Data Driven Road Safety Transformation-Canada
Group 1
Capstone Project

Abigail Sunil Dsilva
Aesha Bhavanisinh Rathod
Ann babu Treeza
Srinivas Venuvanka

Data Driven Road Safety Transformation-Canada

Abigail Sunil Dsilva, Aesha Bhavanisinh Rathod, Ann Babu Treeza, Srinivas Venuvanka

ABSTRACT

Traffic collisions are a major public concern globally, as they are the leading cause of death for children and young adults, impacting millions of lives and leading to significant economic loss. Canada's vision for road safety is ambitious, aiming to boast the safest roads in the world by reducing collision injuries and fatalities. Our project employs the National Collision Database (NCDB) for Canada, spanning from 2000 to 2019, to predict and understand the causation of fatalities on Canadian roads. Through exploratory data analysis and machine learning techniques, including data mining methodologies, we delve into association rules and identify key factors contributing to fatal outcomes.

Our findings highlight the deadliness of head-on collisions, particularly in non-intersection areas where traffic control systems are absent. Furthermore, we have uncovered that most fatalities occur under non-extreme weather and road conditions. Surprisingly, "collision configuration" and "used safety devices" emerge as the most critical features, surpassing other factors such as vehicle year, time of collision, and demographic details of the individuals involved.

With this knowledge, we propose to design two supervised learning classification models for future work, aiming to enhance emergency response and road safety proactively. By focusing on the efficiency of first responders, including paramedics, firefighters, and police, we plan to optimize resource allocation during high-risk periods and improve overall preparedness. Our analysis also extends to the condition of road surfaces, providing timely alerts to authorities for necessary repairs.

Our exploratory and predictive analyses underscore the significance of road design and traffic safety education. By integrating a data-driven approach, we aim to reduce motor vehicle collisions, create safer roads, and save lives, reinforcing the notion that every life matters.

***KEYWORDS*:**
Motor Vehicle Collisions, Road safety; Data analysis, NCDB

1. INTRODUCTION:

The analysis of collision data in Canada is critical due to its considerable economic and personal impact. In 2020, the social cost of road collisions reached $35.98 billion, accounting for 1.92% of the GDP, highlighting the need for effective road safety strategies. Additionally, these collisions impose a substantial financial burden on individuals, costing an average of $946.65 per capita and $1,334.00 per licensed driver. [1]

From 2000 to 2019, the staggering toll of over 6.9 million road accidents led to nearly 47,000 fatalities and more than 3.6 million injuries. These figures reflect a profound loss of human life, considerable suffering, and a significant economic burden. Our initiative, titled "Data-Driven Road Safety Transformation," emerges from this critical situation, underscoring the importance of each life and the extensive impact road accidents have on communities. [2]

Building on the robust foundation of the National Collision Database (NCDB) [3] and informed by the practices outlined in Manitoba's Traffic Collision Statistics Report, our project extends the analysis to a national scale. By examining a broad range of factors from collision timings to weather conditions, and leveraging insights on reportable collisions and their causes, we aim to identify patterns and risk factors at a more granular level. This comprehensive approach is geared towards developing targeted strategies for reducing traffic collisions across Canada, enhancing overall road safety. [4]

With this thorough analysis, we plan to identify high-risk conditions and times, shaping the direction of specific preventative strategies. We also aim to enhance the readiness and efficiency of emergency response teams like paramedics, firefighters, and police officers through predictive modeling. This strategy is designed to prepare for and respond to high-risk incidents effectively, ensuring timely and efficient medical care for victims, especially during the busiest hours. Moreover, our project extends its focus to improving road safety. By assessing road surface conditions and other environmental factors, we aim to provide actionable insights for necessary road repairs and adjustments. Our project's scope is not limited to merely addressing the aftermath of road accidents; we are committed to proactive measures to prevent their occurrence.

An analysis of the data reveals critical insights: males are more frequently involved in these accidents than females, with 3.6M male involvements compared to 2.8M female. Notably, the age group most affected is 21 to 30 years, recording 1.3M cases, followed by the 31 to 40 age group with 1M cases.

The months of August and July see the highest incidence of accidents, while April has the least. Interestingly, most accidents occur during weekends and in the peak hours between 3 to 8 PM. Additionally, most of these accidents happen on straight and level roads, followed by curved roads.

These findings highlight the necessity of tailoring our strategies to these specific demographics, timings, and road conditions. In response, our project will develop targeted interventions aimed at these high-risk groups and situations. By doing so, we hope to make a substantial impact in reducing road accidents and enhancing overall road safety. These findings highlight the necessity of tailoring our strategies to these specific demographics, timings, and road conditions. In response, our project will develop targeted interventions aimed at these high-risk groups and situations. By doing so, we hope to make a substantial impact in reducing road accidents and enhancing overall road safety. [5]

In conclusion, our data-driven approach aspires to forge safer roads and a more resilient emergency response system, significantly enhancing all road users' safety and quality of life.

2. METHODOLOGY

Our methodology for the "Data-Driven Road Safety Transformation" project incorporates data acquisition, preprocessing, exploratory data analysis, and visualization using Python and Tableau to analyze road traffic collisions in Canada based on the National Collision Database (NCDB) datasets.

Data Acquisition: We sourced 20 distinct datasets from the Canada Open Data Portal, covering the period from 1999 to 2019. These datasets include all police-reported motor vehicle collisions on public roads in Canada, as captured in the NCDB. The data encompasses a wide array of variables related to fatal and injury collisions.

Data Merging and Loading: To consolidate the datasets, we employed Python scripts. We utilized the panda's library to read and merge individual Excel files from the specified directory, adding a 'C_YEAR' column to each dataset to maintain the year-wise distinction. The merged dataset was then saved as a CSV file for efficient handling. For data analysis, we utilized libraries such as NumPy, matplotlib, seaborn, and Google Colab for Python execution and data loading.

Data Preprocessing: The preprocessing stage involved renaming columns for clarity and handling null values in several important features. Techniques included replacing placeholder values with NaN or median values where appropriate and converting categorical data to numerical where necessary. Data types were adjusted for accurate statistical analysis.

Data Cleaning and Transformation: We focused on transforming various features into more analyzable formats. This included extracting year information from strings, converting categorical data to numerical values, and handling outliers and missing values effectively.

Exploratory Data Analysis (EDA): Our EDA involved univariate, bivariate, and multivariate analysis using Python. We inspected data types, checked for null values, and conducted a statistical summary to understand data distribution and characteristics. This process included visualizing data distributions and relationships using histograms, line graphs, and bar graphs.

Data Visualization: To augment our analysis and provide interactive insights, we employed Tableau alongside Python for data visualization. We developed an interactive dashboard that allowed users to dynamically filter and explore the data across different dimensions such as time, vehicle type, weather conditions, and road surfaces. This interactive approach enabled a more nuanced understanding of the factors contributing to road traffic accidents.

Interpretation and Validation: Our interpretation of the data involved an intricate analysis of patterns and trends in road traffic collisions. By leveraging the interactive dashboard, we were able to isolate the effects of individual factors, drawing comprehensive conclusions from the data. The findings were validated against established road safety research to ensure accuracy and relevance.

Statistical Analysis and Insights: The statistical analysis involved examining correlations, frequency distributions, and average values to derive meaningful insights. We explored relationships between weather conditions, road surfaces, vehicle types, and collision configurations. This analysis helped in understanding the most prevalent scenarios leading to collisions.

3. DATA ANALYSIS

From Fig 3.1, it's evident that there has been a total of 6,913,204 reported accidents between the year 2000 to 2019. This considerable number underlines the importance of road safety and accident prevention measures.

A key observation from fig 3.1 is the declining trend in the annual number of accidents. In 2000, there were 155.84K reported accidents, a number that gradually decreased to 108.60K by 2021. This decline of nearly 30% over two decades is significant and may suggest the positive impact of various road safety measures and policies implemented over the years. Two key factors contributing to Canada's success in road safety compared to the US include a preference for smaller vehicles and higher gas taxes. Canadians tend to choose somewhat smaller models of SUVs and trucks, which may be contributing to a lower rate of pedestrian and cyclist fatalities. Additionally, higher gas prices in Canada, partly due to taxes, seem to encourage less driving and the adoption of different travel habits compared to the US, potentially leading to fewer road accidents. [6] It could also reflect advancements in vehicle safety technologies, improved road conditions, and increased public awareness about safe driving practices.

Passenger severity	% of Total
Fatality	0.72%
Injury	56.01%
No Injury	43.27%

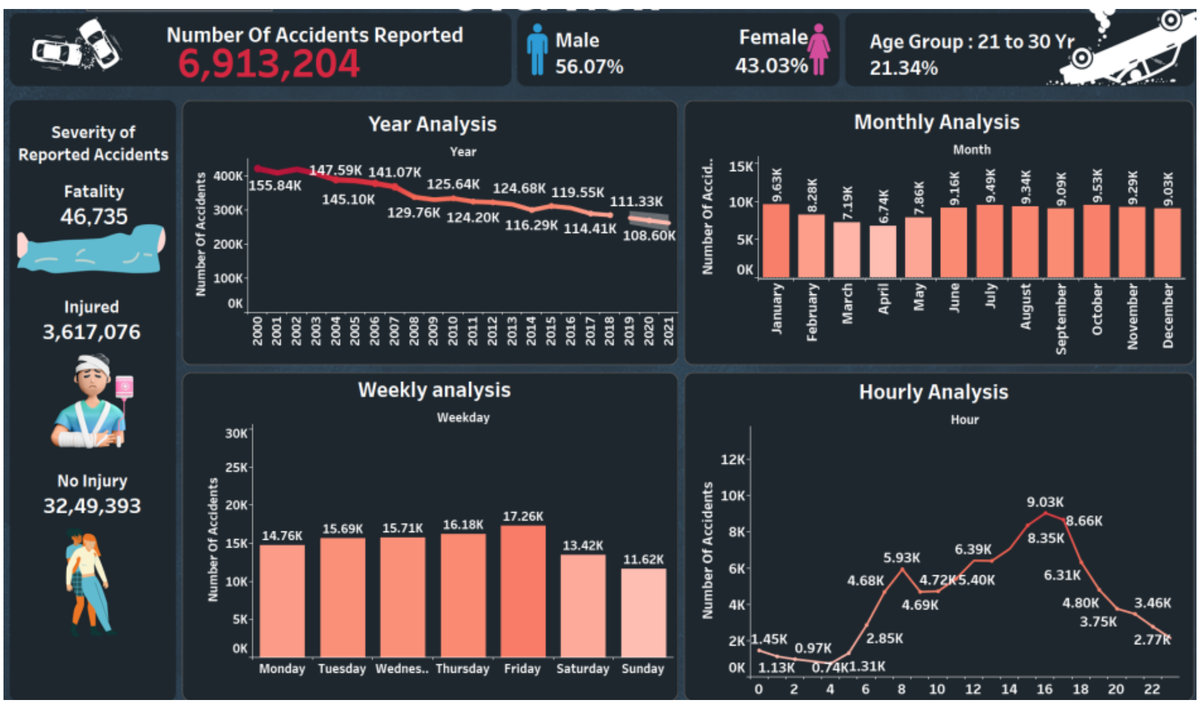


Fig 3.1

The demographic analysis of the accident data brings additional critical insights. A notable finding is that most of the accidents involved male drivers, who accounted for 56.07% of the total incidents. This statistic points to a gender disparity in road accidents and may indicate the need for targeted safety campaigns or interventions focused on male drivers. A notable finding is that most of the accidents involved male drivers, who accounted for 56.07% of the total incidents. [7]

Gender	Count of Case
Male	3,697,197
Female	2,896,561

Furthermore, the most affected age group in these accidents was the 21 to 30 years old demographic, representing 21.34% of the total incidents. This age group, typically characterized by relatively new and less experienced drivers, indicates a critical area for enhanced road safety education and stricter enforcement of traffic laws. It may also reflect lifestyle factors and driving habits prevalent in this age group, which could be addressed through tailored awareness and training programs. Road crashes are a leading cause of death among teenagers, with those aged 20 to 30 particularly vulnerable; alcohol and/or drugs play a role in 24% of these incidents. Males, especially at 19 years old, are more frequently involved, often due to risky behaviors like impaired driving. To mitigate this, it's recommended to intensify awareness and educational campaigns targeted at young drivers, focusing on the dangers of impaired driving, and promoting safer driving practices. [8]

Passenger Age (group)	Count of Cases
1 to 10	326,842
10 to 20	1,043,267
21 to 30	1,375,181
31 to 40	1,089,056
41 to 50	1,041,903
51 to 60	788,437
61 to 70	438,768
71 to 80	245,892
81 to 90	90,109
91 to 100	6,750

In terms of severity, there were 46,735 fatalities and a concerning number of injuries at 3,617,076, indicating a with higher traffic volumes. Hourly analysis reveals that accidents peak during evening rush hours, with the highest frequency at 6 PM, underscoring the need for enhanced.

The monthly analysis across the dashboards shows consistent patterns, with higher accident numbers in August, July, January and fewer in April. Weekly data reflects a higher number of accidents on weekdays, particularly on Fridays, which could alter safety measures during these hours.

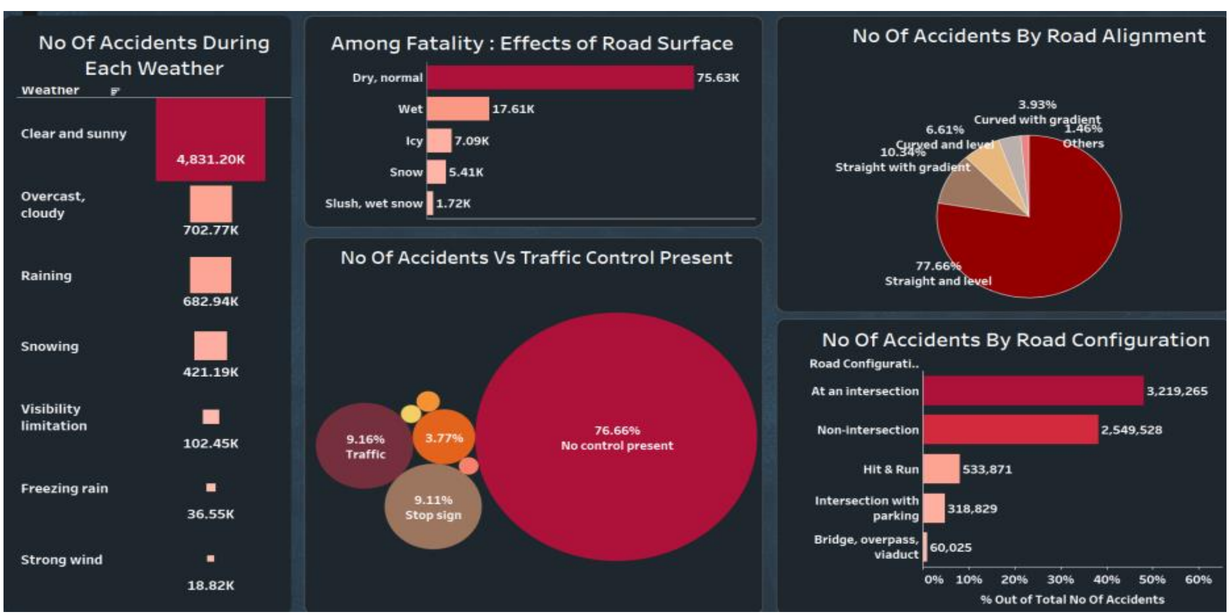


Fig 3.2

Collectively, these insights from the dashboards can inform policymakers and stakeholders in traffic management to focus on high-risk times and demographics for targeted interventions.

Fig 3.2 presents data on traffic accidents in relation to various conditions and scenarios. Most accidents occurred in clear and sunny weather, with over 4.83 million incidents, suggesting that driver complacency or other factors such as high traffic volumes during good weather conditions might play a significant role. In contrast, fewer accidents happened during adverse weather conditions like overcast/cloudy, rainy, and snowy weather, which together accounted for less than 1.8 million incidents.

When examining the effect of road surface conditions on fatalities, most occurred on dry, normal roads (75.63K), indicating that dangerous driving behavior may not be limited to adverse conditions. The presence of traffic control seems to significantly impact the occurrence of accidents, with a vast majority (76.66%) happening in areas with no traffic control present, highlighting the potential benefits of traffic regulation devices.

The presence of traffic control seems to significantly impact the occurrence of accidents, with a vast majority (76.66%) happening in areas with no traffic control present, highlighting the potential benefits of traffic regulation devices.

Looking at road alignment, accidents predominantly occurred on straight and level roads (77.66%), which might be due to higher speeds or inattention on these stretches. In terms of road configuration, intersections are the most common sites for accidents, with over 3.2 million incidents, more than double the number of non-intersection related accidents, which indicates that intersections are critical points for traffic safety interventions.

This data can be instrumental for traffic safety authorities to prioritize safety measures in clear weather conditions, enhance traffic control presence, and focus on intersection safety to reduce traffic accidents.

fig.3.3 indicates trends and distributions of traffic accidents by vehicle type, passenger position, and vehicle model year. Over the years, motorcycles and mopeds have been involved in the highest number of accidents, followed by bicycles, indicating that two-wheeled vehicles are particularly vulnerable on the roads. Trucks and vans also show a significant presence in accident statistics. When analyzing accidents by passenger position, the driver's seat is

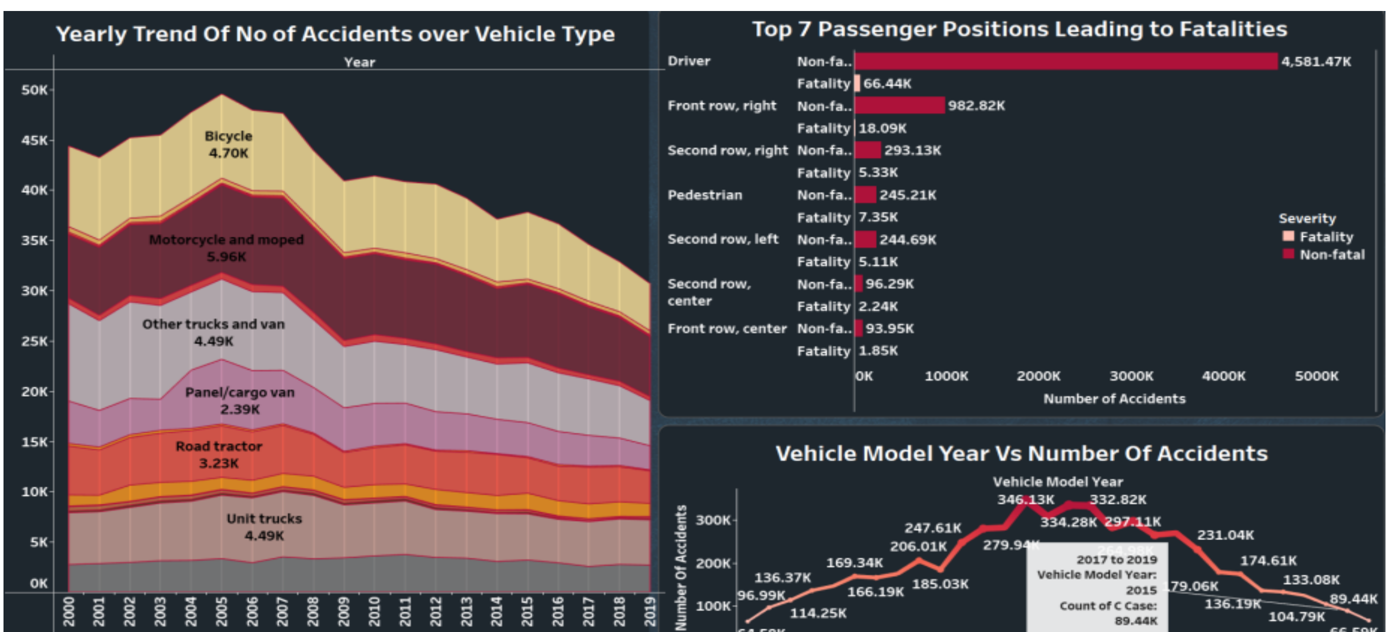


Fig 3.3

associated with the highest number of fatalities and non-fatal accidents, underscoring the high risk for vehicle operators. The front row, right seat (commonly the front passenger seat), is the second most frequent position for fatalities, which could point towards the impact of side collisions, critical need for ongoing road safety education and infrastructure improvement. However, many accidents resulted in no injuries, which suggests that not all reported incidents were severe.

Pedestrians also feature prominently in fatality statistics, which highlights the need for better pedestrian safety measures. The yearly trends of accidents over vehicle types seem to have varied fluctuations without a clear increasing or decreasing pattern, suggesting that multiple factors contribute to these accidents over time. In terms of vehicle model years, there is a noticeable decline in accidents involving newer models, possibly reflecting improvements in vehicle safety features. The data from this dashboard can guide targeted safety campaigns and regulatory measures to improve road safety, especially for vulnerable vehicle types and road users.

Fig 3.4 shows driver behavior, including overconfidence and distraction, increases accident risks on straight, non-intersection roads. Alcohol, drugs, and fatigue further exacerbate these dangers. Vehicle factors, such as older cars' lack of safety features and the vulnerability of two-wheeled vehicles, also contribute significantly.

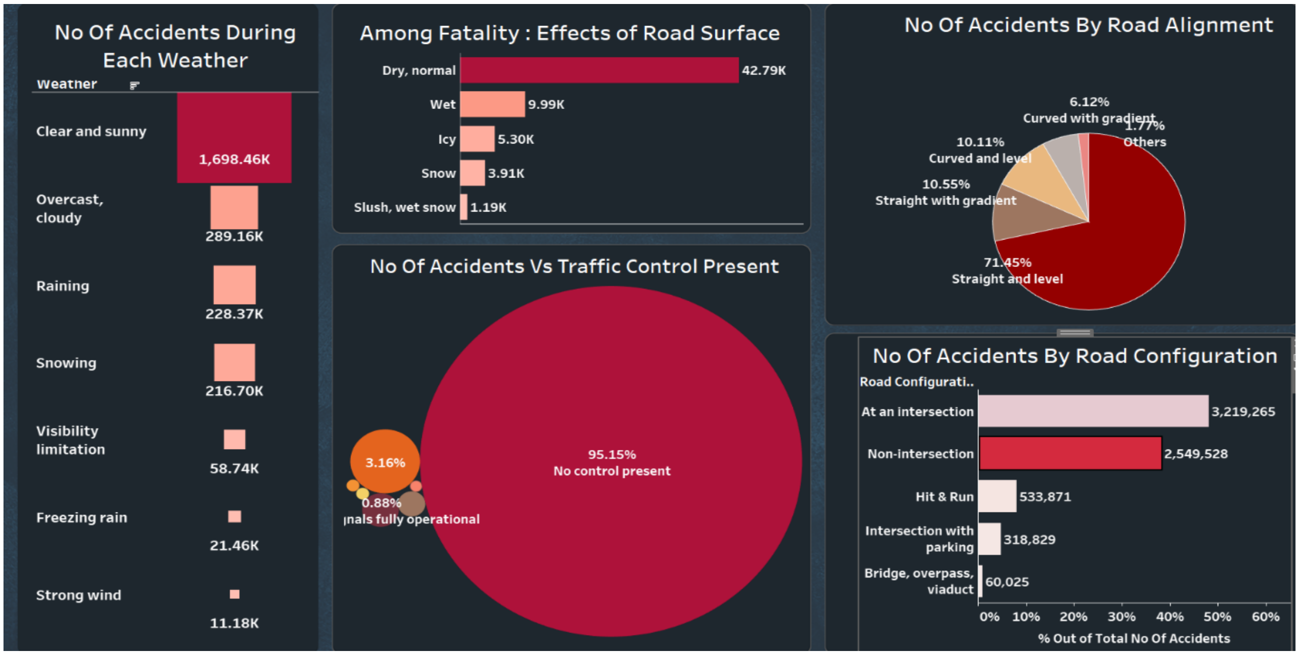


fig 3.4

Road characteristics, like the absence of traffic control, poor maintenance, and inadequate signage, increase the likelihood of accidents. Environmental factors, including weather and visibility, affect safety on these roads. Traffic patterns and volume, especially during rush hours, can create hazardous conditions.

Human elements, such as driver age and gender, play a crucial role. Younger, less experienced drivers and males are more prone to accidents. The involvement of bicyclists and pedestrians, particularly on roads without specific safety measures, is also a critical consideration.

4. INTERPRETATION & LIMITATION

Incomplete Dataset: The initial dataset does not include province-wise accident records for the 20-year span, necessitating separate data acquisition from each province, adding to the complexity of the project.

Data Format and Integration Challenges: The data received from provinces is in a non-standardized format, comprising results and findings rather than raw data. This necessitates extra effort for interpretation and integration into the overall analysis.

Year-wise Data Segregation: Data for each year was provided in separate Excel worksheets. The team had to first merge 20 years' worth of data using Python, before they could begin deriving insights and creating visualizations. This process added an additional layer of data handling complexity.

Generalizability of Findings: Insights and recommendations are specific to the dataset's geographic and temporal context and may not be applicable in other settings.

5. RECOMMENDATIONS:

Control Peak Time Collisions: To mitigate traffic congestion and accidents during peak times, several effective measures can be implemented: adjusting traffic light timings and speed limits to improve traffic flow and safety; employing quick response strategies like privacy screens and freeway patrols for incidents, alongside ramp metering and active traffic management to control vehicle entry on highways. Enhancing non-motorized transport infrastructure, like segregated bike lanes, and promoting carpooling can significantly reduce the number of vehicles on roads. Additionally, increasing public awareness about traffic safety and using data analytics for smart traffic management are crucial for preempting and addressing potential accident hotspots. [9]

Traffic Control Setup*:* Install traffic controls at high-risk spots such as non-intersections and curved roads. Place additional traffic management systems at intersections with adjacent parking.

Healthcare Strategy: To effectively manage peak time accidents, healthcare systems should increase medical personnel and resources, including ambulance services and hospital bed availability, during high-risk periods. Emergency facilities must prioritize critical care spaces like operating rooms and intensive care units, and schedule non-urgent procedures outside peak times to free up resources for emergencies. Additionally, conducting public awareness campaigns and strategically preparing healthcare services for heightened accident rates can significantly enhance emergency response and patient care during these periods.

Police: Seasonal Hazard Education*:* Increase educational efforts about driving risks during months with historically higher accident rates.

Road Authorities*:* To reduce traffic accidents, it's crucial to improve traffic control, especially on straight and level roads in clear conditions, and enhance road surfaces, as most fatalities occur on dry roads. Implementing engineering solutions such as improved signage and lane markings, along with promoting vehicle safety features, driver education, and data-driven law enforcement, will significantly boost road safety.

For Users - Seat Safety Instruction: Educate drivers and front-row passengers on optimal seat positioning and seat belt usage. Stress the importance of using seat belts correctly to reduce fatalities.

6. FUTURE WORK:

Employ K-Means Clustering to identify high-risk scenarios, which allows for proactive accident prevention measures. Utilize Time Series Analysis to predict trends and assist in the development of effective prevention strategies. Apply K-Means Clustering again to pinpoint potential high-risk scenarios accurately.

Implement Random Forests for a comprehensive analysis of risk factors, providing insightful data to inform road safety measures.

Upon receipt of the provincial data, which is currently presented in non-standardized formats with summarized results rather than raw figures, we will undertake the necessary efforts to interpret and standardize this information. This will allow for its integration into our broader data analysis and predictive modeling. Doing so is expected to yield more comprehensive insights into collision patterns across individual provinces:

7. APPENDIX

Python Code

Code for merging datasets for all years - 2000 to 2019¶

```
import os import pandas as pd

folder_path = r"C:\AeshaDAB\Sem3\Capstone1\Dataset" file_paths = [os.path.join(folder_path, file) for file in os.listdir(folder_path) if file.endswith(".xlsx")]

dataframes = []

for file_path in file_paths:

# Check if the file exists

if not os.path.exists(file_path):

   print(f"File '{file_path}' not found.")

else:

   print(f"Reading data from '{file_path}'")

   year = os.path.splitext(os.path.basename(file_path))[0]

   df = pd.read_excel(file_path)

   df['C_YEAR'] = year

   dataframes.append(df)


if dataframes:

# Merge all DataFrames into a single DataFrame

merged_data = pd.concat(dataframes, ignore_index=True)


# Save the merged DataFrame to a CSV file

merged_data.to_csv("merged_years_mvc.csv", index=False)


# Read the merged DataFrame from the CSV file

merged_data = pd.read_csv("merged_years_mvc.csv")


# Get unique years from the 'C_YEAR' column

unique_years = merged_data['C_YEAR'].unique()
```

num_unique_years = len(unique_years)

print(f"Number of unique years: {num_unique_years}")

else: print("No dataframes were created. Please check your files and paths.")

Loading the Dataset & Required Libraries¶

In [ ]:

from google.colab import drive

drive.mount('/content/drive')

Mounted at /content/drive

In [ ]:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

In [ ]:

file_path = '/content/drive/MyDrive/merged_years_mvc.csv'

df = pd.read_csv(file_path)

df.head(2).T

#df

<ipython-input-3-6a287497560f>:2: DtypeWarning: Columns (1,2,5,12) have mixed types. Specify dtype option on import or set low_memory=False.

df = pd.read_csv(file_path)

Out[ ]:

	0	1
C_YEAR	y_2000_en	y_2000_en
C_MNTH	1	1
C_WDAY	1	1
C_HOUR	16	16

	0	1
C_SEV	2	2
C_VEHS	2	2
C_CONF	21	21
C_RCFG	UU	UU
C_WTHR	1	1
C_RSUR	1	1
C_RALN	1	1
C_TRAF	18	18
V_ID	1	2
V_TYPE	1	1
V_YEAR	UUUU	UUUU
P_ID	1	1
P_SEX	M	F
P_AGE	33	32
P_PSN	11	11
P_ISEV	2	1
P_SAFE	2	NN
P_USER	1	1
C_CASE	151401	151401

Data Inspection¶

In [ ]:

#datacopy = df

In [ ]:

data = df

In [ ]:

```
#Inspecting the datatype of each feature

data.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 6913204 entries, 0 to 6913203

Data columns (total 23 columns):

 #   Column  Dtype

---  ------  -----

 0   C_YEAR  object

 1   C_MNTH  object

 2   C_WDAY  object

 3   C_HOUR  object

 4   C_SEV   int64

 5   C_VEHS  object

 6   C_CONF  object

 7   C_RCFG  object

 8   C_WTHR  object

 9   C_RSUR  object

 10  C_RALN  object

 11  C_TRAF  object

 12  V_ID    object

 13  V_TYPE  object

 14  V_YEAR  object

 15  P_ID    object

 16  P_SEX   object

 17  P_AGE   object

 18  P_PSN   object

 19  P_ISEV  object

 20  P_SAFE  object

 21  P_USER  object

 22  C_CASE  int64
```

dtypes: int64(2), object(21)

memory usage: 1.2+ GB

In [ ]:

```
#Viewing data dimensions

data.shape
```

Out[ ]:

(6913204, 23)

There are 6.9 million records, and 23 features.

Rename The Columns¶

In [ ]:

```
#Renaming the columns for better understanding

data = data.rename(columns={

   'C_YEAR': 'Year',

   'C_MNTH': 'Month',

   'C_WDAY': 'Weekday',

   'C_HOUR': 'Hour',

   'C_SEV': 'Severity',

   'C_VEHS': 'Num_vehicles',

   'C_CONF': 'Collision_configuration',

   'C_RCFG': 'Road_configuration',

   'C_WTHR': 'Weather_condition',

   'C_RSUR': 'Road_surface',

   'C_RALN': 'Road_alignment',

   'C_TRAF': 'Traffic_control',

   'V_ID': 'vehicle_id',

   'V_TYPE': 'Vehicle_type',

   'V_YEAR': 'Vehicle_year',

   'P_ID': 'Person_id',

   'P_SEX': 'Person_sex',

   'P_AGE': 'Person_age',
```

'P_PSN': 'Person_position',

'P_ISEV': 'Person_injury_severity',

'P_SAFE': 'Safety_device_used',

'P_USER': 'Road_user_type',

'C_CASE': 'Collision_case',

})

Statistical Summary¶

In [ ]:

#Summary Statistics

data.describe(include='all').T

Out[ ]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Year	6913204	20	y_2000_en	422075	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Month	6913204.0	17.0	8.0	640546.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Weekday	6913204.0	15.0	5.0	675079.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Hour	6913204	25	16	616745	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Severity	6913204.0	NaN	NaN	NaN	1.983509	0.127354	1.0	2.0	2.0	2.0	2.0
Num_vehicles	6913204.0	86.0	2.0	2588234.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Collision_configuration	6913204	20	21	2081140	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Road_configuration	6913204	12	2	3219265	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Weather_condition	6913204	9	1	4831197	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Road_surface	6913204	11	1	4559289	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Road_alignment	6913204	8	1	4977754	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Traffic_control	6913204	19	18	3571219	NaN	NaN	NaN	NaN	NaN	NaN	NaN
vehicle_id	6913204	164	1	2340067	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Vehicle_type	6913204	20	1	5739898	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Vehicle_year	6913204	116	UUUU	362375	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Person_id	6913204	96	1	4959635	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Person_sex	6913204	4	M	3697197	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Person_age	6913204	101	UU	447220	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Person_position	6913204	16	11	4647907	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Person_injury_severity	6913204	5	2	3617076	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Safety_device_used	6913204	10	2	4903542	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Road_user_type	6913204	6	1	4329482	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Collision_case	6913204.0	NaN	NaN	NaN	1457657.782408	760672.067969	151316.0	795588.0	1455185.0	2114928.25	2786266.0

In [ ]:

```
categorical_df = pd.DataFrame(data[categorical_attributes])
```

```
# Set the plot size
plt.figure(figsize=(15, 10))

# Loop through each categorical attribute and create count plots
for i, col in enumerate(categorical_df.columns):
    plt.subplot(4, 3, i+1)  # Adjust the subplot layout as per your number of attributes
    sns.countplot(x=col, data=categorical_df)
    plt.xticks(rotation='vertical')
    plt.xlabel(col)
    plt.ylabel('Count')

plt.tight_layout()  # Adjust the spacing between subplots if necessary
plt.show()


---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-30-6ae69ab7f5bd> in <cell line: 1>()
----> 1 categorical_df = pd.DataFrame(data[categorical_attributes])
      2
      3 # Set the plot size
      4 plt.figure(figsize=(15, 10))
      5

NameError: name 'categorical_attributes' is not defined
```

In [ ]:

```
# Checking for null values
null_values = data.isnull().sum()
null_values
```

Out[ ]:

```
Year                    0
Month                   0
Weekday                 0
Hour                    0
Severity                0
Num_vehicles            0
Collision_configuration 0
Road_configuration      0
Weather_condition       0
Road_surface            0
Road_alignment          0
Traffic_control         0
vehicle_id              0
Vehicle_type            0
Vehicle_year            0
Person_id               0
Person_sex              0
Person_age              0
Person_position         0
Person_injury_severity  0
Safety_device_used      0
Road_user_type          0
Collision_case          0
dtype: int64
```

In [ ]:

```
#Count of unique values in each feature
data.nunique()
```

Out[ ]:

```
Year                    20
Month                   17
```

```
Weekday                    15
Hour                       25
Severity                    2
Num_vehicles               86
Collision_configuration    20
Road_configuration         12
Weather_condition           9
Road_surface               11
Road_alignment              8
Traffic_control            19
vehicle_id                164
Vehicle_type               20
Vehicle_year              116
Person_id                  96
Person_sex                  4
Person_age                101
Person_position            16
Person_injury_severity      5
Safety_device_used         10
Road_user_type              6
Collision_case        2634951
dtype: int64
```

In [ ]:

```
#We are not converting U's because, that means its a hit and run case


data.replace(['U', 'UU', 'UUUU'], 0, inplace=True)


#convert ',N', 'Q', 'NN', 'NNNN', 'QQ', 'QQQQ', 'QQQQ', 'NNN' to np.nan
placeholders_for_nan = [',N', 'Q', 'NN', 'NNNN', 'QQ', 'QQQQ', 'QQQQ', 'NNN']
data.replace(placeholders_for_nan, np.nan, inplace=True)
```

In [ ]:

```
#sum of null values in each feature
data.isnull().sum()
```

Out[ ]:

```
Year                      0
Month                     0
Weekday                   0
Hour                      0
Severity                  0
Num_vehicles              0
Collision_configuration   351708
Road_configuration        176414
Weather_condition         16749
Road_surface              203519
Road_alignment            31353
Traffic_control           96489
vehicle_id                0
Vehicle_type              310410
Vehicle_year              317721
Person_id                 12334
Person_sex                0
Person_age                19779
Person_position           66112
Person_injury_severity    0
Safety_device_used        773948
Road_user_type            0
Collision_case            0
dtype: int64
```

Handling Null Values in Important Features¶

Person_age

In [ ]:

```
age_median = data['Person_age'].median()
#Filled NaN values in 'Person_age' with the calculated median
data['Person_age'].fillna(age_median, inplace=True)
data.isnull().sum()
```

Out[ ]:

```
Year                      0
Month                     0
Weekday                   0
Hour                      0
Severity                  0
Num_vehicles              0
Collision_configuration   351708
Road_configuration        176414
Weather_condition         16749
Road_surface              203519
Road_alignment            31353
Traffic_control           96489
vehicle_id                0
Vehicle_type              310410
Vehicle_year              0
Person_id                 12334
Person_sex                0
Person_age                0
Person_position           66112
Person_injury_severity    0
Safety_device_used        0
Road_user_type            0
Collision_case            0
dtype: int64
```

In [ ]:

```
#changing 'Person_age' datatype from object to integer
data['Person_age'] = data['Person_age'].astype(int)
```

Safety Device Used

In [ ]:

```
data['Safety_device_used'].unique()
```

Out[ ]:

```
array(['2', nan, '12', '9', '13', '1', 0, '11', '10'], dtype=object)
```

In [ ]:

```
#we will replace nan with 20 here (categorical value): which means people might not have used any safety devices
data['Safety_device_used'].fillna(50, inplace=True)
data['Safety_device_used'].isnull().sum()
```

Out[ ]:

```
0
```

In [ ]:

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

Out[ ]:

```
Year                     0
Month                    0
Weekday                  0
Hour                     0
Severity                 0
Num_vehicles             0
Collision_configuration  351708
Road_configuration       176414
Weather_condition        16749
Road_surface             203519
Road_alignment           31353
Traffic_control          96489
```

```
vehicle_id                0
Vehicle_type         310410
Vehicle_year         317721
Person_id             12334
Person_sex                0
Person_age                0
Person_position       66112
Person_injury_severity    0
Safety_device_used        0
Road_user_type            0
Collision_case            0
dtype: int64
```

Year (extract the year from string)

In [ ]:

```
# Handling the 'Year' column: Extracting the year from the string
data['Year']= data['Year'].str.extract('(\d+)')
data['Year']
```

Out[ ]:

```
0          2000
1          2000
2          2000
3          2000
4          2000
           ...
6913199    2019
6913200    2019
6913201    2019
6913202    2019
6913203    2019
Name: Year, Length: 6913204, dtype: object
```

Vehicle_year

In [ ]:

```
vehicle_year_median = data['Vehicle_year'].median()
#Filled NaN values in 'Vehicle_year' with the calculated median
data['Vehicle_year'].fillna(vehicle_year_median, inplace=True)
```

Converting categorical datatypes to numerical¶

In [ ]:

```
#data['Year'] = pd.to_numeric(data['Year']).astype(int)
data['Hour'] = pd.to_numeric(data['Hour']).astype(int)
data['Weekday'] = pd.to_numeric(data['Weekday']).astype(int)
data['Month'] = pd.to_numeric(data['Month']).astype(int)
data['Num_vehicles'] = pd.to_numeric(data['Num_vehicles']).astype(int)
#data['vehicle_id'] = pd.to_numeric(data['vehicle_id']).astype(int)
data['Vehicle_year'] = pd.to_numeric(data['Vehicle_year']).astype(int)
#data['Person_id'] = pd.to_numeric(data['Person_id']).astype(int)
```

In [ ]:

```
data.dtypes
```

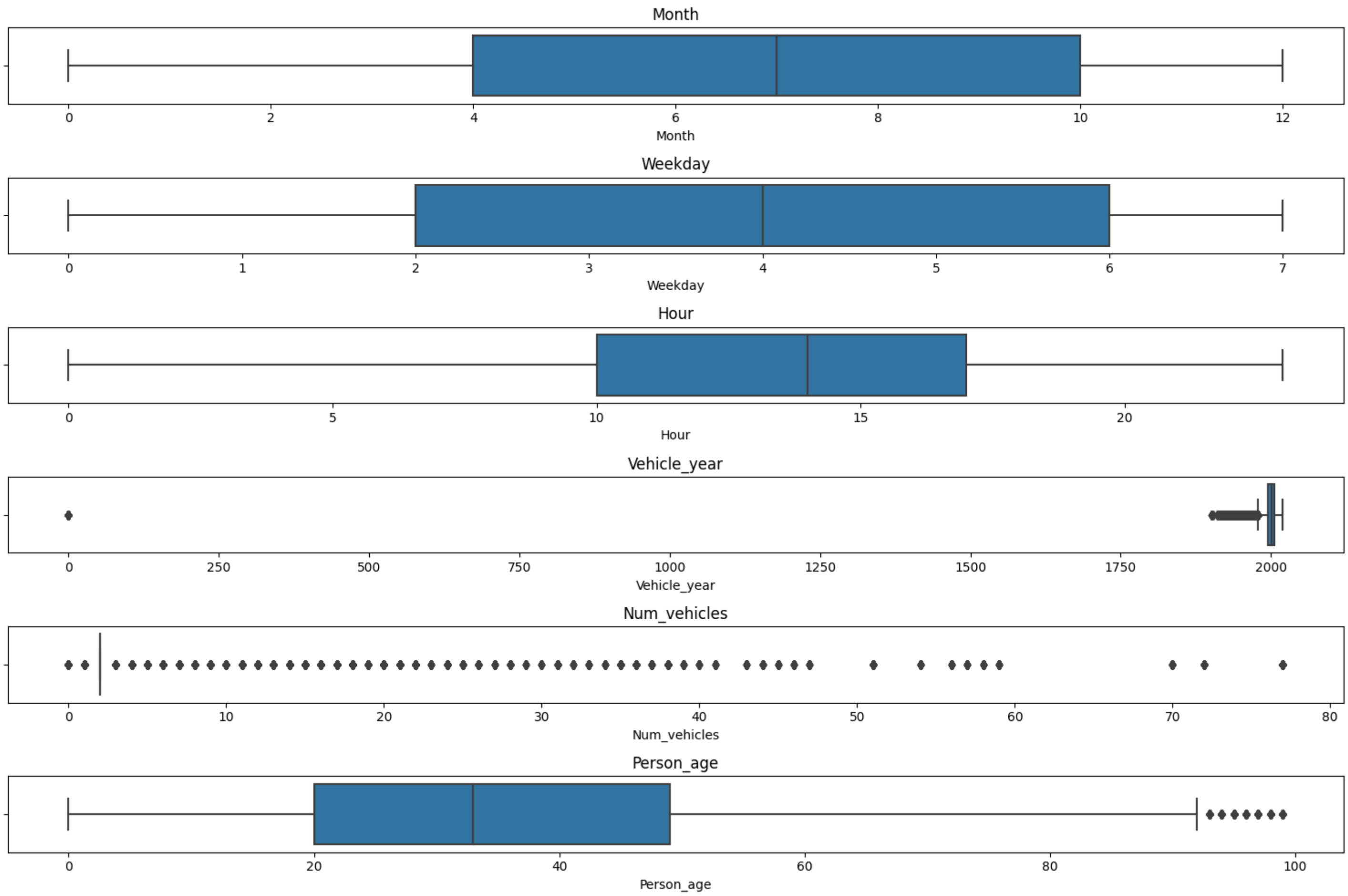
Out[ ]:

```
Year                    object
Month                    int64
Weekday                  int64
Hour                     int64
Severity                 int64
Num_vehicles             int64
Collision_configuration    object
Road_configuration        object
Weather_condition         object
Road_surface              object
Road_alignment            object
Traffic_control           object
```

```
vehicle_id              object
Vehicle_type            object
Vehicle_year             int64
Person_id               object
Person_sex              object
Person_age               int64
Person_position         object
Person_injury_severity   object
Safety_device_used      object
Road_user_type          object
Collision_case           int64
dtype: object
```

In [ ]:

```
#Filtering the integer columns based on the specified list
filtered_integer_columns = ['Month', 'Weekday', 'Hour','Vehicle_year', 'Num_vehicles',  'Person_age']

plt.figure(figsize=(15, 10))

for i, column in enumerate(filtered_integer_columns, 1):# for index, value in enumerate(iterable,starat = 0):
    plt.subplot(len(filtered_integer_columns), 1, i)
    sns.boxplot(x=data[column])
    plt.title(column)

plt.tight_layout()
plt.show()
```



null_values in month¶

In [ ]:

```
data.loc[data['Month'] > 12] = np.nan
#month_median = data['Month'].median()
data['Month'].fillna(13, inplace=True)
data['Month'].unique()
```

Out[ ]:

```
array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12.,  0.])
```

In [ ]:

```
#month_median
```

In [ ]:

```
data.shape
```

Out[ ]:

(6913204, 23)

In [ ]:

Null values in vehicle year¶

In [ ]:

```
data['Vehicle_year'].unique()
#we have a zero, which is a outlier
```

Out[ ]:

```
array([   0., 2001., 2000., 1997., 1989., 1999., 1993., 1990., 1991.,
       1980., 1994., 1988., 1995., 1996., 1982., 1998., 1985., 1986.,
       1992., 1987., 1984., 1979., 1978., 1983., 1975., 1976., 1974.,
       1981., 1967., 1977., 1961., 1964., 1972., 1956., 1971., 1973.,
       1969., 1970., 1947., 1957., 1963., 1966., 1955., 1950., 1962.,
       1917., 1948., 1968., 1914., 1913., 1940., 1920., 1951., 1965.,
       1932., 1925., 1953., 1927., 2002., 1939., 1960., 1941., 1945.,
       1938., 1949., 1954., 1916., 1933., 1935., 1929., 1937., 1959.,
       1930., 1923., 1928., 1944., 1958., 1942., 1919., 1926., 1931.,
       1918., 2003., 1952., 1946., 1924., 1922., 1901., 1915., 1934.,
       1903., 2004., 2005., 1904., 2006., 1912., 2007., 2008., 1943.,
       1911., 2009., 2010., 2011., 1936., 2012., 1910., 1921., 2013.,
       2014., 2015., 2016., 2017., 2018., 2019., 2020.])
```

In [ ]:

```
#Filtering out the years and removing outliers
data = data[(data['Vehicle_year'] > 0) & (data['Vehicle_year'] <= 2019)]
unique_vehicle_years = data['Vehicle_year'].unique()
unique_vehicle_years
```

Out[ ]:

```
array([2001., 2000., 1997., 1989., 1999., 1993., 1990., 1991., 1980.,
       1994., 1988., 1995., 1996., 1982., 1998., 1985., 1986., 1992.,
       1987., 1984., 1979., 1978., 1983., 1975., 1976., 1974., 1981.,
```

```
    1967., 1977., 1961., 1964., 1972., 1956., 1971., 1973., 1969.,
    1970., 1947., 1957., 1963., 1966., 1955., 1950., 1962., 1917.,
    1948., 1968., 1914., 1913., 1940., 1920., 1951., 1965., 1932.,
    1925., 1953., 1927., 2002., 1939., 1960., 1941., 1945., 1938.,
    1949., 1954., 1916., 1933., 1935., 1929., 1937., 1959., 1930.,
    1923., 1928., 1944., 1958., 1942., 1919., 1926., 1931., 1918.,
    2003., 1952., 1946., 1924., 1922., 1901., 1915., 1934., 1903.,
    2004., 2005., 1904., 2006., 1912., 2007., 2008., 1943., 1911.,
    2009., 2010., 2011., 1936., 2012., 1910., 1921., 2013., 2014.,
    2015., 2016., 2017., 2018., 2019.])
```

Null values in vehicle type¶

In [ ]:

```
print(f"unique values : {data['Vehicle_type'].unique()}")
print(f"Total null_values in vehicle type {data['Vehicle_type'].isnull().sum()}")
#we have many nans
# 0 means hit and run
# will replace the null values with a different number (since this is categorical column we are not using any numbers, and replacing the nans with 25)
```

```
unique values : [nan '1' 0 '16' '5' '8' '7' '22' '6' '11' '17' '20' '9' '23' '14' '10'
 '18' '21' '19']
Total null_values in vehicle type 304022
```

In [ ]:

```
data['Vehicle_type'].fillna(50, inplace=True)
data['Vehicle_type'].unique()
```

Out[ ]:

```
array([50, '1', 0, '16', '5', '8', '7', '22', '6', '11', '17', '20', '9',
    '23', '14', '10', '18', '21', '19'], dtype=object)
```

In [ ]:

```
print(f"Total null_values in vehicle type {data['Vehicle_type'].isnull().sum()}")
```

```
Total null_values in vehicle type 0
```

Null values in Collision Cofiguration¶

In [ ]:

print(f"unique values : {data['Collision_configuration'].unique()}")

print(f"Total null_values in Collision configuration{data['Collision_configuration'].isnull().sum()}")

#we have many nans

0 means hit and run

will replace the null values with a different number (since this is categorical column we are not using any numbers)

#50 means different category : Unknown.

unique values : [nan '2' '21' '3' 0 '35' '4' '1' '6' '33' '31' '24' '22' '32' '23' '41'

'5' '34' '36' '25']

Total null_values in Collision configuration318788

In [ ]:

data['Collision_configuration'].fillna(50, inplace=True)

#50 means different category : Unknown.

Null values in Road Surface¶

In [ ]:

print(f"unique values : {data['Road_surface'].unique()}")

print(f"Total null_values in Road_surface {data['Road_surface'].isnull().sum()}")

#we have many nans

0 means hit and run

will replace the null values with a different number (since this is categorical column we are not using any numbers)

#50 means different category : Unknown.

data['Road_surface'].fillna(50, inplace=True)

print(f"Total null_values in Road_surface after cleaning is {data['Road_surface'].isnull().sum()}")

#50 means different category : Unknown.

unique values : ['2' '3' '1' '5' '4' 0 nan '6' '7' '9' '8']

Total null_values in Road_surface 200570

Total null_values in Road_surface after cleaning is 0

In [ ]:

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

Out[ ]:

```
Year                       0
Month                      0
Weekday                    0
Hour                       0
Severity                   0
Num_vehicles               0
Collision_configuration    0
Road_configuration         164897
Weather_condition          16101
Road_surface               0
Road_alignment             29762
Traffic_control            92256
vehicle_id                 0
Vehicle_type               0
Vehicle_year               0
Person_id                  11598
Person_sex                 0
Person_age                 0
Person_position            63496
Person_injury_severity     0
Safety_device_used         0
Road_user_type             0
Collision_case             0
dtype: int64
```

Null values in Traffic Control¶

In [ ]:

```
print(f"unique values : {data['Traffic_control'].unique()}")
```

print(f"Total null_values in Traffic_control {data['Traffic_control'].isnull().sum()}")

#we have many nans

0 means hit and run

will replace the null values with a different number (since this is categorical column we are not using any numbers)

#50 means different category : Unknown.

data['Traffic_control'].fillna(50, inplace=True)

print(f"Total null_values in Traffic_control after cleaning is {data['Traffic_control'].isnull().sum()}")

#50 means different category : Unknown.

unique values : ['18' '1' 0 '3' '6' '11' '8' '10' '15' '4' '13' '2' '5' nan '16' '17' '7'

'9' '12']

Total null_values in Traffic_control 92256

Total null_values in Traffic_control after cleaning is 0

Null Values in Road Configuration¶

In [ ]:

print(f"unique values : {data['Road_configuration'].unique()}")

print(f"Total null_values in Road_configuration {data['Road_configuration'].isnull().sum()}")

#we have many nans

0 means hit and run

will replace the null values with a different number (since this is categorical column we are not using any numbers)

#50 means different category : Unknown.

data['Road_configuration'].fillna(50, inplace=True)

print(f"Total null_values in Road_configuration after cleaning is {data['Road_configuration'].isnull().sum()}")

#50 means different category : Unknown.

unique values : [0 '2' nan '1' '5' '4' '6' '3' '8' '7' '9' '10']

Total null_values in Road_configuration 164897

Total null_values in Road_configuration after cleaning is 0

Null Values in Person_position¶

In [ ]:

print(f"unique values : {data['Person_position'].unique()}")

print(f"Total null_values in Person_position {data['Person_position'].isnull().sum()}")

#we have many nans

0 means hit and run

will replace the null values with a different number (since this is categorical column we are not using any numbers)

#50 means different category : Unknown.

data['Person_position'].fillna(50, inplace=True)

print(f"Total null_values in Person_position after cleaning is {data['Person_position'].isnull().sum()}")

#50 means different category : Unknown.

unique values : ['99' '11' nan '13' '12' '21' '22' '23' 0 '98' '96' '32' '33' '31' '97']

Total null_values in Person_position 63496

Total null_values in Person_position after cleaning is 0

Drop the null values¶

In [ ]:

#Dropping the null values

data_clean = data.dropna()

data_clean.shape

Out[ ]:

(6492680, 23)

Univariate & Bivariate Analysis:¶

In [ ]:

#Summary Statistics

data.describe().T

Out[ ]:

	count	mean	std	min	25%	50%	75%	max
Month	6549929.0	6.699383e+00	3.451154	0.0	4.0	7.0	10.0	12.0
Weekday	6549929.0	4.002606e+00	1.931725	0.0	2.0	4.0	6.0	7.0

	count	mean	std	min	25%	50%	75%	max
Hour	6549929.0	1.355844e+01	5.288638	0.0	10.0	14.0	17.0	23.0
Severity	6549929.0	1.983477e+00	0.127475	1.0	2.0	2.0	2.0	2.0
Num_vehicles	6549929.0	2.054547e+00	1.299170	0.0	2.0	2.0	2.0	77.0
Vehicle_year	6549929.0	2.001152e+03	7.688022	1901.0	1996.0	2001.0	2006.0	2019.0
Person_age	6549929.0	3.533600e+01	19.813944	0.0	21.0	33.0	49.0	99.0
Collision_case	6549929.0	1.460722e+06	761217.311964	151316.0	798775.0	1458941.0	2120909.0	2786266.0

In [ ]:

```
data_clean = data.dropna()

data_clean.shape
```

Out[ ]:

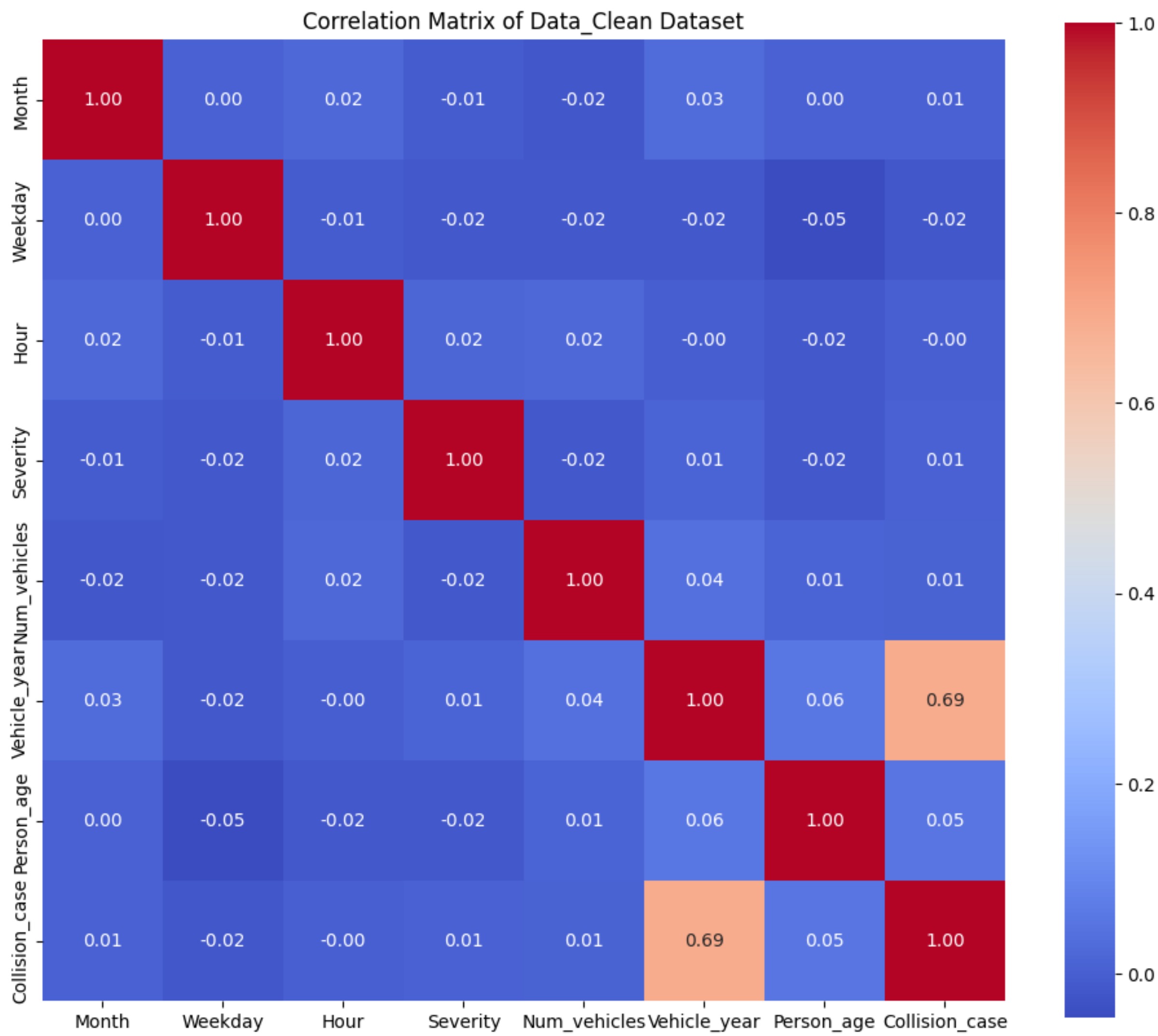
(6492680, 23)

Correlation matrix¶

In [ ]:

```
correlation_matrix = data_clean.corr()

plt.figure(figsize=(12, 10))

sns.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm', square=True)

plt.title('Correlation Matrix of Data_Clean Dataset')

plt.show()
```

```
<ipython-input-66-8218755c7c4e>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

  correlation_matrix = data_clean.corr()
```



Number of Unique Cases¶

In [ ]:

```
unique_collision_cases_sum = data_clean['Collision_case'].nunique()
unique_collision_cases_sum
```

Out[ ]:

2560782

Year with maximum number of unique cases (Sorted in descending order)¶

In [ ]:

```
unique_cases_per_year = data_clean.groupby('Year')['Collision_case'].nunique()
unique_cases_per_year.sort_values(ascending=False)
```

Out[ ]:

```
Year
2002    151141
2000    150056
2003    147919
2001    146380
2005    143331
2004    142504
2006    140337
2007    136607
2008    125548
2010    122219
2012    121614
2009    121272
2011    121025
2013    119127
2015    117803
2016    116538
2014    113344
2017    111519
2018    109584
2019    102914
Name: Collision_case, dtype: int64
```

Hour with maximum number of unique cases (Sorted in descending order)¶

In [ ]:

```
unique_cases_per_hour = data_clean.groupby('Hour')['Collision_case'].nunique()
unique_cases_per_hour.sort_values(ascending=False)
```

Out[ ]:

```
Hour
16.0    216507
17.0    208441
```

15.0   203258

14.0   165569

18.0   156661

12.0   154995

13.0   153310

8.0   140341

11.0   130619

19.0   118069

10.0   112298

9.0   107904

7.0   106143

20.0   93194

21.0   87364

22.0   71649

0.0   67204

6.0   59781

23.0   56553

2.0   35483

1.0   34918

3.0   30846

5.0   27487

4.0   22188

Name: Collision_case, dtype: int64

Month with maximum number of unique cases (Sorted in descending order)

In [ ]:

```
unique_cases_per_month = data_clean.groupby('Month')['Collision_case'].nunique()
unique_cases_per_month.sort_values(ascending=False)
```

Out[ ]:

Month

8.0   230469

```
7.0     229305
12.0    229233
10.0    227656
11.0    226297
6.0     226175
9.0     224721
1.0     222168
5.0     203701
2.0     188920
3.0     180834
4.0     171209
0.0         94
Name: Collision_case, dtype: int64
```

Weekday with maximum number of unique cases (Sorted in descending order)

In [ ]:

```
unique_cases_per_weekday = data_clean.groupby('Weekday')['Collision_case'].nunique()
unique_cases_per_weekday.sort_values(ascending=False)
```

Out[ ]:

```
Weekday
5.0    431478
4.0    392243
3.0    373651
2.0    370748
6.0    351332
1.0    350362
7.0    290767
0.0       201
Name: Collision_case, dtype: int64
```

In [ ]:

```
unique_cases_per_for_collisionconfiguration =
data_clean.groupby('Collision_configuration')['Collision_case'].nunique()

unique_cases_per_for_collisionconfiguration.sort_values(ascending=False)
```

Out[ ]:

```
Collision_configuration
21    625006
35    344135
6     338104
36    184885
33    156030
4     150438
50    136063
2     128656
3     110396
0      79816
22     72884
31     72489
1      32307
41     31731
23     26813
24     21457
32     19178
34     13691
5      10330
25      6373
Name: Collision_case, dtype: int64
```

In [ ]:

```
#Checking the Severity level

severity_counts = data_clean['Severity'].value_counts()

severity_counts
```

Out[ ]:

```
2.0    6385227
1.0     107453
Name: Severity, dtype: int64
```

In [ ]:

```
data_clean.isnull().sum()
```

Out[ ]:

```
Year                    0
Month                   0
Weekday                 0
Hour                    0
Severity                0
Num_vehicles            0
Collision_configuration 0
Road_configuration      0
Weather_condition       0
Road_surface            0
Road_alignment          0
Traffic_control         0
vehicle_id              0
Vehicle_type            0
Vehicle_year            0
Person_id               0
Person_sex              0
Person_age              0
Person_position         0
Person_injury_severity  0
Safety_device_used      0
Road_user_type          0
Collision_case          0
```

dtype: int64

In [ ]:

data_clean.shape

Out[ ]:

(6492680, 23)

Percentage of data dropped

In [ ]:

(1 - (6492680/6913204))*100

Out[ ]:

6.082910326384116

In [ ]:

data

Out[ ]:

	Year	Month	Weekday	Hour	Severity	Num_vehicles	Collision_configuration	Road_configuration	Weather_condition	Road_surface	...	Vehicle_type	Vehicle_year	Person_id	Person_sex	Person_age	Person_position	Person_injury_severity	Safety_device_used	Road_user_type	Collision_case
3	2000	1.0	1.0	17.0	2.0	1.0	50	0	5	2	...	50	2001.0	1	F	16.0	99	2	50	3	151441.0
4	2000	1.0	1.0	17.0	2.0	1.0	50	0	5	2	...	50	2001.0	2	F	16.0	99	2	50	3	151441.0
6	2000	1.0	1.0	10.0	2.0	1.0	2	0	4	3	...	50	2001.0	1	F	31.0	99	2	50	3	151460.0
8	2000	1.0	1.0	10.0	2.0	1.0	2	0	4	3	...	50	2001.0	1	M	61.0	99	2	50	3	151461.0

	Year	Month	Weekday	Hour	Severity	Num_vehicles	Collision_configuration	Road_configuration	Weather_condition	Road_surface	...	Vehicle_type	Vehicle_year	Person_id	Person_sex	Person_age	Person_position	Person_injury_severity	Safety_device_used	Road_user_type	Collision_case
10	2000	1.0	1.0	8.0	2.0	3.0	21	0	1	2	...	50	2000.0	1	N	33.0	11	1	50	0	151509.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6913198	2019	12.0	7.0	23.0	2.0	2.0	35	2	2	2	...	1	2016.0	1	F	39.0	11	1	2	1	2785585.0
6913199	2019	12.0	7.0	23.0	2.0	2.0	35	2	2	2	...	1	2016.0	2	M	38.0	12	2	2	2	2785585.0
6913200	2019	12.0	7.0	23.0	2.0	2.0	35	2	2	2	...	1	2011.0	1	M	30.0	11	2	2	1	2785585.0
6913202	2019	0.0	0.0	17.0	1.0	1.0	1	50	NaN	5	...	1	2007.0	1	M	50.0	11	1	1	1	2786255.0
6913203	2019	0.0	0.0	17.0	1.0	1.0	1	50	NaN	5	...	50	2001.0	1	F	76.0	99	3	50	3	2786255.0

6549929 rows × 23 columns

In [ ]:

```
data.to_csv('/content/drive/MyDrive/capstone.csv')
```

Univariate Plots : Histogram Plots of Person_age, Year, Month, Weekday, Hour

In [ ]:

```
plt.figure(figsize=(20, 22))
important_continous_values = ['Person_age', 'Year', 'Month', 'Weekday', 'Hour']


for i, column in enumerate(important_continous_values, 1):
    plt.subplot(len(important_continous_values), 1, i)
    data[column].hist()
    plt.title(f'Histogram of {column}')
    plt.xlabel(column)
    plt.ylabel('Frequency')

    plt.tight_layout()


plt.show()
```

Histogram of Person_age:

Shows age distribution of persons in incidents.

Majority aged 20-30 years.

Frequency decreases with increasing age.

Histogram of Year:

Incident frequency from 2000 to 2019.

Relatively stable with fluctuations.

Notable decrease after 2010.

Histogram of Month:

Incident distribution across months.

Significant peak might be indicating seasonal effects.

Maximum number of cases are in August, and then in Second

Histogram of Weekday:

Uniform incident frequency across the week.

Slight daily variations.

Histogram of Hour:

Non-uniform distribution across hours of the day.

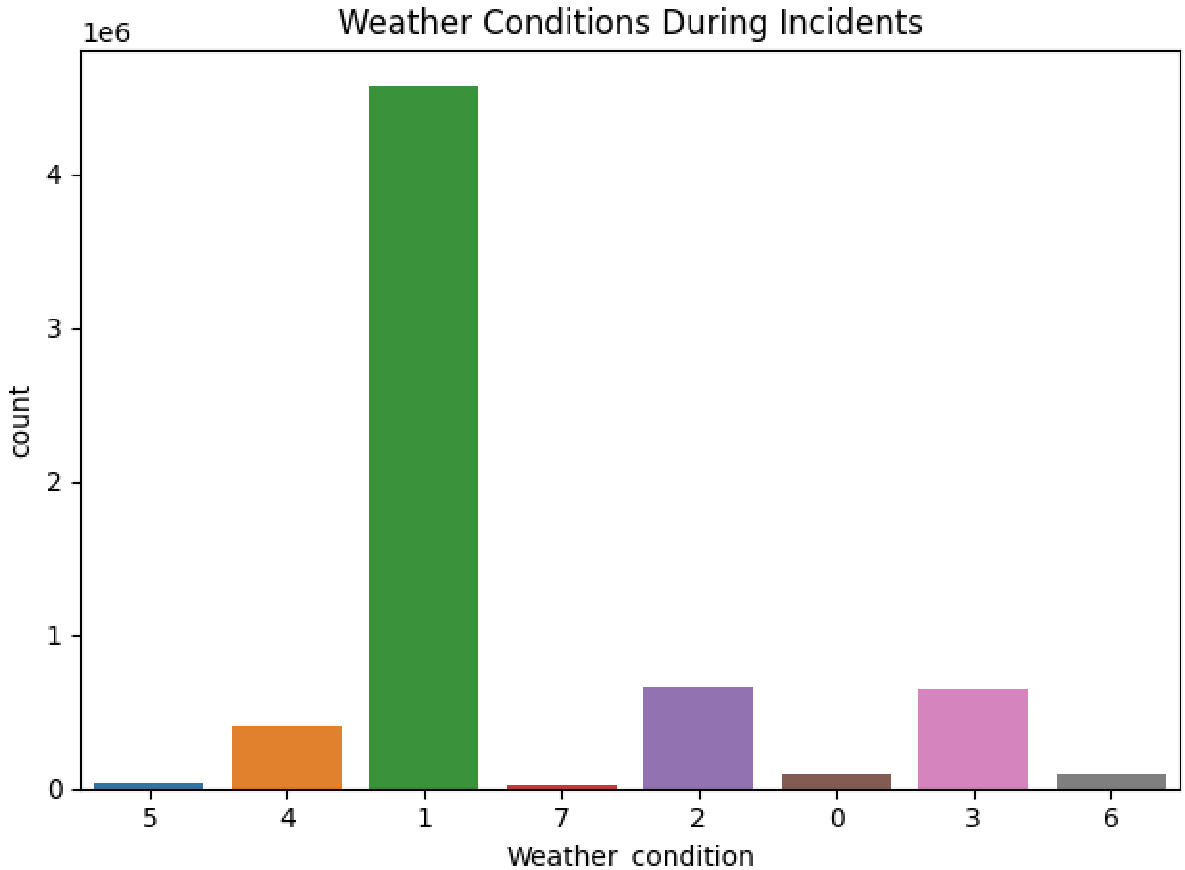
Peaks during certain hours, likely rush or late-night hours.

In [ ]:

```
#!pip install summarytools

#from summarytools import dfSummary

#dfSummary(data)
```

In [ ]:

```
#Checking the weather conditions

sns.countplot(x='Weather_condition', data=data)

plt.title('Weather Conditions During Incidents')


plt.tight_layout()

plt.show()
```



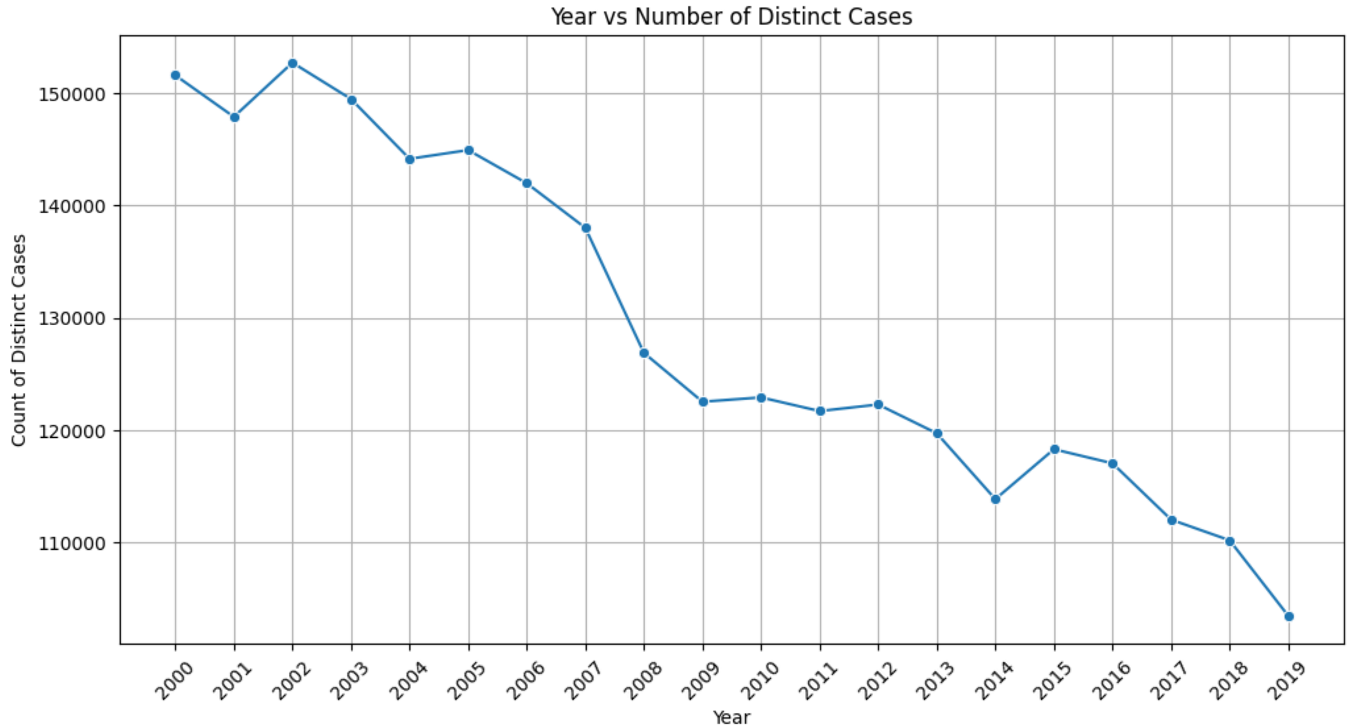
Most of the accidents have occured in clear and sunny weather,secondly in raining weather condition and thirdly in overcast, cloudy but no precipitation.

Bivariate analysis¶

Count of Cases over the Years

In [ ]:

```
yearly_case_count = data.groupby('Year')['Collision_case'].nunique().reset_index()

plt.figure(figsize=(12, 6))

sns.lineplot(x='Year', y='Collision_case', data=yearly_case_count, marker='o')

plt.title('Year vs Number of Distinct Cases')

plt.xlabel('Year')

plt.ylabel('Count of Distinct Cases')

plt.xticks(rotation=45)

plt.grid(True)

plt.show()
```

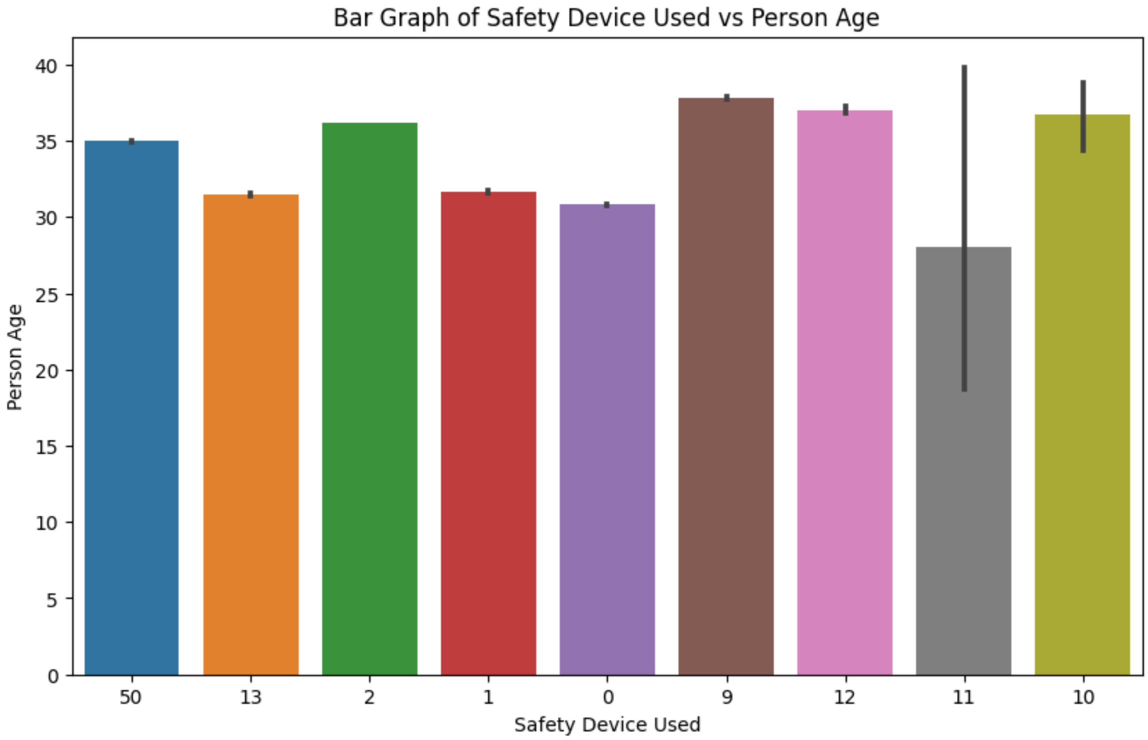


We can observe a gradual decrease in the number of accidents throughtout the years.

Bar Graph of Safety Device Used vs Passenger Age

In [ ]:

```
plt.figure(figsize=(10, 6))
sns.barplot(x='Safety_device_used', y='Person_age', data=data_clean)


plt.title('Bar Graph of Safety Device Used vs Person Age')
plt.xlabel('Safety Device Used')
plt.ylabel('Person Age')
plt.show()
```



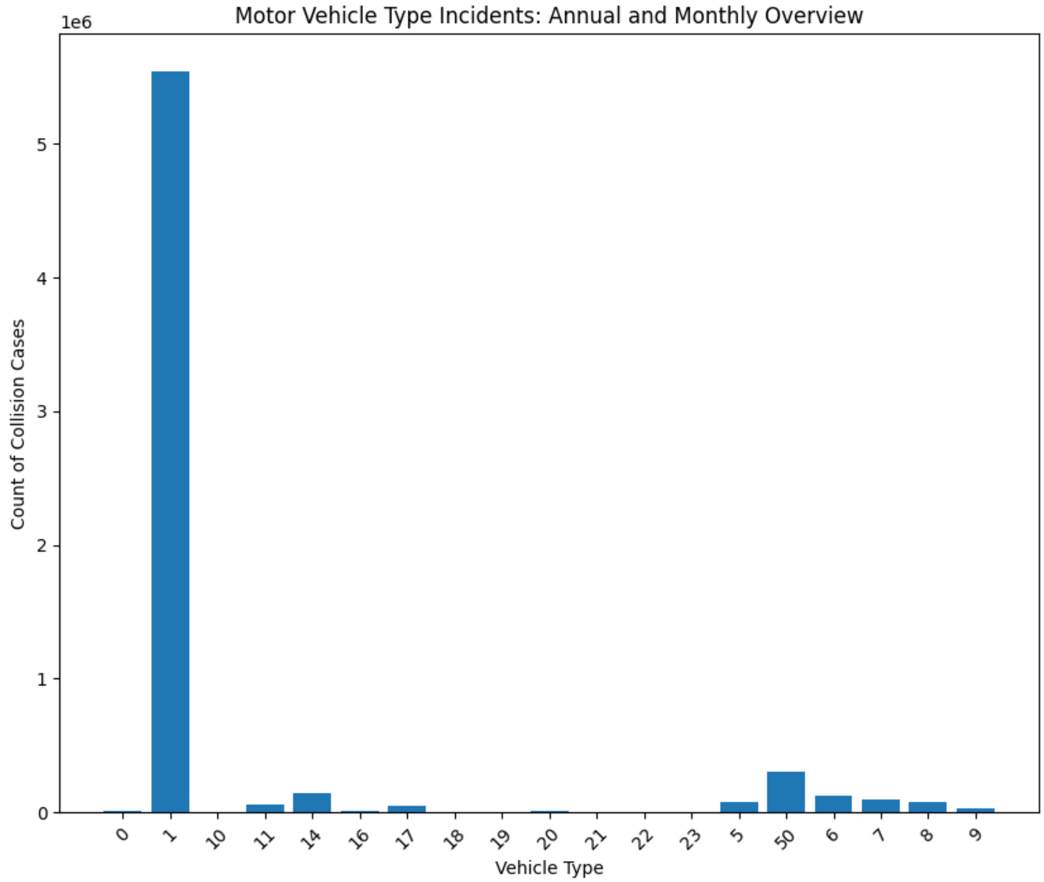
Reflective clothing and the combination of helmets and reflective clothing - are used by an older age group on average, suggesting that more experienced individuals might prioritize visibility and protection.

There is a consistent trend of people in their early to mid-30s not using safety devices or only using standard safety devices, indicating this age group may be less inclined to use specialized safety equipment.

3. Collision Case counts for various vehicle type

In [ ]:

```
data['Collision_case'] = pd.to_numeric(data['Collision_case'], errors='coerce')

data['Vehicle_type'] = data['Vehicle_type'].astype(str)

collision_counts = data.groupby('Vehicle_type')['Collision_case'].count()

plt.figure(figsize=(10, 8))

plt.bar(collision_counts.index, collision_counts.values)

plt.xlabel('Vehicle Type')

plt.ylabel('Count of Collision Cases')

plt.title('Motor Vehicle Type Incidents: Annual and Monthly Overview')

plt.xticks(rotation=45)

plt.show()
```



Light Duty Vehicles top collision stats; likely from being most common on roads.

Fewer mishaps with big rigs and buses hint at less road time or safer driving.

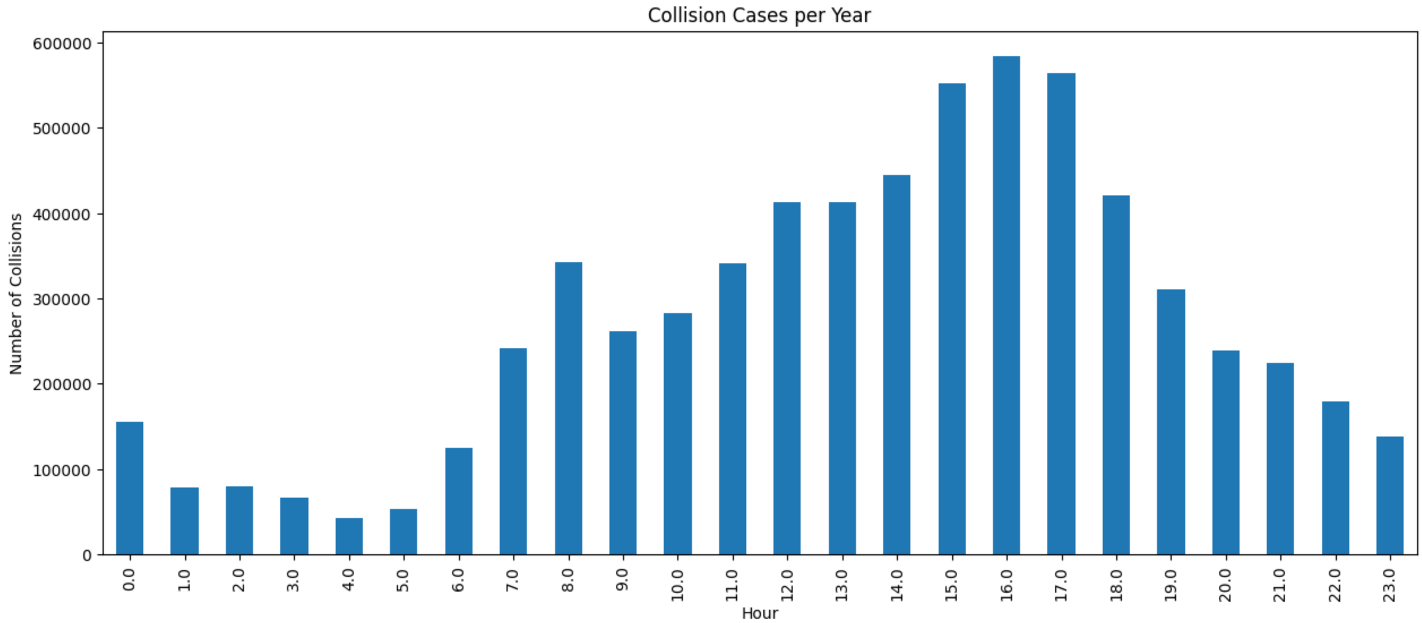
Bikes and motorbikes see moderate trouble, balancing numbers and risk.

Rare incidents with farm gear and fire engines point to scarce road use or stricter safety.

Number of collision for each hour

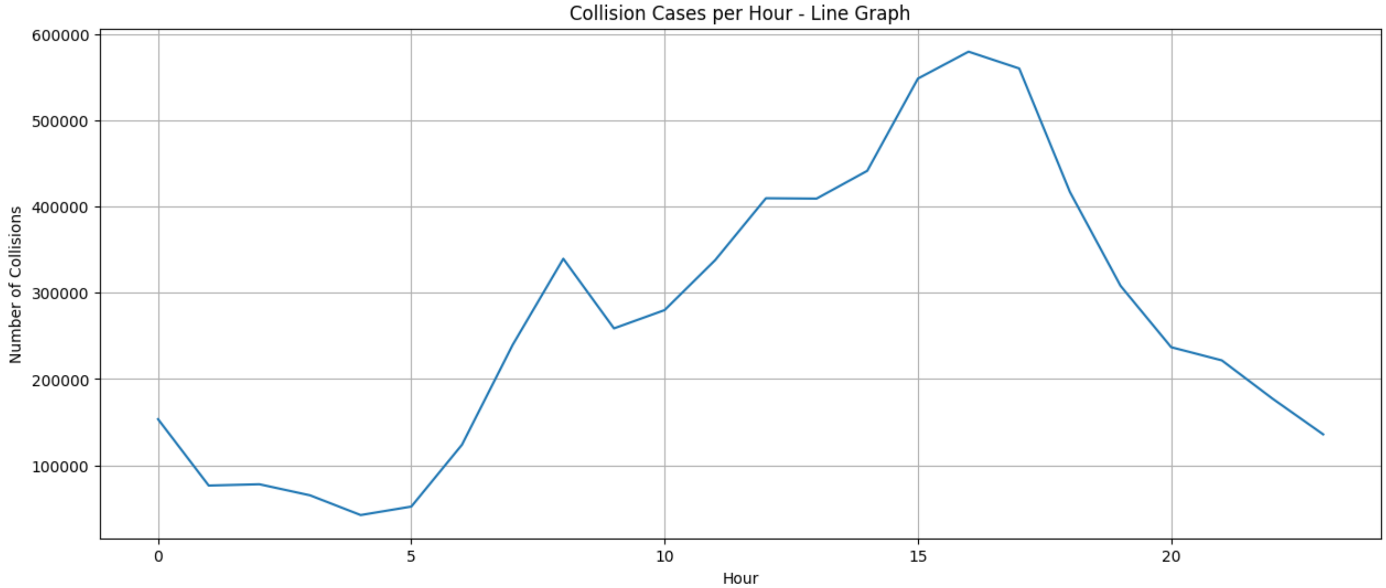
In [ ]:

```
Collision_counts = data.groupby('Hour').size()

plt.figure(figsize=(15,6))

Collision_counts.plot(kind='bar')

plt.title('Collision Cases per Year')

plt.xlabel('Hour')

plt.ylabel('Number of Collisions')

plt.show()
```



In [ ]:

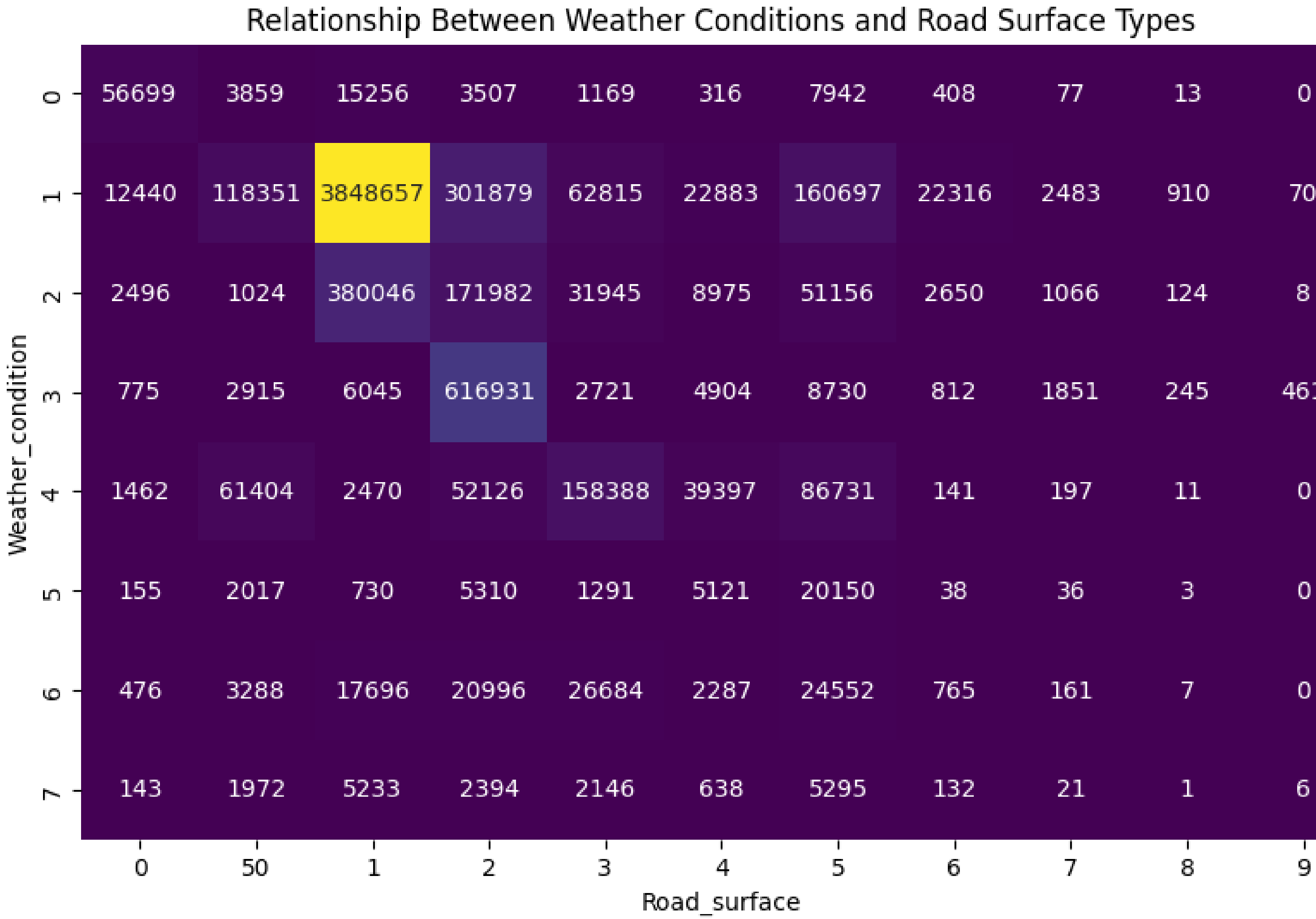
```
Collision_counts = data_clean.groupby('Hour').size()

plt.figure(figsize=(15, 6))

Collision_counts.plot(kind='line')

plt.title('Collision Cases per Hour - Line Graph')

plt.xlabel('Hour')

plt.ylabel('Number of Collisions')

plt.grid(True)

plt.show()
```



Heatmap to check the relationship between Weather Conditions and Road Surface Type

In [ ]:

```
weather_road_crosstab = pd.crosstab(data_clean['Weather_condition'], data_clean['Road_surface'])

plt.figure(figsize=(12, 6))

sns.heatmap(weather_road_crosstab, annot=True, fmt='d', cmap='viridis')

plt.title('Relationship Between Weather Conditions and Road Surface Types')

plt.show()
```



Checking the Average Number of vehicles involved in Collision Cases

In [ ]:

```
num_vehicles_collision_group = data_clean.groupby('Collision_case')['Num_vehicles'].mean().reset_index()

plt.figure(figsize=(12, 6))

sns.barplot(x='Collision_case', y='Num_vehicles', data=num_vehicles_collision_group)

plt.title('Average Number of Vehicles Involved by Collision Case')

plt.show()
```

8. REFERENCES

[1] "Government of Canada," [Online]. Available: https://tc.canada.ca/en/road-transportation/statistics-data/statistics-data-road-safety/2020-statistics-social-costs-collisions-canada.

[2] "statcan.gc.ca," [Online]. Available: https://www150.statcan.gc.ca/n1/daily-quotidien/221117/dq221117d-eng.htm.

[3] "open.canada.ca," [Online]. Available: https://open.canada.ca/data/en/dataset/1eb9eba7-71d1-4b30-9fb1-30cbdab7e63a.

[4] "www.mpi.mb.ca," [Online]. Available: https://www.mpi.mb.ca/Documents/TCSR2021.pdf.

[5] [Online]. Available: https://www.ricekendig.com/collision-risks-when-traveling-during-peak-hours/.

[6] [Online]. Available: https://www.bloomberg.com/news/articles/2022-07-01/why-canada-isn-t-having-a-traffic-safety-crisis.

[7] [Online]. Available: https://www.iihs.org/topics/fatality-statistics/detail/males-and-females.

[8] [Online]. Available: https://madd.ca/pages/programs/youth-services/statistics-links/.

[9] "Breaking the Bottlenecks," [Online]. Available: https://www.caa.ca/app/uploads/2021/01/Congestion-solutions-Summary-ENG-V2.pdf.