# Hands-on Activity 10.1 Data Analysis using Python

Intended Learning Outcome

Perform descriptive and correlation analysis to to analyze the dataset. Interpret the results of descriptive and correlation analysis Resources

Personal Computer Jupyter Notebook Internet Connection Instruction

1. Gather a dataset regarding your identified problem for the ASEAN Data Science Explorer. Make sure that the dataset includes multiple variables.
2. Load the dataset into pandas dataframe.
3. Prepare the data by applying appropriate data preprocessing techniques.
4. Analyze the data using descriptive analysis.
5. Perform correlation analysis.
6. Interpret the results based on the descriptive and correlation analysis.
7. Submit the PDF file.

importing all the necessary

```
import pandas as pd  # Importing a library called pandas and naming it as 'pd' for easier use.
import numpy as np   # Importing a library called numpy and naming it as 'np'.
import matplotlib as mp  # Importing a library called matplotlib and naming it as 'mp'.
import seaborn as sb   # Importing a library called seaborn and naming it as 'sb'.

d1 = pd.read_csv('dt.csv')  # Reading a CSV file named 'dt.csv' and storing its content in a variable called 'd1'.
d1  # Displaying the content of 'd1', which is the data from the CSV file.
```


```
{"summary":"{\n  \"name\": \"d1\",\n  \"rows\": 36,\n  \"fields\": [\n{\n      \"column\": \"City\",\n      \"properties\": {\n\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n\"samples\": [\n          \"New York\",\n          \"Los Angeles\",\n\"Chicago\"\n        ],\n        \"semantic_type\": \"\",\n\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Vehicle Type\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 3,\n        \"samples\": [\n          \"Car\",\n          \"Bus\",\n          \"Truck\"\n],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n}\n    },\n    {\n      \"column\": \"Weather\",\n
```

\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Rainy\",\n          \"Snowy\",\n          \"Sunny\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Economic Condition\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Stable\",\n          \"Declining\",\n          \"Growing\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Day
Of Week\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 7,\n        \"samples\": [\n
\"Monday\",\n          \"Tuesday\",\n          \"Saturday\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Hour Of Day\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3,\n        \"min\": 8,\n
\"max\": 18,\n        \"num_unique_values\": 9,\n        \"samples\":
[\n          13,\n          12,\n          14\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Speed\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 10,\n
\"min\": 28,\n        \"max\": 65,\n        \"num_unique_values\":
30,\n        \"samples\": [\n          37,\n          54,\n
28\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Is
Peak Hour\",\n      \"properties\": {\n        \"dtype\":
\"boolean\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          false,\n          true\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Random Event Occurred\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          true,\n
false\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Energy Consumption\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8,\n        \"min\": 40,\n
\"max\": 70,\n        \"num_unique_values\": 26,\n        \"samples\":
[\n          65,\n          46\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Traffic Density\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"Medium\",\n          \"High\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"d1"}

```
d2 = pd.read_csv('tr.csv')  # Reading another CSV file named 'tr.csv' and storing its content in a variable called 'd2'.
d2  # Displaying the content of 'd2', which is the data from the new CSV file.
```

{"summary":"{\n  \"name\": \"d2\",\n  \"rows\": 2976,\n  \"fields\":
[\n    {\n      \"column\": \"Time\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 96,\n
\"samples\": [\n          \"8:00:00 PM\",\n          \"7:15:00 PM\",\n
\"6:15:00 PM\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Date\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 8,\n        \"min\": 1,\n        \"max\": 31,\n
\"num_unique_values\": 31,\n        \"samples\": [\n          6,\n
25,\n          2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Day
of the week\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 7,\n        \"samples\":
[\n          \"Tuesday\",\n          \"Wednesday\",\n
\"Sunday\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CarCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 45,\n        \"min\": 6,\n
\"max\": 180,\n        \"num_unique_values\": 172,\n
\"samples\": [\n          82,\n          36,\n          106\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BikeCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 12,\n        \"min\": 0,\n
\"max\": 70,\n        \"num_unique_values\": 71,\n        \"samples\":
[\n          15,\n          0,\n          57\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"BusCount\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 14,\n
\"min\": 0,\n        \"max\": 50,\n        \"num_unique_values\": 51,\
n        \"samples\": [\n          45,\n          37,\n          19\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"TruckCount\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
10,\n        \"min\": 0,\n        \"max\": 40,\n
\"num_unique_values\": 41,\n        \"samples\": [\n          13,\n
21,\n          10\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 60,\n        \"min\": 21,\n        \"max\": 279,\n
\"num_unique_values\": 239,\n        \"samples\": [\n          184,\n
67,\n          72\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Traffic Situation\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          \"normal\",\n          \"high\",\n          \"low\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"d2"}

Concatenating the two dataframes into one

```
# Concatenating the dataframes side by side
concatenated_df = pd.concat([d1, d2], axis=1)
# Storing the combined data from d1 and d2 into a new DataFrame named
'concatenated_df'.

# Writing the concatenated dataframe to a new CSV file without
including row indices
concatenated_df.to_csv('df.csv', index=False)
# Saving the combined data from 'concatenated_df' to a CSV file named
'df.csv', without including row numbers.

df = pd.read_csv('df.csv')
df
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 2976,\n  \"fields\":
[\n    {\n      \"column\": \"City\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"New York\",\n          \"Los Angeles\",\n
\"Chicago\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Vehicle Type\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          \"Car\",\n          \"Bus\",\n          \"Truck\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Weather\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Rainy\",\n          \"Snowy\",\n          \"Sunny\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Economic Condition\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Stable\",\n          \"Declining\",\n          \"Growing\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Day
Of Week\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 7,\n        \"samples\": [\n
\"Monday\",\n          \"Tuesday\",\n          \"Saturday\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Hour Of Day\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3.4142558277233497,\n        \"min\":
8.0,\n        \"max\": 18.0,\n        \"num_unique_values\": 9,\n
\"samples\": [\n          13.0,\n          12.0,\n          14.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Speed\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
10.609369922796082,\n        \"min\": 28.0,\n        \"max\": 65.0,\n
```

```
\"num_unique_values\": 30,\n        \"samples\": [\n          37.0,\n
54.0,\n          28.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Is
Peak Hour\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          false,\n          true\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Random Event Occurred\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          true,\n
false\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Energy Consumption\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8.538986222865436,\n        \"min\":
40.0,\n        \"max\": 70.0,\n        \"num_unique_values\": 26,\n
\"samples\": [\n          65.0,\n          46.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Traffic Density\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Medium\",\n          \"High\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Time\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 96,\n
\"samples\": [\n          \"8:00:00 PM\",\n          \"7:15:00 PM\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Date\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 8,\n
\"min\": 1,\n        \"max\": 31,\n        \"num_unique_values\": 31,\
n        \"samples\": [\n          6,\n          25\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Day of the week\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 7,\n        \"samples\": [\n
\"Tuesday\",\n          \"Wednesday\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"CarCount\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 45,\n
\"min\": 6,\n        \"max\": 180,\n        \"num_unique_values\":
172,\n        \"samples\": [\n          82,\n          36\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"BikeCount\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
12,\n        \"min\": 0,\n        \"max\": 70,\n
\"num_unique_values\": 71,\n        \"samples\": [\n          15,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BusCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 14,\n        \"min\": 0,\n
```

```
\"max\": 50,\n        \"num_unique_values\": 51,\n        \"samples\":
[\n          45,\n          37\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"TruckCount\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 10,\n        \"min\": 0,\n
\"max\": 40,\n        \"num_unique_values\": 41,\n        \"samples\":
[\n          13,\n          21\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Total\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 60,\n        \"min\": 21,\n
\"max\": 279,\n        \"num_unique_values\": 239,\n
\"samples\": [\n          184,\n          67\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Traffic Situation\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"normal\",\n          \"high\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


Checking for missing values to be able to see if needed for cleaning the data

```
df.isnull().sum()  # Check and count the number of missing values
(NaN) in each column of the DataFrame 'df'.
```

```
City                      2940
Vehicle Type              2940
Weather                   2940
Economic Condition        2940
Day Of Week               2940
Hour Of Day               2940
Speed                     2940
Is Peak Hour              2940
Random Event Occurred     2940
Energy Consumption        2940
Traffic Density           2940
Time                         0
Date                         0
Day of the week              0
CarCount                     0
BikeCount                    0
BusCount                     0
TruckCount                   0
Total                        0
Traffic Situation            0
dtype: int64
```

Removing missing values due to there's a row that has none values in it

```python
# Removing missing values
df.dropna(inplace=True)

# Check if there are any remaining missing values
print(df.isnull().sum())
```

```
City                        0
Vehicle Type                0
Weather                     0
Economic Condition          0
Day Of Week                 0
Hour Of Day                 0
Speed                       0
Is Peak Hour                0
Random Event Occurred       0
Energy Consumption          0
Traffic Density             0
Time                        0
Date                        0
Day of the week             0
CarCount                    0
BikeCount                   0
BusCount                    0
TruckCount                  0
Total                       0
Traffic Situation           0
dtype: int64
```

```python
df.dtypes
```

```
City                         object
Vehicle Type                 object
Weather                      object
Economic Condition           object
Day Of Week                  object
Hour Of Day                 float64
Speed                       float64
Is Peak Hour                 object
Random Event Occurred        object
Energy Consumption          float64
Traffic Density              object
Time                         object
Date                          int64
Day of the week              object
CarCount                      int64
BikeCount                     int64
BusCount                      int64
TruckCount                    int64
Total                         int64
```

```
Traffic Situation         object
dtype: object
```

Created a preprocessing data that a dictionary to to convert the intended data

```
def preprocessing(df, conversions):

    for col, dtype in conversions.items():
        if dtype == 'int':
            df[col] = df[col].astype(int)
        elif dtype == 'category':
            df[col] = df[col].astype('category')
        else:
            df[col] = df[col].astype(str)
    return df

#Convert specified columns in the DataFrame 'df' to the desired data
types

conversions = {
    'Hour Of Day': 'int',
    'Day of the week': 'category',
    'Date': 'int'

}

# Call preprocessing function
df_processed = preprocessing(df, conversions)

# Check the data types after conversion
print(df_processed.dtypes)
```

```
City                        object
Vehicle Type                object
Weather                     object
Economic Condition          object
Day Of Week                 object
Hour Of Day                  int64
Speed                      float64
Is Peak Hour                object
Random Event Occurred       object
Energy Consumption         float64
Traffic Density             object
Time                        object
Date                         int64
Day of the week           category
CarCount                     int64
BikeCount                    int64
BusCount                     int64
TruckCount                   int64
```

```
Total                        int64
Traffic Situation           object
dtype: object
```

```
df
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 36,\n  \"fields\": [\n
{\n      \"column\": \"City\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"New York\",\n          \"Los Angeles\",\n
\"Chicago\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Vehicle Type\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          \"Car\",\n          \"Bus\",\n          \"Truck\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Weather\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Rainy\",\n          \"Snowy\",\n          \"Sunny\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Economic Condition\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Stable\",\n          \"Declining\",\n          \"Growing\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Day
Of Week\",\n      \"properties\": {\n        \"dtype\": \"category\",\
n        \"num_unique_values\": 7,\n        \"samples\": [\n
\"Monday\",\n          \"Tuesday\",\n          \"Saturday\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Hour Of Day\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3,\n        \"min\": 8,\n
\"max\": 18,\n        \"num_unique_values\": 9,\n        \"samples\":
[\n          13,\n          12,\n          14\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Speed\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 10.609369922796082,\
n        \"min\": 28.0,\n        \"max\": 65.0,\n
\"num_unique_values\": 30,\n        \"samples\": [\n          37.0,\n
54.0,\n          28.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Is
Peak Hour\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          false,\n          true\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Random Event Occurred\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          true,\n
```

```
false\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Energy Consumption\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 8.538986222865434,\n        \"min\":
40.0,\n        \"max\": 70.0,\n        \"num_unique_values\": 26,\n
\"samples\": [\n          65.0,\n          46.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Traffic Density\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Medium\",\n          \"High\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Time\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 36,\n
\"samples\": [\n          \"8:45:00 AM\",\n          \"3:15:00 AM\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Date\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 10,\n        \"max\": 10,\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          10\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Day of the week\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n
\"Tuesday\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CarCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 38,\n        \"min\": 31,\n
\"max\": 150,\n        \"num_unique_values\": 26,\n
\"samples\": [\n          34\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"BikeCount\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 13,\n        \"min\": 0,\n
\"max\": 39,\n        \"num_unique_values\": 19,\n        \"samples\":
[\n          0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BusCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 16,\n        \"min\": 1,\n
\"max\": 49,\n        \"num_unique_values\": 21,\n        \"samples\":
[\n          4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"TruckCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 5,\n        \"min\": 0,\n
\"max\": 16,\n        \"num_unique_values\": 12,\n        \"samples\":
[\n          11\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 62,\n        \"min\": 39,\n        \"max\": 212,\n
\"num_unique_values\": 28,\n        \"samples\": [\n          49\n
```

```
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Traffic Situation\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n          \"low\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


```
df.describe()  # Generate descriptive statistics for the numeric
columns in the DataFrame 'df'.
```

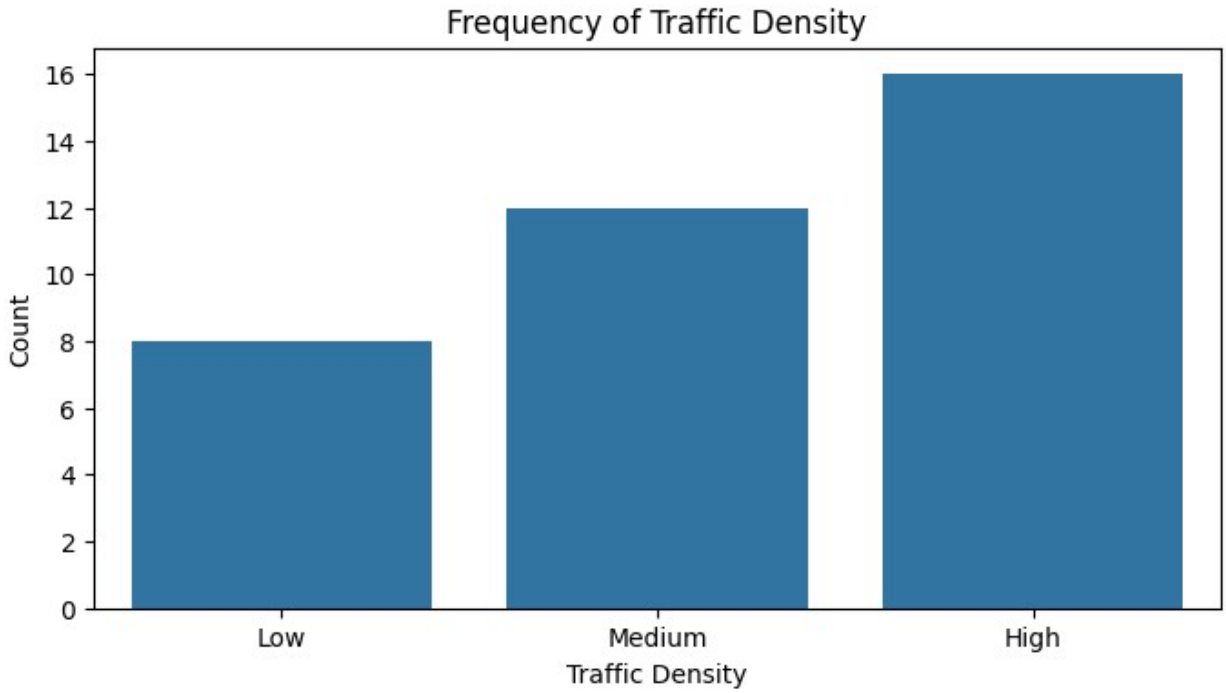

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Hour Of Day\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 9.757002678734931,\n
\"min\": 3.4142558277233497,\n        \"max\": 36.0,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          36.0,\n
13.0,\n          16.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Speed\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 17.09340491616186,\n        \"min\": 10.609369922796082,\n
\"max\": 65.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          47.888888888888886,\n          49.5,\n
36.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Energy Consumption\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 18.318701517957596,\n        \"min\":
8.538986222865434,\n        \"max\": 70.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
52.666666666666664,\n          50.0,\n          36.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Date\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 10.309496315810694,\n
\"min\": 0.0,\n        \"max\": 36.0,\n        \"num_unique_values\":
3,\n        \"samples\": [\n          36.0,\n          10.0,\n
0.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CarCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 42.46062056617112,\n        \"min\":
31.0,\n        \"max\": 150.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          82.38888888888889,\n          67.0,\n
36.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BikeCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 14.81800737819519,\n        \"min\":
0.0,\n        \"max\": 39.0,\n        \"num_unique_values\": 7,\n
\"samples\": [\n          36.0,\n          14.86111111111111,\n
27.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"BusCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 16.608286113053865,\n        \"min\":
1.0,\n        \"max\": 49.0,\n        \"num_unique_values\": 8,\n
```

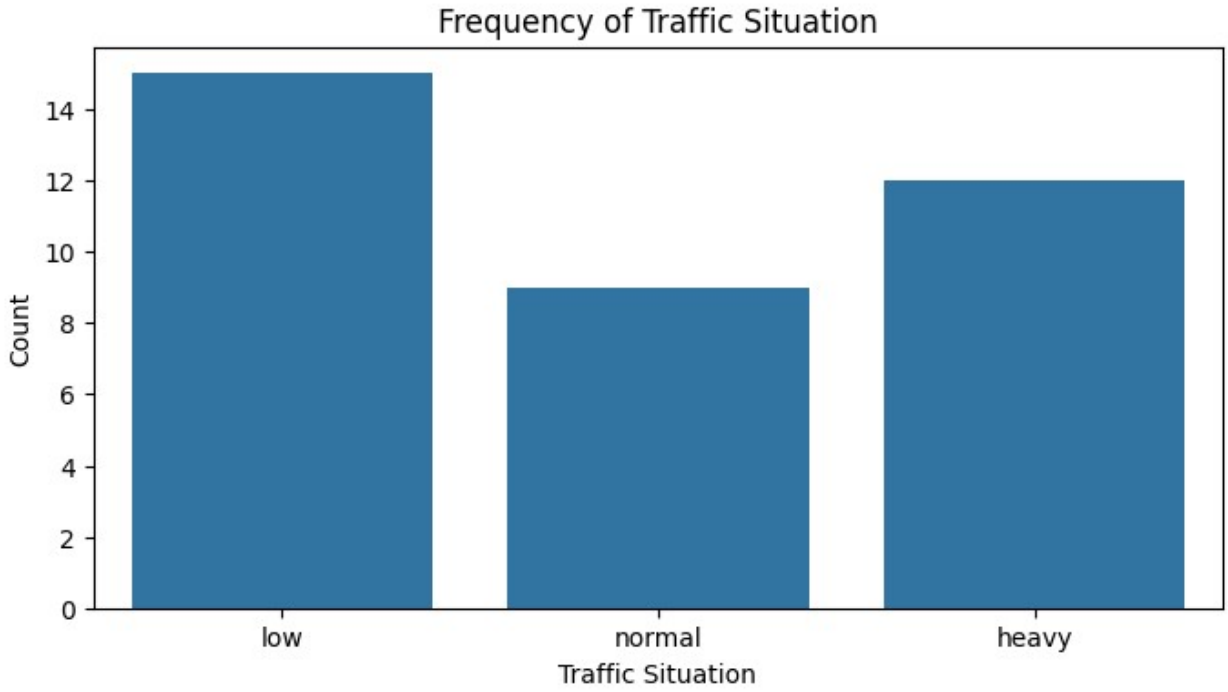
```
\"samples\": [\n          17.305555555555557,\n          8.5,\n
36.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"TruckCount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 11.878851288751378,\n        \"min\":
0.0,\n        \"max\": 36.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          5.111111111111111,\n          3.5,\n
36.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 66.96789077164226,\n        \"min\": 36.0,\n        \"max\":
212.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
119.66666666666667,\n          111.5,\n          36.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```


PLOTTING THE DATA

```
# order of traffic density levels
traffic_density_order = ['Low', 'Medium', 'High']

# order of traffic situation levels
traffic_situation_order = ['low', 'normal', 'heavy']

# Group the data by 'Traffic Density' and 'Traffic Situation', and
count
grouped_data_density = df.groupby('Traffic Density')['Traffic
Density'].count()
grouped_data_situation = df.groupby('Traffic Situation')['Traffic
Situation'].count()

# Plot the bar plot for traffic density
plt.figure(figsize=(8, 4))
sns.barplot(x=grouped_data_density.index,
y=grouped_data_density.values, order=traffic_density_order)
plt.title('Frequency of Traffic Density')
plt.xlabel('Traffic Density')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()

# Plot the bar plot for traffic situation
plt.figure(figsize=(8, 4))
sns.barplot(x=grouped_data_situation.index,
y=grouped_data_situation.values, order=traffic_situation_order)
plt.title('Frequency of Traffic Situation')
plt.xlabel('Traffic Situation')
plt.ylabel('Count')
plt.xticks(rotation=0)
```

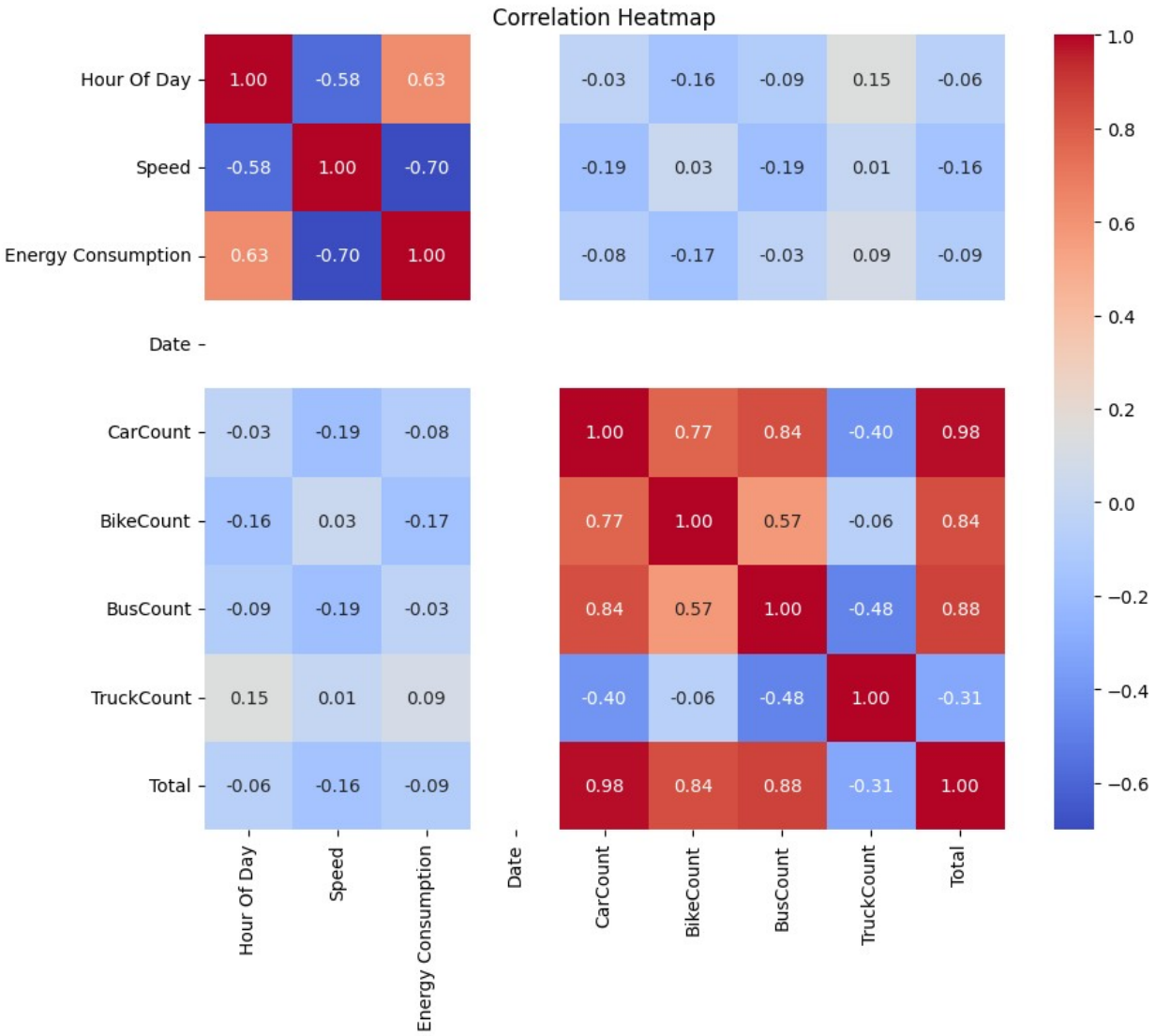
```
plt.show()
```


Frequency of Traffic Density
Count
0
2
4
6
8
10
12
14
16
Low
Medium
High
Traffic Density


Frequency of Traffic Situation
Count
0
2
4
6
8
10
12
14
low
normal
heavy
Traffic Situation

# correlation analysis.

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


numerical_columns = df.select_dtypes(include=['int', 'float']).columns

correlation_matrix = df[numerical_columns].corr()

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f", annot_kws={"size": 10})
plt.title('Correlation Heatmap')
plt.show()
```



Objective: To investigate the relationship between traffic density (number of vehicles on a specific road) and traffic situations (frequency and severity of congestion) in relation to SDG 11, which focuses on creating sustainable cities and communities, particularly in optimizing transportation and reducing road congestion.

Analysis:

Traffic Density and SDG 11:

Observation: Higher traffic density indicates a greater number of vehicles on a specific road, which often leads to congestion and inefficient transportation systems.

Correlation with SDG 11: Improving transportation efficiency and reducing road congestion are essential components of SDG 11. By managing and optimizing traffic density, we can contribute to creating more accessible and sustainable urban areas, aligning with the objectives of SDG 11.

Traffic Situations and SDG 11:

Observation: Frequent and severe traffic situations (congestion) hinder accessibility, increase travel time, and contribute to environmental pollution and inefficiency in transportation systems.

Correlation with SDG 11: Addressing traffic situations effectively is important for achieving the goals of SDG 11. By implementing this measures to reduce congestion and improve traffic flow, we can enhance the accessibility, sustainability, and resilience transportation.

Conclusion:

Analysis suggests a significant correlation between traffic density, traffic situations, and SDG 11. To advance towards sustainable cities and communities as outlined in SDG 11, it is imperative to focus on optimizing transportation systems, managing traffic density, and addressing congestion effectively. By doing so, we can promote accessibility, reduce road congestion, and contribute to the overall sustainability and livability of urban areas.