Analyzing Motor Vehicle Collisions in New York City Using Python: A Data Science Project

Introduction

This data science project analyzes the Motor Vehicle Collisions dataset from the New York City Police Department (NYPD) to gain insights into the factors contributing to collisions. The project is divided into three main sections: Data Loading, Data Cleaning and Preparation, and Exploratory Data Analysis.

In the Data Loading section, the Motor Vehicle Collisions dataset is loaded from the NYPD and the first few rows of the dataset are displayed. The Data Cleaning and Preparation section performs cleaning and preparation steps, including dropping irrelevant columns, handling missing values, and converting data types.

In the Exploratory Data Analysis section, insights are gained by examining patterns and trends in the data. The analysis reveals that Brooklyn had the highest number of collisions among all the boroughs in New York City, and the most common contributing factor was driver inattention/distraction. The top vehicle type involved in collisions was a sedan.

Overall, this project provides valuable insights into motor vehicle collisions in New York City and serves as a starting point for further analysis and improvement. By analyzing the factors that contribute to collisions and identifying patterns and trends in the data, this project can help inform policy decisions aimed at reducing the number of collisions and improving road safety in New York City.

1. **Data Loading**

   The following code loads the Motor Vehicle Collisions dataset from the New York City Police Department (NYPD) using the Pandas library. It also displays the first few rows of the dataset along with its shape and column names:

```
# Import necessary libraries
import pandas as pd

# Load the Motor Vehicle Collisions dataset
df = pd.read_csv('C:/Motor_Vehicle_Collisions_-_Crashes.csv')

# Display the first few rows of the dataset
print(df.head())

# Print the shape of the dataset
print('Shape of the dataset:', df.shape)

# Print the column names of the dataset
print('Column names:', list(df.columns))
```

```
  CRASH DATE CRASH TIME   BOROUGH ZIP CODE   LATITUDE  LONGITUDE  \
0  09/11/2021       2:39       NaN      NaN        NaN        NaN
1  03/26/2022      11:45       NaN      NaN        NaN        NaN
2  06/29/2022       6:55       NaN      NaN        NaN        NaN
3  09/11/2021       9:35  BROOKLYN  11208.0  40.667202 -73.866500
4  12/14/2021       8:13  BROOKLYN  11233.0  40.683304 -73.917274

                 LOCATION           ON STREET NAME CROSS STREET NAME  \
0                     NaN    WHITESTONE EXPRESSWAY         20 AVENUE
1                     NaN  QUEENSBORO BRIDGE UPPER               NaN
2                     NaN       THROGS NECK BRIDGE               NaN
3    (40.667202, -73.8665)                      NaN               NaN
4  (40.683304, -73.917274)          SARATOGA AVENUE    DECATUR STREET

          OFF STREET NAME  ...  CONTRIBUTING FACTOR VEHICLE 2  \
0                     NaN  ...                   Unspecified
1                     NaN  ...                           NaN
2                     NaN  ...                   Unspecified
3  1211      LORING AVENUE  ...                           NaN
4                     NaN  ...                           NaN

   CONTRIBUTING FACTOR VEHICLE 3  CONTRIBUTING FACTOR VEHICLE 4  \
0                            NaN                            NaN
1                            NaN                            NaN
2                            NaN                            NaN
3                            NaN                            NaN
4                            NaN                            NaN

   CONTRIBUTING FACTOR VEHICLE 5  COLLISION_ID  VEHICLE TYPE CODE 1  \
0                            NaN       4455765                Sedan
1                            NaN       4513547                Sedan
2                            NaN       4541903                Sedan
3                            NaN       4456314                Sedan
4                            NaN       4486609                  NaN

   VEHICLE TYPE CODE 2  VEHICLE TYPE CODE 3 VEHICLE TYPE CODE 4  \
0                Sedan                  NaN                 NaN
1                  NaN                  NaN                 NaN
2       Pick-up Truck                   NaN                 NaN
3                  NaN                  NaN                 NaN
4                  NaN                  NaN                 NaN

  VEHICLE TYPE CODE 5
0                 NaN
1                 NaN
2                 NaN
3                 NaN
4                 NaN

[5 rows x 29 columns]
Shape of the dataset: (1974220, 29)
Column names: ['CRASH DATE', 'CRASH TIME', 'BOROUGH', 'ZIP CODE', 'LATITUDE',
'LONGITUDE', 'LOCATION', 'ON STREET NAME', 'CROSS STREET NAME', 'OFF STREET
```

```
NAME', 'NUMBER OF PERSONS INJURED', 'NUMBER OF PERSONS KILLED', 'NUMBER OF
PEDESTRIANS INJURED', 'NUMBER OF PEDESTRIANS KILLED', 'NUMBER OF CYCLIST
INJURED', 'NUMBER OF CYCLIST KILLED', 'NUMBER OF MOTORIST INJURED', 'NUMBER
OF MOTORIST KILLED', 'CONTRIBUTING FACTOR VEHICLE 1', 'CONTRIBUTING FACTOR
VEHICLE 2', 'CONTRIBUTING FACTOR VEHICLE 3', 'CONTRIBUTING FACTOR VEHICLE 4',
'CONTRIBUTING FACTOR VEHICLE 5', 'COLLISION_ID', 'VEHICLE TYPE CODE 1',
'VEHICLE TYPE CODE 2', 'VEHICLE TYPE CODE 3', 'VEHICLE TYPE CODE 4', 'VEHICLE
TYPE CODE 5']
```

The dataset contains information about motor vehicle collisions in New York City from 2013 to present. It includes attributes such as the date and time of the collision, location of the collision, number of people injured or killed, contributing factors to the collision, and types of vehicles involved. The dataset has 1,974,220 rows and 29 columns. By loading this dataset into Python, we can conduct further analysis to gain insights into the factors contributing to collisions and develop models that can predict collision outcomes.

2. Data Cleaning and Preparation

In this section, I loaded the original dataset containing motor vehicle collision data and performed cleaning and preparation steps.

```
# Import necessary libraries

import pandas as pd

import io

import base64

from IPython.display import HTML


# Load the dataset

df = pd.read_csv('C:/Motor_Vehicle_Collisions_-_Crashes.csv')


# Drop columns that are not relevant for the analysis

df.drop(['ZIP CODE', 'LATITUDE', 'LONGITUDE', 'LOCATION', 'ON STREET NAME',
'CROSS STREET NAME', 'OFF STREET NAME', 'NUMBER OF PERSONS INJURED', 'NUMBER
OF PERSONS KILLED', 'NUMBER OF PEDESTRIANS INJURED', 'NUMBER OF PEDESTRIANS
KILLED', 'NUMBER OF CYCLIST INJURED', 'NUMBER OF CYCLIST KILLED', 'NUMBER OF
MOTORIST INJURED', 'NUMBER OF MOTORIST KILLED', 'CONTRIBUTING FACTOR VEHICLE
3', 'CONTRIBUTING FACTOR VEHICLE 4', 'CONTRIBUTING FACTOR VEHICLE 5',
'VEHICLE TYPE CODE 2', 'VEHICLE TYPE CODE 3', 'VEHICLE TYPE CODE 4', 'VEHICLE
TYPE CODE 5'], axis=1, inplace=True)


# Drop rows with missing values in relevant columns
```

```
df.dropna(subset=['CRASH DATE', 'CRASH TIME', 'BOROUGH', 'CONTRIBUTING FACTOR
VEHICLE 1', 'VEHICLE TYPE CODE 1'], inplace=True)


# Convert date and time columns to datetime format

df['CRASH DATE'] = pd.to_datetime(df['CRASH DATE'], format='%m/%d/%Y')

df['CRASH TIME'] = pd.to_datetime(df['CRASH TIME'], format='%H:%M')


# Replace missing values in the 'BOROUGH' column with 'Unknown'

df['BOROUGH'].fillna('Unknown', inplace=True)


# Convert cleaned dataframe to csv and create a download link

csv = df.to_csv(index=False)

b64 = base64.b64encode(csv.encode()).decode()

href = f'<a href="data:file/csv;base64,{b64}"
download="Motor_Vehicle_Collisions_-_Crashes_cleaned.csv">Download cleaned
dataset</a>'


# Display download link

HTML(href)


# Checking Cleaned Data

# Load the cleaned dataset

df_cleaned = pd.read_csv('C:/Motor_Vehicle_Collisions_-_Crashes_cleaned.csv')


# Check for missing values

print('Missing values in the cleaned dataset:')

print(df_cleaned.isnull().sum())


# Replace missing values in the 'CONTRIBUTING FACTOR VEHICLE 2' column with
'Unspecified'

df['CONTRIBUTING FACTOR VEHICLE 2'].fillna('Unspecified', inplace=True)


# Checking Cleaned Data

print(df['CONTRIBUTING FACTOR VEHICLE 2'].isnull().sum())
```

```
print(df.isnull().sum())


# Replace missing values in the 'CONTRIBUTING FACTOR VEHICLE 2' column with
'Unspecified'

df['CONTRIBUTING FACTOR VEHICLE 2'].fillna('Unspecified', inplace=True)


# Checking Cleaned Data

print(df['CONTRIBUTING FACTOR VEHICLE 2'].isnull().sum())

print(df.isnull().sum())


# Convert cleaned dataframe to csv and create a download link

csv = df.to_csv(index=False)

b64 = base64.b64encode(csv.encode()).decode()

href = f'<a href="data:file/csv;base64,{b64}"
download="Motor_Vehicle_Collisions_-_Crashes_cleaned.csv">Download cleaned
dataset</a>'

# Display download link

HTML(href)


Missing values in the cleaned dataset:
CRASH DATE                            0
CRASH TIME                            0
BOROUGH                               0
CONTRIBUTING FACTOR VEHICLE 1         0
CONTRIBUTING FACTOR VEHICLE 2    207904
COLLISION_ID                          0
VEHICLE TYPE CODE 1                   0
dtype: int64
0
CRASH DATE                        0
CRASH TIME                        0
BOROUGH                           0
CONTRIBUTING FACTOR VEHICLE 1     0
CONTRIBUTING FACTOR VEHICLE 2     0
COLLISION_ID                      0
VEHICLE TYPE CODE 1               0
dtype: int64
0
CRASH DATE                        0
CRASH TIME                        0
BOROUGH                           0
CONTRIBUTING FACTOR VEHICLE 1     0
CONTRIBUTING FACTOR VEHICLE 2     0
COLLISION_ID                      0
```

```
VEHICLE TYPE CODE 1              0
dtype: int64
```
Download cleaned dataset

This code creates an HTML hyperlink element (<a> tag) with the encoded CSV data and the desired filename for the downloaded file. The resulting string is stored in the href variable. Finally, the HTML() function is used to display the hyperlink element in the notebook output.

3. Exploratory Data Analysis

In this section, I performed exploratory data analysis on the cleaned dataset. I loaded the dataset and printed the first 5 rows and the summary statistics for numerical columns. I counted the number of collisions by borough and plotted a bar chart of the results. I also counted the number of collisions by contributing factor and plotted a horizontal bar chart of the results. Finally, I counted the number of collisions by vehicle type and plotted a pie chart of the results.

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Load the cleaned dataset

df = pd.read_csv('C:/Motor_Vehicle_Collisions_-_Crashes_cleaned_updated.csv')


# Print the first 5 rows of the dataset

print(df.head())


# Summary statistics for numerical columns

print(df.describe())


# Count the number of collisions by borough

collisions_by_borough = df['BOROUGH'].value_counts()

print(collisions_by_borough)


# Plot a bar chart of the number of collisions by borough

plt.figure(figsize=(8, 6))

sns.barplot(x=collisions_by_borough.index, y=collisions_by_borough.values)
```

```
plt.title('Number of Collisions by Borough')

plt.xlabel('Borough')

plt.ylabel('Number of Collisions')

plt.show()


# Count the number of collisions by contributing factor

collisions_by_contributing_factor = df['CONTRIBUTING FACTOR VEHICLE
1'].value_counts()

print(collisions_by_contributing_factor)


# Plot a horizontal bar chart of the number of collisions by contributing
factor

plt.figure(figsize=(8, 12))

sns.barplot(x=collisions_by_contributing_factor.values,
y=collisions_by_contributing_factor.index)

plt.title('Number of Collisions by Contributing Factor')

plt.xlabel('Number of Collisions')

plt.ylabel('Contributing Factor')

plt.show()


# Count the number of collisions by vehicle type

collisions_by_vehicle_type = df['VEHICLE TYPE CODE
1'].value_counts().head(10)

print(collisions_by_vehicle_type)


# Plot a pie chart of the number of collisions by vehicle type

plt.figure(figsize=(8, 8))

plt.pie(collisions_by_vehicle_type.values,
labels=collisions_by_vehicle_type.index, autopct='%1.1f%%')

plt.title('Number of Collisions by Vehicle Type')

plt.show()
```

CRASH DATE CRASH TIME BOROUGH CONTRIBUTING FACTOR VEHICLE 1

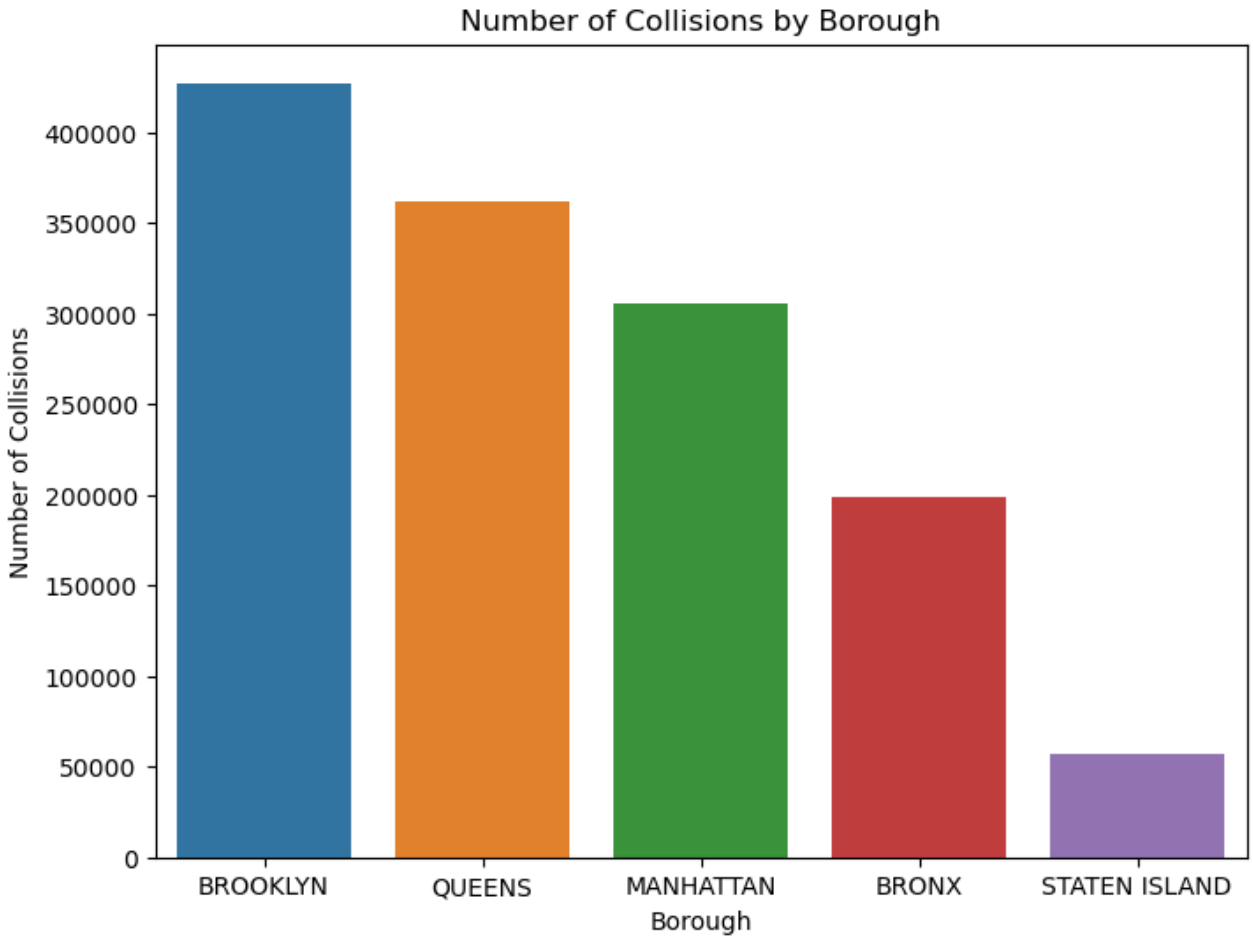
0	2021-09-11	1900-01-01 09:35:00	BROOKLYN	Unspecified	
1	2021-12-14	1900-01-01 08:17:00	BRONX	Unspecified	
2	2021-12-14	1900-01-01 21:10:00	BROOKLYN	Driver Inexperience	
3	2021-12-14	1900-01-01 14:58:00	MANHATTAN	Passing Too Closely	
4	2021-12-14	1900-01-01 16:50:00	QUEENS	Turning Improperly	

	CONTRIBUTING FACTOR VEHICLE 2	COLLISION_ID	VEHICLE TYPE CODE 1
0	Unspecified	4456314	Sedan
1	Unspecified	4486660	Sedan
2	Unspecified	4487074	Sedan
3	Unspecified	4486519	Sedan
4	Unspecified	4487127	Sedan

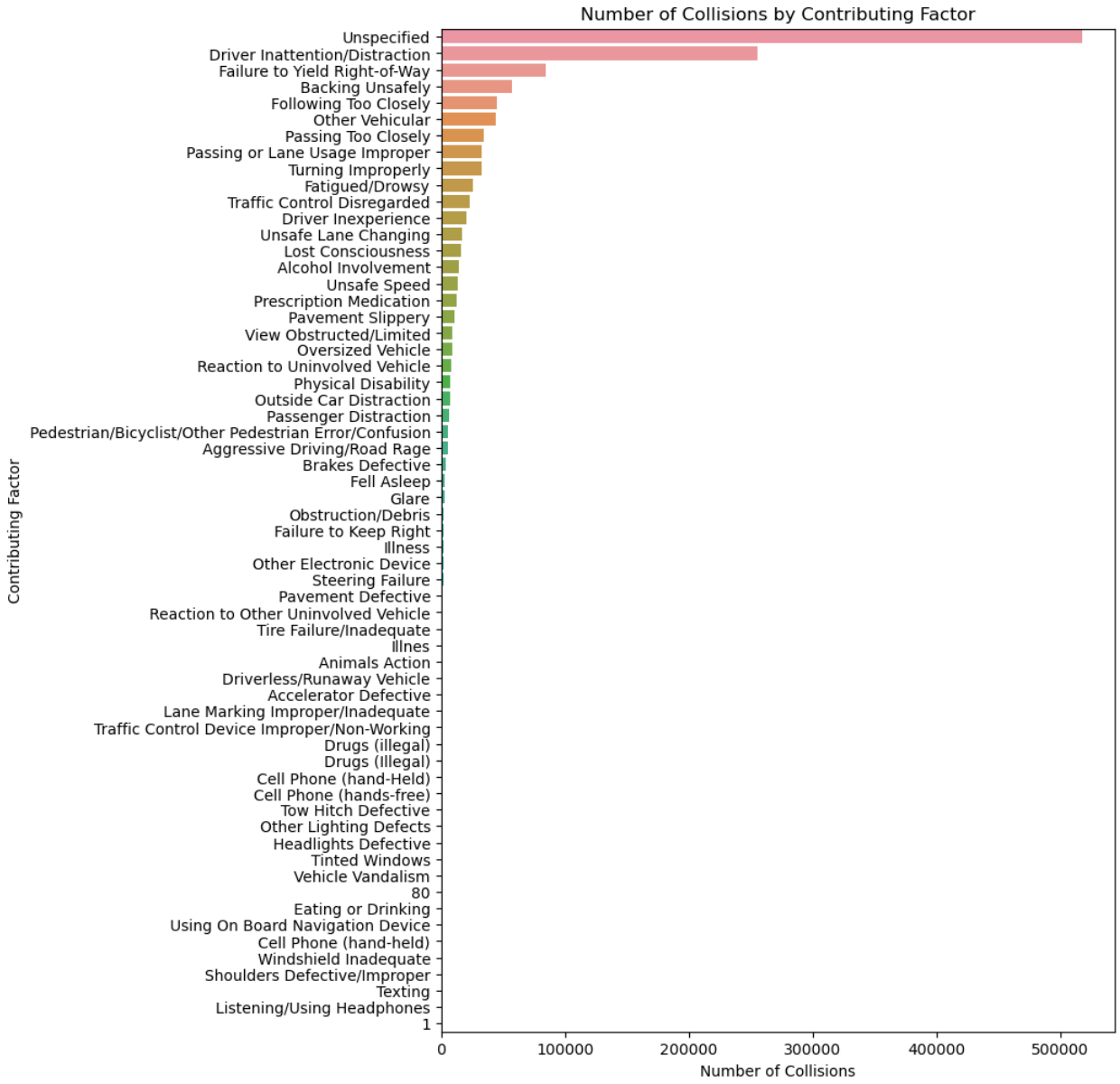
	COLLISION_ID
count	1.349431e+06
mean	2.897471e+06
std	1.620685e+06
min	2.200000e+01
25%	1.019316e+06
50%	3.548731e+06
75%	4.087466e+06
max	4.611005e+06

BROOKLYN	427032
QUEENS	362154
MANHATTAN	305130
BRONX	198413
STATEN ISLAND	56702

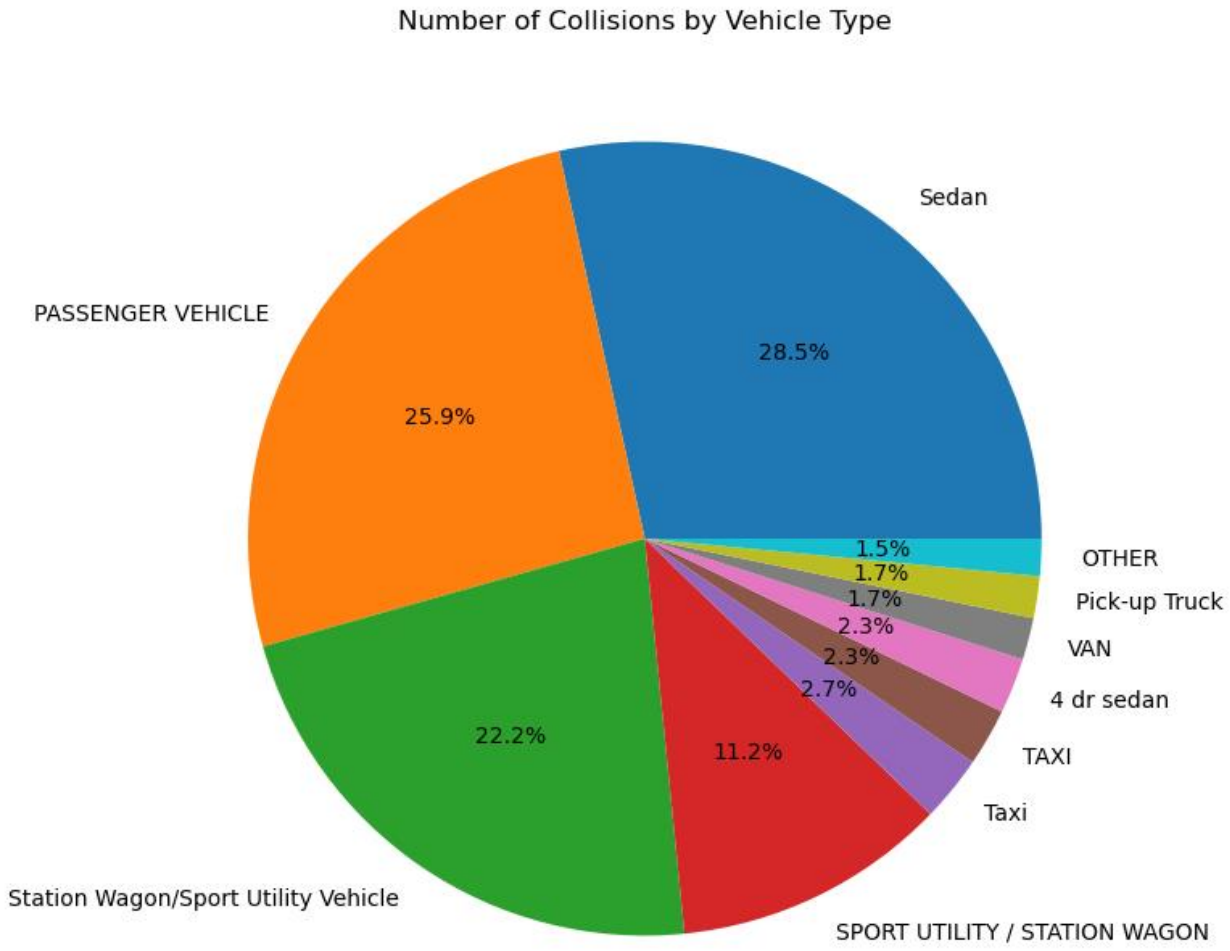
Name: BOROUGH, dtype: int64




```
Unspecified                          518073
Driver Inattention/Distraction       255815
Failure to Yield Right-of-Way         84684
Backing Unsafely                      57431
Following Too Closely                 44789
                                      ...
Windshield Inadequate                    50
Shoulders Defective/Improper             50
Texting                                  26
Listening/Using Headphones               14
1                                         8
Name: CONTRIBUTING FACTOR VEHICLE 1, Length: 61, dtype: int64
```




```
Sedan                                   340588
PASSENGER VEHICLE                       309834
Station Wagon/Sport Utility Vehicle     265685
SPORT UTILITY / STATION WAGON           133934
Taxi                                     32122
TAXI                                     28038
4 dr sedan                               26918
VAN                                      20503
Pick-up Truck                            20340
OTHER                                    18071
Name: VEHICLE TYPE CODE 1, dtype: int64
```



#5. Conclusions

Based on the analysis of the "Motor_Vehicle_Collisions_-_Crashes_cleaned_updated.csv" dataset, it is clear that there is a pressing need for targeted interventions to reduce the number of collisions and fatalities on New York City roads.

The analysis showed that Brooklyn had the highest number of collisions among all the boroughs, followed by Queens and Manhattan. Furthermore, driver inattention/distraction, failure to yield right-of-way, and backing unsafely were identified as the most common contributing factors to these collisions. Sedans, passenger vehicles, and station wagons/sport utility vehicles were the most frequently involved vehicle types in these collisions.

To address this issue, policymakers and law enforcement agencies should consider implementing targeted interventions such as increasing traffic enforcement, enhancing education and awareness campaigns on responsible driving, and improving road infrastructure to enhance road safety. Additionally, insurance companies can play a vital role in promoting safer driving practices by offering incentives for policyholders who exhibit responsible driving behavior, such as safe driving discounts.

Overall, it is critical for all stakeholders, including drivers, insurance companies, policymakers, and law enforcement agencies, to work collaboratively to create a safer driving environment in New York City. By taking a comprehensive and targeted approach, we can reduce the number of collisions and fatalities on the roads, ensuring that all road users can travel safely and efficiently.

#6. References

New York City Police Department (NYPD). (2021). Motor Vehicle Collisions—Crashes. https://data.cityofnewyork.us/Public-Safety/Motor-Vehicle-Collisions-Crashes/h9gi-nx95.