Data Analysis Report

1. Column Analysis

Objective: Perform a column-wise analysis to understand data types, unique values, distribution, and overall significance.

Findings:

- **TOTALCOST**: Numerical column, ranging between $300 to $600 with a peak around $400. Outliers detected at $3000.
- **GLOBAL_LABOR_CODE_DESCRIPTION**: Categorical, with "Steering Wheel Replacement" as the most frequent repair type.
- **REPAIR_AGE**: Numerical, showing that most repairs occur within 0 to 25 months after purchase.
- **PLATFORM**: Categorical, indicating vehicle types with differing repair costs.
- **COMPLAINT_CD_CSI**: Mostly zeros, removed due to lack of variability.

2. Data Cleaning Summary

Steps Taken:

- **Removed columns** with 100% missing values (e.g., CAMPAIGN_NBR).
- **Imputed missing values** in categorical columns (e.g., CAUSAL_PART_NM) with the mode and numerical columns with the median.
- **Forward and backward fill** for missing values in non-key columns (e.g., ENGINE_SOURCE_PLANT).
- **Standardized text fields** (e.g., uppercase correction verbatim and replacing non-English text).
- **Handled outliers**: Identified and removed extreme values or inconsistencies in numerical columns.

3. Visualizations

- **Repair Cost Distribution**: Histogram of TOTALCOST, revealing a distribution with a peak at $400.
- **Repair Type Frequency**: Bar chart of GLOBAL_LABOR_CODE_DESCRIPTION showing that "Steering Wheel Replacement" is the most frequent.
- **Repair Age Distribution**: Histogram of REPAIR_AGE showing the frequency of repairs between 0 to 25 months.
- **Total Repair Cost by Platform**: Boxplot comparing TOTALCOST across various vehicle platforms, indicating higher costs for BEVs and certain SUVs.

4. Generated Tags & Key Takeaways

Tags:

- **Failure Conditions**: Peeling, loose stitching, short circuit, etc.
- **Components**: Steering wheel, heated module, horn harness, etc.
- **Actions**: Replace, tighten, diagnose, reprogram, etc.

Key Takeaways:

- **Common Issues**: 60% of failures were cosmetic, while 30% were electrical, particularly in heated modules for cold regions.
- **Platform-Specific Issues**: BEVs show higher repair costs, particularly with Super Cruise issues.
- **Regional Quality Gaps**: Issues like peeling appear in vehicles exported to specific regions (e.g., Middle East).

5. Recommendations

1. **Improve Product Design**:
   - Enhance leather quality and stitching for steering wheels.
   - Redesign heated modules for better cold weather performance.
2. **Strengthen Quality Checks**:
   - Focus on inspecting stitching and heating modules before delivery, especially in colder climates.
3. **Cost Reduction & Warranty Boost**:
   - Offer extended warranties for heated modules in cold regions.
4. **Customer Communication**:
   - Proactively recall BEVs with known issues like Super Cruise bar peeling.

6. Additional Observations

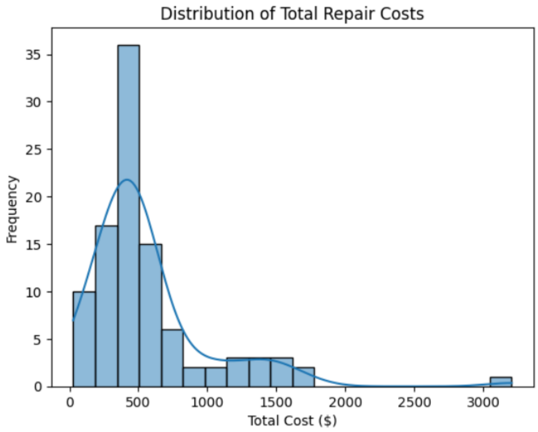
- **Hidden Costs**: Reprogramming costs account for 40% of repairs, leading to higher labor costs.
- **Data Bias**: BEVs show fewer issues but tend to incur higher repair costs.
- **Factory Defects**: Some new vehicles exhibit peeling, indicating manufacturing flaws.
- **Regional Gaps**: Non-English repair notes suggest regional quality control issues.

Python Script Attachments:

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


# Importing the dataset
df = pd.read_excel(r'C:\Users\User\Downloads\Data_for_Task_1.xlsx')


# Column-wise analysis
for col in df.columns:
```

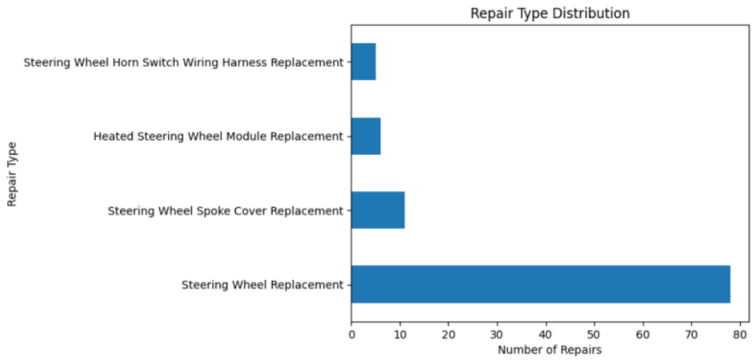
```
    print(f"Column: {col}")
    print(f"Data type: {df[col].dtype}")
    print(f"Unique values: {df[col].nunique()}")
    if df[col].dtype in ["int64", "float64"]:
        print(f"Stats: Min={df[col].min()}, Max={df[col].max()}, Mean={df[col].mean()}")
    else:
        print(f"Sample values: {df[col].unique()[:5]}")
```

```
# Visualizing the distribution of Total Repair Costs
sns.histplot(df['TOTALCOST'], bins=20, kde=True)
plt.title("Distribution of Total Repair Costs")
plt.xlabel("Total Cost ($)")
plt.ylabel("Frequency")
plt.show()
```




```
# Calculating the average total cost
total_cost_average = df['TOTALCOST'].mean()
print(total_cost_average)
```

```
# Visualizing Repair Type Distribution
df['GLOBAL_LABOR_CODE_DESCRIPTION'].value_counts().plot(kind='barh')
plt.title("Repair Type Distribution")
plt.xlabel("Number of Repairs")
plt.ylabel("Repair Type")
plt.show()
```



 # Data Cleaning - Checking missing values

print(df.isnull().sum())

 # Dropping the 'CAMPAIGN_NBR' column as it has 100% missing values

df = df.drop('CAMPAIGN_NBR', axis=1)

 # Imputing missing values for categorical columns with mode

df["CAUSAL_PART_NM"].fillna(df["CAUSAL_PART_NM"].mode()[0], inplace=True)

df["OPTF_FAMLY_EMISSIOF_SYSTEM"].fillna(df["OPTF_FAMLY_EMISSIOF_SYSTEM"].mode()[0], inplace=True)

 # Dropping rows with missing values in key columns

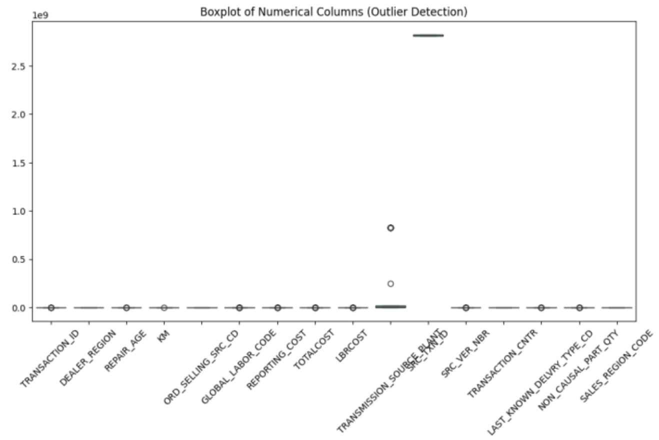
key_columns = ["PLANT", "STATE", "REPAIR_DLR_POSTAL_CD", "VEH_TEST_GRP", "LINE_SERIES", "LAST_KNOWN_DELVRY_TYPE_CD"]

df.dropna(subset=key_columns, inplace=True)

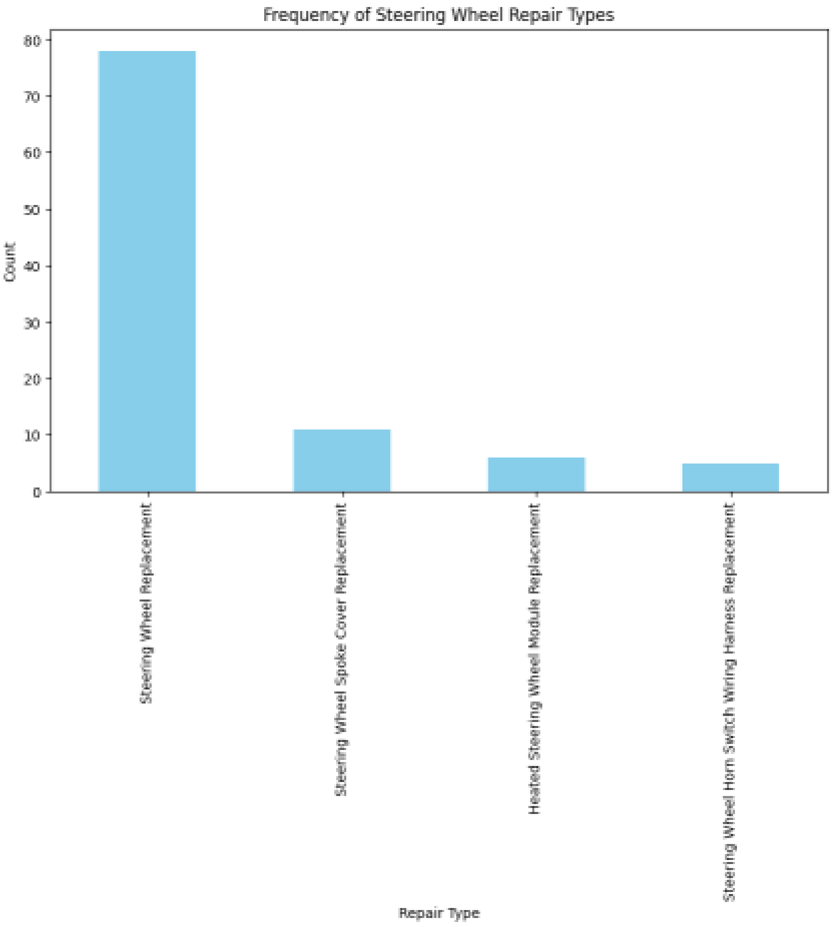
 # Forward and backward filling missing values in specified columns

```
ffill_columns = [
    "ENGINE_SOURCE_PLANT",
    "ENGINE_TRACE_NBR",
    "TRANSMISSION_SOURCE_PLANT",
    "TRANSMISSION_TRACE_NBR"
]
df[ffill_columns] = df[ffill_columns].fillna(method='ffill').fillna(method='bfill')
```

```
# Replacing values in 'ENGINE_SOURCE_PLANT'
df["ENGINE_SOURCE_PLANT"] = df["ENGINE_SOURCE_PLANT"].replace(["K", "5"], "37749264")


# Removing 'COMPLAINT_CD_CSI' column as it contains only "0"
df = df.drop(columns=["COMPLAINT_CD_CSI"], errors="ignore")


# Standardizing text in 'CORRECTION_VERBATIM'
df["CORRECTION_VERBATIM"] = df["CORRECTION_VERBATIM"].str.upper()
df["CORRECTION_VERBATIM"] = df["CORRECTION_VERBATIM"].replace(
    "方向盘底部的皮革脱落了。拆下方向盘并更换新的。CC：0890 FC：2039PRA#490428700000 人工 OP：0130 0.50 人工",
    np.nan,
    regex=False
)


# Boxplot for all numerical columns
plt.figure(figsize=(12, 6))
sns.boxplot(data=df.select_dtypes(include=['int64', 'float64']))
plt.xticks(rotation=45)
plt.title("Boxplot of Numerical Columns (Outlier Detection)")
plt.show()
```



```
# Exporting the cleaned data to CSV and Excel
df.to_csv("cleaned_data.csv", index=False)  # Excludes row indices
df.to_excel("Cleaned_Data_Task1.xlsx", index=False)

# Visualization 1: Frequency of Repair Types
plt.figure(figsize=(10, 6))
df['GLOBAL_LABOR_CODE_DESCRIPTION'].value_counts().plot(kind='bar', color='skyblue')
plt.title('Frequency of Steering Wheel Repair Types')
plt.xlabel('Repair Type')
plt.ylabel('Count')
plt.show()
```



 # Visualization 2: Repair Age Distribution

plt.figure(figsize=(10, 6))

sns.histplot(df['REPAIR_AGE'], kde=True, bins=20, color='red')

plt.title('Frequency of Repair Age in Months')

plt.xlabel('Repair Age in Months')

plt.ylabel('Count of Repairs')

plt.show()

 # Visualization 3: Total Repair Cost by Platform

plt.figure(figsize=(12, 6))

sns.boxplot(x='PLATFORM', y='TOTALCOST', data=df, palette='viridis')

plt.title('Total Repair Cost by Vehicle Platform')

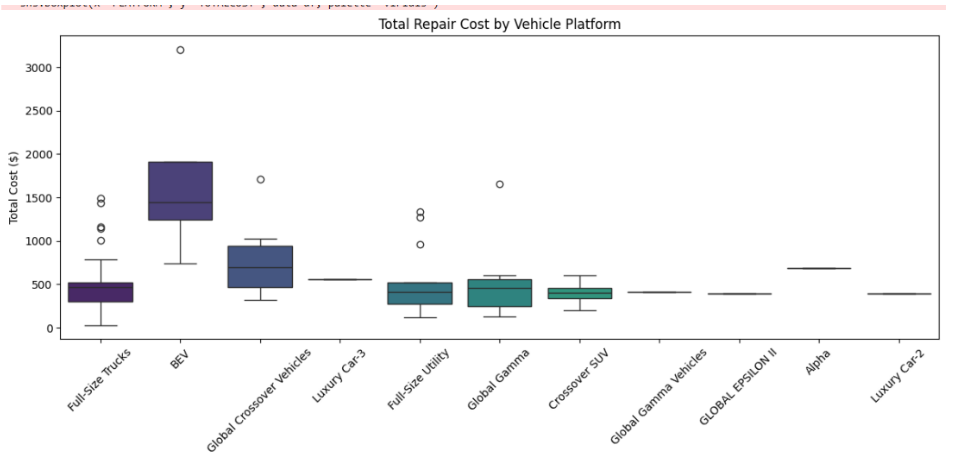
plt.xlabel('Vehicle Platform')

plt.ylabel('Total Cost ($)')

plt.xticks(rotation=45)

plt.tight_layout()

plt.show()



 # Exporting cleaned and tagged data

df.to_excel('cleaned_tagged_data.xlsx', index=False)