Experiment 6

Aim: Perform Classification Modeling:
a) Choose a classifier for a classification problem.
b) Evaluate the performance of the classifier.
Perform Classification using the following 3 classifiers:

1. K-Nearest Neighbors (KNN)
2. Naive Bayes
3. Decision Tree

Performance:

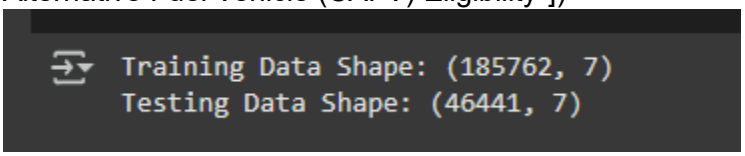
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.tree import plot_tree

df = pd.read_csv('set3.csv')
print(df.head())
print(df.info())
```

```
    VIN (1-10)    County      City State  Postal Code  Model Year    Make  \
0  2T3YL4DV0E      King  Bellevue    WA      98005.0        2014  TOYOTA
1  5YJ3E1EB6K      King   Bothell    WA      98011.0        2019   TESLA
2  5UX43EU02S  Thurston   Olympia    WA      98502.0        2025     BMW
3  JTMAB3FV5R  Thurston   Olympia    WA      98513.0        2024  TOYOTA
4  5YJYGDEE8M    Yakima     Selah    WA      98942.0        2021   TESLA

         Model                   Electric Vehicle Type  \
0         RAV4          Battery Electric Vehicle (BEV)
1      MODEL 3          Battery Electric Vehicle (BEV)
2           X5  Plug-in Hybrid Electric Vehicle (PHEV)
3  RAV4 PRIME  Plug-in Hybrid Electric Vehicle (PHEV)
4      MODEL Y          Battery Electric Vehicle (BEV)

   Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0           Clean Alternative Fuel Vehicle Eligible           103.0
1           Clean Alternative Fuel Vehicle Eligible           220.0
2           Clean Alternative Fuel Vehicle Eligible            40.0
3           Clean Alternative Fuel Vehicle Eligible            42.0
4  Eligibility unknown as battery range has not b...             0.0
```

```
   Base MSRP  Legislative District  DOL Vehicle ID  \
0        0.0                 41.0       186450183
1        0.0                  1.0       478093654
2        0.0                 35.0       274800718
3        0.0                  2.0       260758165
4        0.0                 15.0       236581355

              Vehicle Location                                Electric Utility  \
0   POINT (-122.1621 47.64441)  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)
1  POINT (-122.20563 47.76144)  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)
2  POINT (-122.92333 47.03779)                          PUGET SOUND ENERGY INC
3  POINT (-122.81754 46.98876)                          PUGET SOUND ENERGY INC
4  POINT (-120.53145 46.65405)                                       PACIFICORP

   2020 Census Tract
0       5.303302e+10
1       5.303302e+10
2       5.306701e+10
3       5.306701e+10
4       5.307700e+10
<class 'pandas.core.frame.DataFrame'>
```

```
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dtype
---  ------                                             --------------   -----
 0   VIN (1-10)                                         232230 non-null  object
 1   County                                             232226 non-null  object
 2   City                                               232226 non-null  object
 3   State                                              232230 non-null  object
 4   Postal Code                                        232226 non-null  float64
 5   Model Year                                         232230 non-null  int64
 6   Make                                               232230 non-null  object
 7   Model                                              232230 non-null  object
 8   Electric Vehicle Type                              232230 non-null  object
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  232230 non-null  object
 10  Electric Range                                     232203 non-null  float64
 11  Base MSRP                                          232203 non-null  float64
 12  Legislative District                               231749 non-null  float64
 13  DOL Vehicle ID                                     232230 non-null  int64
 14  Vehicle Location                                   232219 non-null  object
 15  Electric Utility                                   232226 non-null  object
 16  2020 Census Tract                                  232226 non-null  float64
dtypes: float64(5), int64(2), object(10)
```

```
df_filtered = df[["Model Year", "Make","Model","Electric
Vehicle Type", "Clean Alternative Fuel Vehicle (CAFV)
Eligibility", "Electric Range", "Base MSRP"]].dropna()

label_encoders = {}
for col in ["Make", "Model", "Electric Vehicle Type"]:
        le = LabelEncoder()
        df_filtered[col] = le.fit_transform(df_filtered[col])
        label_encoders[col] = le

target_encoder = LabelEncoder()
df_filtered["CAFV Eligibility"] =
target_encoder.fit_transform(df_filtered["Clean
Alternative Fuel Vehicle (CAFV) Eligibility"])
```

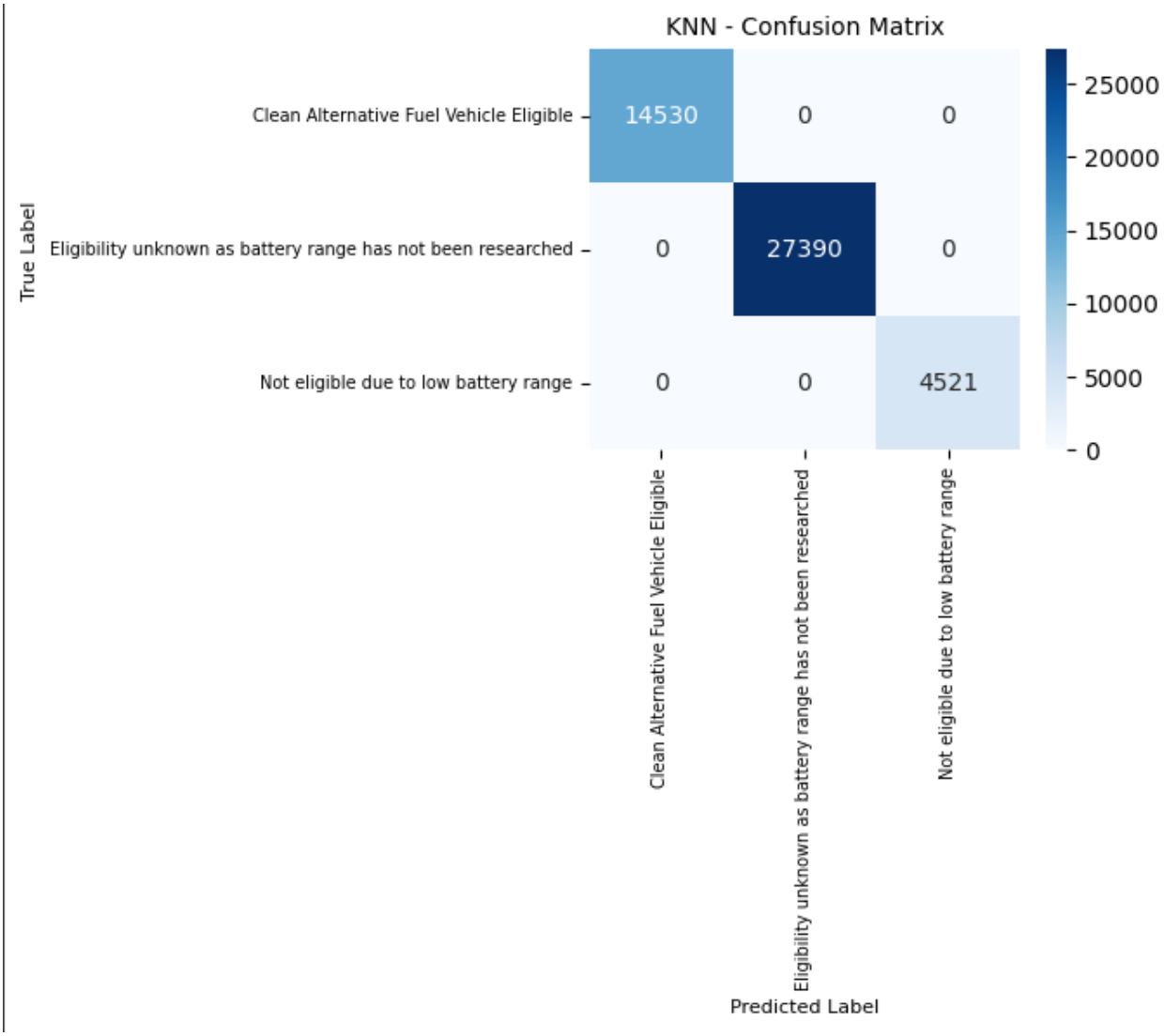


The code splits the dataset into 80% training and 20% testing sets using train_test_split() Finally, it prints the shapes of the training and testing sets.

Training and Evaluating K-Nearest Neighbours (KNN):-

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)

print("\nK-Nearest Neighbors (KNN) Performance:")
print(f"Accuracy: {accuracy_score(y_test, y_pred_knn):.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred_knn,
target_names=target_encoder.classes_))

plt.figure(figsize=(4, 3))
cm = confusion_matrix(y_test, y_pred_knn)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=target_encoder.classes_,
yticklabels=target_encoder.classes_)
plt.xlabel("Predicted Label", fontsize=8)
plt.ylabel("True Label", fontsize=8)
plt.title("KNN - Confusion Matrix", fontsize=10)
plt.xticks(fontsize=7)
plt.yticks(fontsize=7)
plt.show()
```

```
K-Nearest Neighbors (KNN) Performance:
Accuracy: 1.0000
Classification Report:
                                                    precision    recall  f1-score   support

                    Clean Alternative Fuel Vehicle Eligible       1.00      1.00      1.00     14530
Eligibility unknown as battery range has not been researched       1.00      1.00      1.00     27390
                      Not eligible due to low battery range       1.00      1.00      1.00      4521

                                                 accuracy                           1.00     46441
                                                macro avg       1.00      1.00      1.00     46441
                                             weighted avg       1.00      1.00      1.00     46441
```



trains a K-Nearest Neighbors (KNN) model with n_neighbors=5 using the training data and makes predictions on the test set.

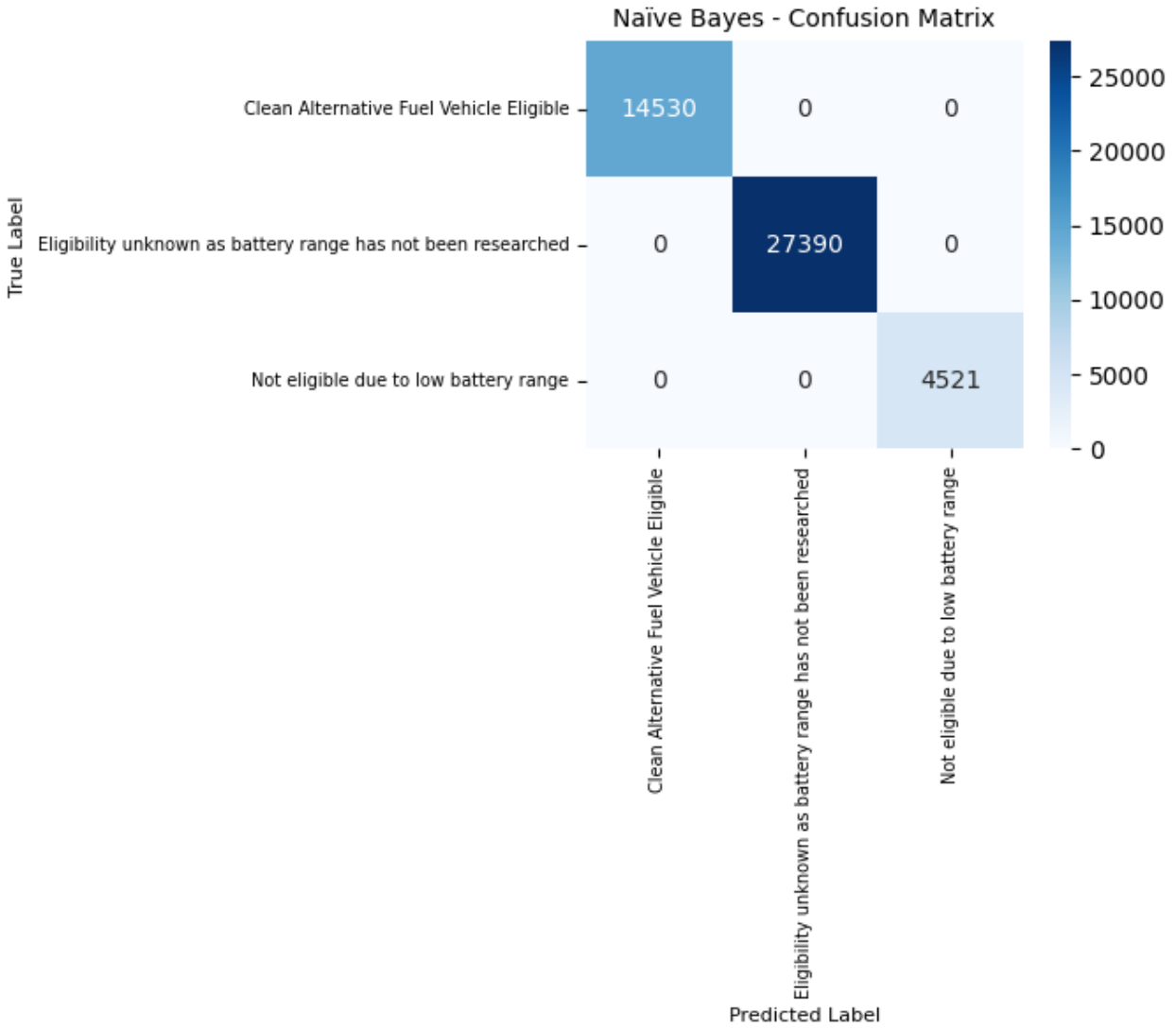
Training and Evaluating Naive Bayes:
nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)

print("\nNaïve Bayes Performance:")
print(f"Accuracy: {accuracy_score(y_test, y_pred_nb):.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred_nb,
target_names=target_encoder.classes_))

plt.figure(figsize=(4, 3))
cm = confusion_matrix(y_test, y_pred_nb)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=target_encoder.classes_,
yticklabels=target_encoder.classes_)
plt.xlabel("Predicted Label", fontsize=8)

```
plt.ylabel("True Label", fontsize=8)
plt.title("Naïve Bayes - Confusion Matrix", fontsize=10)
plt.xticks(fontsize=7)
plt.yticks(fontsize=7)
plt.show()
```

```
Naïve Bayes Performance:
Accuracy: 1.0000
Classification Report:
                                                    precision    recall  f1-score   support

                     Clean Alternative Fuel Vehicle Eligible       1.00      1.00      1.00     14530
Eligibility unknown as battery range has not been researched       1.00      1.00      1.00     27390
                       Not eligible due to low battery range       1.00      1.00      1.00      4521

                                                    accuracy                           1.00     46441
                                                   macro avg       1.00      1.00      1.00     46441
                                                weighted avg       1.00      1.00      1.00     46441
```

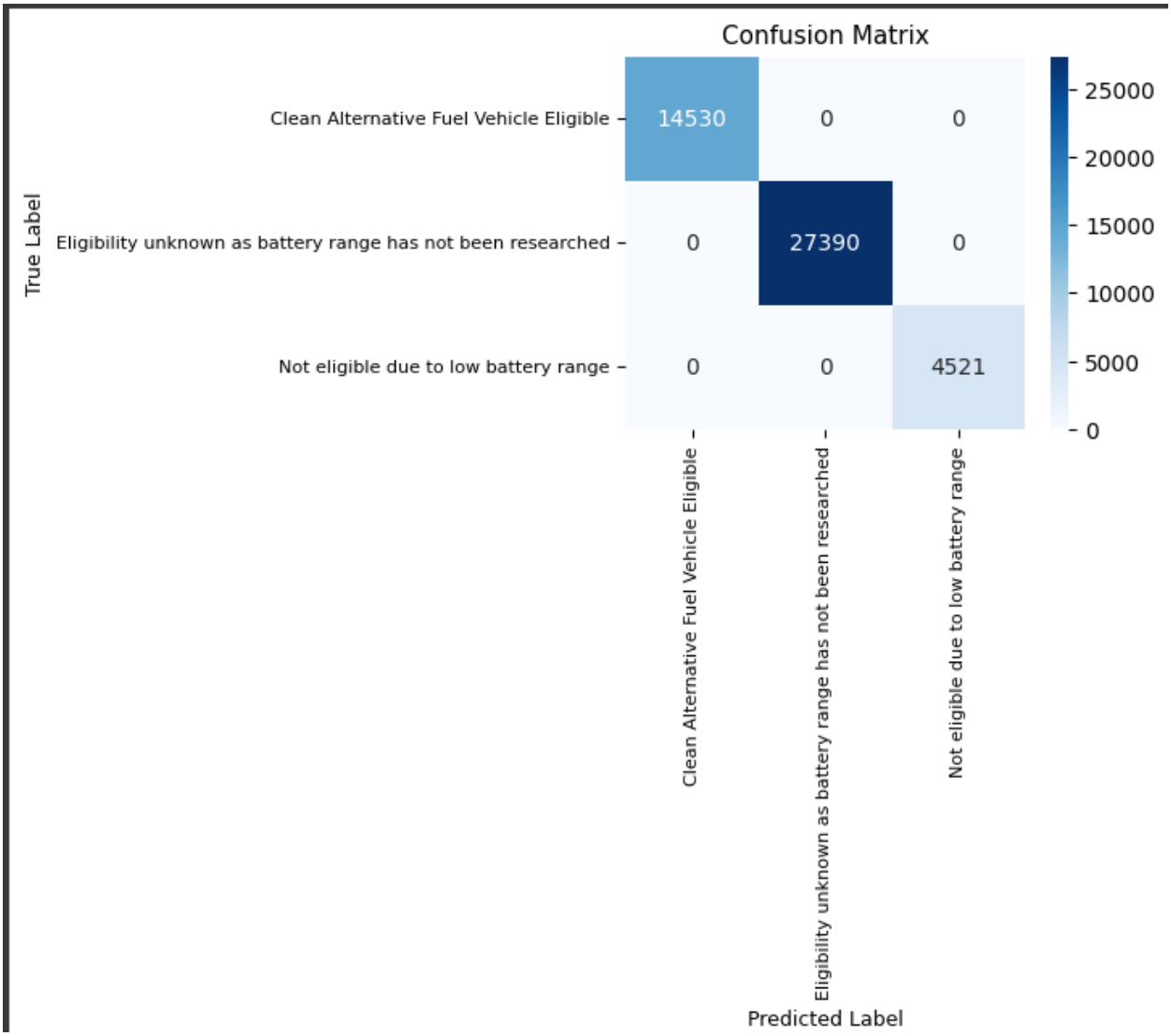


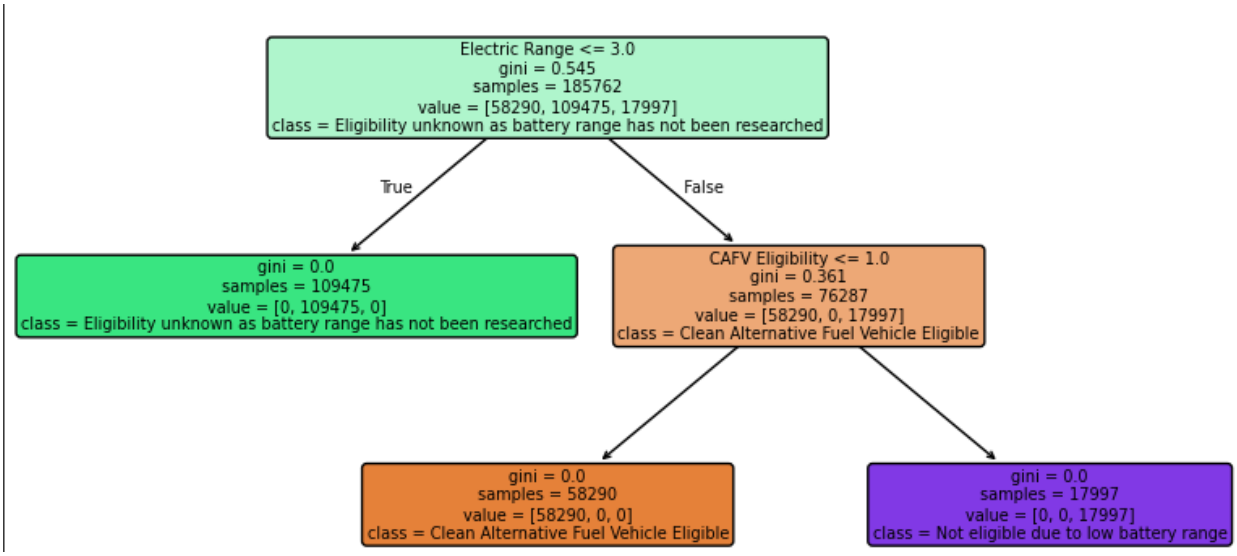
trains a Naïve Bayes classifier using the training dataset and makes predictions on the test set.

Decision Tree Model Training, Evaluation and Visualization:-

```
 dt = DecisionTreeClassifier(random_state=42, max_depth=3,
min_samples_split=10, min_samples_leaf=5)
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)

print("\nDecision Tree Performance:")
print(f"Accuracy: {accuracy_score(y_test, y_pred_dt):.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred_dt,
target_names=target_encoder.classes_))

plt.figure(figsize=(4, 3))
cm = confusion_matrix(y_test, y_pred_dt)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
xticklabels=target_encoder.classes_,
yticklabels=target_encoder.classes_)
plt.xlabel("Predicted Label", fontsize=9)
plt.ylabel("True Label", fontsize=9)
plt.title("Confusion Matrix", fontsize=11)
plt.xticks(fontsize=8)
plt.yticks(fontsize=8)
plt.show()

plt.figure(figsize=(10, 5))
plot_tree(dt, filled=True, feature_names=X.columns,
class_names=target_encoder.classes_, fontsize=7, rounded=True)
plt.title("Decision Tree Visualization", fontsize=11)
plt.show()
```

```
Decision Tree Performance:
Accuracy: 1.0000
Classification Report:
                                                    precision    recall  f1-score   support

                     Clean Alternative Fuel Vehicle Eligible       1.00      1.00      1.00     14530
Eligibility unknown as battery range has not been researched       1.00      1.00      1.00     27390
                       Not eligible due to low battery range       1.00      1.00      1.00      4521

                                            accuracy                           1.00     46441
                                           macro avg       1.00      1.00      1.00     46441
                                        weighted avg       1.00      1.00      1.00     46441
```


Confusion Matrix
Clean Alternative Fuel Vehicle Eligible
Eligibility unknown as battery range has not been researched
Not eligible due to low battery range
True Label
14530
0
0
0
27390
0
0
0
4521
25000
20000
15000
10000
5000
0
Clean Alternative Fuel Vehicle Eligible
Eligibility unknown as battery range has not been researched
Not eligible due to low battery range
Predicted Label


Electric Range <= 3.0
gini = 0.545
samples = 185762
value = [58290, 109475, 17997]
class = Eligibility unknown as battery range has not been researched
True
False
gini = 0.0
samples = 109475
value = [0, 109475, 0]
class = Eligibility unknown as battery range has not been researched
CAFV Eligibility <= 1.0
gini = 0.361
samples = 76287
value = [58290, 0, 17997]
class = Clean Alternative Fuel Vehicle Eligible
gini = 0.0
samples = 58290
value = [58290, 0, 0]
class = Clean Alternative Fuel Vehicle Eligible
gini = 0.0
samples = 17997
value = [0, 0, 17997]
class = Not eligible due to low battery range

trains a Decision Tree classifier on the training dataset and makes predictions on the test set.

Model Performance Comparison:-

```
model_performances = {
        "KNN": accuracy_score(y_test, y_pred_knn),
    "Naïve Bayes": accuracy_score(y_test, y_pred_nb),
    "Decision Tree": accuracy_score(y_test, y_pred_dt)
                                    }

print("\nModel Performance Summary:")
      for model, acc in model_performances.items():
            print(f"{model}: Accuracy = {acc:.4f}")
```

```
Model Performance Summary:
KNN: Accuracy = 1.0000
Naïve Bayes: Accuracy = 1.0000
Decision Tree: Accuracy = 1.0000
```

The code compares the accuracy scores of different machine learning models, including K-Nearest Neighbors (KNN), Naïve Bayes and Decision Tree. These accuracy values are saved in a dictionary, model_performances, and then printed in a structured format to provide a quick overview of how well each model performed on the test dataset. This step helps in identifying the most effective model for classification.

Conclusion:

In this experiment, we performed classification modeling using K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree classifiers. All three models achieved perfect accuracy (100%), indicating that the dataset was highly separable and well-suited for classification. The Decision Tree provided clear and interpretable decision boundaries, KNN demonstrated flawless proximity-based classification, and even Naïve Bayes despite its simplistic probabilistic assumptions, was able to perfectly classify the data. These results suggest that the chosen features were highly informative and that any of the tested models could be effectively deployed for this classification task.