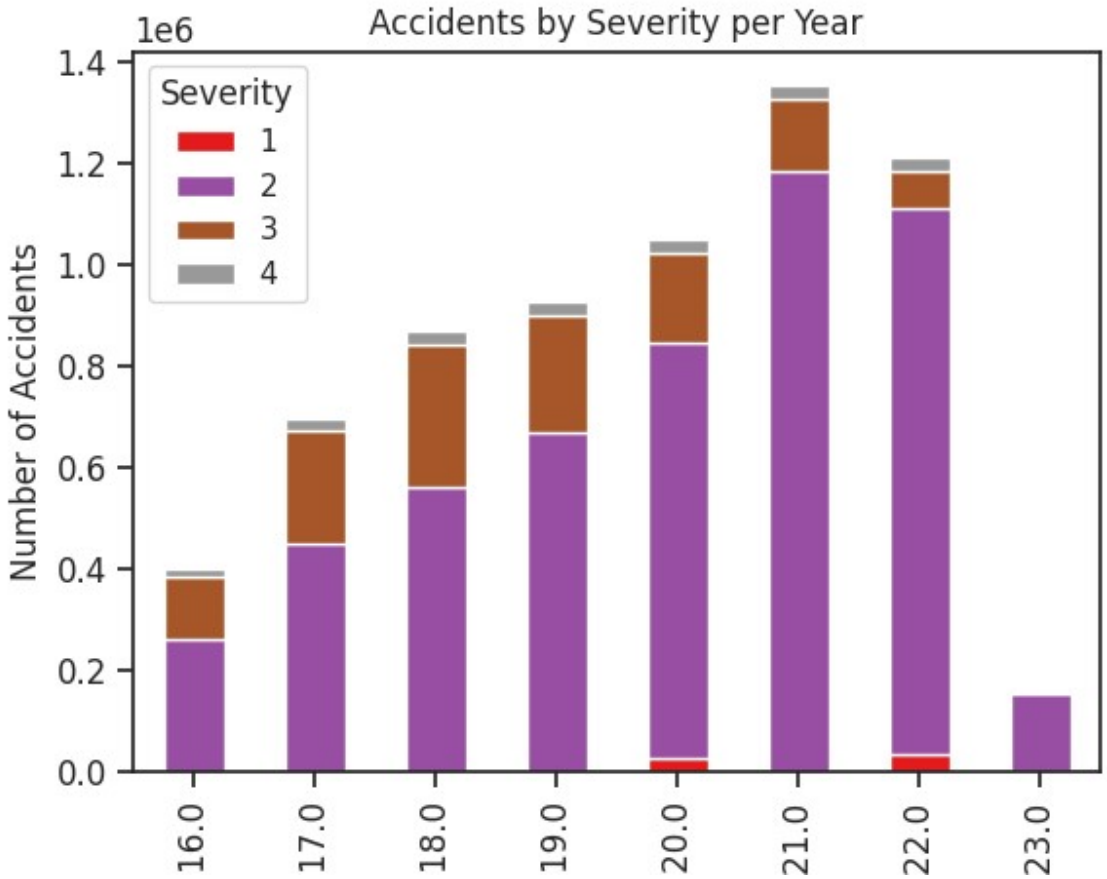
15. Road Accident Percentage for different months in US (2016 - March 2023)

    Road accident percentage is lowest in the summer months, ENJOY YOUR SUMMER!!

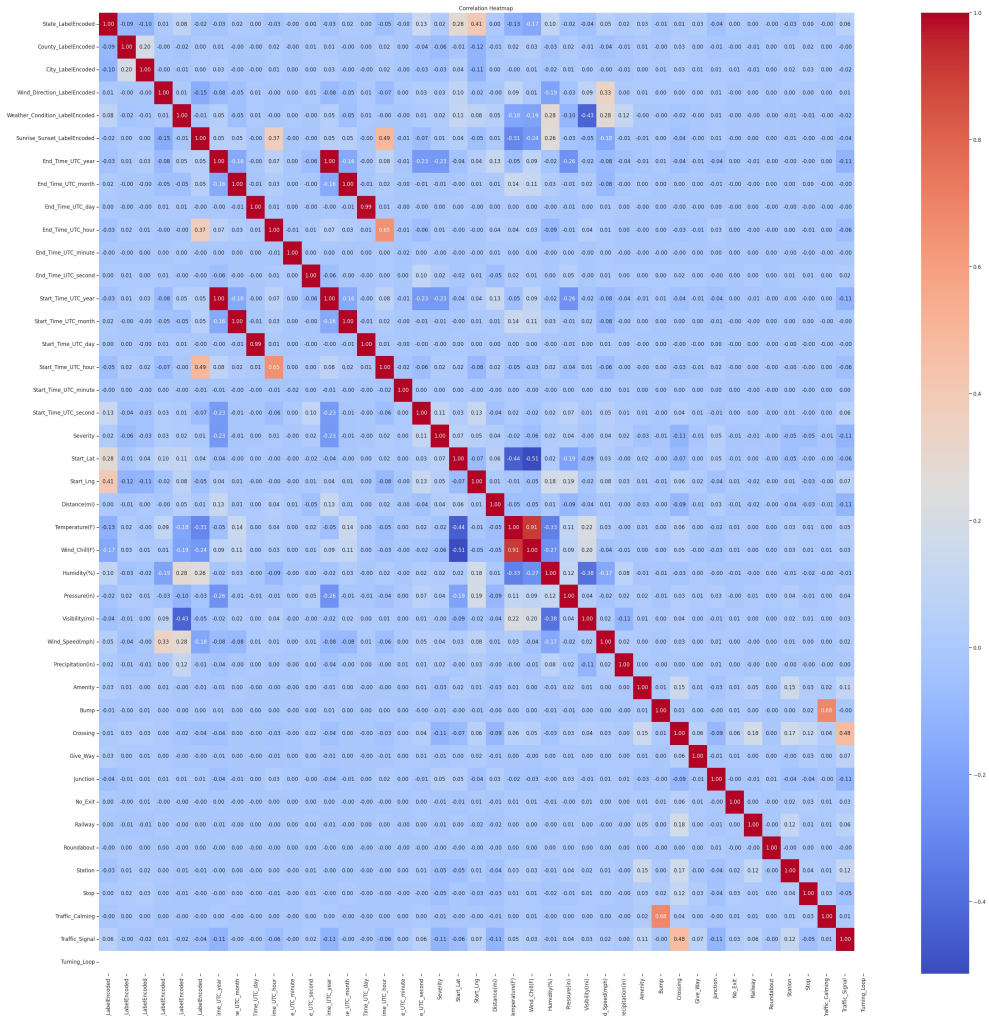


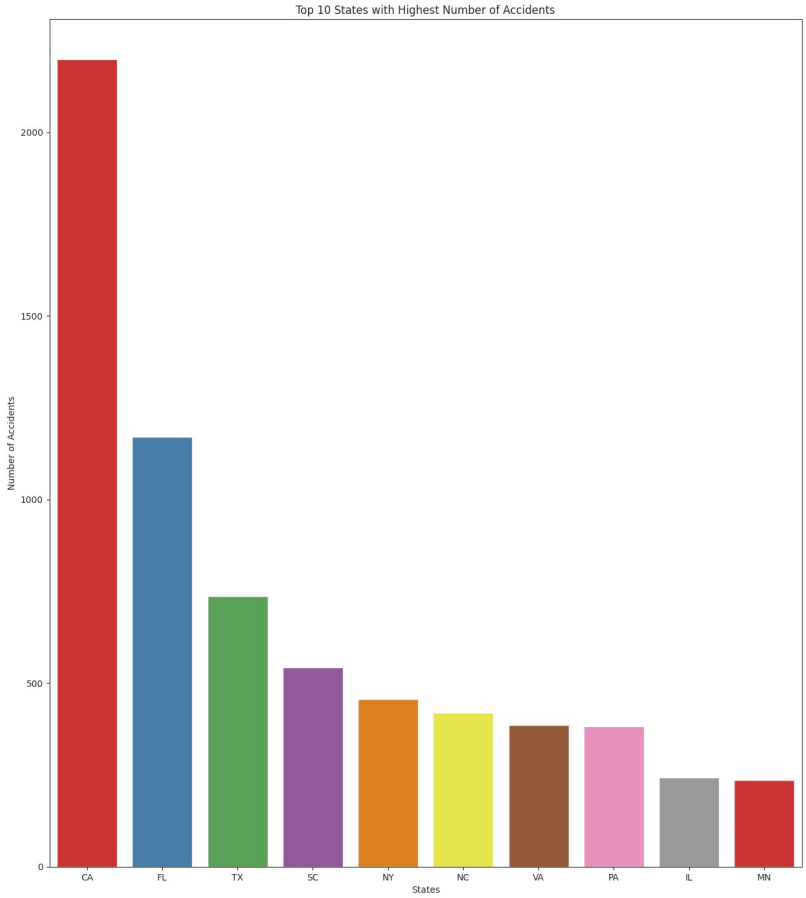
16. Accidents by Severity per Year

We can observe upward trend, apart from severity 3 accidents.

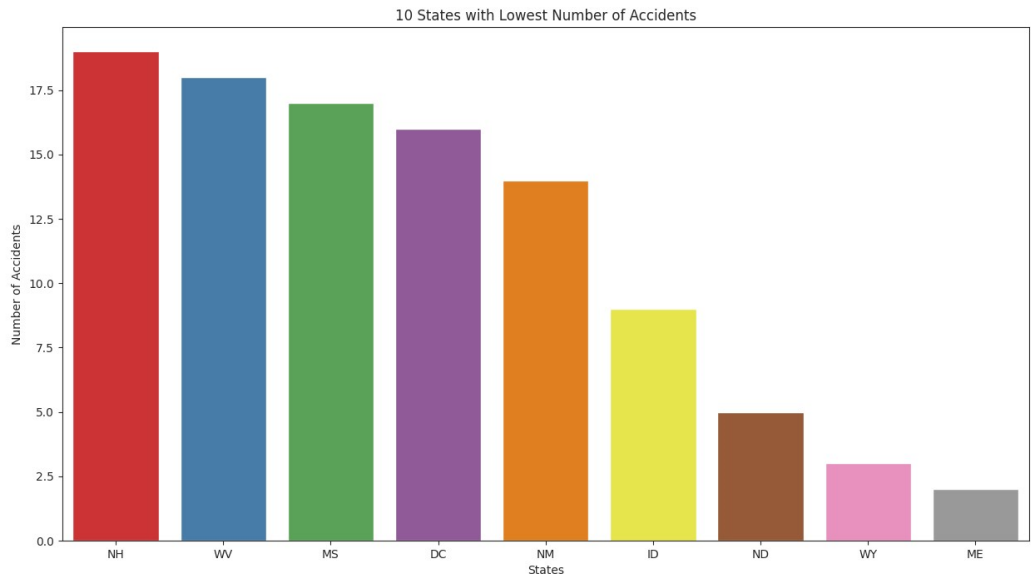


17. Correlation Heatmap

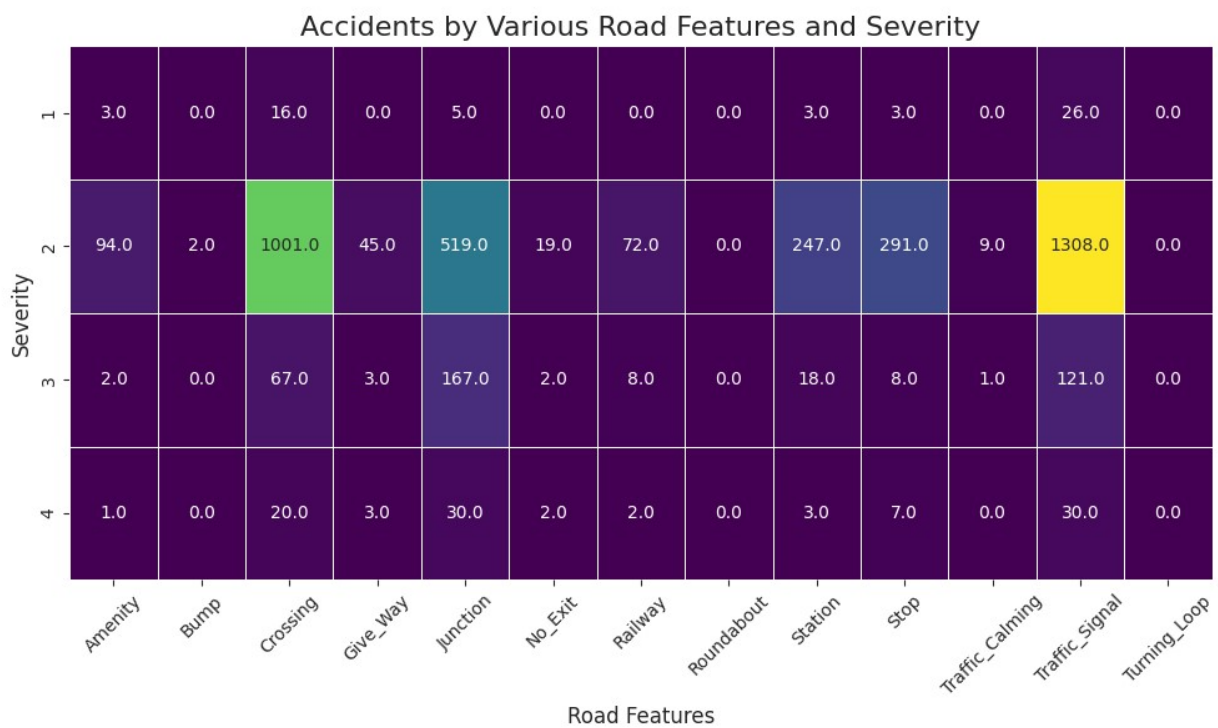

Correlation Heatmap
State_LabelEncoded
County_LabelEncoded
City_LabelEncoded
Wind_Direction_LabelEncoded
Weather_Condition_LabelEncoded
Sunrise_Sunset_LabelEncoded
End_Time_UTC_year
End_Time_UTC_month
End_Time_UTC_day
End_Time_UTC_hour
End_Time_UTC_minute
End_Time_UTC_second
Start_Time_UTC_year
Start_Time_UTC_month
Start_Time_UTC_day
Start_Time_UTC_hour
Start_Time_UTC_minute
Start_Time_UTC_second
Severity
Start_Lat
Start_Lng
Distance(mi)
Temperature(F)
Wind_Chill(F)
Humidity(%)
Pressure(in)
Visibility(mi)
Wind_Speed(mph)
Precipitation(in)
Amenity
Bump
Crossing
Give_Way
Junction
No_Exit
Railway
Roundabout
Station
Stop
Traffic_Calming
Traffic_Signal
Turning_Loop
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4



18. Conversely, this chart identifies states with the lowest accident rates, offering insights into successful safety strategies and the opportunity for improvement in higher-risk regions.

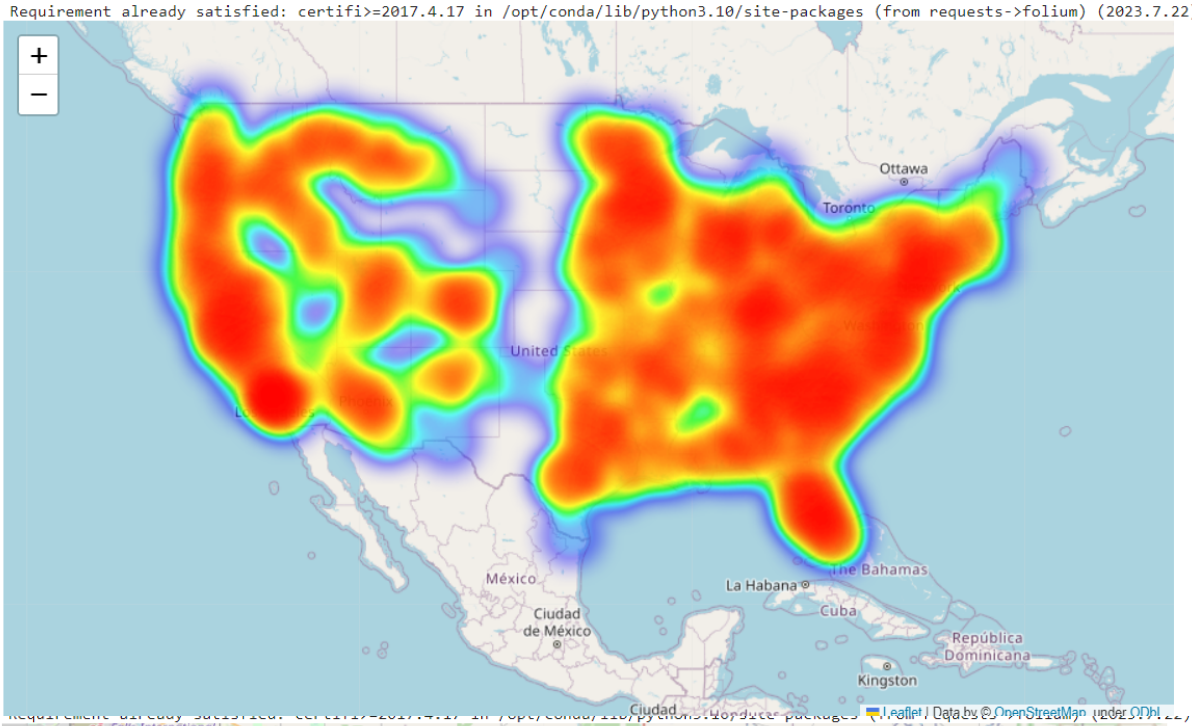


19. This heatmap explores the number of accidents across different road features, helping to prioritize areas for safety infrastructure improvement, such as better signage or crossing enhancements.

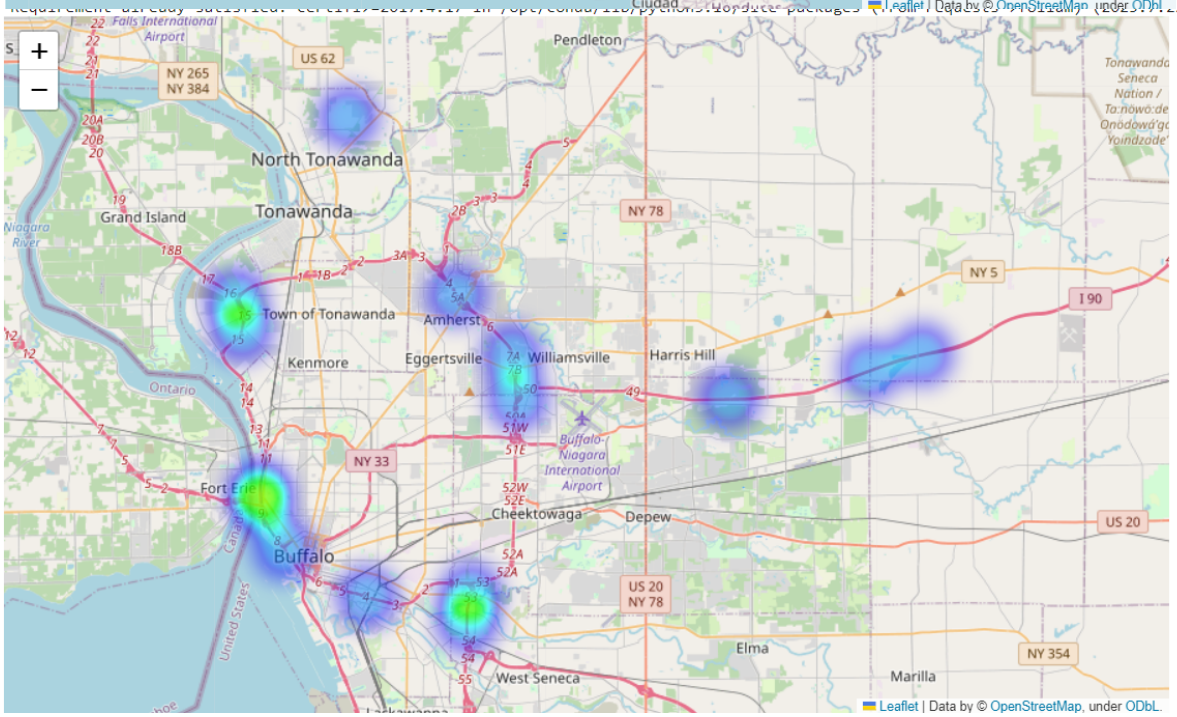


20. Heatmap of Accidents

```
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests->folium) (2023.7.22)
```

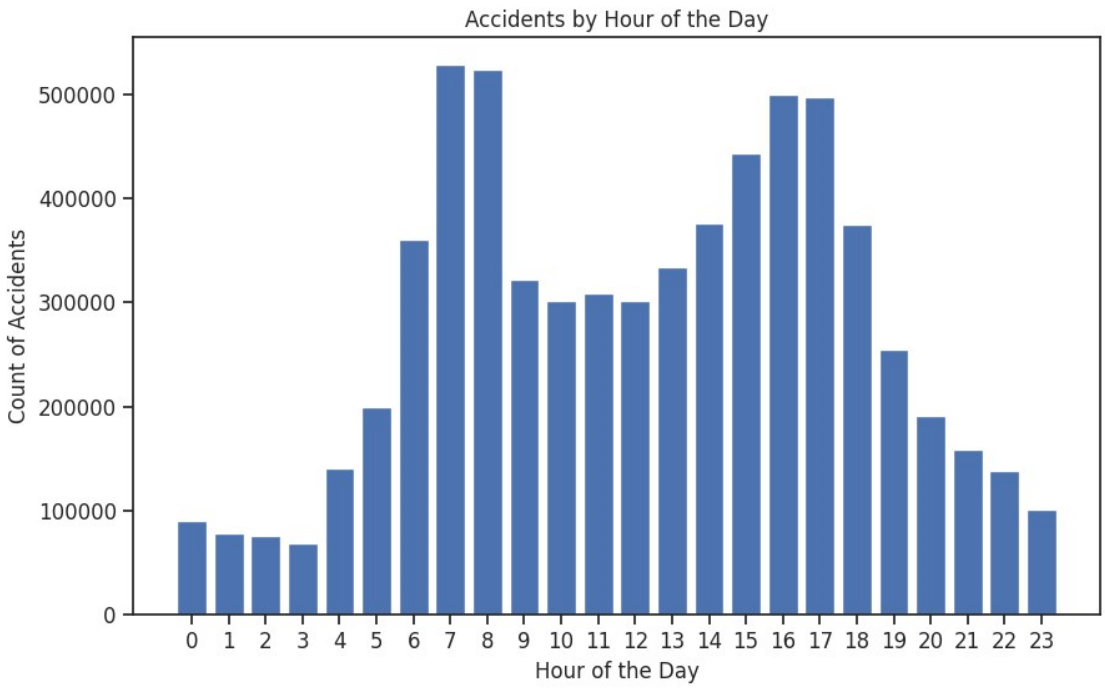


[177…

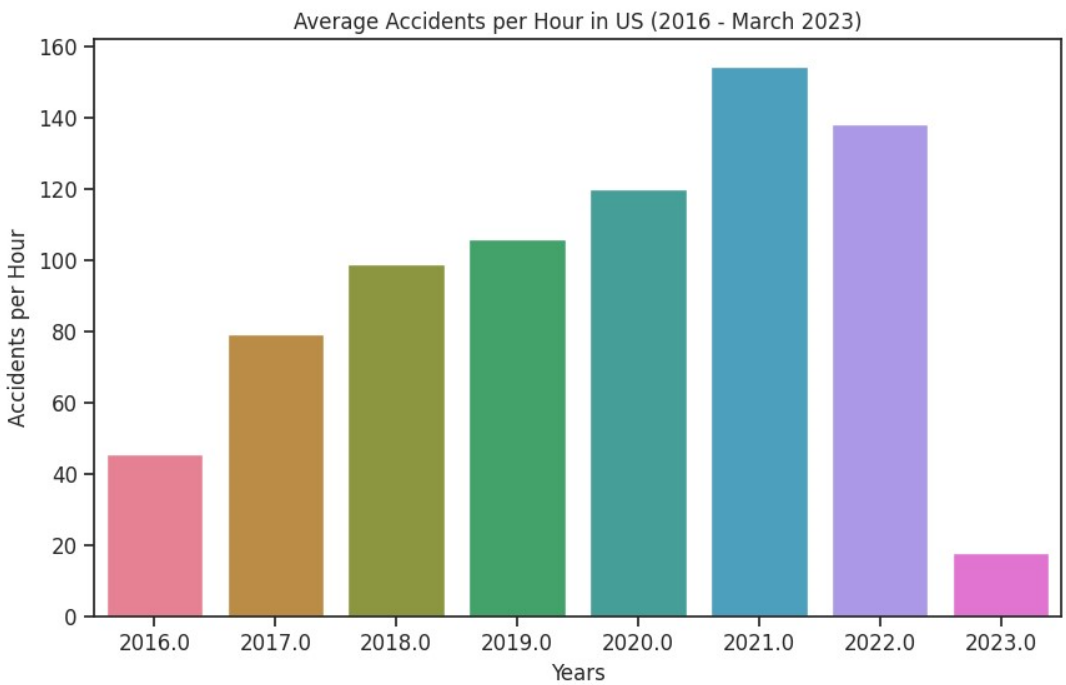


21. Accidents by Hour

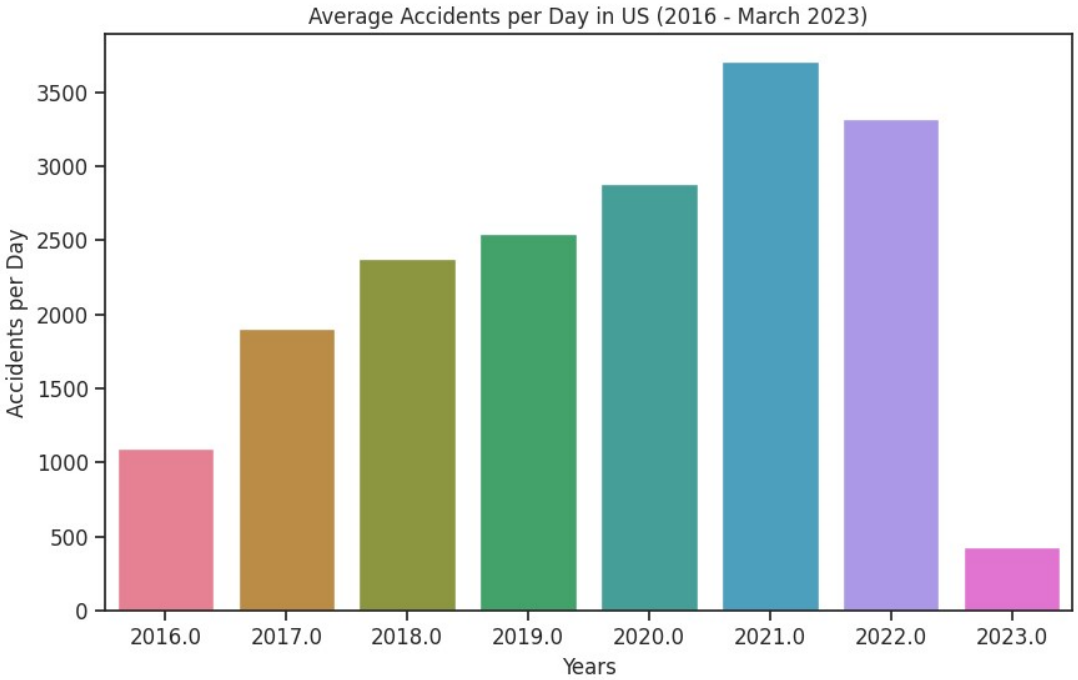
    We can see that this peaks before 9 and somewhere around 5, maybe because of 9 – 5 job.



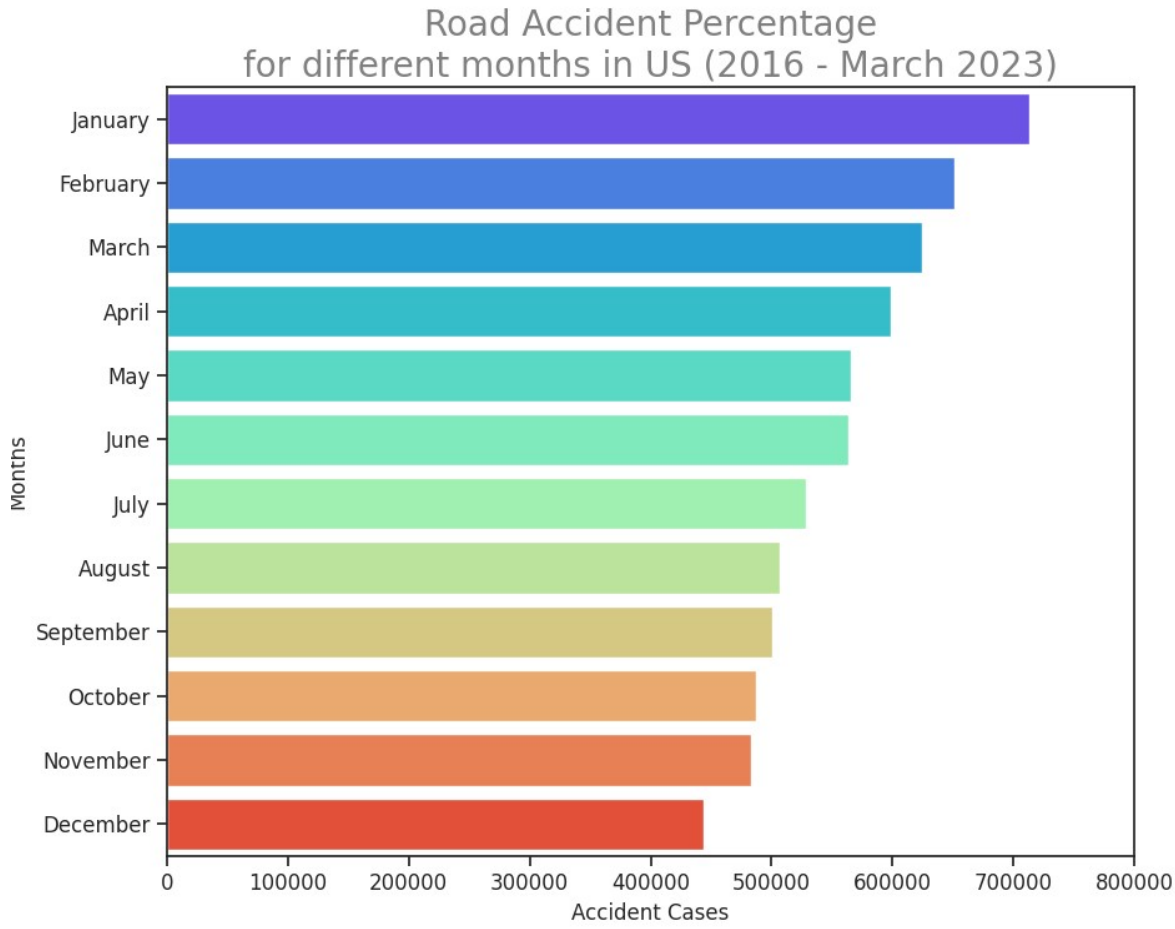
22. Average hour



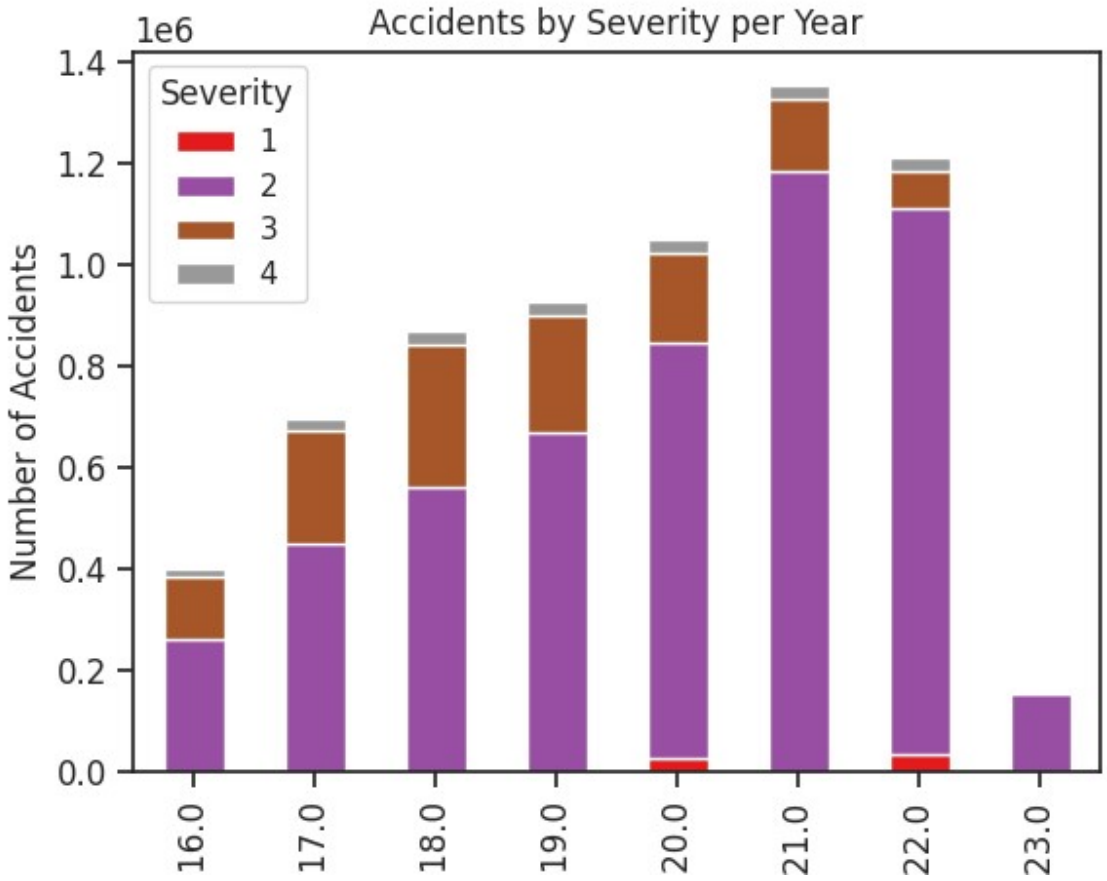
23. Average day



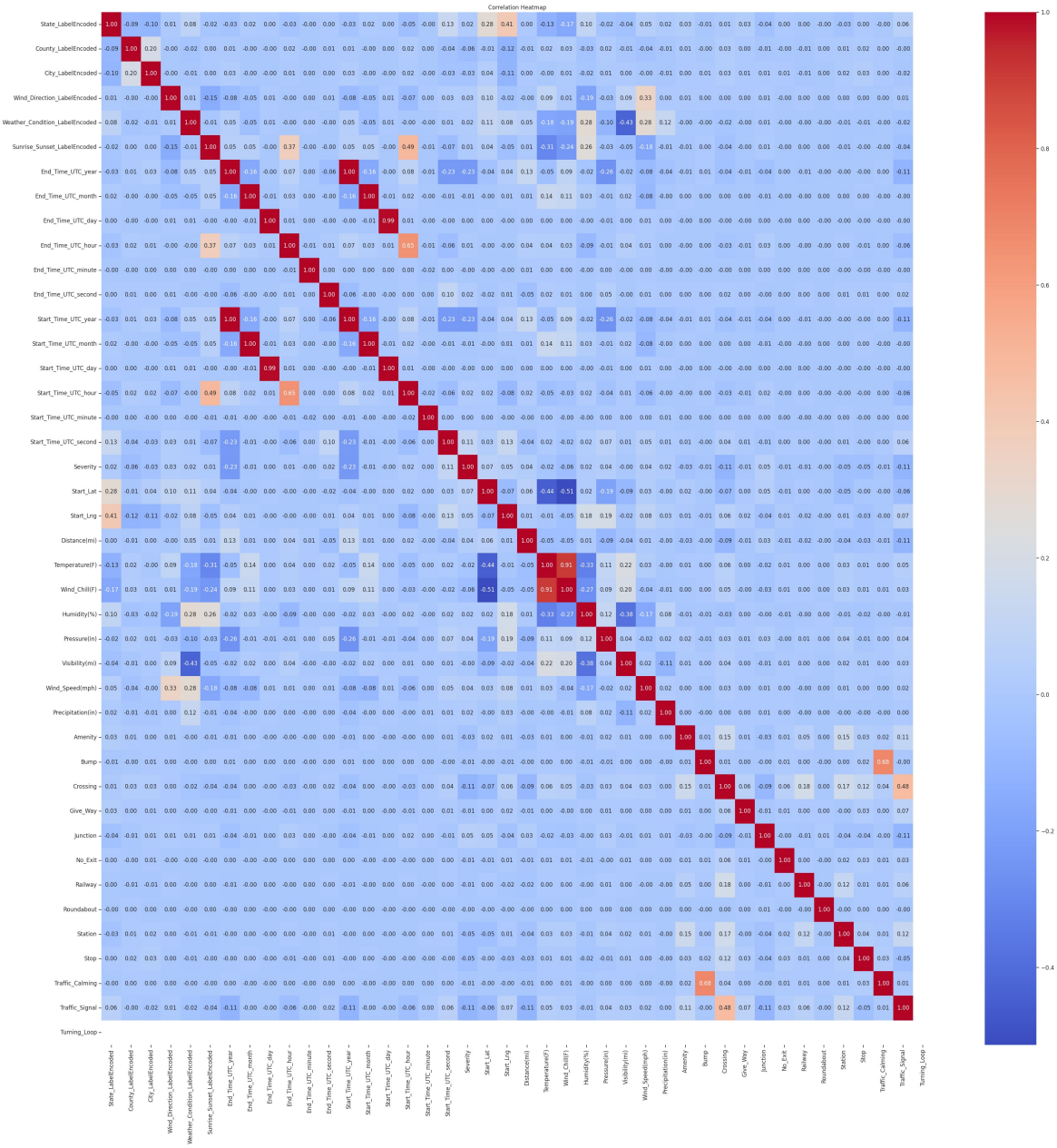
24. Road Accident Percentage for different months in US (2016 - March 2023)14.



25. Accidents by Severity per Year



26. Correlation Heatmap



REFERENCES:

https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents/data

https://medium.com/@ishitaguharoy/us-accidents-exploratory-data-analysis-with-python-27d7e1ab837f

https://arxiv.org/abs/1909.09638

https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=2085&context=etd