In [4]:
```
!pip install folium
```

```
Collecting folium
  Obtaining dependency information for folium from https://files.pythonhosted.org/pac
kages/ae/6d/18a7546e1748ecdd6ed7cd00d3f183faf1df08bd4f5e5e0eb3e72458b862/folium-0.17.
0-py2.py3-none-any.whl.metadata
  Downloading folium-0.17.0-py2.py3-none-any.whl.metadata (3.8 kB)
Collecting branca>=0.6.0 (from folium)
  Obtaining dependency information for branca>=0.6.0 from https://files.pythonhosted.
org/packages/75/ca/6074ab4a04dd1a503201c18091b3426f3709670115fae316907a97f98d75/branc
a-0.7.2-py3-none-any.whl.metadata
  Downloading branca-0.7.2-py3-none-any.whl.metadata (1.5 kB)
Requirement already satisfied: jinja2>=2.9 in c:\users\praveen t\anaconda3\lib\site-p
ackages (from folium) (3.1.2)
Requirement already satisfied: numpy in c:\users\praveen t\anaconda3\lib\site-package
s (from folium) (1.24.3)
Requirement already satisfied: requests in c:\users\praveen t\anaconda3\lib\site-pack
ages (from folium) (2.31.0)
Requirement already satisfied: xyzservices in c:\users\praveen t\anaconda3\lib\site-p
ackages (from folium) (2022.9.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\praveen t\anaconda3\lib\si
te-packages (from jinja2>=2.9->folium) (2.1.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\praveen t\anacond
a3\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\praveen t\anaconda3\lib\site-
packages (from requests->folium) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\praveen t\anaconda3\lib
\site-packages (from requests->folium) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\praveen t\anaconda3\lib
\site-packages (from requests->folium) (2023.7.22)
Downloading folium-0.17.0-py2.py3-none-any.whl (108 kB)
   ---------------------------------------- 0.0/108.4 kB ? eta -:--:--
   ---------------------------------------- 108.4/108.4 kB 6.1 MB/s eta 0:00:00
Downloading branca-0.7.2-py3-none-any.whl (25 kB)
Installing collected packages: branca, folium
Successfully installed branca-0.7.2 folium-0.17.0
```

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import folium
from folium.plugins import HeatMap
```

In [2]:
```
data = pd.read_csv(r"C:\Users\Praveen T\Downloads\US_Accidents_March23.csv\US_Acciden
data
```

Out[2]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_
0	A-1	Source2	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	NaN	
1	A-2	Source2	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	NaN	
2	A-3	Source2	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	NaN	
3	A-4	Source2	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	NaN	
4	A-5	Source2	2	2016-02-08 07:39:07	2016-02-08 08:09:07	39.627781	-84.188354	NaN	
...	...	...	...	...	...	...	...	...	
7728389	A-7777757	Source1	2	2019-08-23 18:03:25	2019-08-23 18:32:01	34.002480	-117.379360	33.99888	-117.3
7728390	A-7777758	Source1	2	2019-08-23 19:11:30	2019-08-23 19:38:23	32.766960	-117.148060	32.76555	-117.1
7728391	A-7777759	Source1	2	2019-08-23 19:00:21	2019-08-23 19:28:49	33.775450	-117.847790	33.77740	-117.8
7728392	A-7777760	Source1	2	2019-08-23 19:00:21	2019-08-23 19:29:42	33.992460	-118.403020	33.98311	-118.3
7728393	A-7777761	Source1	2	2019-08-23 18:52:06	2019-08-23 19:21:31	34.133930	-117.230920	34.13736	-117.2

7728394 rows × 46 columns

```
In [3]: data.head()
```

Out[3]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(m
0	A-1	Source2	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	NaN	NaN	0.0
1	A-2	Source2	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	NaN	NaN	0.0
2	A-3	Source2	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	NaN	NaN	0.0
3	A-4	Source2	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	NaN	NaN	0.0
4	A-5	Source2	2	2016-02-08 07:39:07	2016-02-08 08:09:07	39.627781	-84.188354	NaN	NaN	0.0

5 rows × 46 columns

In [4]:
```python
data.tail()
```

Out[4]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_L
7728389	A-7777757	Source1	2	2019-08-23 18:03:25	2019-08-23 18:32:01	34.00248	-117.37936	33.99888	-117.370
7728390	A-7777758	Source1	2	2019-08-23 19:11:30	2019-08-23 19:38:23	32.76696	-117.14806	32.76555	-117.153
7728391	A-7777759	Source1	2	2019-08-23 19:00:21	2019-08-23 19:28:49	33.77545	-117.84779	33.77740	-117.857
7728392	A-7777760	Source1	2	2019-08-23 19:00:21	2019-08-23 19:29:42	33.99246	-118.40302	33.98311	-118.395
7728393	A-7777761	Source1	2	2019-08-23 18:52:06	2019-08-23 19:21:31	34.13393	-117.23092	34.13736	-117.239

5 rows × 46 columns

In [5]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7728394 entries, 0 to 7728393
Data columns (total 46 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   ID                     object 
 1   Source                 object 
 2   Severity               int64  
 3   Start_Time             object 
 4   End_Time               object 
 5   Start_Lat              float64
 6   Start_Lng              float64
 7   End_Lat                float64
 8   End_Lng                float64
 9   Distance(mi)           float64
 10  Description            object 
 11  Street                 object 
 12  City                   object 
 13  County                 object 
 14  State                  object 
 15  Zipcode                object 
 16  Country                object 
 17  Timezone               object 
 18  Airport_Code           object 
 19  Weather_Timestamp      object 
 20  Temperature(F)         float64
 21  Wind_Chill(F)          float64
 22  Humidity(%)            float64
 23  Pressure(in)           float64
 24  Visibility(mi)         float64
 25  Wind_Direction         object 
 26  Wind_Speed(mph)        float64
 27  Precipitation(in)      float64
 28  Weather_Condition      object 
 29  Amenity                bool   
 30  Bump                   bool   
 31  Crossing               bool   
 32  Give_Way               bool   
 33  Junction               bool   
 34  No_Exit                bool   
 35  Railway                bool   
 36  Roundabout             bool   
 37  Station                bool   
 38  Stop                   bool   
 39  Traffic_Calming        bool   
 40  Traffic_Signal         bool   
 41  Turning_Loop           bool   
 42  Sunrise_Sunset         object 
 43  Civil_Twilight         object 
 44  Nautical_Twilight      object 
 45  Astronomical_Twilight  object 
dtypes: bool(13), float64(12), int64(1), object(20)
memory usage: 2.0+ GB
```

In [6]:

```
data.describe()
```

Out[6]:

	Severity	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(mi)	Temp
count	7.728394e+06	7.728394e+06	7.728394e+06	4.325632e+06	4.325632e+06	7.728394e+06	7.5
mean	2.212384e+00	3.620119e+01	-9.470255e+01	3.626183e+01	-9.572557e+01	5.618423e-01	6.1
std	4.875313e-01	5.076079e+00	1.739176e+01	5.272905e+00	1.810793e+01	1.776811e+00	1.9
min	1.000000e+00	2.455480e+01	-1.246238e+02	2.456601e+01	-1.245457e+02	0.000000e+00	-8.9
25%	2.000000e+00	3.339963e+01	-1.172194e+02	3.346207e+01	-1.177543e+02	0.000000e+00	4.9
50%	2.000000e+00	3.582397e+01	-8.776662e+01	3.618349e+01	-8.802789e+01	3.000000e-02	6.4
75%	2.000000e+00	4.008496e+01	-8.035368e+01	4.017892e+01	-8.024709e+01	4.640000e-01	7.6
max	4.000000e+00	4.900220e+01	-6.711317e+01	4.907500e+01	-6.710924e+01	4.417500e+02	2.0

In [7]:

```python
# Convert Start_Time and End_Time to datetime
data['Start_Time'] = pd.to_datetime(data['Start_Time'])
data['End_Time'] = pd.to_datetime(data['End_Time'])

# Extract hour, day of week, and month
data['hour'] = data['Start_Time'].dt.hour
data['day_of_week'] = data['Start_Time'].dt.day_name()
data['month'] = data['Start_Time'].dt.month

data.head()
```

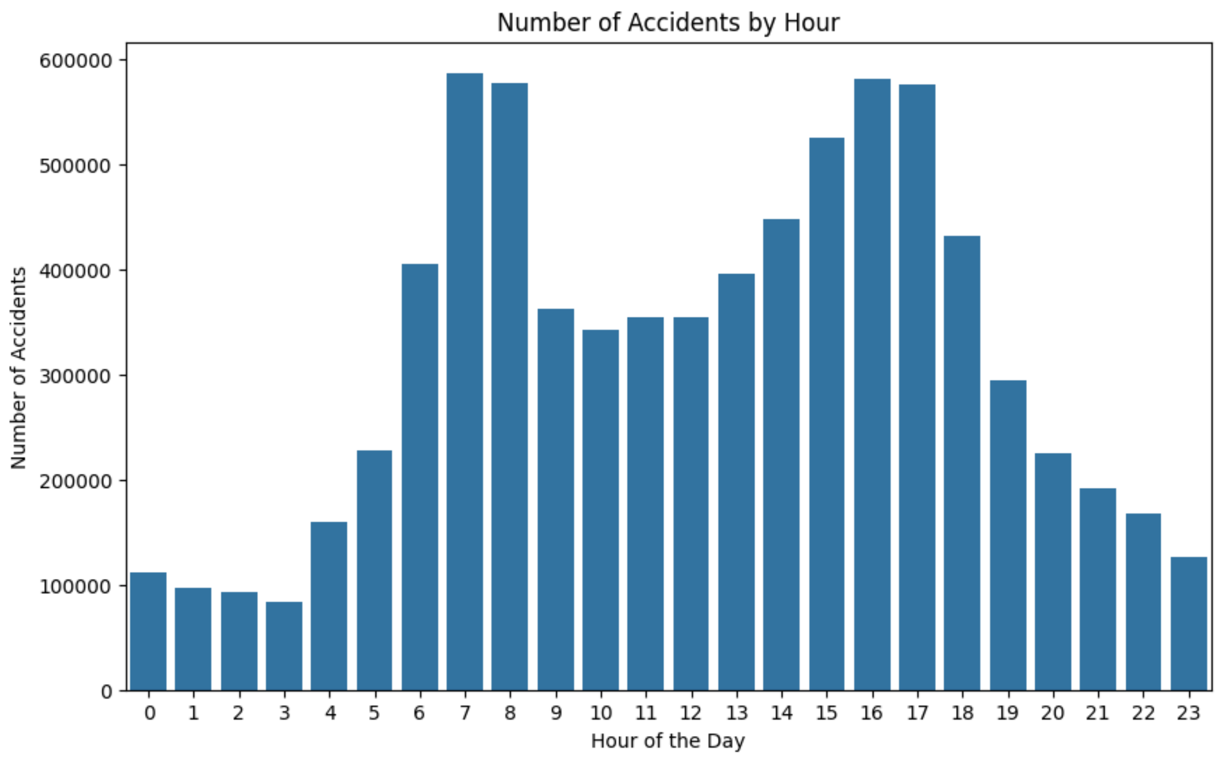
Out[7]:

	ID	Source	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(m
0	A-1	Source2	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	NaN	NaN	0.0
1	A-2	Source2	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	NaN	NaN	0.0
2	A-3	Source2	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	NaN	NaN	0.0
3	A-4	Source2	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	NaN	NaN	0.0
4	A-5	Source2	2	2016-02-08 07:39:07	2016-02-08 08:09:07	39.627781	-84.188354	NaN	NaN	0.0

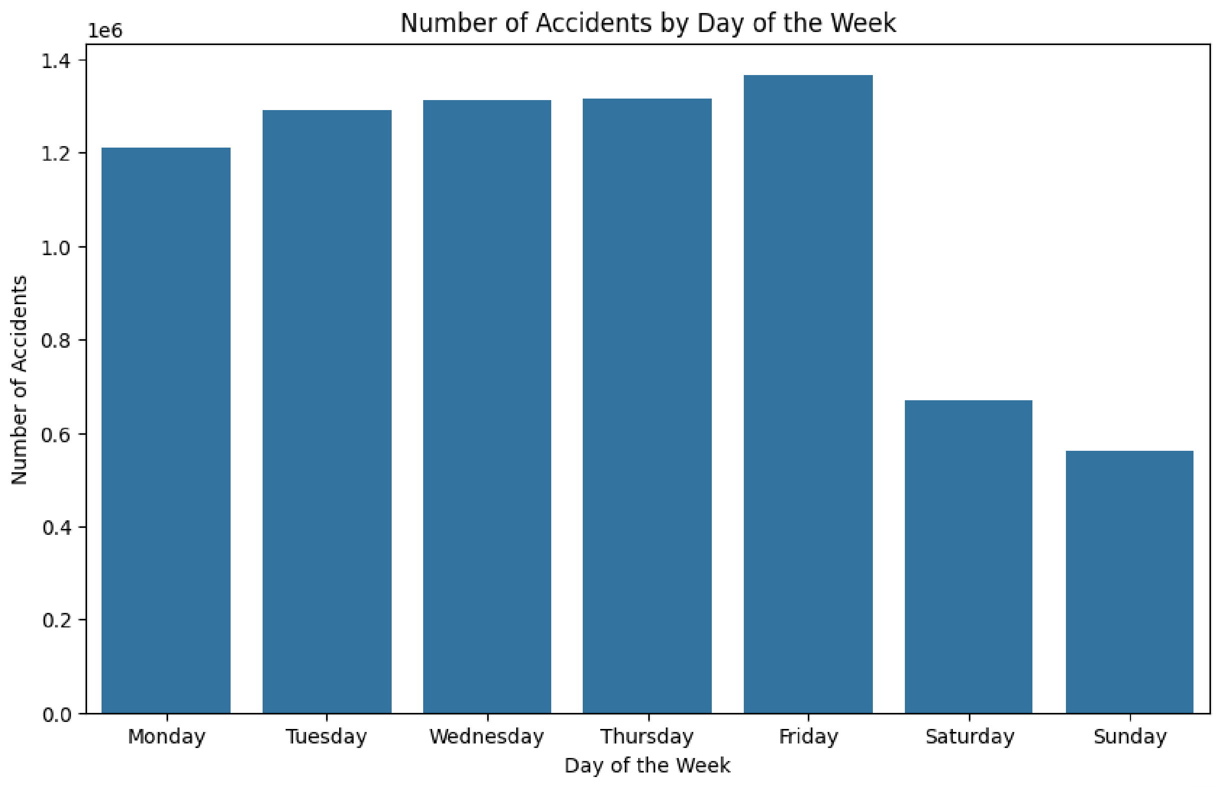
5 rows × 49 columns

In [8]:

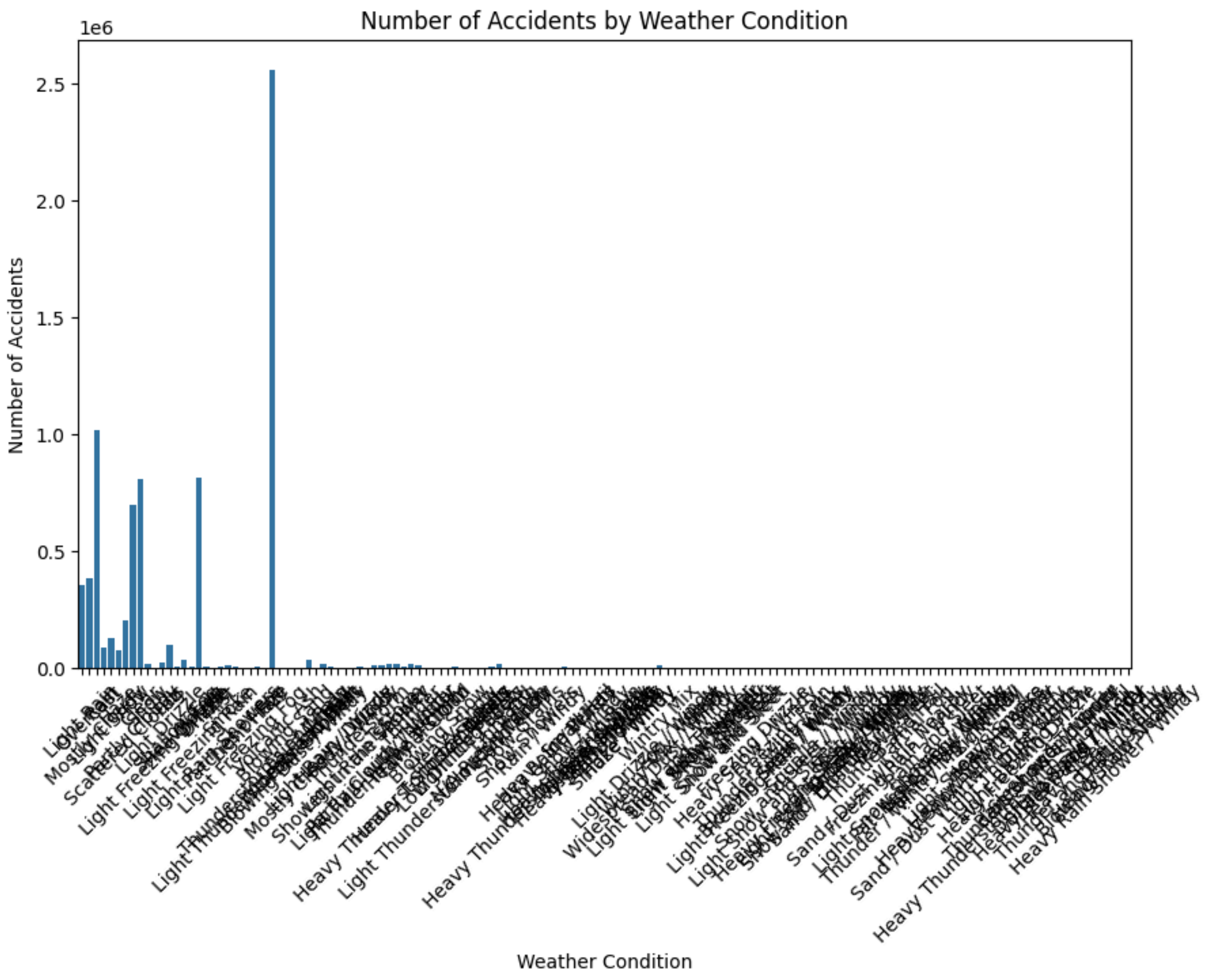
```python
# Accidents by hour
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='hour')
plt.title('Number of Accidents by Hour')
plt.xlabel('Hour of the Day')
plt.ylabel('Number of Accidents')
plt.show()
```



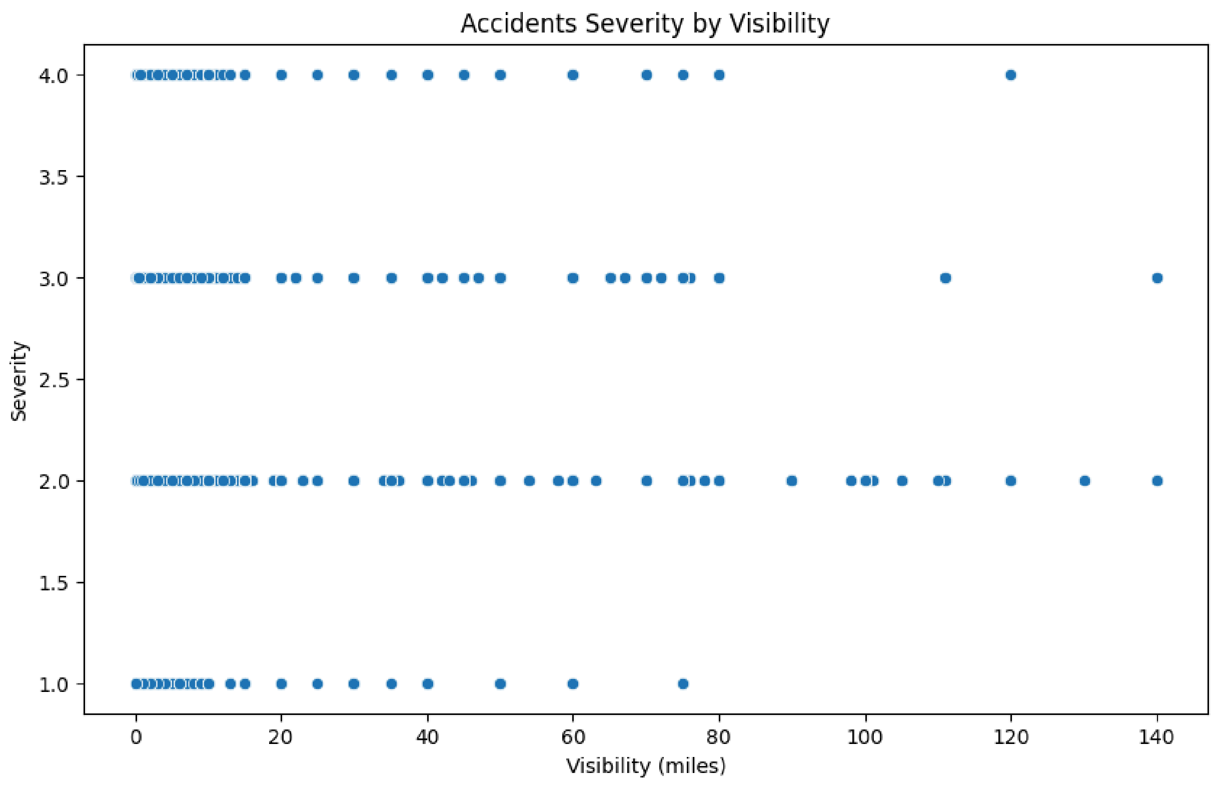
In [9]:
```
# Accidents by day of the week
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='day_of_week', order=['Monday', 'Tuesday', 'Wednesday', 'T
plt.title('Number of Accidents by Day of the Week')
plt.xlabel('Day of the Week')
plt.ylabel('Number of Accidents')
plt.show()
```



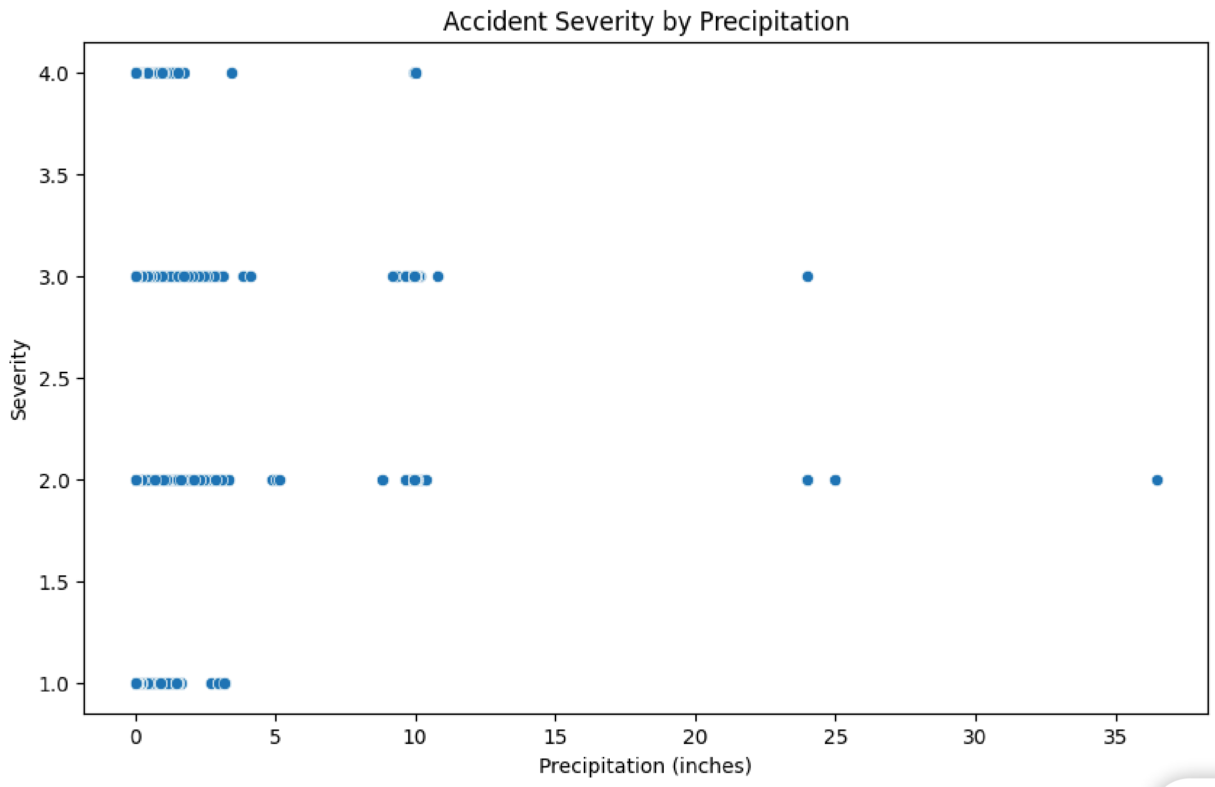
In [10]:
```
# Accidents by weather condition
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='Weather_Condition')
plt.title('Number of Accidents by Weather Condition')
plt.xlabel('Weather Condition')
plt.ylabel('Number of Accidents')
plt.xticks(rotation=45)
plt.show()
```



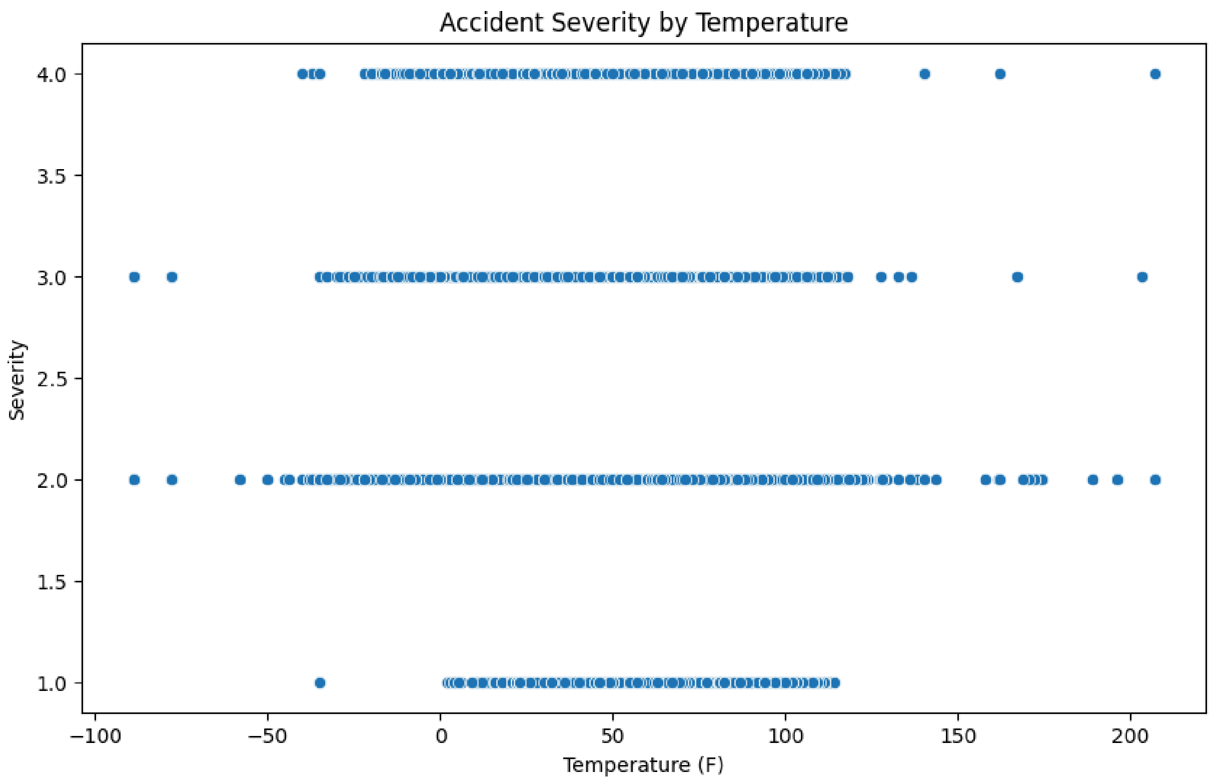
In [11]:
```
# Accidents by visibility
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x='Visibility(mi)', y='Severity')
plt.title('Accidents Severity by Visibility')
plt.xlabel('Visibility (miles)')
plt.ylabel('Severity')
plt.show()
```



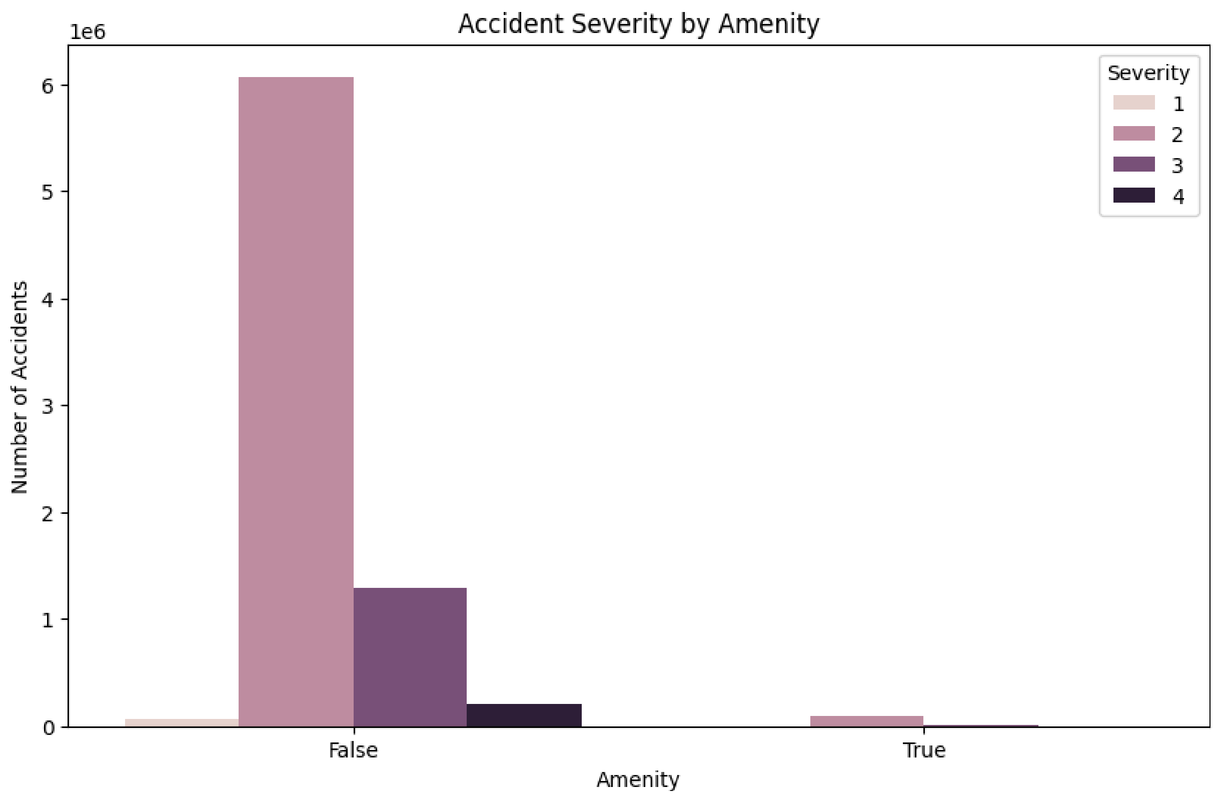
```
In [12]: # Accidents by precipitation
         plt.figure(figsize=(10, 6))
         sns.scatterplot(data=data, x='Precipitation(in)', y='Severity')
         plt.title('Accident Severity by Precipitation')
         plt.xlabel('Precipitation (inches)')
         plt.ylabel('Severity')
         plt.show()
```

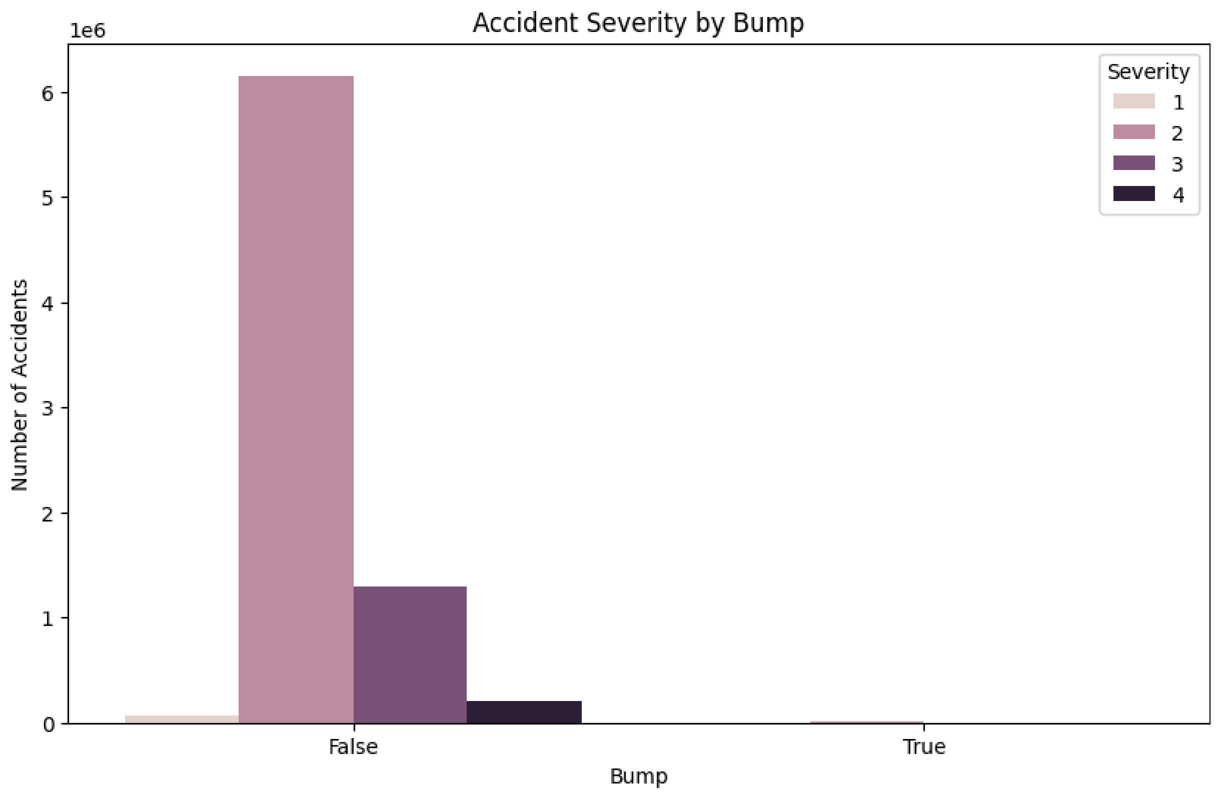


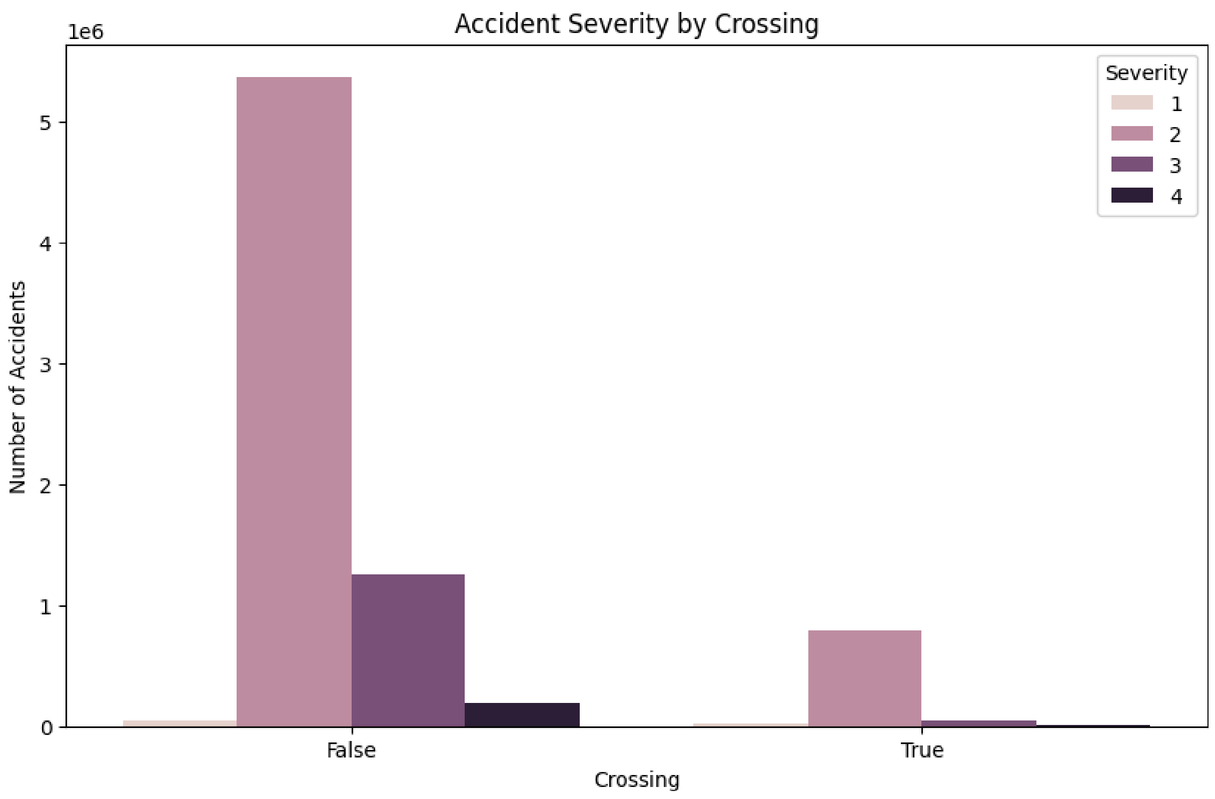
```
In [14]: # Accidents by temperature
         plt.figure(figsize=(10, 6))
         sns.scatterplot(data=data, x='Temperature(F)', y='Severity')
         plt.title('Accident Severity by Temperature')
         plt.xlabel('Temperature (F)')
         plt.ylabel('Severity')
         plt.show()
```

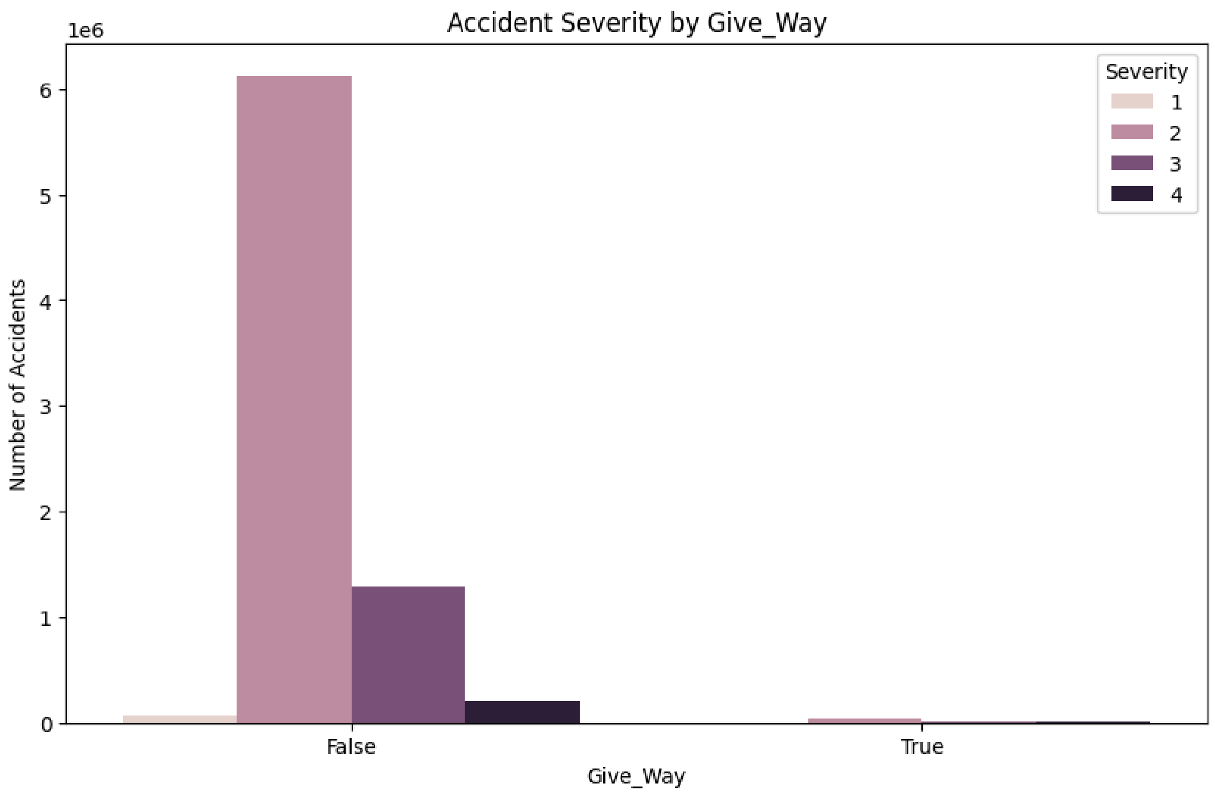


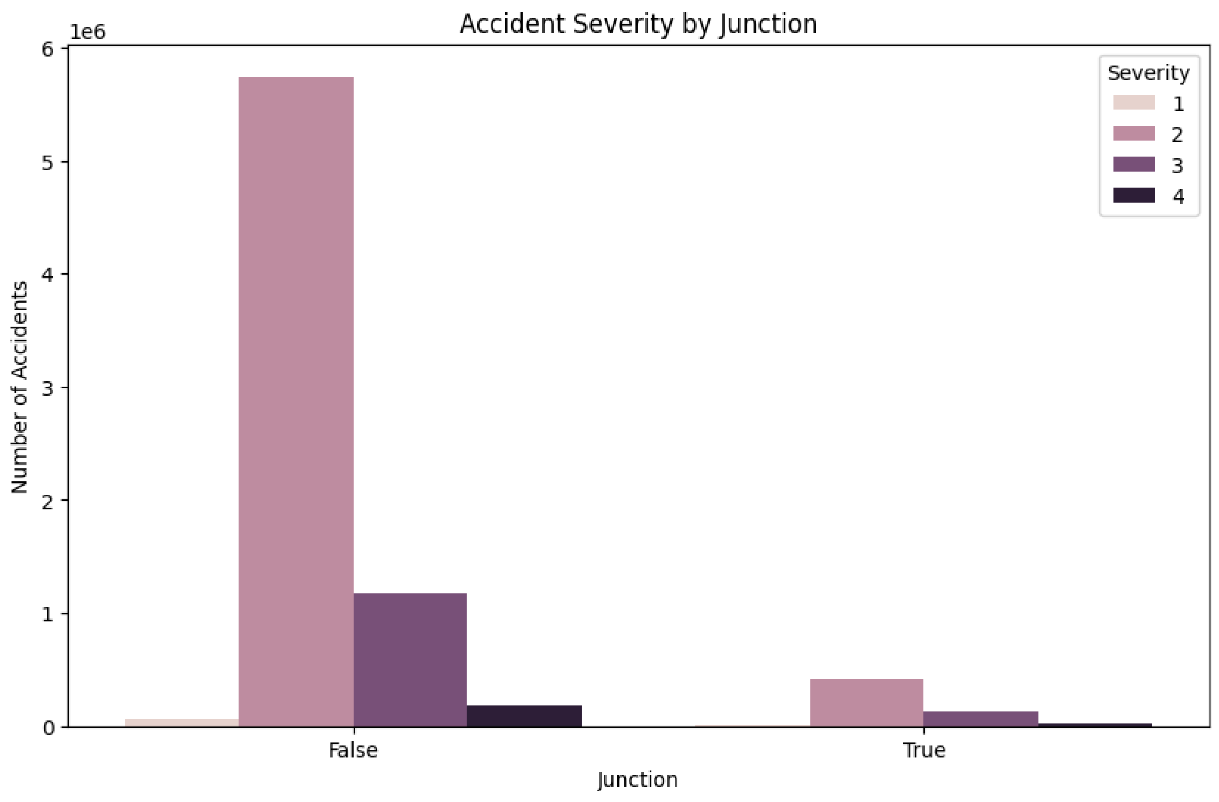
In [13]:
```
# List of road condition features
road_conditions = ['Amenity', 'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit',
                   'Railway', 'Roundabout', 'Station', 'Stop', 'Traffic_Calming',
                   'Traffic_Signal', 'Turning_Loop']

# Plotting accident severity by road conditions
for condition in road_conditions:
    plt.figure(figsize=(10, 6))
    sns.countplot(data=data, x=condition, hue='Severity')
    plt.title(f'Accident Severity by {condition}')
    plt.xlabel(condition)
    plt.ylabel('Number of Accidents')
    plt.legend(title='Severity')
    plt.show()
```

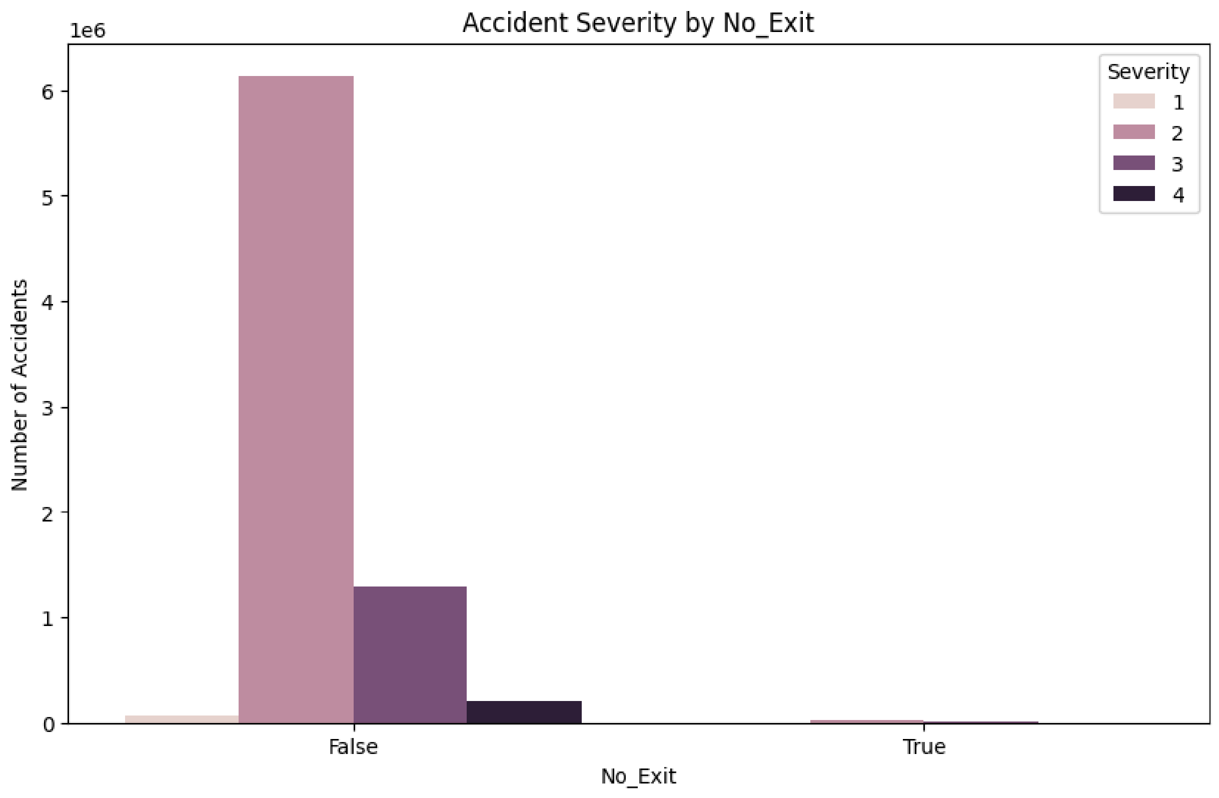

Accident Severity by Amenity
1e6
Number of Accidents
6
5
4
3
2
1
0
False
True
Amenity
Severity
1
2
3
4

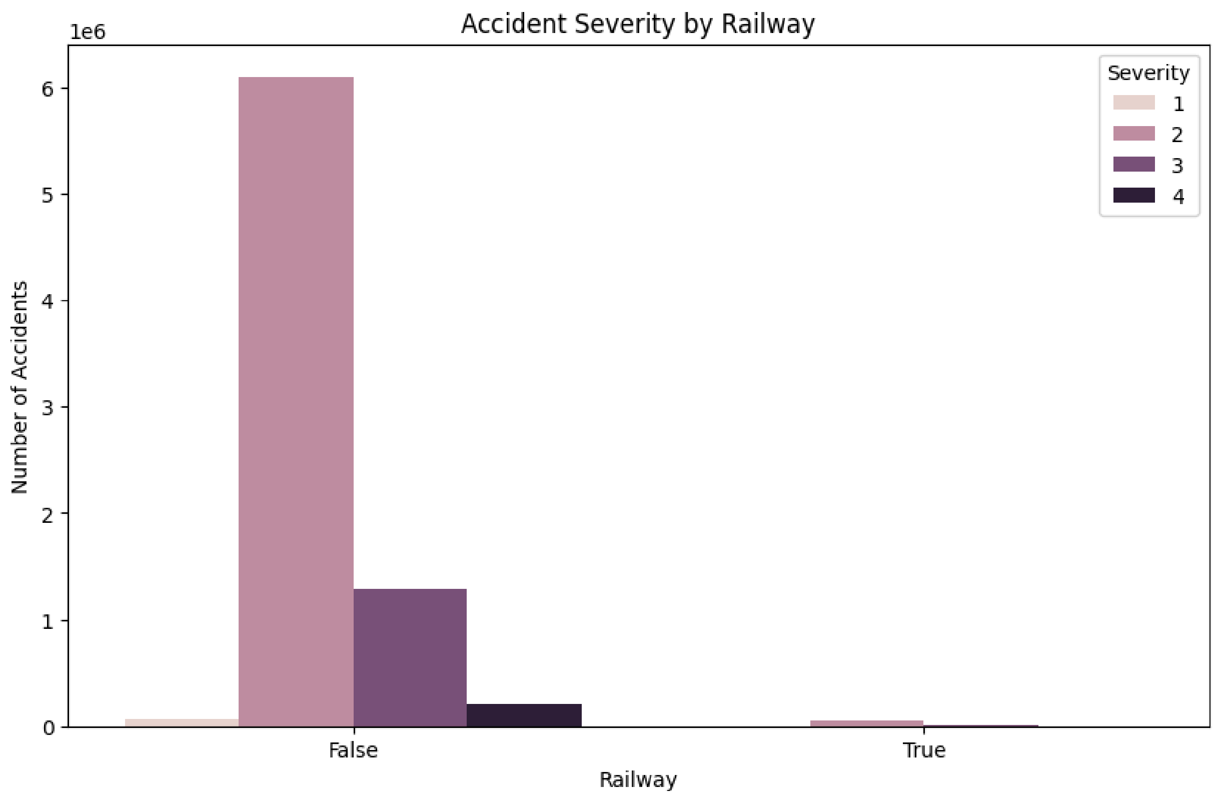

Accident Severity by Bump
1e6
Number of Accidents
6
5
4
3
2
1
0
False
True
Bump
Severity
1
2
3
4


Accident Severity by Crossing
1e6
0
1
2
3
4
5
Number of Accidents
Severity
1
2
3
4
False
True
Crossing

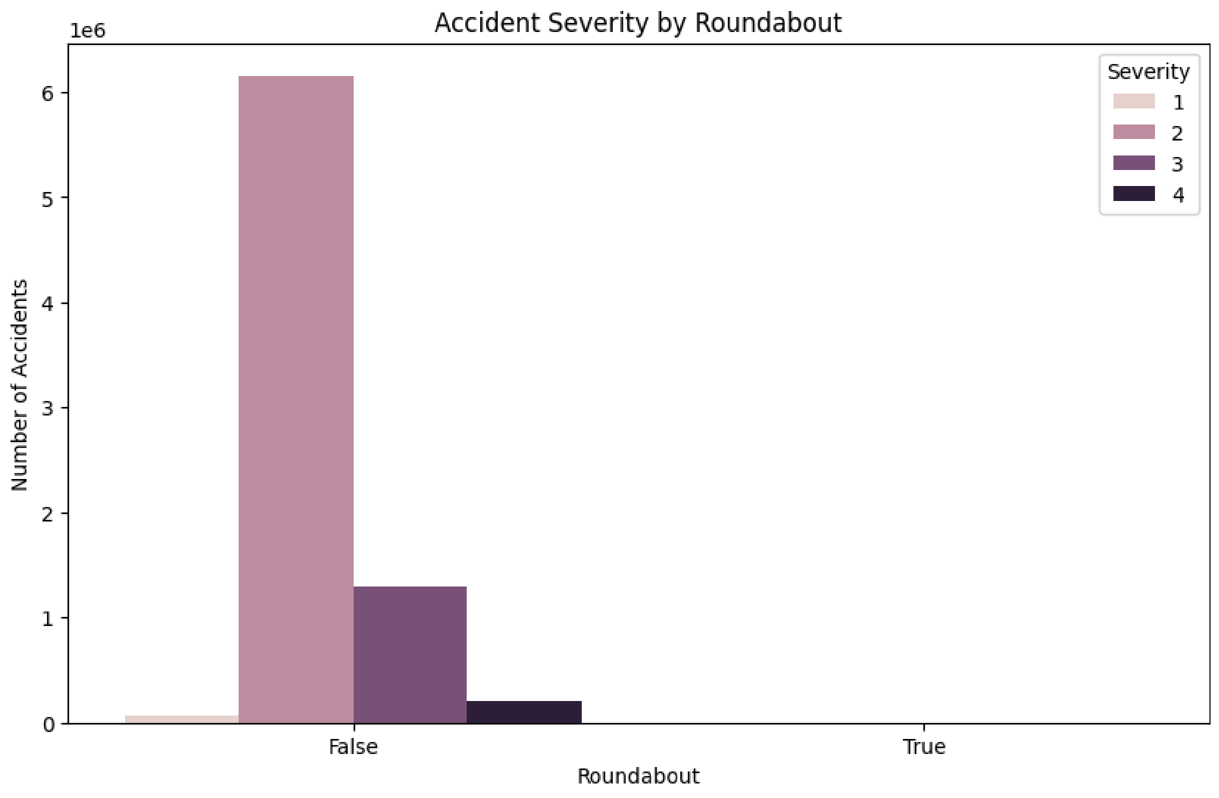

Accident Severity by Give_Way
1e6
0
1
2
3
4
5
6
Number of Accidents
Severity
1
2
3
4
False
True
Give_Way


Accident Severity by Junction
1e6
6
5
4
3
2
1
0
Number of Accidents
Severity
1
2
3
4
False
True
Junction

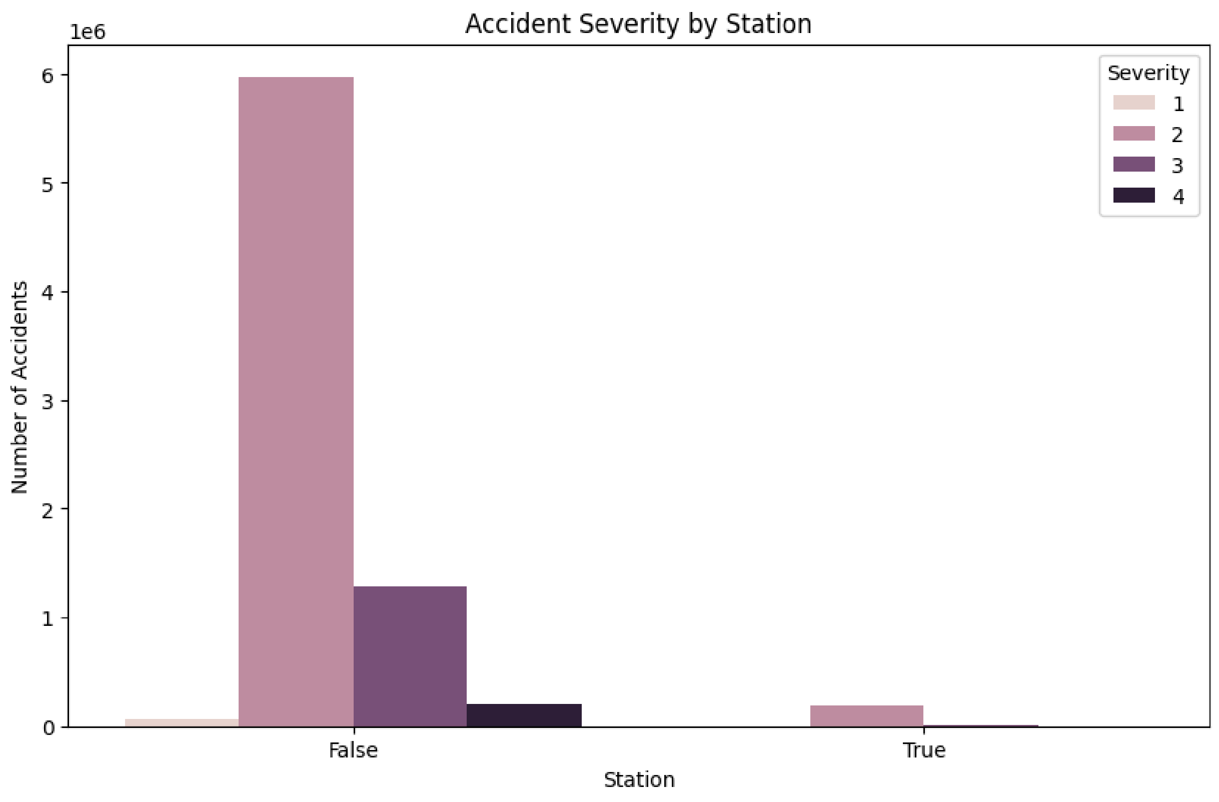

Accident Severity by No_Exit
1e6
6
5
4
3
2
1
0
Number of Accidents
Severity
1
2
3
4
False
True
No_Exit

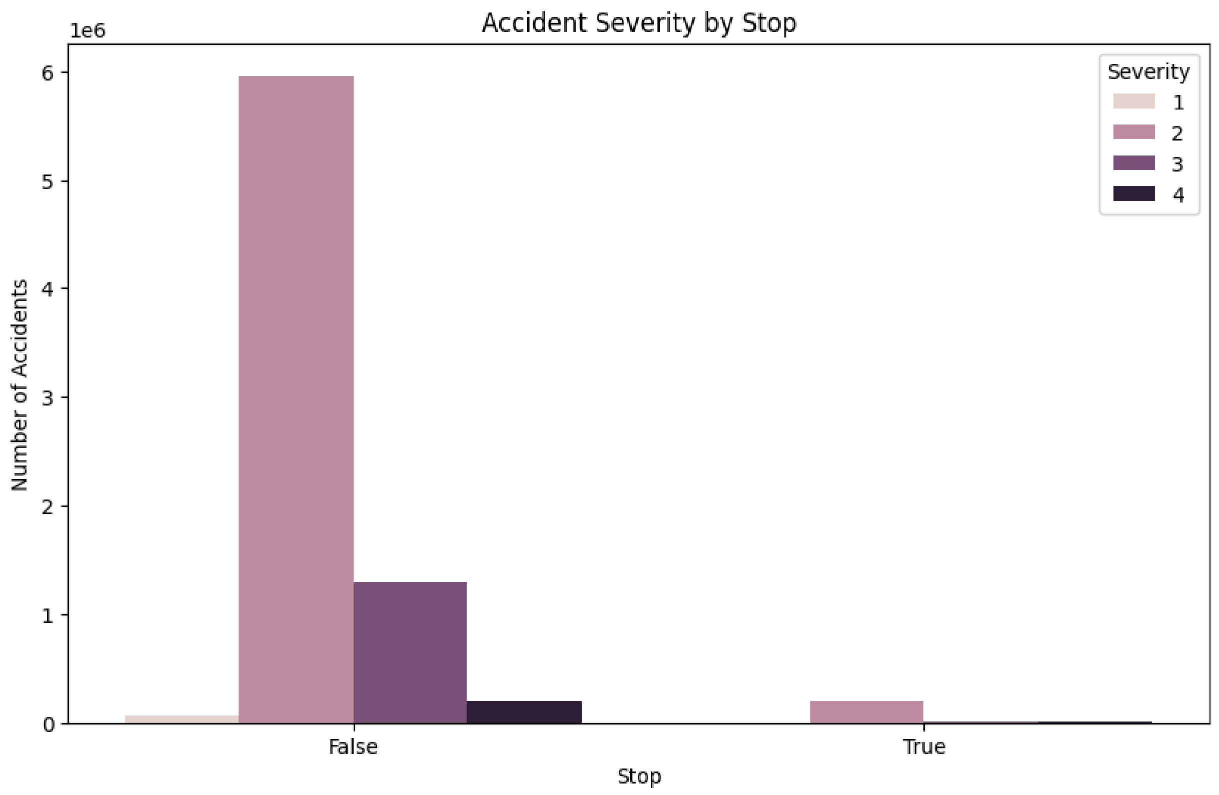


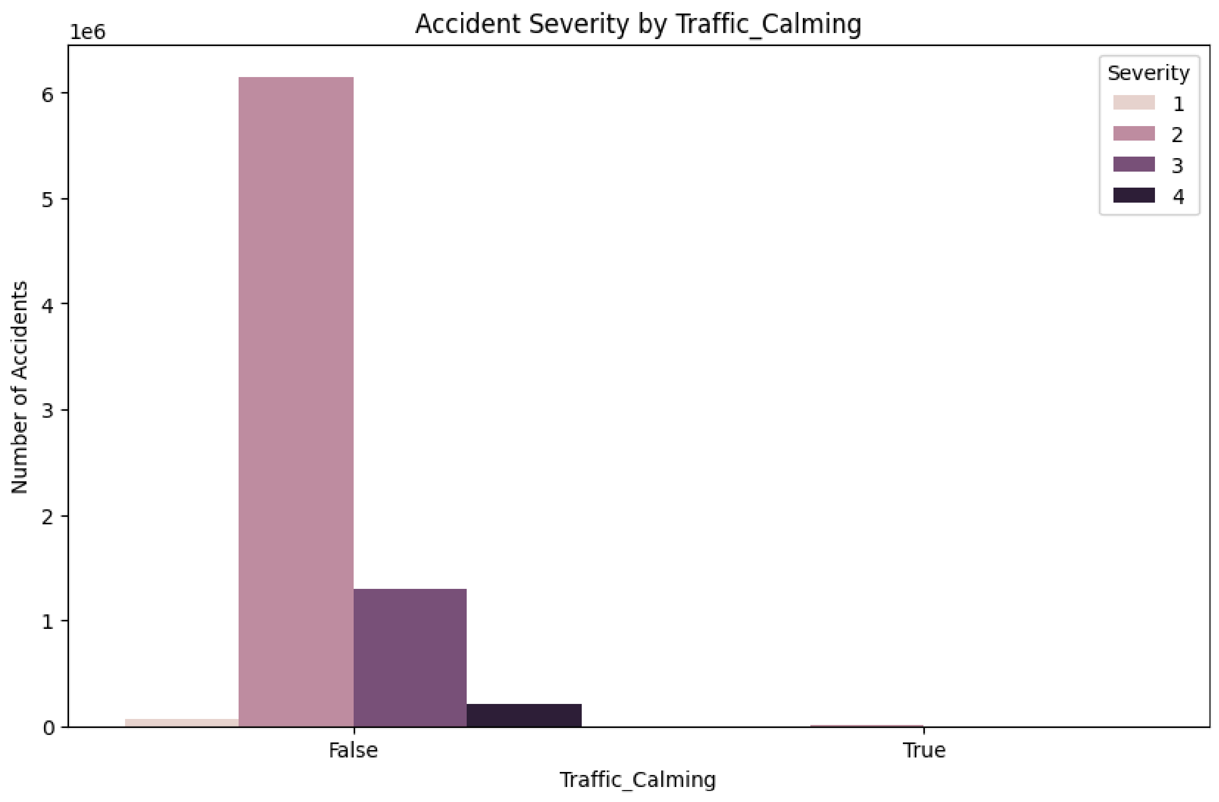
Accident Severity by Railway
1e6
0
1
2
3
4
5
6
Number of Accidents
False
True
Railway
Severity
1
2
3
4

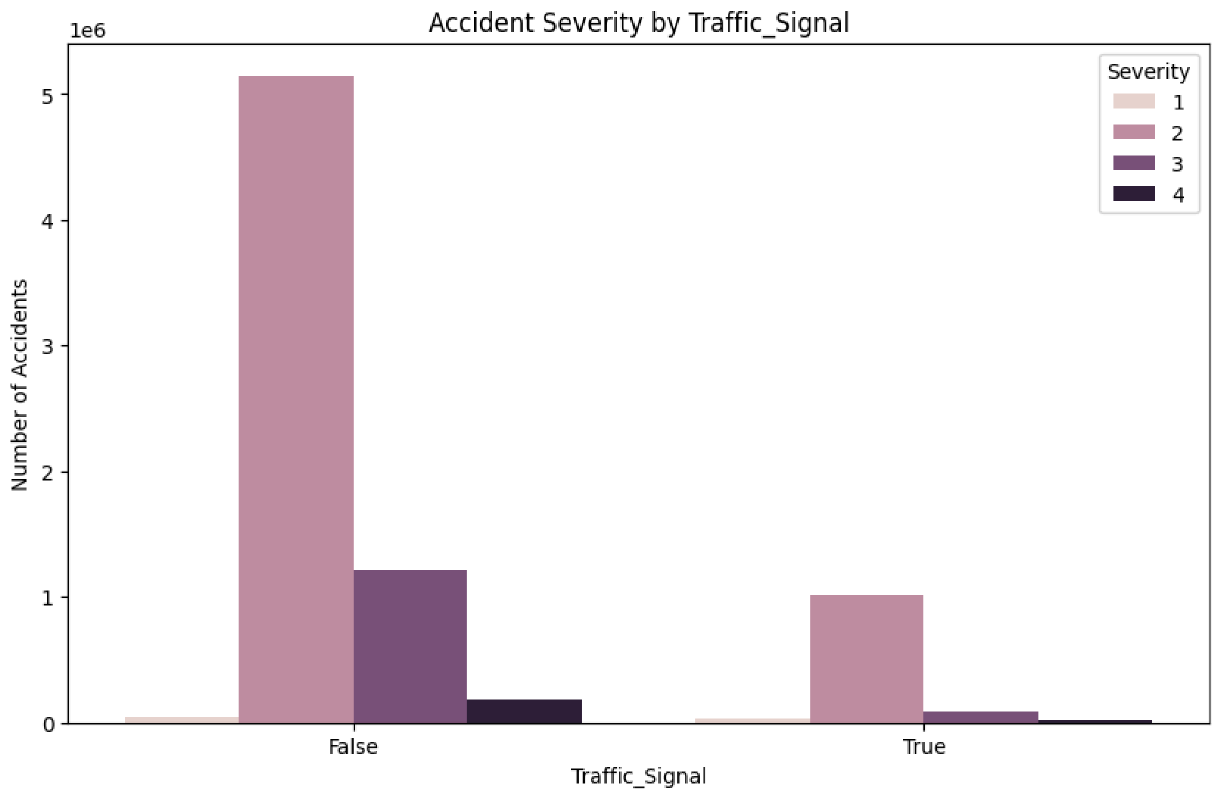


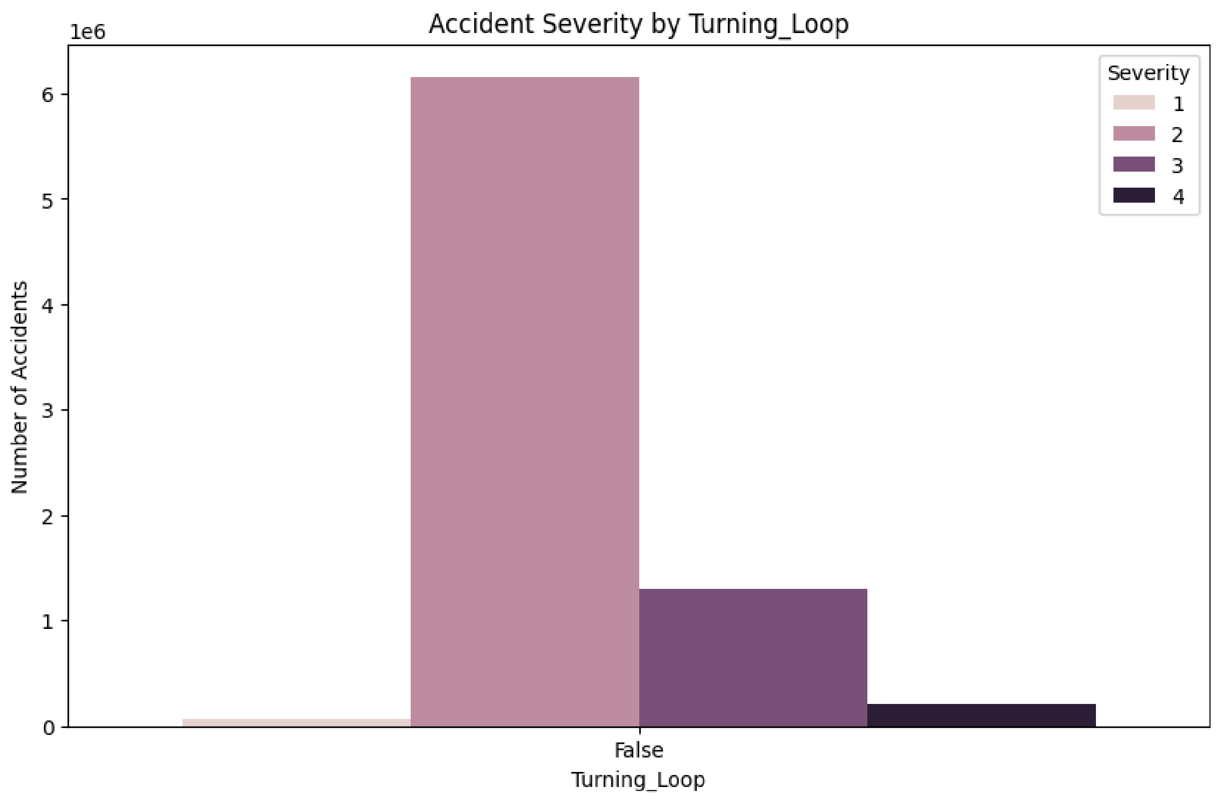
Accident Severity by Roundabout
1e6
0
1
2
3
4
5
6
Number of Accidents
False
True
Roundabout
Severity
1
2
3
4


Accident Severity by Station
1e6
6
5
4
3
2
1
0
Number of Accidents
False
True
Station
Severity
1
2
3
4


Accident Severity by Stop
1e6
6
5
4
3
2
1
0
Number of Accidents
False
True
Stop
Severity
1
2
3
4


Accident Severity by Traffic_Calming
1e6
0
1
2
3
4
5
6
Number of Accidents
False
True
Traffic_Calming
Severity
1
2
3
4


Accident Severity by Traffic_Signal
1e6
0
1
2
3
4
5
Number of Accidents
False
True
Traffic_Signal
Severity
1
2
3
4



In [15]:
```
# Create a base map
m = folium.Map(location=[data['Start_Lat'].mean(), data['Start_Lng'].mean()], zoom_st

# Add accident data to the map
heat_data = [[row['Start_Lat'], row['Start_Lng']] for index, row in data.iterrows() i
HeatMap(heat_data).add_to(m)

# Save the map as an HTML file
m.save('accident_hotspots.html')
```

In [ ]:
```
```