```
# Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Step 1: datset
from google.colab import files
uploaded = files.upload()  # This will prompt you to upload the CSV file
```

Choose Files 63_Railway…2013-14.csv

- **63_Railway_Key_Statistics_1950-51_to_2013-14.csv**(text/csv) - 2135 bytes, last modified: 12/4/2024 - 100% done

```
data = pd.read_csv(next(iter(uploaded)))
```

```
print("Dataset preview:")
print(data.head(5))
```

```
Dataset preview:
       Year  \
0  1950-51
1  1955-56
2  1960-61
3  1961-62
4  1962-63

   Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)  \
0                                                 44
1                                                 46
2                                                 51
3                                                 52
4                                                 54

   Broad Gauge- Net Load - (Tonnes)  \
0                               489
1                               537
2                               656
3                               657
4                               699

   Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)  \
0                                              1068
1                                              1146
2                                              1354
3                                              1358
4                                              1405

   Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)  \
0                                                 33
1                                                 37
2                                                 40
3                                                 40
4                                                 40

   Metre Gauge-Net Load - (Tonnes)  \
0                              185
1                              246
2                              298
3                              311
4                              323

   Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)
0                                                435
1                                                537
2                                                648
3                                                663
4                                                679
```

```
# Step 3: Data Cleaning
# Check for missing values
print("\nMissing Values:")
print(data.isnull().sum())
```

```

Missing Values:
Year                                                                  0
```

```
Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)    0
Broad Gauge- Net Load - (Tonnes)                                         0
Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)               0
Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)     0
Metre Gauge-Net Load - (Tonnes)                                          0
Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)             0
dtype: int64
```

```
data.dropna(inplace=True) #drop null values
```

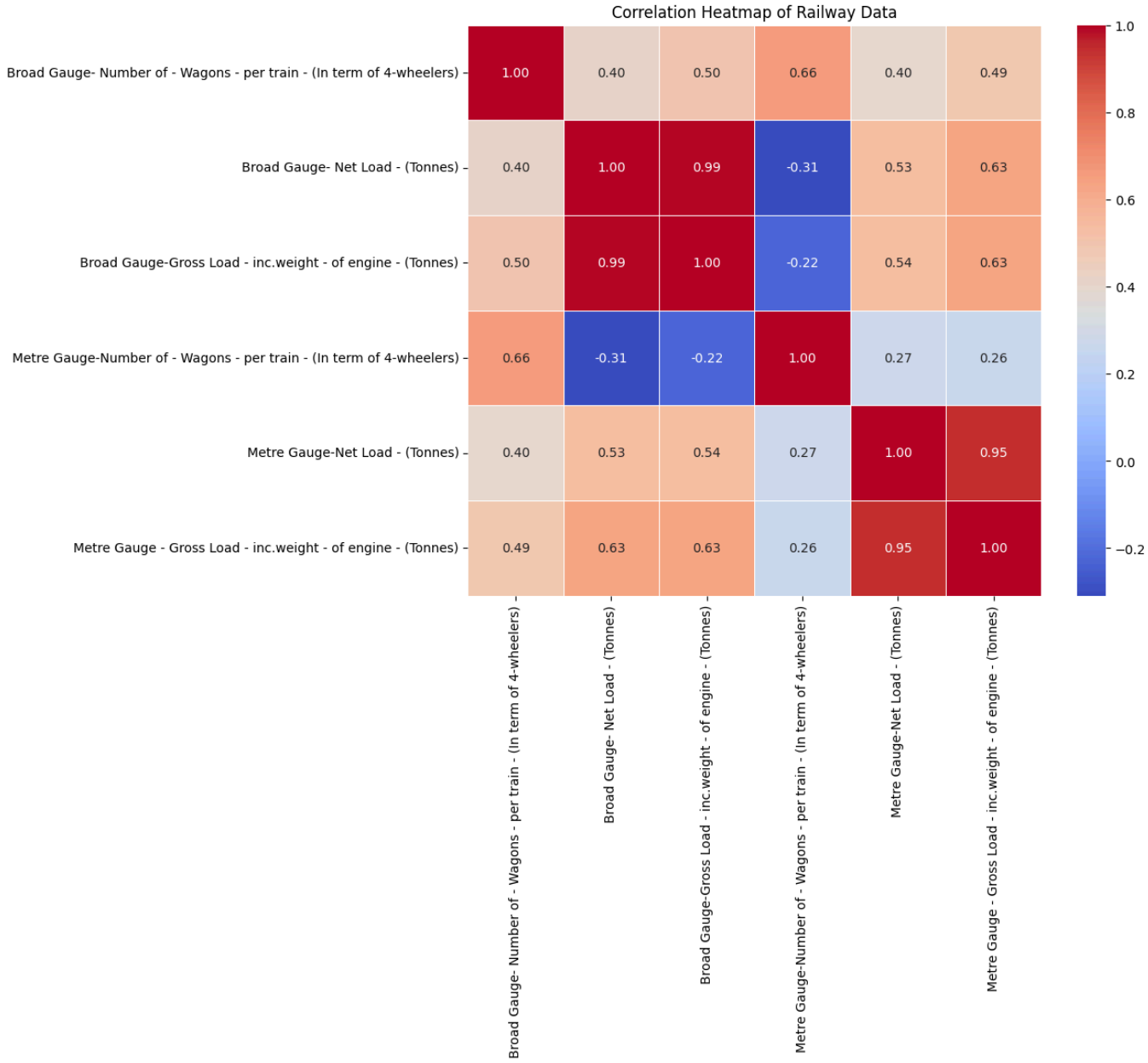
```
# Fill or drop missing values (example: fill with mean for numeric columns)
for column in data.columns:
    if data[column].dtype in ['float64', 'int64']:  # Numeric columns
        data[column].fillna(data[column].mean(), inplace=True)

print("\nAfter Cleaning Missing Values:")
print(data.isnull().sum())
```

```
After Cleaning Missing Values:
Year                                                                     0
Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)    0
Broad Gauge- Net Load - (Tonnes)                                         0
Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)               0
Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)     0
Metre Gauge-Net Load - (Tonnes)                                          0
Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)             0
dtype: int64
<ipython-input-7-b64588d3eb1f>:4: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignme
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  data[column].fillna(data[column].mean(), inplace=True)
```

```
# Inspect the DataFrame
print("DataFrame preview:")
print(data.head())

# Check data types
print("\nData Types:")
print(data.dtypes)

# Select numeric columns
numeric_data = data.select_dtypes(include=['float64', 'int64'])
print("\nNumeric Data preview:")
print(numeric_data.head())

# Check if numeric_data is empty
if numeric_data.empty:
    print("No numeric columns found in the data!")
else:
    # Compute and print correlation matrix
    correlation_matrix = numeric_data.corr()
    print("\nCorrelation Matrix:")
    print(correlation_matrix)
```

```
Metre Gauge-Net Load - (Tonnes)                                                    0.395110
Metre Gauge - Gross Load - inc.weight - of engi...                                 0.490670

                                                        Broad Gauge- Net Load - (Tonnes)  \
Broad Gauge- Number of - Wagons - per train - (...                           0.403431
Broad Gauge- Net Load - (Tonnes)                                             1.000000
Broad Gauge-Gross Load - inc.weight - of engine...                           0.992095
Metre Gauge-Number of - Wagons - per train - (I...                          -0.308417
Metre Gauge-Net Load - (Tonnes)                                              0.534991
Metre Gauge - Gross Load - inc.weight - of engi...                           0.633379

                                                        Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)  \
Broad Gauge- Number of - Wagons - per train - (...                                                 0.498443
Broad Gauge- Net Load - (Tonnes)                                                                   0.992095
Broad Gauge-Gross Load - inc.weight - of engine...                                                 1.000000
Metre Gauge-Number of - Wagons - per train - (I...                                                -0.222475
Metre Gauge-Net Load - (Tonnes)                                                                    0.535961
Metre Gauge - Gross Load - inc.weight - of engi...                                                 0.633140

                                                        Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)  \
Broad Gauge- Number of - Wagons - per train - (...                                                 0.659533
Broad Gauge- Net Load - (Tonnes)                                                                  -0.308417
Broad Gauge-Gross Load - inc.weight - of engine...                                                -0.222475
Metre Gauge-Number of - Wagons - per train - (I...                                                 1.000000
Metre Gauge-Net Load - (Tonnes)                                                                    0.274374
Metre Gauge - Gross Load - inc.weight - of engi...                                                 0.258102

                                                        Metre Gauge-Net Load - (Tonnes)  \
Broad Gauge- Number of - Wagons - per train - (...                           0.395110
Broad Gauge- Net Load - (Tonnes)                                             0.534991
Broad Gauge-Gross Load - inc.weight - of engine...                           0.535961
Metre Gauge-Number of - Wagons - per train - (I...                           0.274374
Metre Gauge-Net Load - (Tonnes)                                              1.000000
Metre Gauge - Gross Load - inc.weight - of engi...                           0.946109

                                                        Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)
Broad Gauge- Number of - Wagons - per train - (...                                                 0.490670
Broad Gauge- Net Load - (Tonnes)                                                                   0.633379
Broad Gauge-Gross Load - inc.weight - of engine...                                                 0.633140
Metre Gauge-Number of - Wagons - per train - (I...                                                 0.258102
Metre Gauge-Net Load - (Tonnes)                                                                    0.946109
Metre Gauge - Gross Load - inc.weight - of engi...                                                 1.000000
```

```
# Step 5: Visualize the Correlation Matrix with a Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap of Railway Data")
plt.show()
```



column name debug

```
# Check the column names in your dataset
print("Column Names in Dataset:")
print(data.columns)

# Remove extra spaces or formatting issues in column names
data.columns = data.columns.str.strip()  # Strip leading/trailing spaces
data.columns = data.columns.str.replace("\s+", " ", regex=True)  # Normalize spaces

# Display cleaned column names
print("\nCleaned Column Names:")
```

```
print(data.columns)

# Verify column names before proceeding with analysis
required_columns = [
    'Broad Gauge-Net Load - (Tonnes)',
    'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)',
    'Metre Gauge-Net Load - (Tonnes)',
    'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)'
]

# Ensure all required columns exist
for col in required_columns:
    if col not in data.columns:
        print(f"Column Missing: {col}")

# Proceed if columns exist
try:
    # Aggregated metrics
    broad_net_load = data['Broad Gauge-Net Load - (Tonnes)'].mean()
    broad_gross_load = data['Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)'].mean()
    metre_net_load = data['Metre Gauge-Net Load - (Tonnes)'].mean()
    metre_gross_load = data['Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)'].mean()

    # Create a DataFrame for aggregated loads
    aggregated_data = pd.DataFrame({
        'Gauge': ['Broad', 'Metre'],
        'Average Net Load': [broad_net_load, metre_net_load],
        'Average Gross Load': [broad_gross_load, metre_gross_load]
    })

    print("\nAggregated Data for Visualization:")
    print(aggregated_data)

    # Bar chart to compare net and gross loads
    plt.figure(figsize=(10, 6))
    sns.barplot(data=aggregated_data.melt(id_vars='Gauge',
                                          var_name='Load Type',
                                          value_name='Load'),
                x='Gauge', y='Load', hue='Load Type', palette='viridis')
    plt.title("Comparison of Net and Gross Loads between Broad and Metre Gauges")
    plt.ylabel("Load (Tonnes)")
    plt.tight_layout()
    plt.show()

except KeyError as e:
    print(f"KeyError: {e}. Please verify the column names in your dataset.")
```

```
Column Names in Dataset:
Index(['Year',
       'Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)',
       'Broad Gauge- Net Load - (Tonnes)',
       'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)',
       'Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)',
       'Metre Gauge-Net Load - (Tonnes)',
       'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)'],
      dtype='object')

Cleaned Column Names:
Index(['Year',
       'Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)',
       'Broad Gauge- Net Load - (Tonnes)',
       'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)',
       'Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)',
       'Metre Gauge-Net Load - (Tonnes)',
       'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)'],
      dtype='object')
Column Missing: Broad Gauge-Net Load - (Tonnes)
KeyError: 'Broad Gauge-Net Load - (Tonnes)'. Please verify the column names in your dataset.
```

```
#debug 1
# Print column names with their exact lengths
print("Column Names and Their Lengths:")
for col in data.columns:
    print(f"'{col}' - Length: {len(col)}")

# Check for differences using list comprehension
print("\nDifferences Detected in Column Names:")
for col in data.columns:
```

```
    if "Broad Gauge-Net Load - (Tonnes)" in col:
        print(f"Matched: '{col}'")
    else:
        print(f"Unmatched: '{col}'")
```

```
Column Names and Their Lengths:
'Year' - Length: 4
'Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)' - Length: 69
'Broad Gauge- Net Load - (Tonnes)' - Length: 32
'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)' - Length: 58
'Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)' - Length: 68
'Metre Gauge-Net Load - (Tonnes)' - Length: 31
'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)' - Length: 60

Differences Detected in Column Names:
Unmatched: 'Year'
Unmatched: 'Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)'
Unmatched: 'Broad Gauge- Net Load - (Tonnes)'
Unmatched: 'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)'
Unmatched: 'Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)'
Unmatched: 'Metre Gauge-Net Load - (Tonnes)'
Unmatched: 'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)'
```

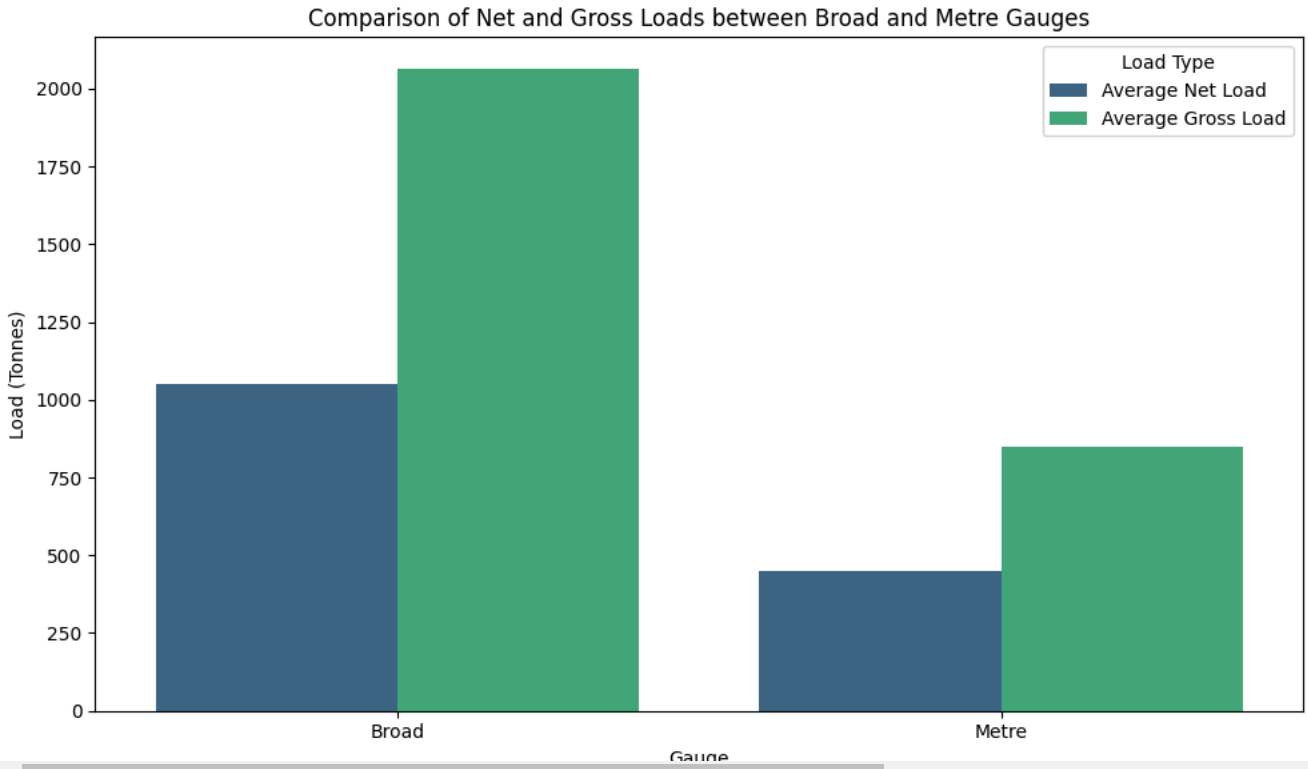
```
#debug2
#After identifying the exact column names, rename them for consistency:
# Rename columns manually to fix any discrepancies
data.rename(columns={
    'Broad Gauge- Net Load - (Tonnes)': 'Broad Gauge-Net Load - (Tonnes)',
    'Broad Gauge-Gross Load - inc.weight - of engine - (Tonnes)': 'Broad Gauge-Gross Load',
    'Metre Gauge-Net Load - (Tonnes)': 'Metre Gauge-Net Load',
    'Metre Gauge - Gross Load - inc.weight - of engine - (Tonnes)': 'Metre Gauge-Gross Load'
}, inplace=True)

print("\nRenamed Columns:")
print(data.columns)
```

```
Renamed Columns:
Index(['Year',
       'Broad Gauge- Number of - Wagons - per train - (In term of 4-wheelers)',
       'Broad Gauge-Net Load - (Tonnes)', 'Broad Gauge-Gross Load',
       'Metre Gauge-Number of - Wagons - per train - (In term of 4-wheelers)',
       'Metre Gauge-Net Load', 'Metre Gauge-Gross Load'],
      dtype='object')
```

Double-click (or enter) to edit

Start coding or generate with AI.

```
#debug3
#final/validaation
# Aggregated metrics
broad_net_load = data['Broad Gauge-Net Load - (Tonnes)'].mean()
broad_gross_load = data['Broad Gauge-Gross Load'].mean()
metre_net_load = data['Metre Gauge-Net Load'].mean()
metre_gross_load = data['Metre Gauge-Gross Load'].mean()

# Create a DataFrame for aggregated loads
aggregated_data = pd.DataFrame({
    'Gauge': ['Broad', 'Metre'],
    'Average Net Load': [broad_net_load, metre_net_load],
    'Average Gross Load': [broad_gross_load, metre_gross_load]
})

# Bar chart to compare net and gross loads
plt.figure(figsize=(10, 6))
sns.barplot(data=aggregated_data.melt(id_vars='Gauge',
                                      var_name='Load Type',
                                      value_name='Load'),
            x='Gauge', y='Load', hue='Load Type', palette='viridis')
plt.title("Comparison of Net and Gross Loads between Broad and Metre Gauges")
plt.ylabel("Load (Tonnes)")
plt.tight_layout()
plt.show()
```



Column Length Debugging: Detect subtle differences in column names. Manual Renaming: Standardize column names using data.rename(). Invisible Characters: Watch for characters like \u00a0 (non-breaking spaces) or extra tabs.

```
#Trend Load over years
plt.figure(figsize=(10, 6))
sns.lineplot(data=data, x='Year', y='Broad Gauge-Net Load - (Tonnes)', label='Broad Gauge - Net Load', color='blue')
sns.lineplot(data=data, x='Year', y='Metre Gauge-Net Load - (Tonnes)', label='Metre Gauge - Net Load', color='orange')
sns.lineplot(data=data, x='Year', y='Broad Gauge-Gross Load', label='Broad Gauge - Gross Load', color='blue', linestyle="--")
sns.lineplot(data=data, x='Year', y='Metre Gauge-Gross Load', label='Metre Gauge - Gross Load', color='orange', linestyle="--")
plt.title("Net and Gross Load Trends Over Years")
plt.xlabel("Year")
plt.ylabel("Load (Tonnes)")
plt.legend()
plt.grid(True)
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-14-d325a142a150> in <cell line: 4>()
      2 plt.figure(figsize=(10, 6))
      3 sns.lineplot(data=data, x='Year', y='Broad Gauge-Net Load - (Tonnes)', label='Broad Gauge - Net Load', color='blue')
----> 4 sns.lineplot(data=data, x='Year', y='Metre Gauge-Net Load - (Tonnes)', label='Metre Gauge - Net Load', color='orange')
      5 sns.lineplot(data=data, x='Year', y='Broad Gauge-Gross Load', label='Broad Gauge - Gross Load', color='blue', linestyle="--")
      6 sns.lineplot(data=data, x='Year', y='Metre Gauge-Gross Load', label='Metre Gauge - Gross Load', color='orange', linestyle="--")

                                    5 frames
/usr/local/lib/python3.10/dist-packages/seaborn/_core/data.py in _assign_variables(self, data, variables)
    230                 else:
    231                     err += "An entry with this name does not appear in `data`."
--> 232                 raise ValueError(err)
    233
    234             else:

ValueError: Could not interpret value `Metre Gauge-Net Load - (Tonnes)` for `y`. An entry with this name does not appear in `data`.
```

Next Explain error

```
# Plot for Broad Gauge: Net and Gross Load over years
plt.figure(figsize=(10, 6))
sns.lineplot(data=data, x='Year', y='Broad Gauge-Net Load - (Tonnes)', label='Broad Gauge - Net Load', color='blue', marker='o')
sns.lineplot(data=data, x='Year', y='Broad Gauge-Gross Load', label='Broad Gauge - Gross Load', color='blue', linestyle="--", marker='o')
plt.title("Broad Gauge - Net and Gross Load Trends Over Years")
plt.xlabel("Year")
plt.ylabel("Load (Tonnes)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```