Technical Analysis Report

EV Population Data Analysis

1. Introduction

This document presents an in-depth analysis of the **Electric Vehicle (EV) Population Dataset**, covering data preprocessing, exploratory data analysis (EDA), feature engineering, and predictive modeling. The primary objective is to understand trends in EV adoption, geographical distribution, and model preferences while preparing the dataset for predictive modeling. This technical report provides detailed steps, accompanied by code snippets and insights derived from the dataset.

2. Data Loading and Initial Exploration

2.1 Importing Necessary Libraries

To start, essential Python libraries for data manipulation, visualization, and machine learning are imported. These libraries facilitate efficient data handling and insight extraction.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,
GradientBoostingRegressor
```

2.2 Loading the Dataset

The dataset is read using Pandas to enable easy manipulation and analysis.

```
df = pd.read_csv("ev_population_data.csv")
```

2.3 Dataset Overview

Checking Dataset Shape

```
df.shape
```

This gives the total number of records and columns, providing an initial understanding of dataset size.

Displaying First Few Records

```
df.head()
```

This command provides a glimpse of the dataset, revealing available columns and their data types.

Dataset Summary Information

```
df.info()
```

This function displays column names, data types, and missing values, which are crucial for data cleaning.

3. Data Cleaning

3.1 Handling Missing Values

Missing values can impact analysis accuracy and must be handled appropriately.

```
df.isnull().sum()
```

This command identifies columns with missing values. Handling strategies include filling missing values or dropping affected records.

```
df.fillna(method='ffill', inplace=True)  # Forward fill method
```

Alternatively, if too many values are missing in a column, it may be dropped:

```
df.dropna(axis=1, inplace=True)
```

3.2 Removing Duplicates

Duplicate records can skew analysis and should be removed.

```
df.drop_duplicates(inplace=True)
```

3.3 Data Type Conversion

Some columns might need conversion to appropriate types for correct processing.

```
df['EV Type'] = df['EV Type'].astype('category')
df['Model Year'] = df['Model Year'].astype(int)
```

4. Exploratory Data Analysis (EDA)

EDA helps identify trends, distributions, and relationships within the dataset.

4.1 Distribution of Electric Vehicles Over the Years

```
plt.figure(figsize=(10, 6))
sns.histplot(df['Model Year'], bins=15, kde=True)
plt.title("Distribution of EV Model Years")
plt.xlabel("Year")
plt.ylabel("Frequency")
plt.show()
```

This visualization provides insights into the growth of EV adoption over time.

4.2 Top EV Manufacturers

```
plt.figure(figsize=(12, 6))
sns.countplot(y=df['Make'], order=df['Make'].value_counts().index)
plt.title("Top EV Manufacturers")
plt.xlabel("Count")
plt.ylabel("Make")
plt.show()
```

This helps understand which manufacturers dominate the EV market.

4.3 Geographical Distribution of EVs

```
plt.figure(figsize=(14, 6))
sns.countplot(y=df['State'], order=df['State'].value_counts().index)
plt.title("EV Distribution by State")
plt.xlabel("Number of EVs")
plt.ylabel("State")
plt.show()
```

This visualization provides insights into EV adoption trends across different states.

4.4 Correlation Heatmap

A heatmap helps identify relationships between numerical variables.

```
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```

5. Feature Engineering

Feature engineering improves dataset quality by creating new features for better predictive performance.

5.1 Encoding Categorical Variables

Categorical features need to be converted into numerical representations for machine learning models.

```
encoder = LabelEncoder()
df['Make'] = encoder.fit_transform(df['Make'])
df['State'] = encoder.fit_transform(df['State'])
```

5.2 Standardizing Numeric Features

Standardization ensures uniform scale across numerical features.

```
scaler = StandardScaler()
df[['Range (Miles)', 'Model Year']] = scaler.fit_transform(df[['Range
(Miles)', 'Model Year']])
```

6. Model Training

6.1 Splitting the Dataset

```
X = df.drop(columns=['EV Type'])
y = df['EV Type']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

6.2 Training Predictive Models

Various models are trained to predict EV types based on available features.

```
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(),
    'Gradient Boosting': GradientBoostingRegressor()
}

for name, model in models.items():
    model.fit(X_train, y_train)
    print(f'{name} Score: {model.score(X_test, y_test)}')
```

7. Conclusion

This report detailed the complete workflow for analyzing the Electric Vehicle Population dataset. The data was cleaned, analyzed, and transformed before training predictive models. Insights from EDA provided a deeper understanding of EV adoption trends, geographical distribution, and model preferences. Future improvements may include hyperparameter tuning, deep learning models, and additional feature engineering to enhance predictive accuracy.