```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVR, SVC
from sklearn.tree import DecisionTreeRegressor, DecisionTreeClassifier
from sklearn.metrics import mean_squared_error, r2_score, accuracy_score, classification_report
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.tree import plot_tree

# Load the data
car_data = pd.read_csv('Car_sales.csv')

# Drop columns not used for modeling
car_data.drop(['Manufacturer', 'Model', 'Latest_Launch'], axis=1, inplace=True)

# Encode categorical feature "Vehicle_type"
label_encoder = LabelEncoder()
car_data['Vehicle_type'] = label_encoder.fit_transform(car_data['Vehicle_type'])

# Split into features and target variable
X = car_data.drop('Sales_in_thousands', axis=1)
y = car_data['Sales_in_thousands']

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Fill missing values with the mean
X_train.fillna(X_train.mean(), inplace=True)
X_test.fillna(X_train.mean(), inplace=True)
```

```
# Initialize models
linear_regression = LinearRegression()
svm_regressor = SVR()
decision_tree_regressor = DecisionTreeRegressor()

# Train the models
linear_regression.fit(X_train, y_train)
svm_regressor.fit(X_train, y_train)
decision_tree_regressor.fit(X_train, y_train)

# Predictions
y_pred_linear = linear_regression.predict(X_test)
y_pred_svm = svm_regressor.predict(X_test)
y_pred_decision_tree = decision_tree_regressor.predict(X_test)
```

```
# Evaluate Linear Regression model
mse_linear = mean_squared_error(y_test, y_pred_linear)
r2_linear = r2_score(y_test, y_pred_linear)
print("Linear Regression Metrics:")
print(f"MSE: {mse_linear:.2f}")
print(f"R2 Score: {r2_linear:.2f}")
print()
```

```
Linear Regression Metrics:
MSE: 6916.40
R2 Score: 0.29
```

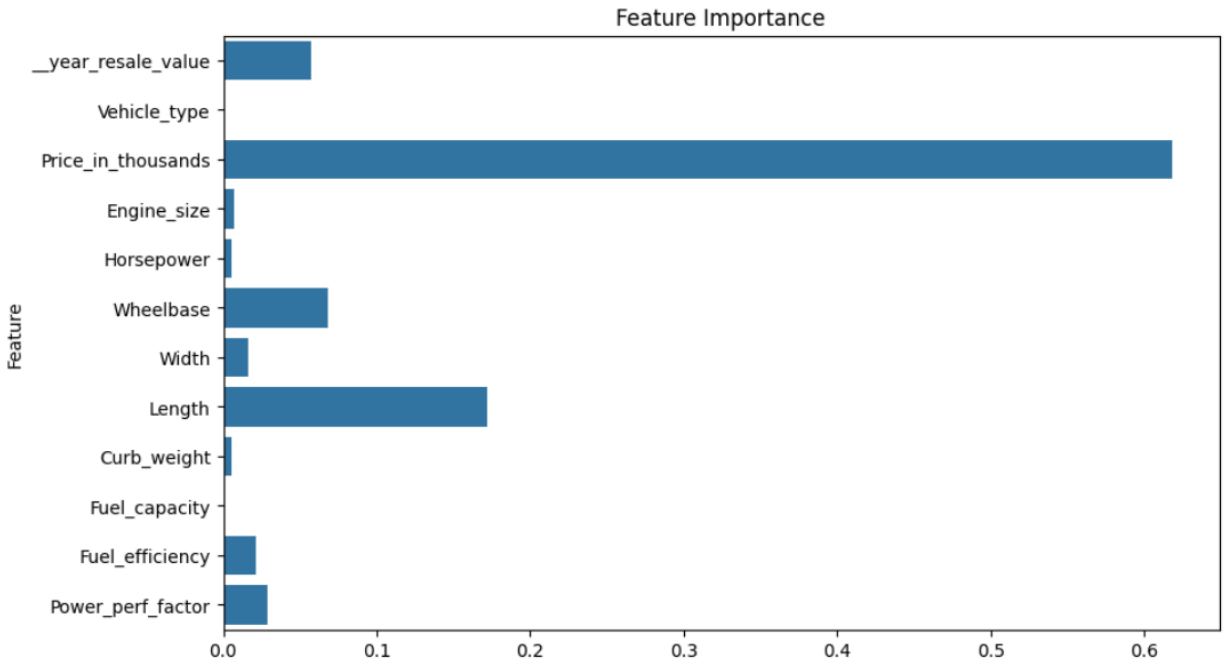
```
# Evaluate SVM model
mse_svm = mean_squared_error(y_test, y_pred_svm)
r2_svm = r2_score(y_test, y_pred_svm)
print("SVM Metrics:")
print(f"MSE: {mse_svm:.2f}")
print(f"R2 Score: {r2_svm:.2f}")
print()
```

```
SVM Metrics:
MSE: 10394.37
R2 Score: -0.07
```

```
# Evaluate Decision Tree model
mse_decision_tree = mean_squared_error(y_test, y_pred_decision_tree)
r2_decision_tree = r2_score(y_test, y_pred_decision_tree)
print("Decision Tree Metrics:")
print(f"MSE: {mse_decision_tree:.2f}")
print(f"R2 Score: {r2_decision_tree:.2f}")
print()
```

```
Decision Tree Metrics:
MSE: 15651.29
R2 Score: -0.61
```

```
# Feature importance for Decision Tree
feature_importance = decision_tree_regressor.feature_importances_
feature_names = X.columns
plt.figure(figsize=(10, 6))
sns.barplot(x=feature_importance, y=feature_names)
plt.title("Feature Importance")
plt.xlabel("Importance")
plt.ylabel("Feature")
plt.show()
```




```
# Visualize Decision Tree
plt.figure(figsize=(20,10))
plot_tree(decision_tree_regressor, filled=True, feature_names=feature_names)
plt.show()
```