Project Name - Regulatory Affairs of Road Accident Data 2020 India _ ML _ FA _ DA projects (Part 1)

Project Type - Data Analysis

Industry - Unified Mentor

Contribution - Individual

Member Name - Hare Krishana Mishra

Task - 1

Project Summary -

Project Description:

The project "Regulatory Affairs of Road Accident Data 2020 India" focuses on the analysis of road accident statistics across 50 million-plus cities in India for the year 2020. The dataset contains vital information, including city names, accident cause categories, subcategories, outcomes, and counts of incidents.

Objective:

The main objective of this project is to analyze, interpret, and visualize accident data to identify patterns and risk factors, with the following goals:

- Understand the distribution of accidents across different cities and cause categories..
- Identify the most dangerous accident causes in terms of fatalities and injuries.
- Compare accident outcomes (e.g., fatalities, serious injuries) across cities and cause types.
- Provide actionable insights that can inform regulatory strategies, traffic control measures, and awareness programs for reducing accidents.

Key Project Details:

Dataset Source: Data.gov.in, covering road accidents in 50 million-plus cities of India (2020).

Data Fields:: Million Plus Cities, Cause Category, Cause Subcategory, Outcome of Incident, Count (number of incidents).

Tools Used: Python, Pandas, Matplotlib, Seaborn, SQL, Excel.

Steps Performed:

- Data loading and inspection.

- Handling missing values.

- Grouping and aggregation for analysis.

- Visualizations (bar plots, pie charts, stacked bars, heatmaps).

Insights Obtained:

- Identification of top accident-prone cities.
- Most common accident causes and subcategories.
- Relationship between causes and outcomes (e.g., fatalities vs. minor injuries).
- Severity patterns in different regions and under different road/weather conditions.

Let's Begin:-

Data Preparation

In [ ]:

```
import pandas as pd
# Load the dataset
df = pd.read_csv('/content/Regulatory Affairs of Road Accident Data 2020 India.csv')
```

In [ ]:

```
# Inspect the first few rows of the dataset
df.head()
```

Out[ ]:

	Million Plus Cities	Cause category	Cause Subcategory	Outcome of Incident	Count
0	Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0
1	Agra	Traffic Control	Flashing Signal/Blinker	Minor Injury	0.0
2	Agra	Traffic Control	Flashing Signal/Blinker	Persons Killed	0.0
3	Agra	Traffic Control	Flashing Signal/Blinker	Total Injured	0.0
4	Agra	Traffic Control	Flashing Signal/Blinker	Total number of Accidents	0.0

In [ ]:

```
# Check for missing values
print(df.isnull().sum())
```

```
Million Plus Cities    0
Cause category         0
Cause Subcategory      0
Outcome of Incident    0
Count                  3
dtype: int64
```

Data Cleaning

In [ ]:

```
# Drop rows with missing values if any
df_cleaned = df.dropna()
```

In [ ]:

```
# Verify the cleaning process
print(df_cleaned.isnull().sum())
```

```
Million Plus Cities    0
Cause category         0
Cause Subcategory      0
Outcome of Incident    0
Count                  0
dtype: int64
```

Exploratory Data Analysis (EDA)

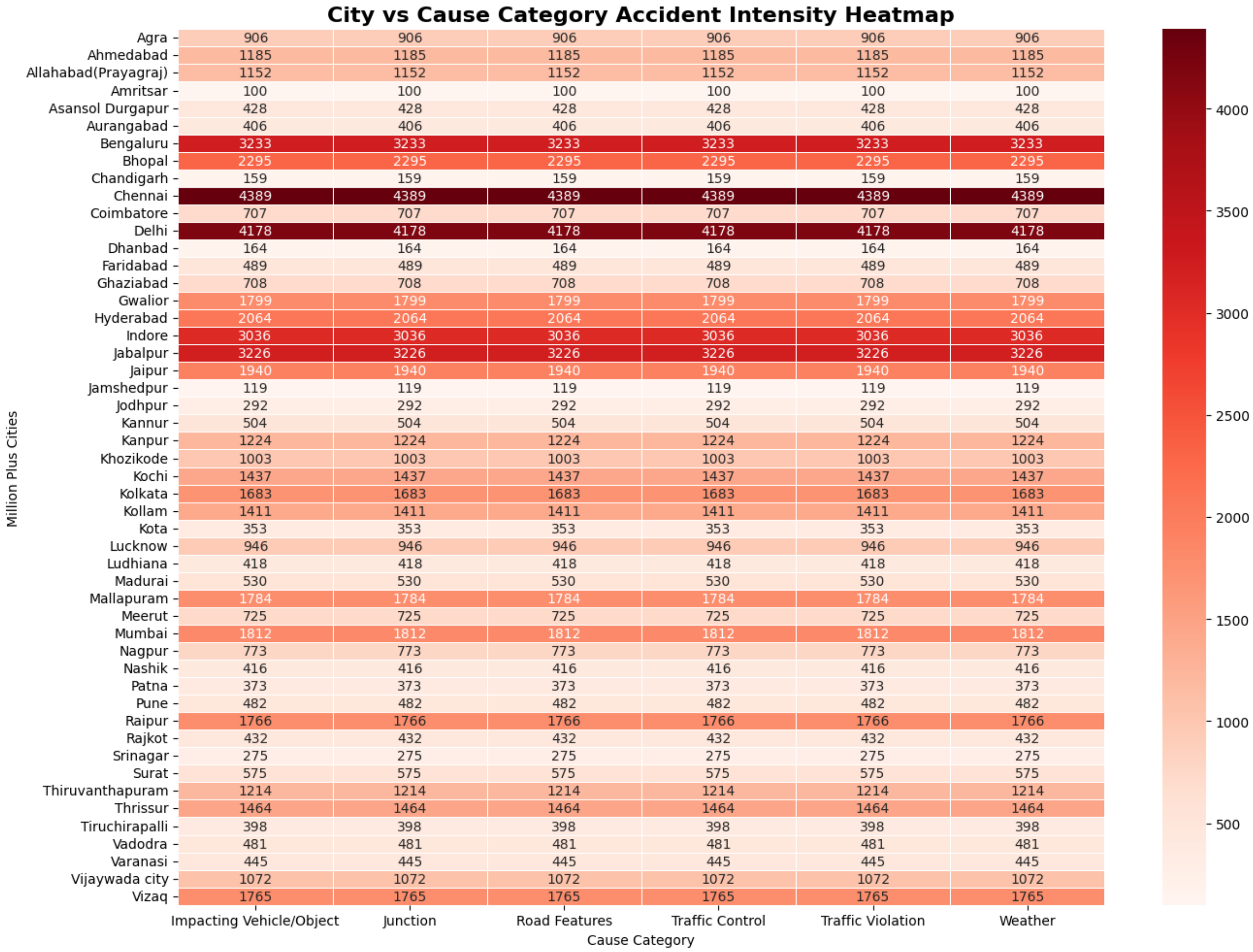
In [ ]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

City vs Cause Category Accident Intensity Heatmap

In [ ]:

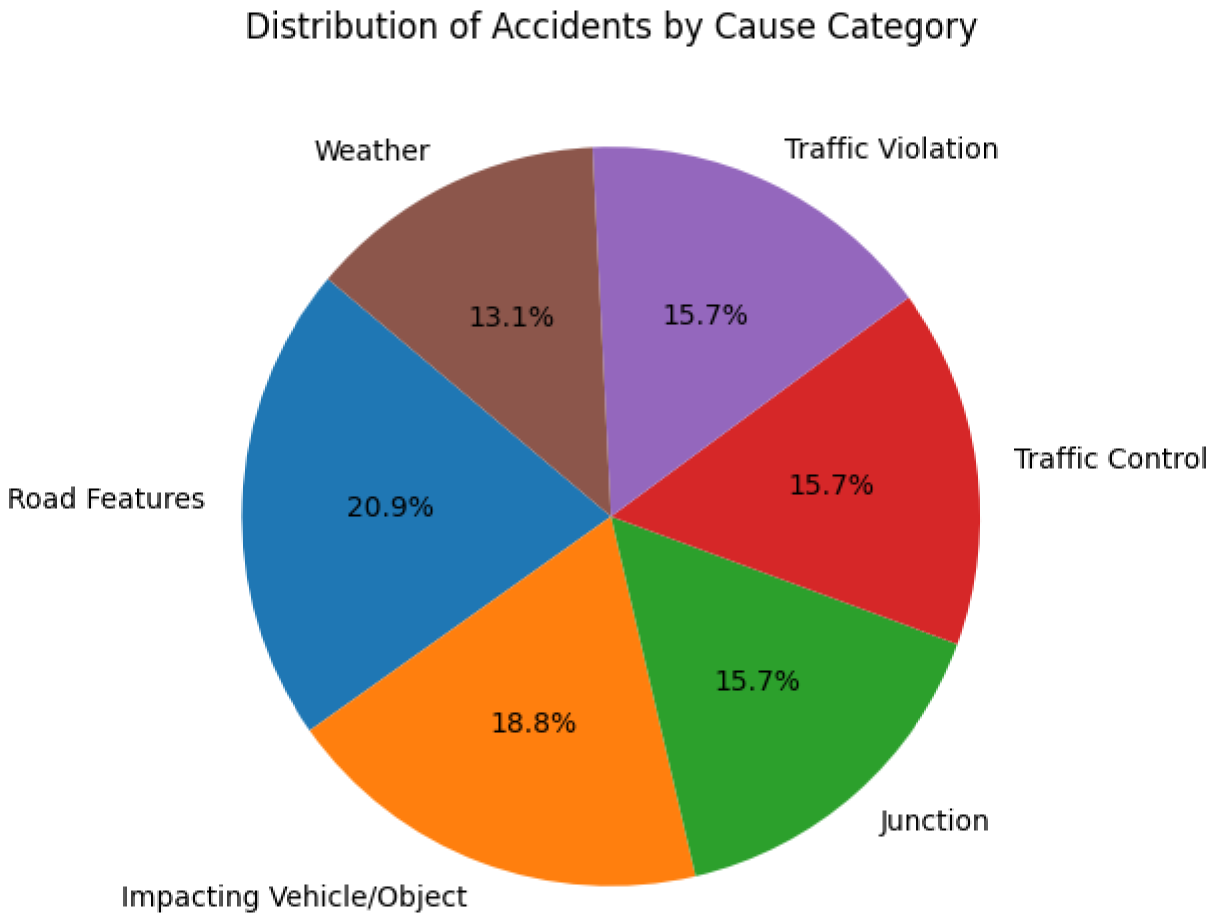
```
# Keep only total number of accidents for intensity analysis
df_total_accidents = df[df['Outcome of Incident'] == 'Total number of Accidents']

# Pivot table for heatmap
heatmap_data = df_total_accidents.pivot_table(
    index='Million Plus Cities',
    columns='Cause category',
    values='Count',
    aggfunc='sum',
    fill_value=0
)

# Plot heatmap
plt.figure(figsize=(14, 10))
sns.heatmap(
    heatmap_data,
    cmap="Reds",
    annot=True,
    fmt=".0f",
    linewidths=0.5
)
plt.title("City vs Cause Category Accident Intensity Heatmap", fontsize=16, fontweight='
plt.xlabel("Cause Category")
plt.ylabel("Million Plus Cities")
plt.tight_layout()
plt.show()
```



Pie chart of Cause category distribution

In [ ]:

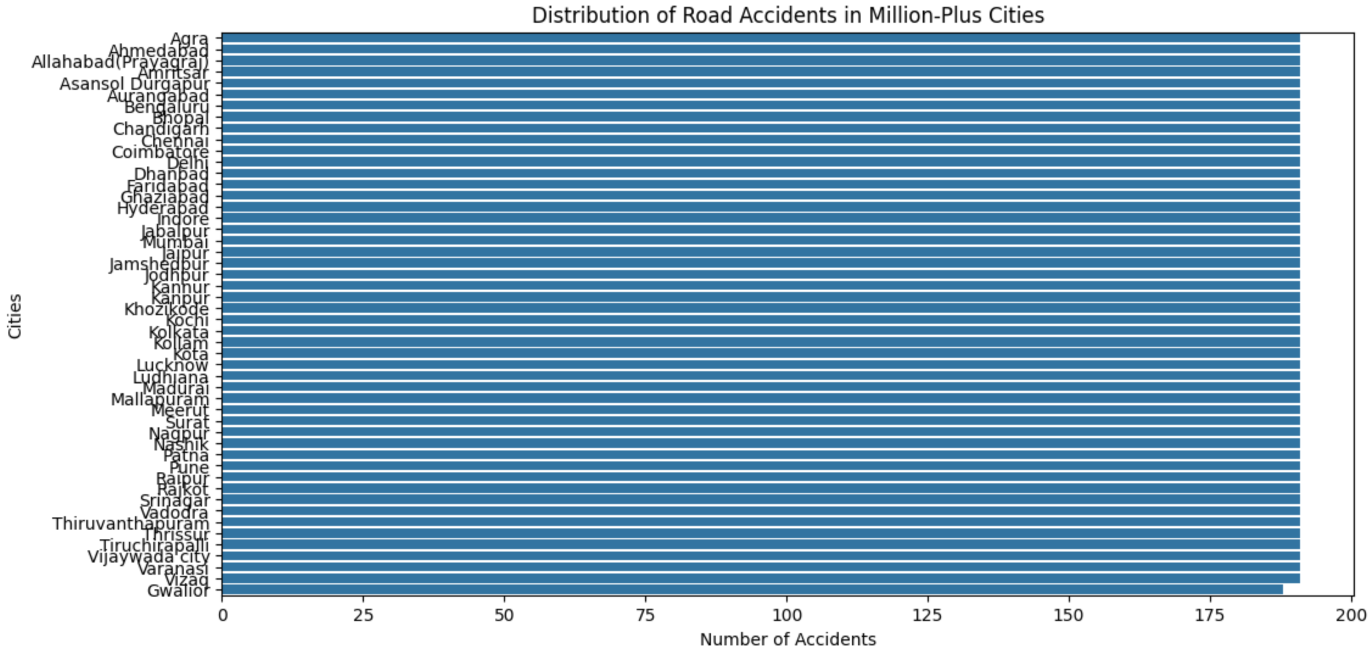
```
cause_counts = df['Cause category'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(cause_counts, labels=cause_counts.index, autopct='%1.1f%%', startangle=140)
plt.title("Distribution of Accidents by Cause Category")
plt.show()
```



Distribution of Road Accidents in Million-Plus Cities

```
In [ ]:
```

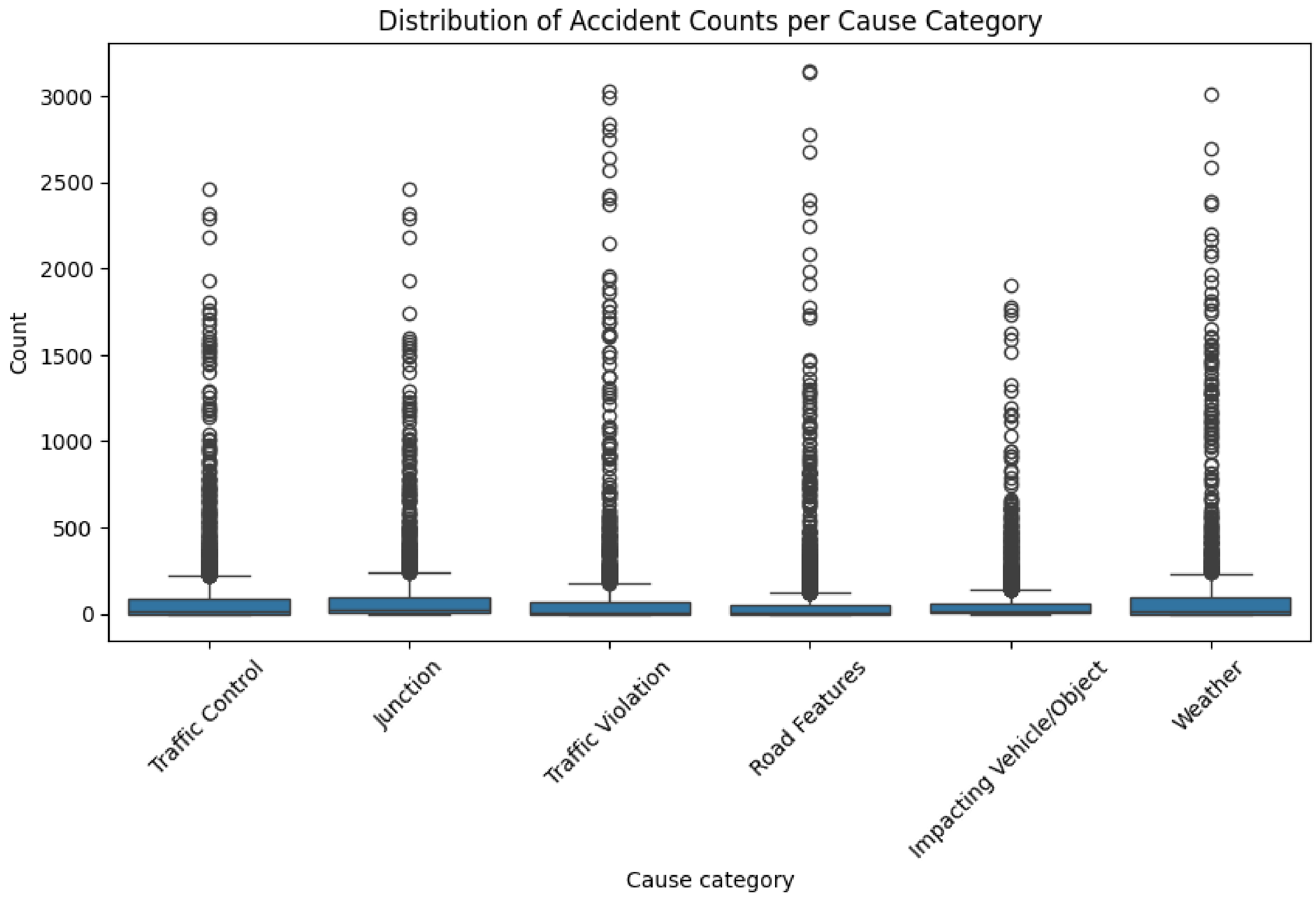
```
# Plot the distribution of accidents by city
plt.figure(figsize=(12,6))
sns.countplot(y='Million Plus Cities', data=df_cleaned,
order=df_cleaned['Million Plus Cities'].value_counts().index)
plt.title('Distribution of Road Accidents in Million-Plus Cities')
plt.xlabel('Number of Accidents')
plt.ylabel('Cities')
plt.show()
```



Spread of Accident Counts by Cause Category

```
In [ ]:
```

```python
plt.figure(figsize=(10, 5))
sns.boxplot(x="Cause category", y="Count", data=df)
plt.xticks(rotation=45)
plt.title("Distribution of Accident Counts per Cause Category")
plt.show()
```

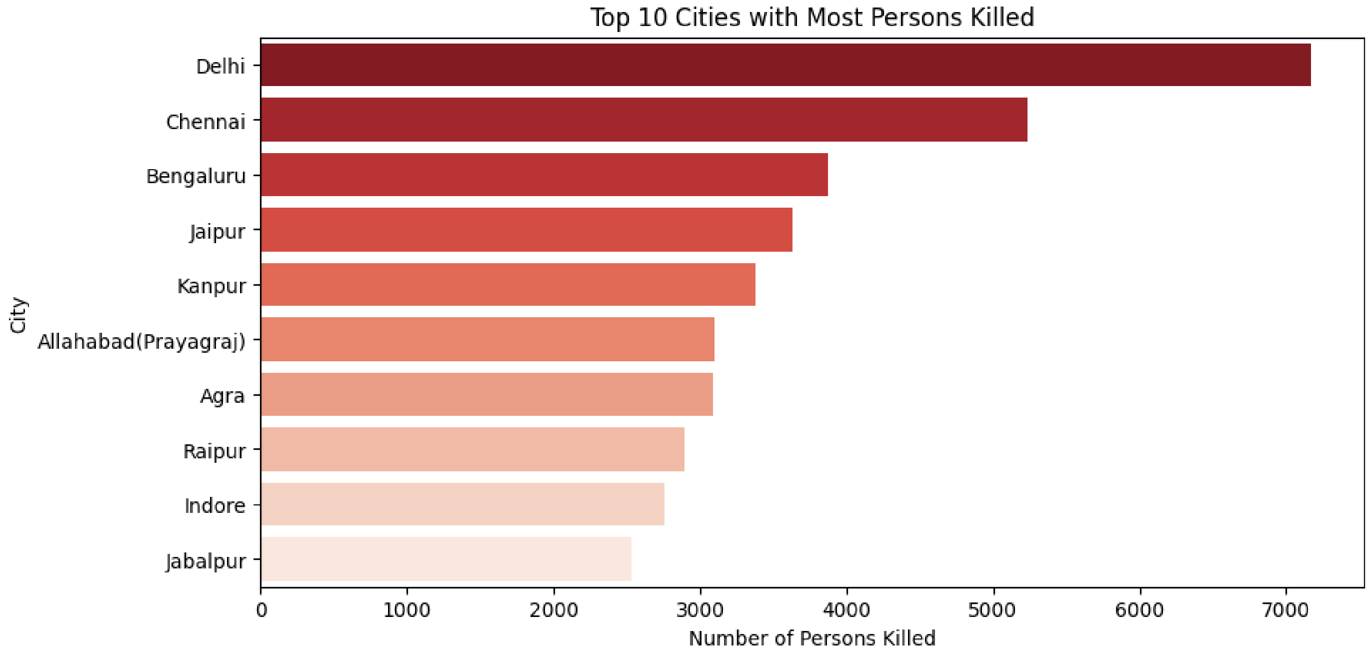


Top 10 cities with most persons killed

In [ ]:

```
persons_killed = df[df['Outcome of Incident'] == 'Persons Killed'] \
    .groupby('Million Plus Cities')['Count'].sum() \
    .sort_values(ascending=False).head(10)

plt.figure(figsize=(10, 5))
sns.barplot(x=persons_killed.values, y=persons_killed.index, palette="Reds_r")
plt.title("Top 10 Cities with Most Persons Killed")
plt.xlabel("Number of Persons Killed")
plt.ylabel("City")
plt.show()
```

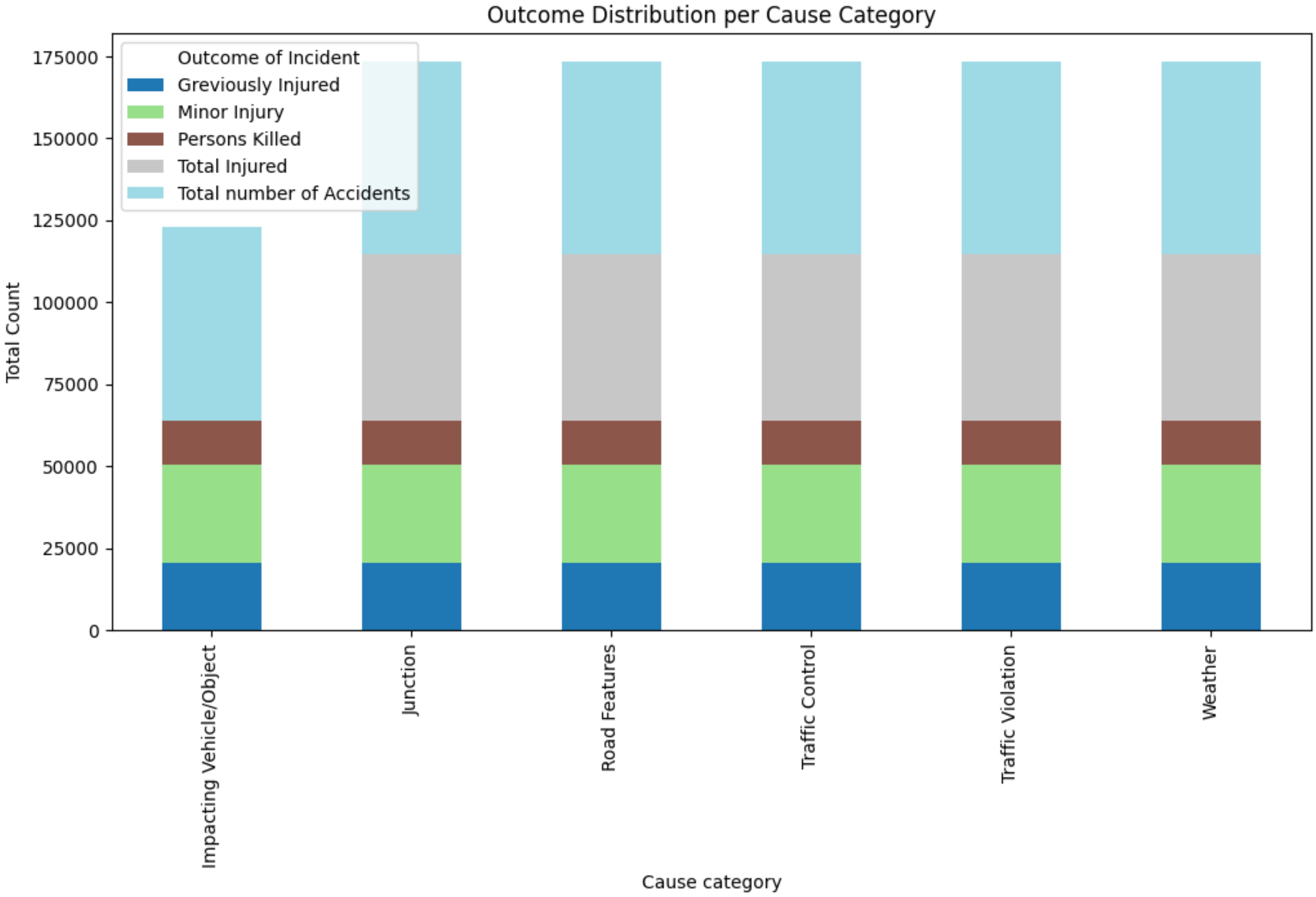
```
/tmp/ipython-input-131148110.py:6: FutureWarning:

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

  sns.barplot(x=persons_killed.values, y=persons_killed.index, palette="Reds_r")
```



Stacked bar of Outcome distribution per Cause category

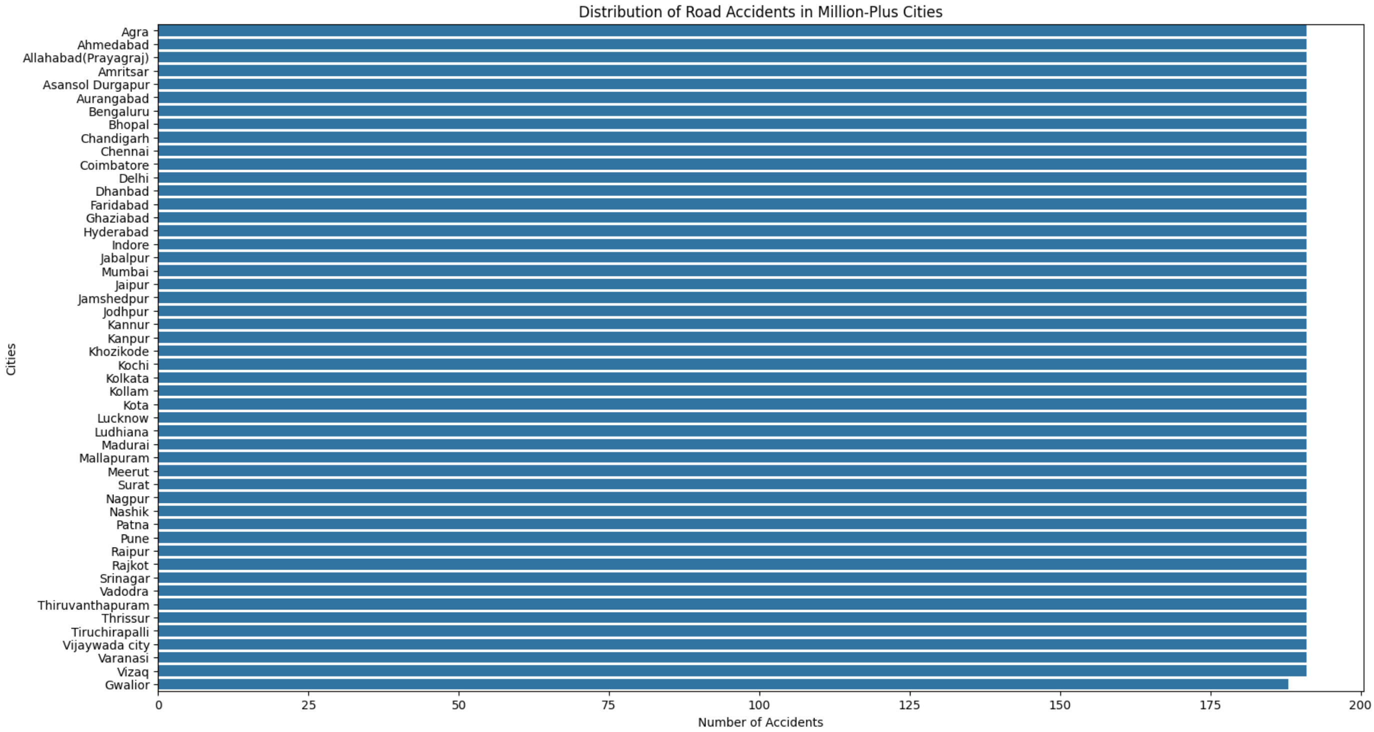
In [ ]:

```
outcome_vs_cause = df.groupby(['Cause category', 'Outcome of Incident'])['Count'].sum().
outcome_vs_cause.plot(kind='bar', stacked=True, figsize=(12, 6), colormap="tab20")
plt.title("Outcome Distribution per Cause Category")
plt.ylabel("Total Count")
plt.show()
```



Distribution of Accidents Across Cities

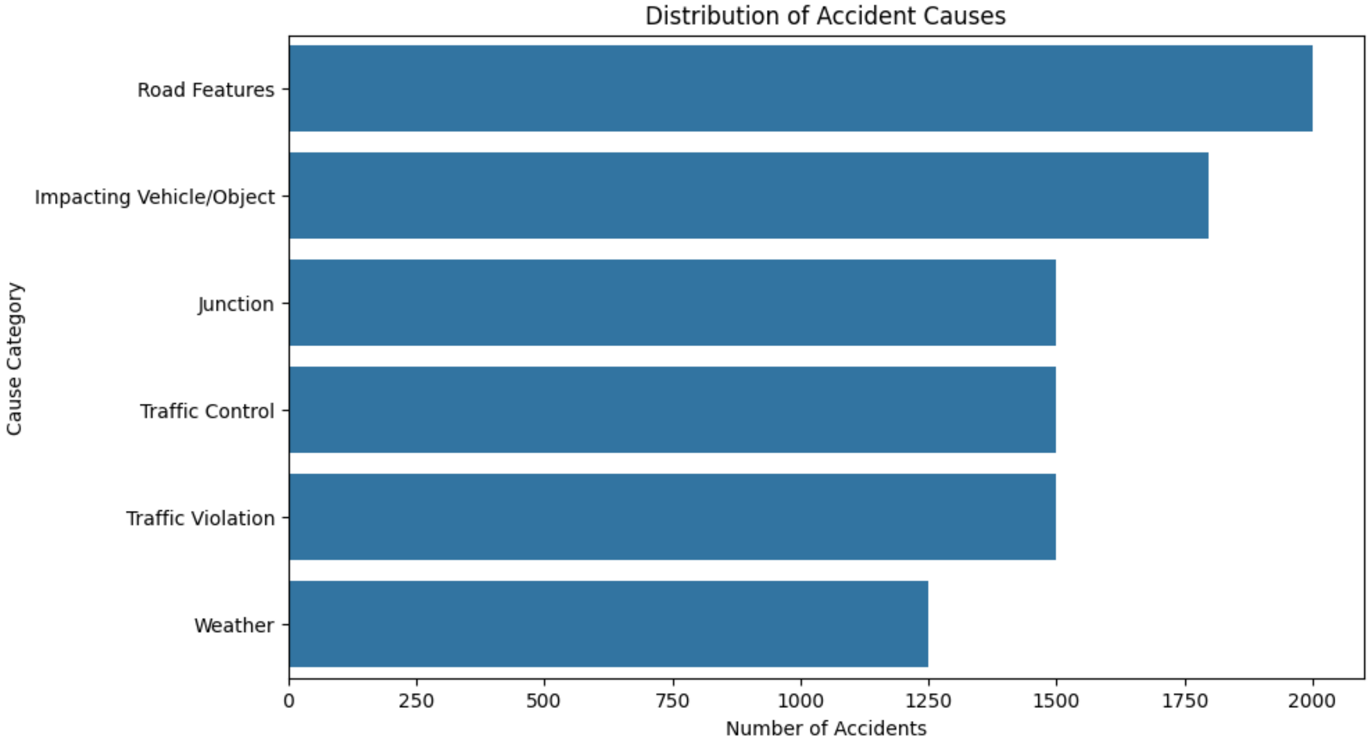
```
In [ ]:
import matplotlib.pyplot as plt
import seaborn as sns
# Plot the distribution of accidents by city
plt.figure(figsize=(18,10))
sns.countplot(y='Million Plus Cities', data=df_cleaned,
order=df_cleaned['Million Plus Cities'].value_counts().index)
plt.title('Distribution of Road Accidents in Million-Plus Cities')
plt.xlabel('Number of Accidents')
plt.ylabel('Cities')
plt.show()
```



Analysis of Accident Causes

In [ ]:

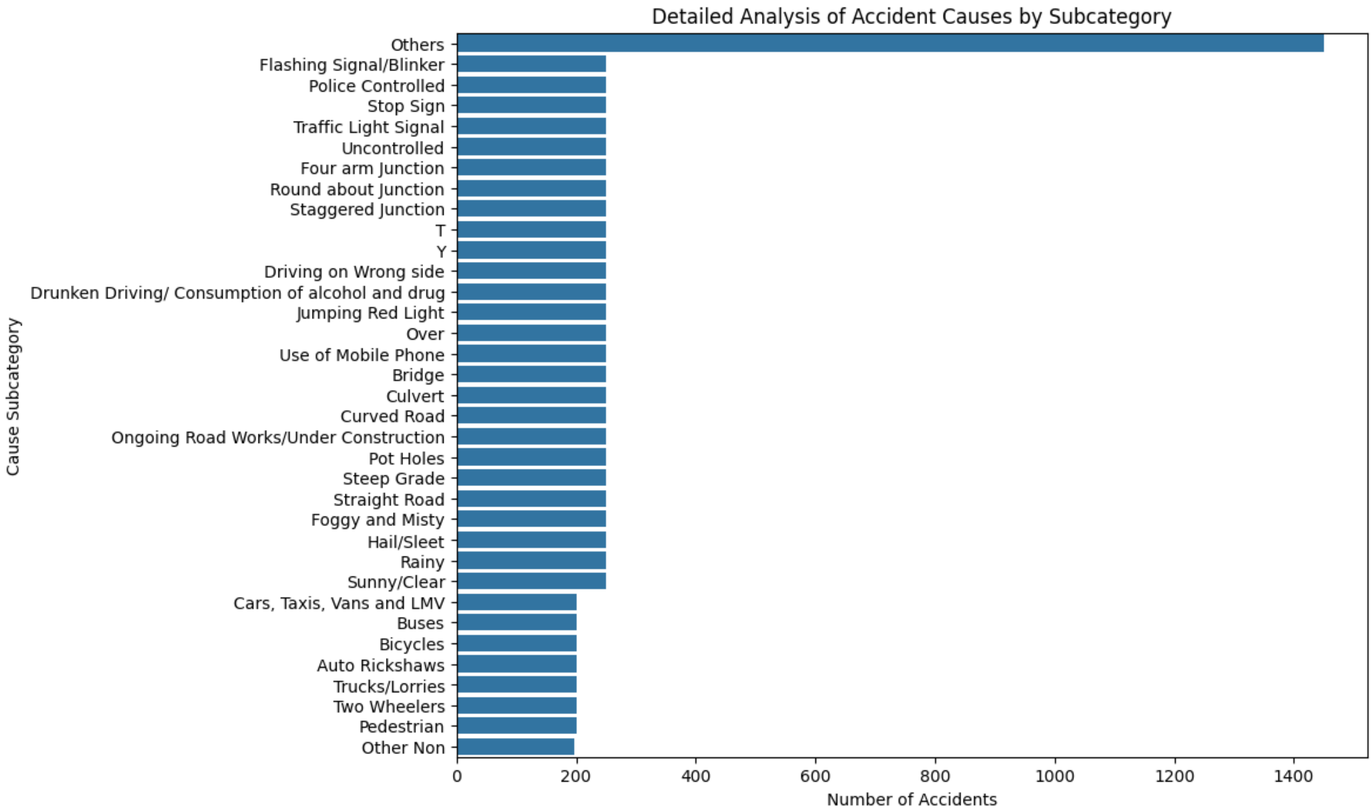
```
# Plot the distribution of accidents by cause category
plt.figure(figsize=(10,6))
sns.countplot(y='Cause category', data=df_cleaned,
order=df_cleaned['Cause category'].value_counts().index)
plt.title('Distribution of Accident Causes')
plt.xlabel('Number of Accidents')
plt.ylabel('Cause Category')
plt.show()
```



Detailed analysis by cause subcategory

In [ ]:

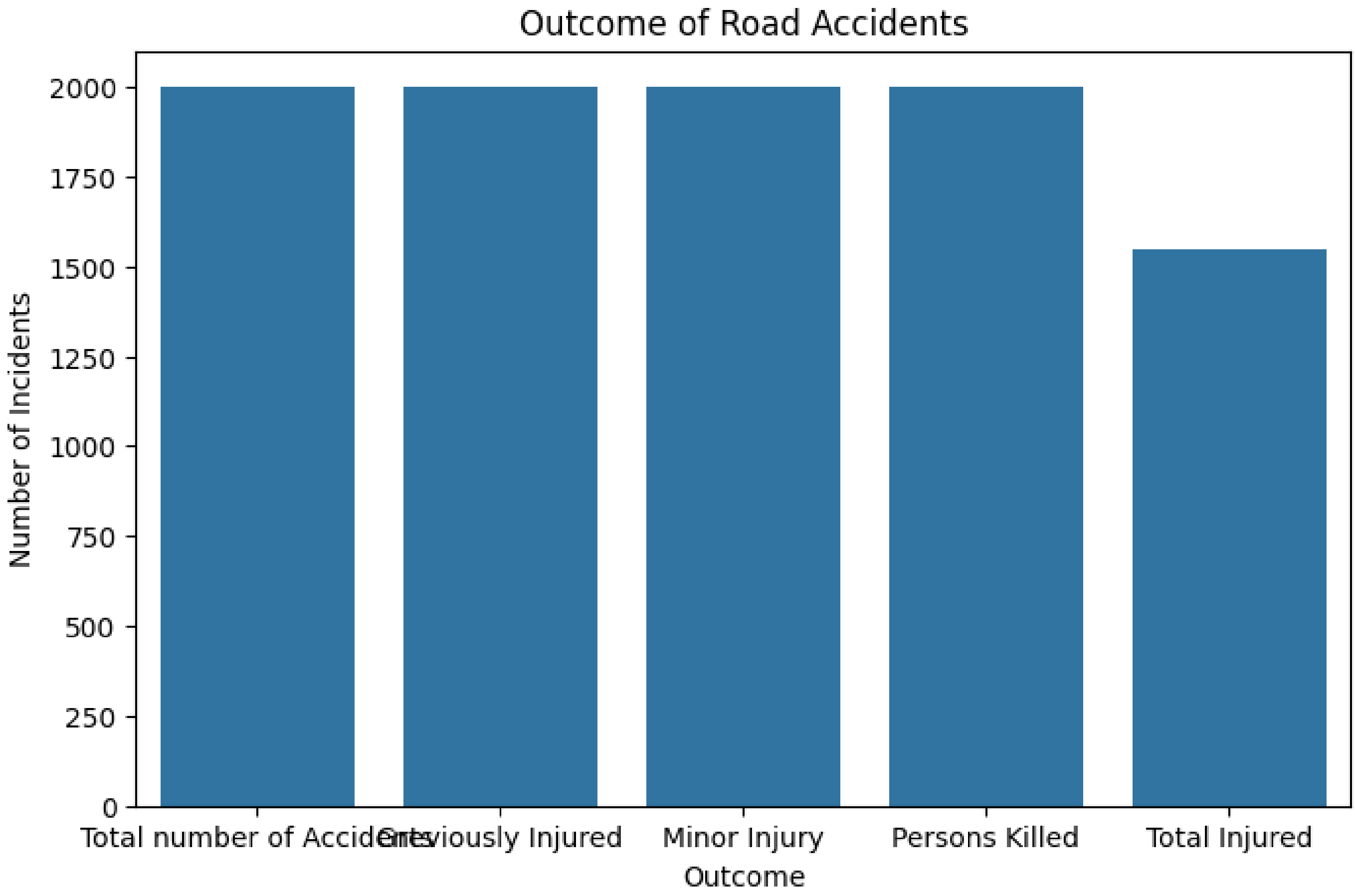
```
plt.figure(figsize=(10,8))
sns.countplot(y='Cause Subcategory', data=df_cleaned,
order=df_cleaned['Cause Subcategory'].value_counts().index)
plt.title('Detailed Analysis of Accident Causes by Subcategory')
plt.xlabel('Number of Accidents')
plt.ylabel('Cause Subcategory')
plt.show()
```



Outcomes of Incidents

In [ ]:

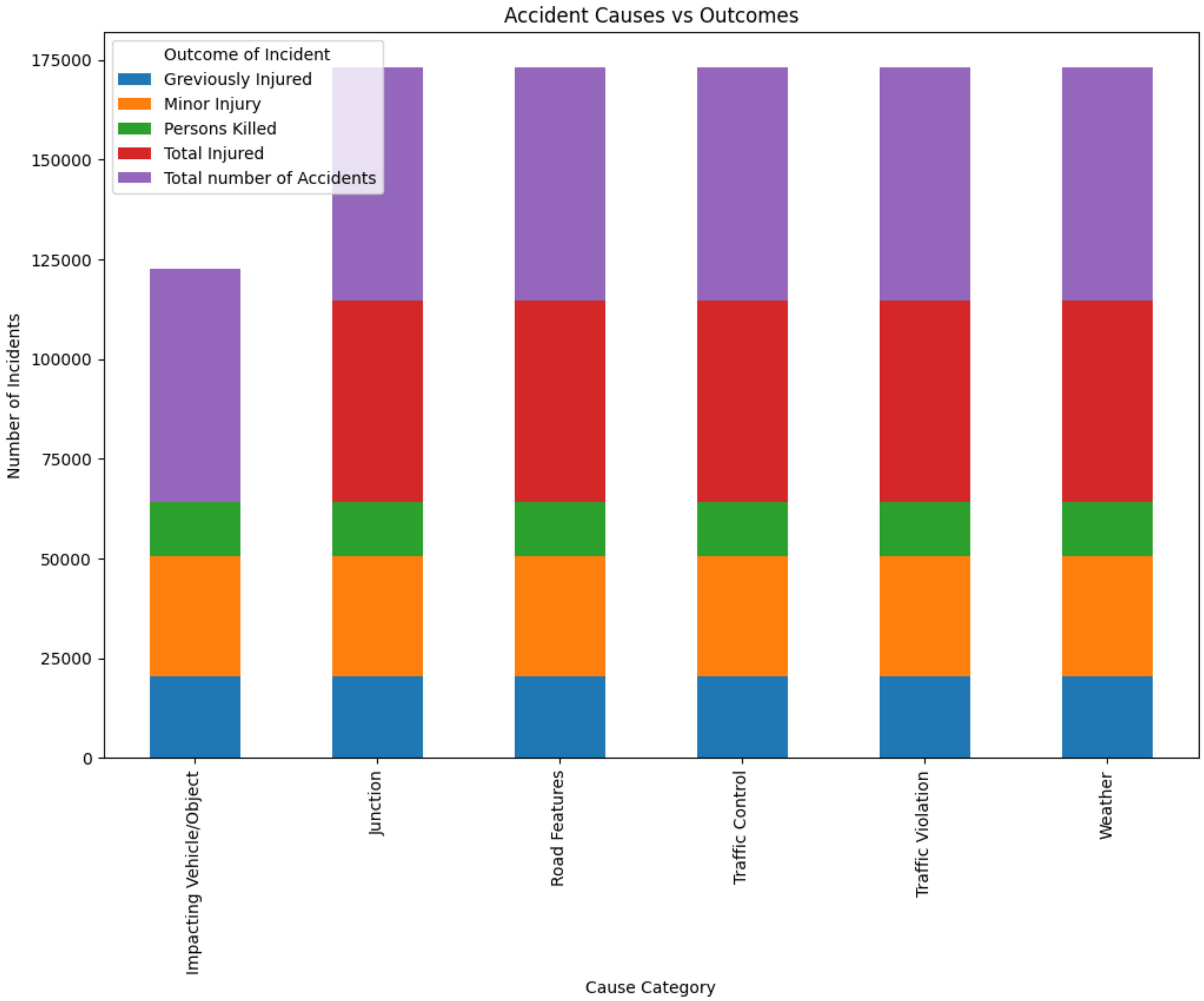
```
# Plot the outcomes of incidents
plt.figure(figsize=(8,5))
sns.countplot(x='Outcome of Incident', data=df_cleaned,
order=df_cleaned['Outcome of Incident'].value_counts().index)
plt.title('Outcome of Road Accidents')
plt.xlabel('Outcome')
plt.ylabel('Number of Incidents')
plt.show()
```



Analyzing Accident Causes vs Outcomes

```
In [ ]:
```

```
# Grouping by cause category and outcome
outcome_vs_cause = df_cleaned.groupby(['Cause category',
'Outcome of Incident'])['Count'].sum().unstack()
# Plot the result
outcome_vs_cause.plot(kind='bar', stacked=True, figsize=(12,8))
plt.title('Accident Causes vs Outcomes')
plt.xlabel('Cause Category')
plt.ylabel('Number of Incidents')
plt.show()
```


Accident Causes vs Outcomes
Outcome of Incident
Greviously Injured
Minor Injury
Persons Killed
Total Injured
Total number of Accidents
Number of Incidents
175000
150000
125000
100000
75000
50000
25000
0
Impacting Vehicle/Object
Junction
Road Features
Traffic Control
Traffic Violation
Weather
Cause Category

Project Name - Regulatory Affairs of Road Accident Data 2020 India _ ML _ FA _ DA projects (Part 2)

Project Type - Data Analysis

Industry - Unified Mentor

Contribution - Individual

Member Name - Hare Krishana Mishra

Task - 2

Project Summary -

Project Description:

This project focuses on analyzing road accident data from 50 million-plus cities in India for the year 2020. The dataset contains details about the type and cause of accidents, their outcomes, and the number of incidents. Using Python, SQL, and Excel, the analysis identifies the most common accident causes, evaluates how these vary across cities, and studies the relationship between causes and accident outcomes such as injuries, fatalities, and total incidents. The project uses exploratory data analysis (EDA) techniques and multiple visualizations to reveal key patterns that can help policymakers and urban planners improve road safety.

Objective:

The main objective of this project is to analyze, interpret, and visualize accident data to identify patterns and risk factors, with the following goals:

- Examine the distribution of road accidents across Indian million-plus cities.
- Identify primary and subcategories of accident causes and their frequency.
- Compare accident outcomes (injuries, fatalities, total accidents) across different cause categories.
- Visualize data to highlight high-risk cities and accident causes

Key Project Details:

Dataset Source: Data.gov.in, covering road accidents in 50 million-plus cities of India (2020).

Number of Records: 9,550

Columns:

- Million Plus Cities – Name of the city

- Cause Category – Broad classification (Traffic Control, Junction, Road Features, Impacting Vehicle/Object, Weather, etc.)

- Cause Subcategory – Specific cause (e.g., Drunken Driving, Pot Holes, Over Speeding, Rainy Weather)
- Outcome of Incident – Result (Grievously Injured, Minor Injury, Persons Killed, Total Accidents, Total Injured)
- Count – Number of incidents for that cause-outcome combination

Tools Used: Python, Pandas, Matplotlib, Seaborn, SQL, Excel.

Key Analysis Performed:

- Accident distribution by city
- Outcome distribution (injuries, deaths, accidents)
- Identification of top accident-prone cities and causes

Let's Begin:-

In [ ]:

```
import pandas as pd
import numpy as np
```

In [ ]:

```
df=pd.read_csv("/content/Regulatory Affairs of Road Accident Data 2020 India.csv")
df
```

Out[ ]:

	Million Plus Cities	Cause category	Cause Subcategory	Outcome of Incident	Count
0	Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0
1	Agra	Traffic Control	Flashing Signal/Blinker	Minor Injury	0.0
2	Agra	Traffic Control	Flashing Signal/Blinker	Persons Killed	0.0
3	Agra	Traffic Control	Flashing Signal/Blinker	Total Injured	0.0
4	Agra	Traffic Control	Flashing Signal/Blinker	Total number of Accidents	0.0
...	...	...	...	...	...
9545	Vizaq	Weather	Sunny/Clear	Greviously Injured	561.0
9546	Vizaq	Weather	Sunny/Clear	Minor Injury	252.0
9547	Vizaq	Weather	Sunny/Clear	Persons Killed	176.0
9548	Vizaq	Weather	Sunny/Clear	Total number of Accidents	1207.0
9549	Vizaq	Weather	Sunny/Clear	Total Injured	813.0

9550 rows × 5 columns

Data Preparation

In [ ]:

```
df.shape  #(rows, columns)
```

Out[ ]:

```
(9550, 5)
```

In [ ]:

```
df.size #9550 rows × 5 columns = 47750
```

Out[ ]:

```
47750
```

In [ ]:

```
df.head()
```

Out[ ]:

	Million Plus Cities	Cause category	Cause Subcategory	Outcome of Incident	Count
0	Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0
1	Agra	Traffic Control	Flashing Signal/Blinker	Minor Injury	0.0
2	Agra	Traffic Control	Flashing Signal/Blinker	Persons Killed	0.0
3	Agra	Traffic Control	Flashing Signal/Blinker	Total Injured	0.0
4	Agra	Traffic Control	Flashing Signal/Blinker	Total number of Accidents	0.0

In [ ]:

```
df.tail()
```

Out[ ]:

	Million Plus Cities	Cause category	Cause Subcategory	Outcome of Incident	Count
9545	Vizaq	Weather	Sunny/Clear	Greviously Injured	561.0
9546	Vizaq	Weather	Sunny/Clear	Minor Injury	252.0
9547	Vizaq	Weather	Sunny/Clear	Persons Killed	176.0
9548	Vizaq	Weather	Sunny/Clear	Total number of Accidents	1207.0
9549	Vizaq	Weather	Sunny/Clear	Total Injured	813.0

In [ ]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9550 entries, 0 to 9549
Data columns (total 5 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Million Plus Cities  9550 non-null   object 
 1   Cause category       9550 non-null   object 
 2   Cause Subcategory    9550 non-null   object 
 3   Outcome of Incident  9550 non-null   object 
 4   Count                9547 non-null   float64
```

```
dtypes: float64(1), object(4)
memory usage: 373.2+ KB
```

In [ ]:

```
df.columns
```

Out[ ]:

```
Index(['Million Plus Cities', 'Cause category', 'Cause Subcategory',
       'Outcome of Incident', 'Count'],
      dtype='object')
```

In [ ]:

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

Out[ ]:

	0
Million Plus Cities	0
Cause category	0
Cause Subcategory	0
Outcome of Incident	0
Count	3

dtype: int64

In [ ]:

```
df.fillna(0)
```

Out[ ]:

	Million Plus Cities	Cause category	Cause Subcategory	Outcome of Incident	Count
0	Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0
1	Agra	Traffic Control	Flashing Signal/Blinker	Minor Injury	0.0
2	Agra	Traffic Control	Flashing Signal/Blinker	Persons Killed	0.0
3	Agra	Traffic Control	Flashing Signal/Blinker	Total Injured	0.0
4	Agra	Traffic Control	Flashing Signal/Blinker	Total number of Accidents	0.0
...	...	...	...	...	...
9545	Vizaq	Weather	Sunny/Clear	Greviously Injured	561.0
9546	Vizaq	Weather	Sunny/Clear	Minor Injury	252.0
9547	Vizaq	Weather	Sunny/Clear	Persons Killed	176.0
9548	Vizaq	Weather	Sunny/Clear	Total number of Accidents	1207.0
9549	Vizaq	Weather	Sunny/Clear	Total Injured	813.0

9550 rows × 5 columns

In [ ]:

```
df["Million Plus Cities"].value_counts()
```

Out[ ]:

	count
Million Plus Cities	
Agra	191
Ahmedabad	191
Allahabad(Prayagraj)	191
Amritsar	191
Asansol Durgapur	191
Aurangabad	191
Bengaluru	191
Bhopal	191
Chandigarh	191
Chennai	191
Coimbatore	191
Delhi	191
Dhanbad	191
Faridabad	191
Ghaziabad	191
Gwalior	191
Hyderabad	191
Indore	191
Jabalpur	191
Jaipur	191
Jamshedpur	191
Jodhpur	191
Kannur	191
Kanpur	191
Khozikode	191
Kochi	191
Kolkata	191
Kollam	191
Kota	191
Lucknow	191
Ludhiana	191
Madurai	191
Mallapuram	191

	count
Million Plus Cities	
Meerut	191
Mumbai	191
Nagpur	191
Nashik	191
Patna	191
Pune	191
Raipur	191
Rajkot	191
Srinagar	191
Surat	191
Thiruvanthapuram	191
Thrissur	191
Tiruchirapalli	191
Vadodra	191
Varanasi	191
Vijaywada city	191
Vizaq	191

dtype: int64

In [ ]:

```
df["Cause category"].value_counts()
```

Out[ ]:

	count
Cause category	
Road Features	2000
Impacting Vehicle/Object	1800
Junction	1500
Traffic Control	1500
Traffic Violation	1500
Weather	1250

dtype: int64

In [ ]:

```
df["Outcome of Incident"].value_counts()
```

Out[ ]:

	count
Outcome of Incident	
Greviously Injured	2000
Minor Injury	2000
Persons Killed	2000
Total number of Accidents	2000
Total Injured	1550

dtype: int64

In [ ]:

```
df=pd.read_csv("/content/Regulatory Affairs of Road Accident Data 2020 India.csv",index_
df
```

Out[ ]:

	Cause category	Cause Subcategory	Outcome of Incident	Count
Million Plus Cities				
Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0
Agra	Traffic Control	Flashing Signal/Blinker	Minor Injury	0.0
Agra	Traffic Control	Flashing Signal/Blinker	Persons Killed	0.0
Agra	Traffic Control	Flashing Signal/Blinker	Total Injured	0.0
Agra	Traffic Control	Flashing Signal/Blinker	Total number of Accidents	0.0
...	...	...	...	...
Vizaq	Weather	Sunny/Clear	Greviously Injured	561.0
Vizaq	Weather	Sunny/Clear	Minor Injury	252.0
Vizaq	Weather	Sunny/Clear	Persons Killed	176.0
Vizaq	Weather	Sunny/Clear	Total number of Accidents	1207.0
Vizaq	Weather	Sunny/Clear	Total Injured	813.0

9550 rows × 4 columns

In [ ]:

```
df.sort_index(ascending=False)
```

Out[ ]:

	Cause category	Cause Subcategory	Outcome of Incident	Count
Million Plus Cities				
Vizaq	Weather	Sunny/Clear	Total Injured	813.0
Vizaq	Junction	Y	Greviously Injured	25.0
Vizaq	Traffic Violation	Over	Minor Injury	277.0

	Cause category	Cause Subcategory	Outcome of Incident	Count
Million Plus Cities				
Vizaq	Traffic Violation	Over	Greviously Injured	590.0
Vizaq	Traffic Violation	Others	Total Injured	304.0
...	...	...	...	...
Agra	Traffic Violation	Use of Mobile Phone	Greviously Injured	8.0
Agra	Traffic Violation	Use of Mobile Phone	Minor Injury	3.0
Agra	Traffic Violation	Use of Mobile Phone	Total number of Accidents	16.0
Agra	Traffic Violation	Use of Mobile Phone	Persons Killed	9.0
Agra	Traffic Control	Flashing Signal/Blinker	Greviously Injured	0.0

9550 rows × 4 columns

Exploratory Data Analysis (EDA)

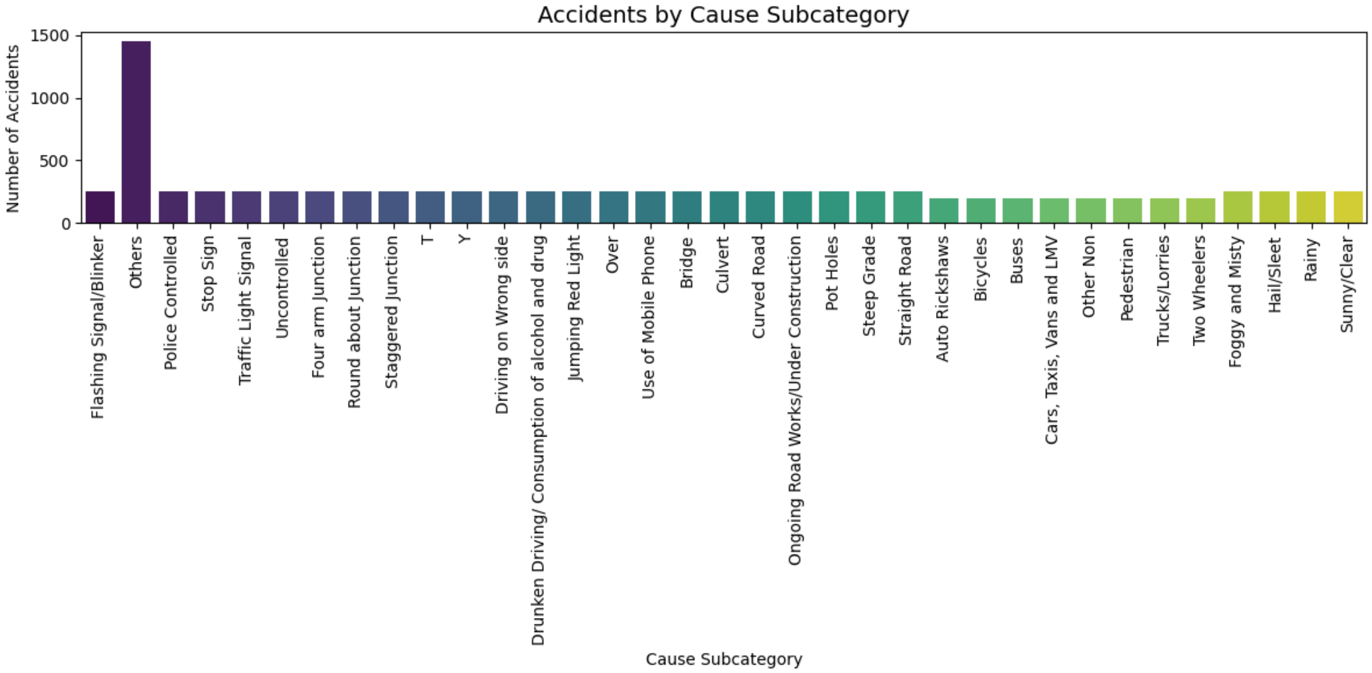
In [ ]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

Frequency of Road Accidents by Specific Cause (2020)

In [ ]:

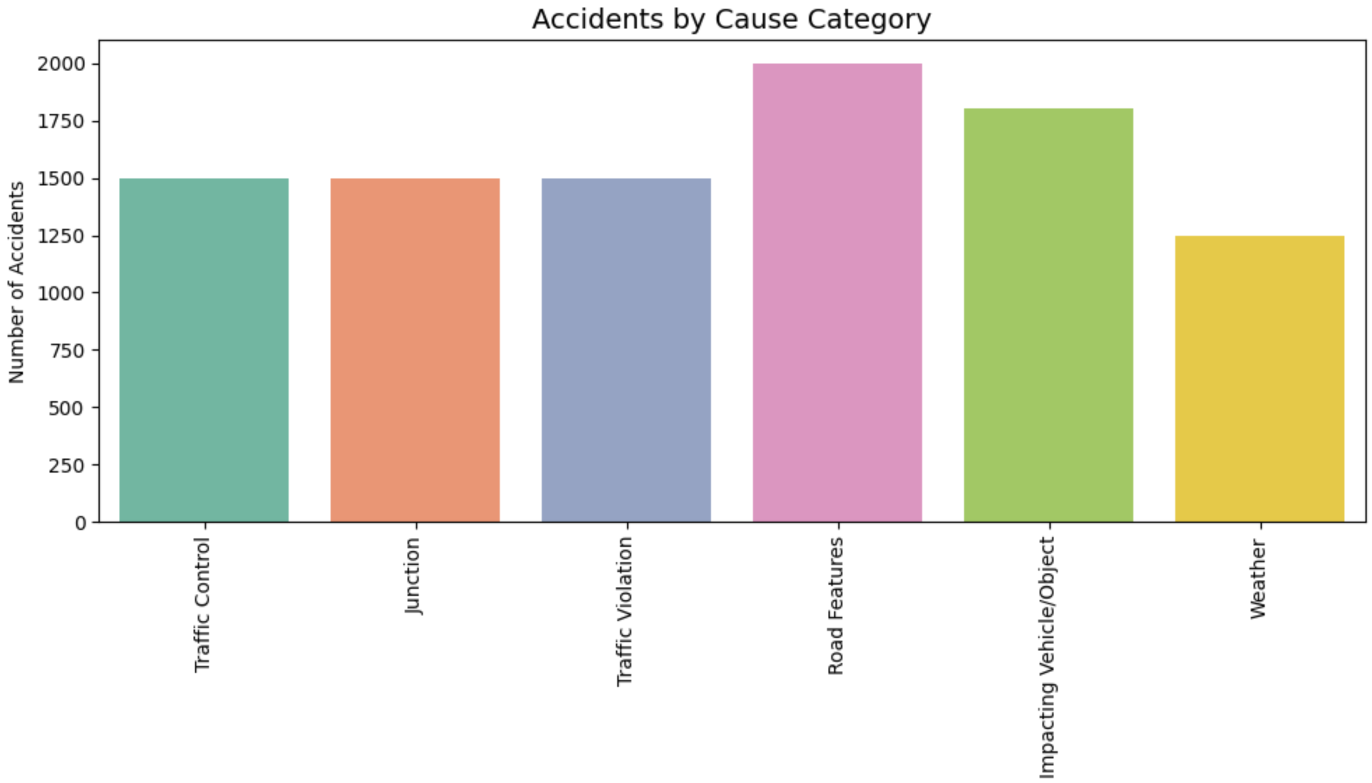
```
plt.figure(figsize=(12,6))
sns.countplot(
    data=df,
    x="Cause Subcategory",
    hue="Cause Subcategory",  # Assign hue to avoid warning
    legend=False,             # Hide the legend
    palette="viridis"
)
plt.xticks(rotation=90)
plt.title("Accidents by Cause Subcategory", fontsize=14)
plt.xlabel("Cause Subcategory")
plt.ylabel("Number of Accidents")
plt.tight_layout()
plt.show()
```


Accidents by Cause Subcategory

Distribution of Road Accidents by Major Cause Category (2020)

In [ ]:

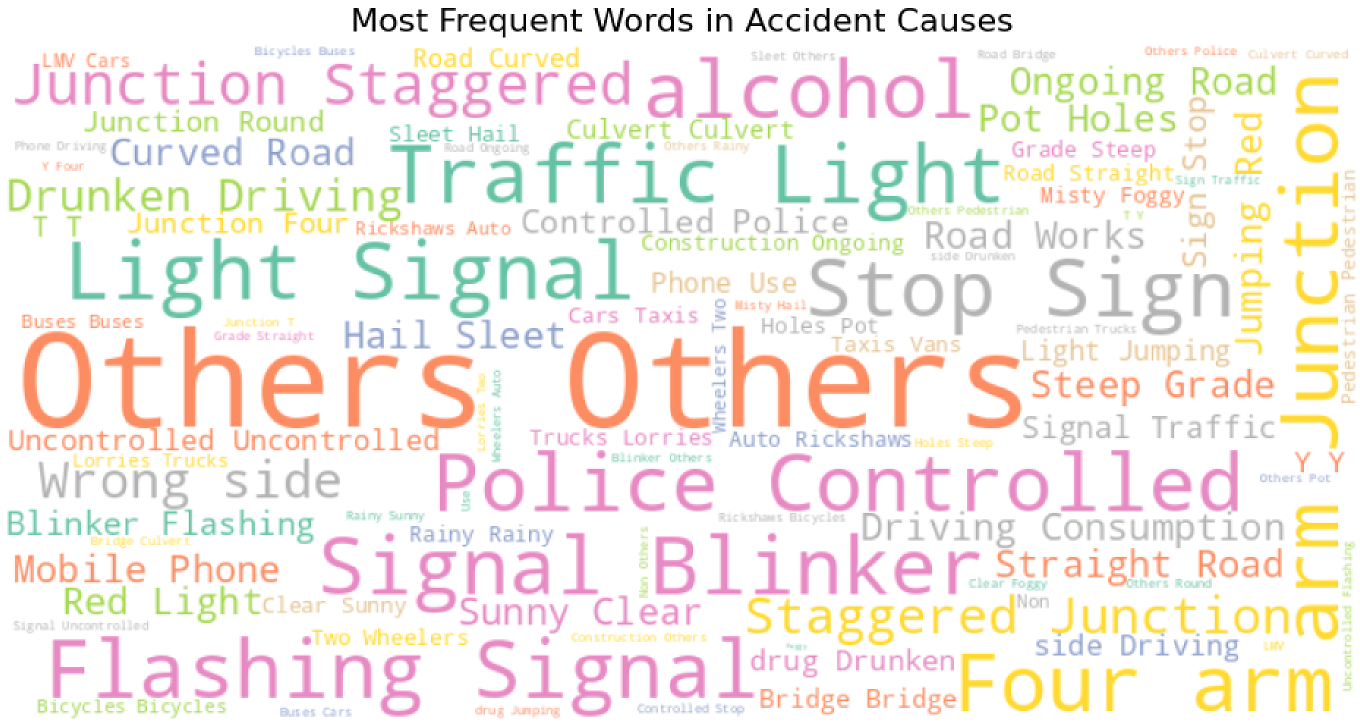
```
plt.figure(figsize=(10,6))
sns.countplot(
    data=df,
    x="Cause category",
    hue="Cause category",   # This assigns colors per category
    legend=False,           # Hide legend since x-axis already shows labels
    palette="Set2"          # Try 'Set2', 'Spectral', 'coolwarm', etc.
)
plt.xticks(rotation=90)
plt.title("Accidents by Cause Category", fontsize=14)
plt.xlabel("Cause Category")
plt.ylabel("Number of Accidents")
plt.tight_layout()
plt.show()
```



Most Common Accident Causes in Indian Cities (2020)

In [ ]:

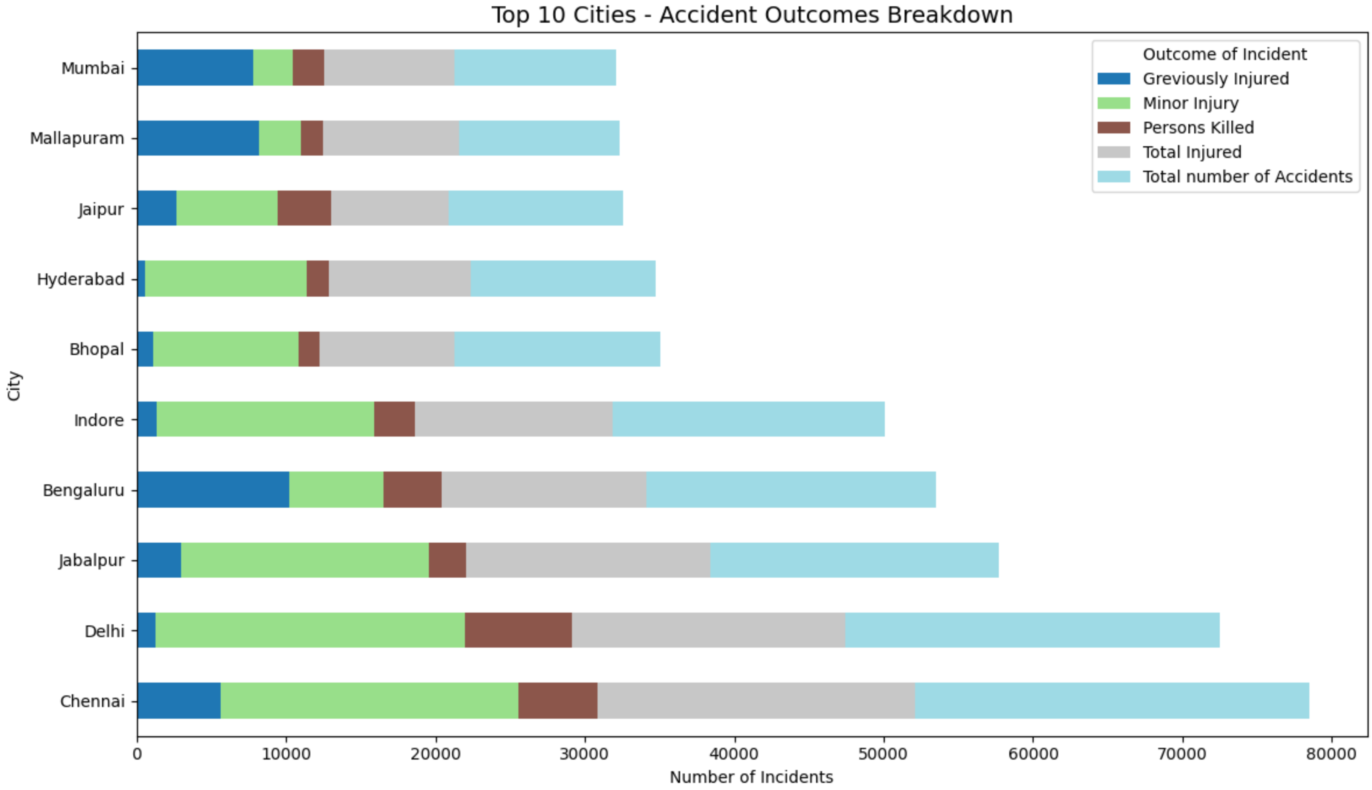
```
from wordcloud import WordCloud

text = " ".join(df["Cause Subcategory"])
wordcloud = WordCloud(width=800, height=400, background_color="white", colormap="Set2").

plt.figure(figsize=(12,6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("Most Frequent Words in Accident Causes", fontsize=16)
plt.show()
```



Top 10 Indian Cities by Accident Outcomes (2020)

In [ ]:

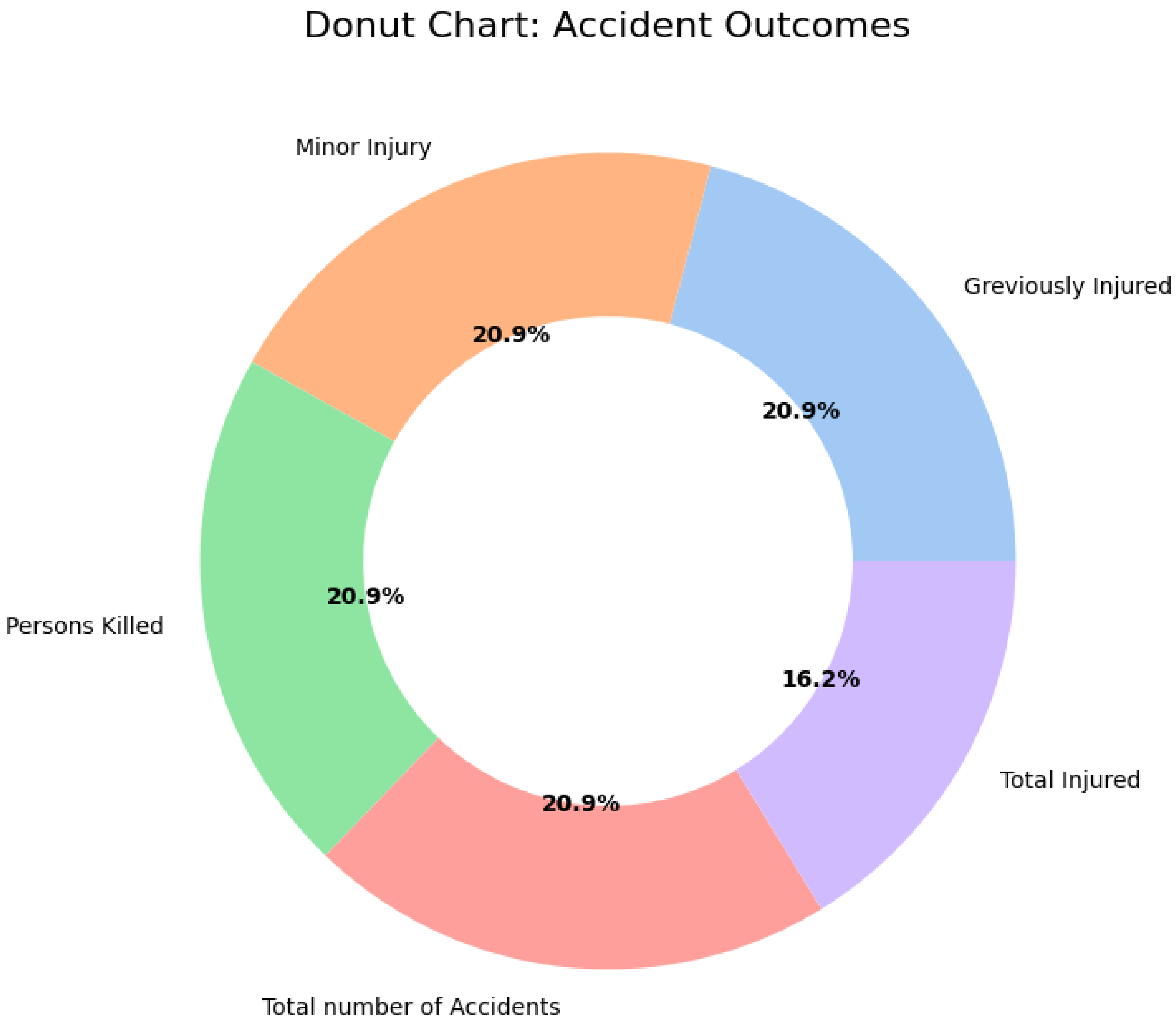
```
city_outcome = df.groupby(["Million Plus Cities", "Outcome of Incident"])["Count"].sum()
top10_cities = city_outcome.sum(axis=1).sort_values(ascending=False).head(10)
top10_data = city_outcome.loc[top10_cities.index]

top10_data.plot(
    kind="barh",
    stacked=True,
    figsize=(12,7),
    colormap="tab20"
)
plt.title("Top 10 Cities - Accident Outcomes Breakdown", fontsize=14)
plt.xlabel("Number of Incidents")
plt.ylabel("City")
plt.tight_layout()
plt.show()
```



Proportion of Road Accident Outcomes in Indian Cities (2020)

In [ ]:

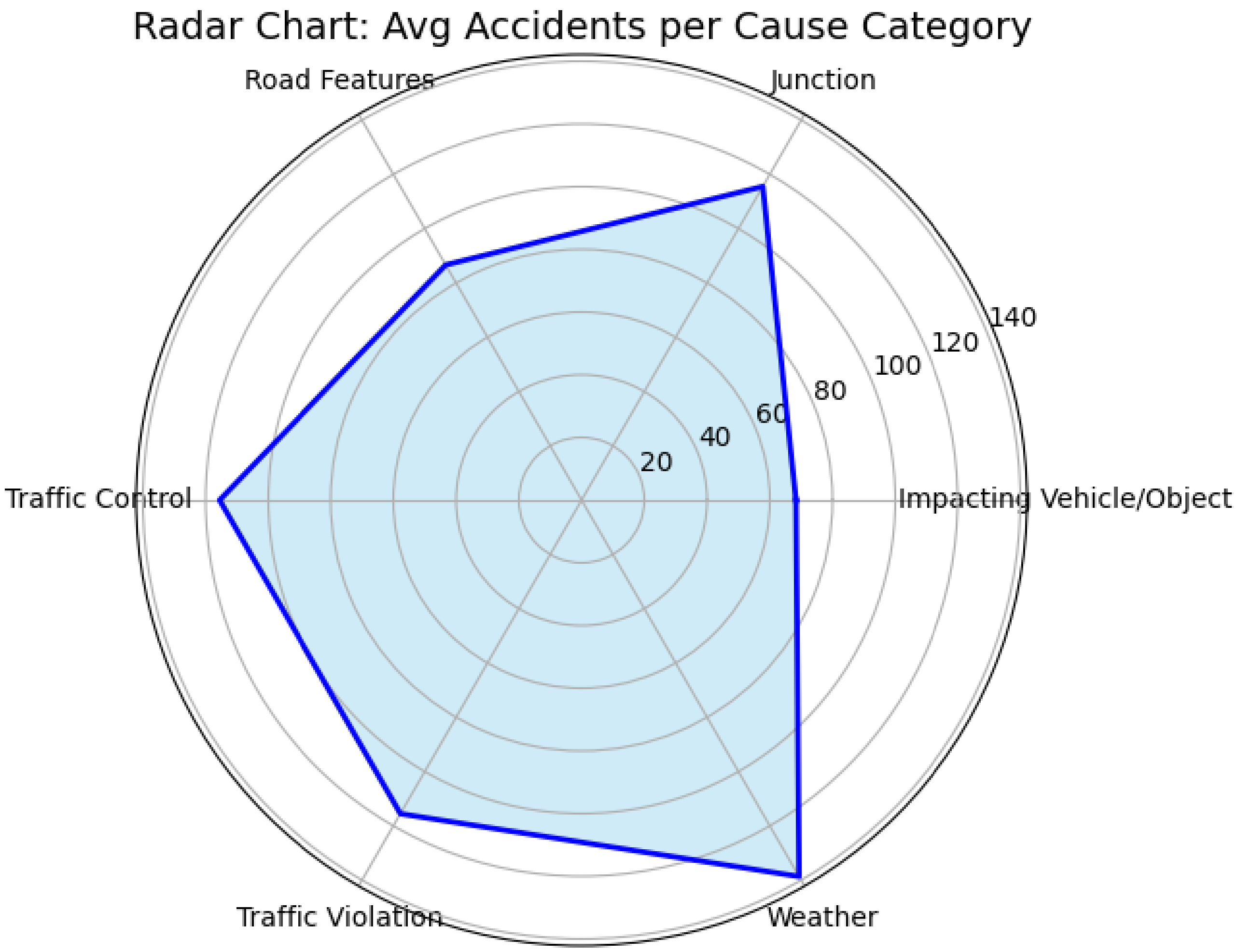
```
outcome_counts = df["Outcome of Incident"].value_counts()

plt.figure(figsize=(8,8))
wedges, texts, autotexts = plt.pie(
    outcome_counts,
    labels=outcome_counts.index,
    autopct="%1.1f%%",
    colors=sns.color_palette("pastel"),
    wedgeprops=dict(width=0.4)  # creates the hole in center
)
plt.setp(autotexts, size=10, weight="bold")
plt.title("Donut Chart: Accident Outcomes", fontsize=16)
plt.show()
```



Average Number of Road Accidents per Major Cause Category (2020)

In [ ]:

```
import numpy as np

avg_cause = df.groupby("Cause category")["Count"].mean()
categories = avg_cause.index
values = avg_cause.values
values = np.append(values, values[0])  # close the circle

angles = np.linspace(0, 2*np.pi, len(categories), endpoint=False).tolist()
angles += angles[:1]

fig, ax = plt.subplots(figsize=(6,6), subplot_kw=dict(polar=True))
ax.plot(angles, values, color='blue', linewidth=2)
ax.fill(angles, values, color='skyblue', alpha=0.4)
ax.set_xticks(angles[:-1])
ax.set_xticklabels(categories)
ax.set_title("Radar Chart: Avg Accidents per Cause Category", fontsize=14)
plt.show()
```



Flow of Road Accident Causes to Outcomes (2020)

In [ ]:

```
import plotly.graph_objects as go

cause_counts = df.groupby(["Cause category", "Outcome of Incident"])["Count"].sum().rese

categories = list(cause_counts["Cause category"].unique()) + list(cause_counts["Outcome
category_map = {cat: i for i, cat in enumerate(categories)}

links = dict(
    source = cause_counts["Cause category"].map(category_map),
    target = cause_counts["Outcome of Incident"].map(category_map),
    value = cause_counts["Count"]
)

fig = go.Figure(go.Sankey(
    node=dict(label=categories, pad=20, thickness=20),
    link=dict(source=links["source"], target=links["target"], value=links["value"])
))
fig.update_layout(title_text="Sankey Diagram: Cause → Outcome", font_size=10)
fig.show()
```