skillcraft-task-4

July 5, 2025

[2]:
```
import zipfile
import pandas as pd
import os

# Unzip the file - Removed as the file is not a valid zip archive
# zip_path = "/content/archive (1).zip"
# extracted_path = "/mnt/data/unzipped_data"

# with zipfile.ZipFile(zip_path, 'r') as zip_ref:
#     zip_ref.extractall(extracted_path)

# Check the extracted files - Removed as no extraction is performed
# extracted_files = os.listdir(extracted_path)
# print("Extracted files:", extracted_files)

# Load CSV (assuming one main CSV file is present)
# for file in extracted_files:
#     if file.endswith(".csv"):
#         data_path = os.path.join(extracted_path, file)
#         df = pd.read_csv(data_path)
#         break

# Assuming the CSV is directly available at this path
data_path = "/content/archive (1).zip" # Assuming the file is a CSV despite the␣
 ↪name

try:
    df = pd.read_csv(data_path)
    print("DataFrame loaded successfully:")
    display(df.head())
except Exception as e:
    print(f"Error loading CSV: {e}")
    print("Please verify if the file is indeed a CSV and provide the correct␣
 ↪path.")
```

```
DataFrame loaded successfully:

   Unnamed: 0 Accident_Index  Location_Easting_OSGR  Location_Northing_OSGR  \
```

```
   Unnamed: 0  Accident_Index  Location_Easting_OSGR  Location_Northing_OSGR  \
0           0  200501BS00001               525680.0                178240.0
1           1  200501BS00002               524170.0                181650.0
2           2  200501BS00003               524520.0                182240.0
3           3  200501BS00004               526900.0                177530.0
4           4  200501BS00005               528060.0                179040.0

   Longitude   Latitude  Police_Force  Accident_Severity  Number_of_Vehicles  \
0  -0.191170  51.489096             1                  2                   1
1  -0.211708  51.520075             1                  3                   1
2  -0.206458  51.525301             1                  3                   2
3  -0.173862  51.482442             1                  3                   1
4  -0.156618  51.495752             1                  3                   1

   Number_of_Casualties  …      Pedestrian_Crossing-Physical_Facilities  \
0                     1  …                                Zebra crossing
1                     1  …  Pedestrian phase at traffic signal junction
2                     1  …        No physical crossing within 50 meters
3                     1  …        No physical crossing within 50 meters
4                     1  …        No physical crossing within 50 meters

                       Light_Conditions          Weather_Conditions  \
0          Daylight: Street light present  Raining without high winds
1  Darkness: Street lights present and lit     Fine without high winds
2  Darkness: Street lights present and lit     Fine without high winds
3          Daylight: Street light present     Fine without high winds
4       Darkness: Street lighting unknown     Fine without high winds

   Road_Surface_Conditions Special_Conditions_at_Site  Carriageway_Hazards  \
0                Wet/Damp                         NaN                  NaN
1                     Dry                         NaN                  NaN
2                     Dry                         NaN                  NaN
3                     Dry                         NaN                  NaN
4                Wet/Damp                         NaN                  NaN

   Urban_or_Rural_Area Did_Police_Officer_Attend_Scene_of_Accident  \
0                    1                                        Yes
1                    1                                        Yes
2                    1                                        Yes
3                    1                                        Yes
4                    1                                        Yes

   LSOA_of_Accident_Location  Year
0                  E01002849  2005
1                  E01002909  2005
2                  E01002857  2005
3                  E01002840  2005
4                  E01002863  2005
```

```
[5 rows x 33 columns]
```

[3]:
```
# Basic Info
df.info()

# Check for null values
df.isnull().sum()

# Preview column names
print(df.columns)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1504150 entries, 0 to 1504149
Data columns (total 33 columns):
 #   Column                                       Non-Null Count    Dtype
---  ------                                       --------------    -----
 0   Unnamed: 0                                   1504150 non-null  int64
 1   Accident_Index                               1504150 non-null  object
 2   Location_Easting_OSGR                        1504049 non-null  float64
 3   Location_Northing_OSGR                       1504150 non-null  float64
 4   Longitude                                    1504049 non-null  float64
 5   Latitude                                     1504150 non-null  float64
 6   Police_Force                                 1504150 non-null  int64
 7   Accident_Severity                            1504150 non-null  int64
 8   Number_of_Vehicles                           1504150 non-null  int64
 9   Number_of_Casualties                         1504150 non-null  int64
 10  Date                                         1504150 non-null  object
 11  Day_of_Week                                  1504150 non-null  int64
 12  Time                                         1504033 non-null  object
 13  Local_Authority_(District)                   1504150 non-null  int64
 14  Local_Authority_(Highway)                    1504150 non-null  object
 15  1st_Road_Class                               1504150 non-null  int64
 16  1st_Road_Number                              1504150 non-null  int64
 17  Road_Type                                    1504150 non-null  object
 18  Speed_limit                                  1504150 non-null  int64
 19  Junction_Control                             901315 non-null   object
 20  2nd_Road_Class                               1504150 non-null  int64
 21  2nd_Road_Number                              1504150 non-null  int64
 22  Pedestrian_Crossing-Human_Control            1504133 non-null  object
 23  Pedestrian_Crossing-Physical_Facilities      1504116 non-null  object
 24  Light_Conditions                             1504150 non-null  object
 25  Weather_Conditions                           1504150 non-null  object
 26  Road_Surface_Conditions                      1504150 non-null  object
 27  Special_Conditions_at_Site                   36582 non-null    object
 28  Carriageway_Hazards                          27250 non-null    object
 29  Urban_or_Rural_Area                          1504150 non-null  int64
 30  Did_Police_Officer_Attend_Scene_of_Accident  1504150 non-null  object
 31  LSOA_of_Accident_Location                    1395912 non-null  object
```

```
 32  Year                                        1504150 non-null  int64
dtypes: float64(4), int64(14), object(15)
memory usage: 378.7+ MB
Index(['Unnamed: 0', 'Accident_Index', 'Location_Easting_OSGR',
       'Location_Northing_OSGR', 'Longitude', 'Latitude', 'Police_Force',
       'Accident_Severity', 'Number_of_Vehicles', 'Number_of_Casualties',
       'Date', 'Day_of_Week', 'Time', 'Local_Authority_(District)',
       'Local_Authority_(Highway)', '1st_Road_Class', '1st_Road_Number',
       'Road_Type', 'Speed_limit', 'Junction_Control', '2nd_Road_Class',
       '2nd_Road_Number', 'Pedestrian_Crossing-Human_Control',
       'Pedestrian_Crossing-Physical_Facilities', 'Light_Conditions',
       'Weather_Conditions', 'Road_Surface_Conditions',
       'Special_Conditions_at_Site', 'Carriageway_Hazards',
       'Urban_or_Rural_Area', 'Did_Police_Officer_Attend_Scene_of_Accident',
       'LSOA_of_Accident_Location', 'Year'],
      dtype='object')
```

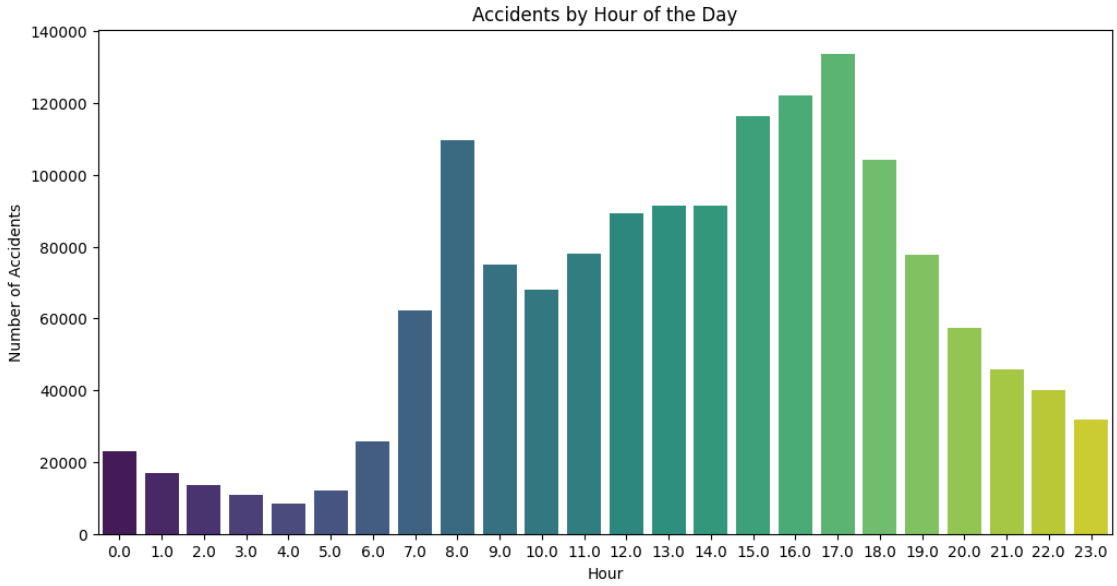
[5]:
```
# Convert time-related column to datetime
df['Time'] = pd.to_datetime(df['Time'], errors='coerce')

# Extract features
df['Hour'] = df['Time'].dt.hour
df['Day'] = df['Time'].dt.day_name()
df['Month'] = df['Time'].dt.month_name()

# Drop rows with nulls in critical columns
df = df.dropna(subset=['Time', 'Weather_Conditions', 'Road_Surface_Conditions'])
```

```
/tmp/ipython-input-5-899517378.py:2: UserWarning: Could not infer format, so
each element will be parsed individually, falling back to `dateutil`. To ensure
parsing is consistent and as-expected, please specify a format.
  df['Time'] = pd.to_datetime(df['Time'], errors='coerce')
```

[6]:
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Hour', palette='viridis')
plt.title('Accidents by Hour of the Day')
plt.xlabel('Hour')
plt.ylabel('Number of Accidents')
plt.show()
```

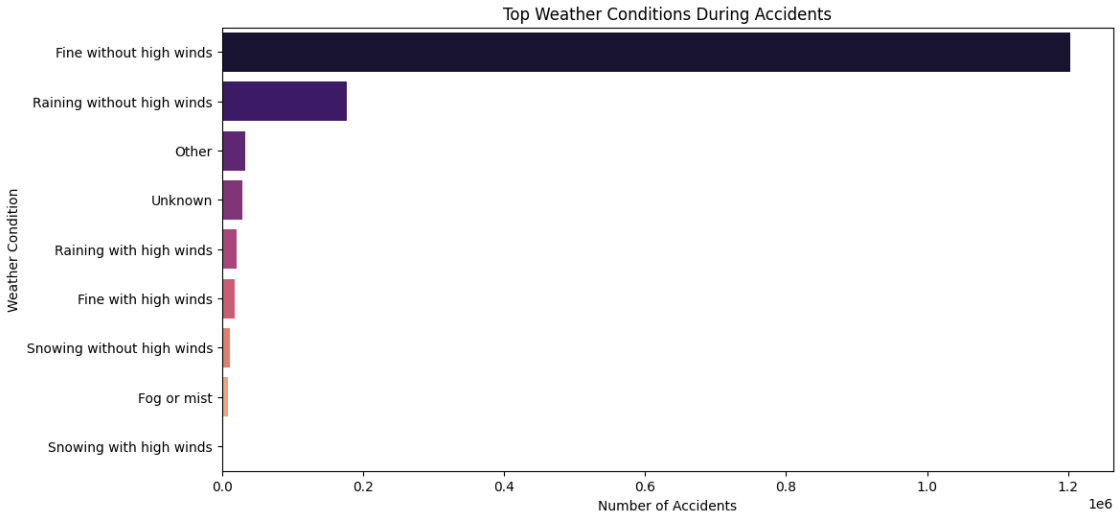
```
/tmp/ipython-input-6-892729279.py:5: FutureWarning:

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

```
  sns.countplot(data=df, x='Hour', palette='viridis')
```



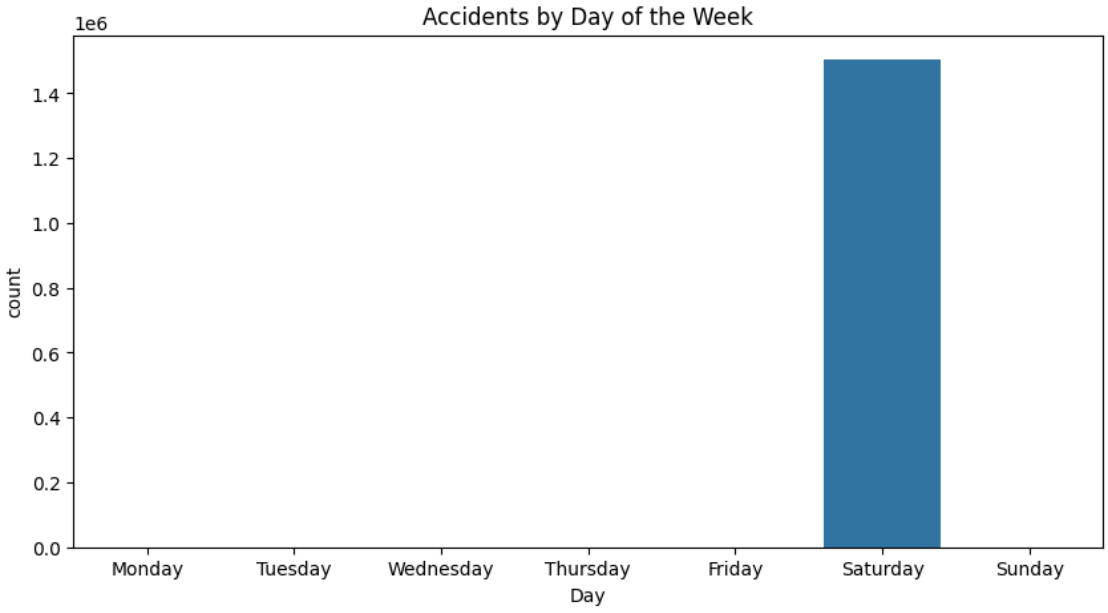
[8]:
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12, 6))
top_weather = df['Weather_Conditions'].value_counts().nlargest(10).index
sns.countplot(data=df[df['Weather_Conditions'].isin(top_weather)],
              y='Weather_Conditions', order=top_weather, palette='magma')
plt.title('Top Weather Conditions During Accidents')
plt.xlabel('Number of Accidents')
plt.ylabel('Weather Condition')
plt.show()
```

```
/tmp/ipython-input-8-1187704211.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.countplot(data=df[df['Weather_Conditions'].isin(top_weather)],
```



[9]:
```
if 'Road_Condition' in df.columns:
    plt.figure(figsize=(10, 5))
    sns.countplot(data=df, y='Road_Condition', palette='coolwarm')
    plt.title('Accidents by Road Condition')
    plt.show()
```

[10]:
```
plt.figure(figsize=(10, 5))
sns.countplot(data=df, x='Day',␣
 ↪order=['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'])
plt.title('Accidents by Day of the Week')
plt.show()
```



[12]:
```
import folium
from folium.plugins import HeatMap

# Filter data with location info
df_map = df[['Latitude', 'Longitude']].dropna().sample(n=1000, random_state=1)
 ↪# Limit for performance

# Create Map
m = folium.Map(location=[df_map['Latitude'].mean(), df_map['Longitude'].
 ↪mean()], zoom_start=6)
HeatMap(df_map.values, radius=10).add_to(m)
m
```

[12]:
```
<folium.folium.Map at 0x7af424a60950>
```