RHOMBIX TECHNOLOGIES

Task (Month-2) Submission

Mohammad Jazib Khan (Data Scientist)

Jk537605@gmail.com

Documentation for Electric Vehicle Population Data Processing

Overview

This script processes the Electric Vehicle Population dataset to prepare it for machine learning applications. It handles missing values, removes outliers, normalizes features, and splits the data into training and testing sets.

Dependencies

The script requires the following Python libraries:

- **pandas**: For data manipulation and analysis.
- **numpy**: For numerical operations.
- **scikit-learn**: For machine learning utilities, including data splitting and scaling.

Dataset

The dataset is assumed to be in CSV format and contains various features related to electric vehicles, such as:

- Country
- City
- Postal Code
- Model Year
- Make
- Model
- Electric Range
- Base MSRP
- Legislative District
- Electric Utility

Note

Ensure that the dataset file **Electric_Vehicle_Population_Data.csv** is available in the same directory as the script or provide the correct path to the file.

Steps Implemented

1. Load the Dataset

The dataset is loaded using "**pandas.read_csv()"**.

2. Handle Missing Values

- The script checks for missing values in the dataset.
- Unnecessary columns (**Unnamed: 3** and **Unnamed: 14**) are dropped.
- Missing values in categorical columns (**Country**, **City**, **Postal Code**, **Legislative District**, and **Electric Utility**) are filled with the mode of each column.

3. Handle Outliers

- Outliers in the **Electric Range** and **Base MSRP** columns are detected using the Interquartile Range (IQR) method.
- Rows containing outliers are removed from the dataset.

4. Normalize or Scale Features

- The **Electric Range** and **Base MSRP** features are normalized using **StandardScaler** to standardize the data, which helps improve model performance.

5. Split the Data

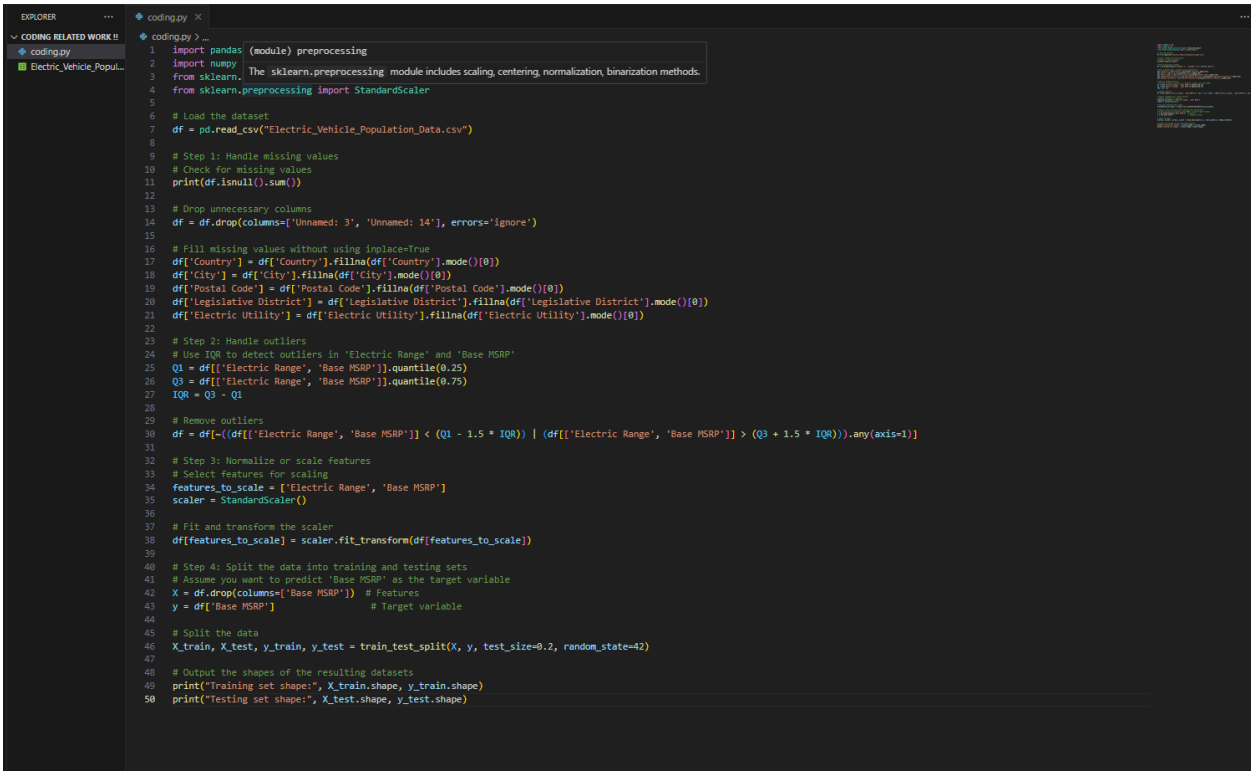
- The processed dataset is split into training and testing sets using an 80-20 split ratio.
- Features (X) and target variable (y) are defined, with **Base MSRP** as the target variable.

6. Output Shapes

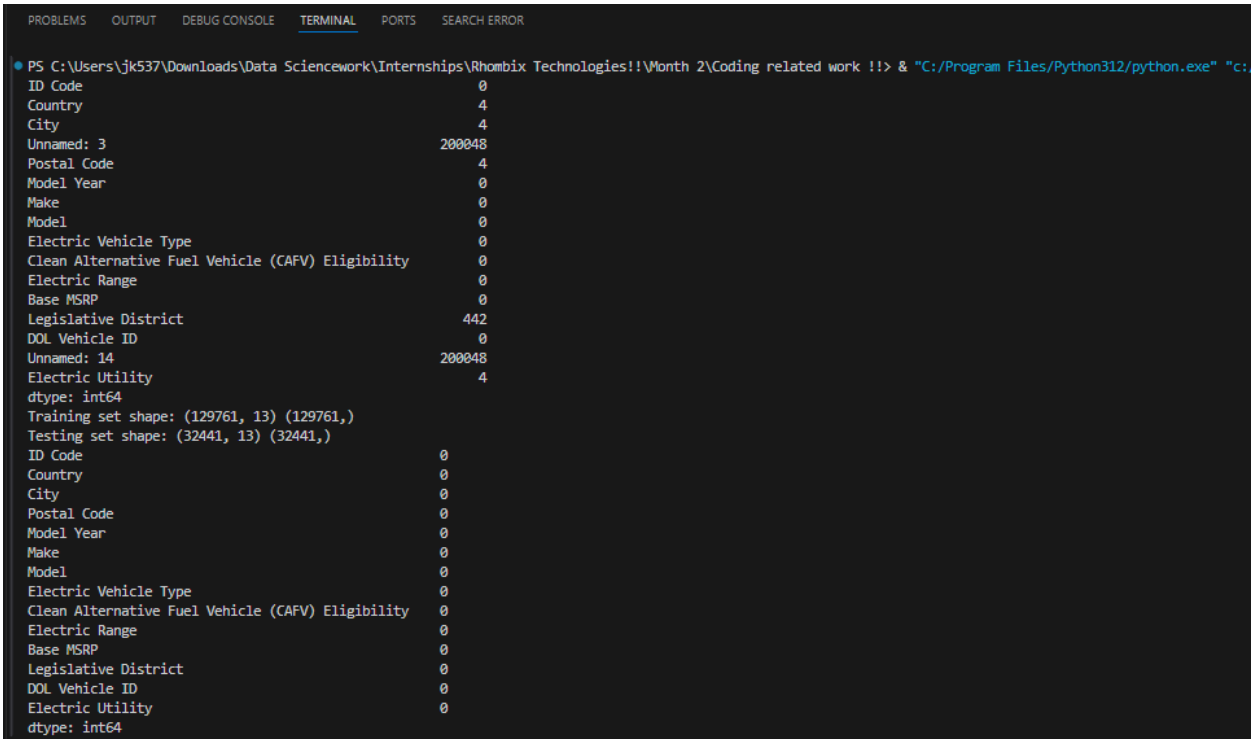
- The shapes of the training and testing datasets are printed to verify the split.

Code Example

Here is the complete code for the implementation and it's output:



```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler

# Load the dataset
df = pd.read_csv("Electric_Vehicle_Population_Data.csv")

# Step 1: Handle missing values
# Check for missing values
print(df.isnull().sum())

# Drop unnecessary columns
df = df.drop(columns=['Unnamed: 3', 'Unnamed: 14'], errors='ignore')

# Fill missing values without using inplace=True
df['Country'] = df['Country'].fillna(df['Country'].mode()[0])
df['City'] = df['City'].fillna(df['City'].mode()[0])
df['Postal Code'] = df['Postal Code'].fillna(df['Postal Code'].mode()[0])
df['Legislative District'] = df['Legislative District'].fillna(df['Legislative District'].mode()[0])
df['Electric Utility'] = df['Electric Utility'].fillna(df['Electric Utility'].mode()[0])

# Step 2: Handle outliers
# Use IQR to detect outliers in 'Electric Range' and 'Base MSRP'
Q1 = df[['Electric Range', 'Base MSRP']].quantile(0.25)
Q3 = df[['Electric Range', 'Base MSRP']].quantile(0.75)
IQR = Q3 - Q1

# Remove outliers
df = df[~((df[['Electric Range', 'Base MSRP']] < (Q1 - 1.5 * IQR)) | (df[['Electric Range', 'Base MSRP']] > (Q3 + 1.5 * IQR))).any(axis=1)]

# Step 3: Normalize or scale features
# Select features for scaling
features_to_scale = ['Electric Range', 'Base MSRP']
scaler = StandardScaler()

# Fit and transform the scaler
df[features_to_scale] = scaler.fit_transform(df[features_to_scale])

# Step 4: Split the data into training and testing sets
# Assume you want to predict 'Base MSRP' as the target variable
X = df.drop(columns=['Base MSRP'])  # Features
y = df['Base MSRP']                 # Target variable

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Output the shapes of the resulting datasets
print("Training set shape:", X_train.shape, y_train.shape)
print("Testing set shape:", X_test.shape, y_test.shape)
```



```
PS C:\Users\jk537\Downloads\Data Sciencework\Internships\Rhombix Technologies!!\Month 2\Coding related work !!> & "C:/Program Files/Python312/python.exe" "c:
ID Code                                             0
Country                                             4
City                                                4
Unnamed: 3                                     200048
Postal Code                                         4
Model Year                                          0
Make                                                0
Model                                               0
Electric Vehicle Type                               0
Clean Alternative Fuel Vehicle (CAFV) Eligibility   0
Electric Range                                      0
Base MSRP                                           0
Legislative District                              442
DOL Vehicle ID                                      0
Unnamed: 14                                    200048
Electric Utility                                    4
dtype: int64
Training set shape: (129761, 13) (129761,)
Testing set shape: (32441, 13) (32441,)
ID Code                                            0
Country                                            0
City                                               0
Postal Code                                        0
Model Year                                         0
Make                                               0
Model                                              0
Electric Vehicle Type                              0
Clean Alternative Fuel Vehicle (CAFV) Eligibility  0
Electric Range                                     0
Base MSRP                                          0
Legislative District                               0
DOL Vehicle ID                                     0
Electric Utility                                   0
dtype: int64
```

Usage

1. Ensure all dependencies are installed.
2. Place the dataset in the same directory as the script or update the file path in the script.
3. Run the script to process the dataset and obtain the training and testing datasets.

Conclusion

This script provides a comprehensive approach to preprocessing the Electric Vehicle Population dataset, making it ready for machine learning tasks. By handling missing values, outliers, and scaling features, it sets a solid foundation for building predictive models.

Future Enhancements

- ***Feature Engineering****:* Consider creating new features that may enhance model performance.
- ***Modeling***: Implement various machine learning algorithms to predict **Base MSRP** and compare their performance.
- ***Visualization***: Add data visualization steps to better understand the dataset and the relationships between features.
- ***Documentation***: Continuously update the documentation as the code evolves and new features are added.