

```
plt.show()
```

Cell was canceled due to an error in a previous cell.

```
[ ]: # Data Manipulation
import numpy as np
import pandas as pd
import warnings
# Visualization
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
from sklearn.tree import export_graphviz
import pydot
#Train Test Split
from sklearn.model_selection import train_test_split

#Scaling
from sklearn.preprocessing import StandardScaler

#Models
import optuna
import xgboost as xgb
from sklearn.dummy import DummyRegressor
from skopt import BayesSearchCV
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import GridSearchCV

#Deep Neural Networks:
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping

#Evaluation
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

```

from sklearn.metrics import explained_variance_score

cars = pd.read_csv(r'C:\Users\rocke\Downloads\used_cars\used_car_cleaned.csv',
                  ↪delimiter=',')
cars2 = pd.read_csv(r'C:\Users\rocke\Downloads\used_cars\archive_
                  ↪(2)\UsedCarsSA_Clean_EN.csv', delimiter=',')
warnings.filterwarnings("ignore")

# print(cars.head())
# print(cars2.head())

# CLEANING CODE

cars2 = cars2.drop(['Origin', 'Color', 'Options', 'Engine_Size', 'Fuel_Type',
                  ↪'Region', 'Negotiable'], axis=1)
cars2 = cars2.rename(columns={'Make': 'car_brand', 'Type': 'car_model', 'Year':
                  ↪'car_model_year', 'Gear_Type': 'car_transmission', 'Mileage': 'car_driven',
                  ↪'Price': 'car_price'})
cars2 = cars2[cars.columns]
cars2 = cars2[cars2['car_price'] != 0]
cars = pd.concat([cars, cars2], ignore_index=True)
print(cars)
cars.to_csv("used_car_data.csv")
# Calculate correlations for selected columns
correlation_columns = ['car_driven', 'car_model_year', 'car_price']
correlation_matrix = cars[correlation_columns].corr()

# Create a heatmap to visualize the correlation matrix
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
sns.pairplot(data=cars, hue='car_transmission')
plt.show()

# price distribution

plt.figure(figsize=(15,8))

plt.subplot(1,2,1)
plt.title('Price Distribution Plot')
sns.histplot(cars['car_price'])
plt.ticklabel_format(useOffset=False, style='plain', axis='x')

plt.subplot(1,2,2)
plt.title('Price Spread')
sns.boxplot(y=cars['car_price'])
plt.ticklabel_format(useOffset=False, style='plain', axis='y')

```

```

plt.show()

#price outliers
expensive = cars.loc[cars['car_price'] > 200000]
cars = cars[cars['car_price'] <= 200000]
cars = cars[cars['car_price'] >= 5000]

plt.figure(figsize=(15,8))

plt.subplot(1,2,1)
plt.title('Price Distribution Plot')
sns.histplot(cars['car_price'])
plt.ticklabel_format(useOffset=False, style='plain', axis='x')

plt.subplot(1,2,2)
plt.title('Price Spread')
sns.boxplot(y=cars['car_price'])
plt.ticklabel_format(useOffset=False, style='plain', axis='y')

plt.show()

# Create a box plot to visualize the distribution of car prices by transmission_
↳type
sns.boxplot(x='car_transmission', y='car_price', data=cars)
plt.show()

plt.figure(figsize=(15,8))

plt.subplot(1,2,1)
plt.title('Year Distribution Plot')
sns.histplot(cars['car_model_year'])
plt.ticklabel_format(useOffset=False, style='plain', axis='x')

plt.subplot(1,2,2)
plt.title('Year Spread')
sns.boxplot(y=cars['car_model_year'])
plt.ticklabel_format(useOffset=False, style='plain', axis='y')

plt.show()

# Create a bar plot to visualize the average car price by car brand
brandPrice = cars.groupby(['car_brand'])['car_price'].mean().reset_index()
sns.barplot(x='car_brand', y='car_price', data=brandPrice)
plt.xticks(rotation=90)
plt.show()

```

```

# milage outliers

plt.subplot(1,2,1)
plt.title('Mileage Distribution Plot')
sns.histplot(cars['car_driven'])
plt.ticklabel_format(useOffset=False, style='plain', axis=('x'))

plt.subplot(1,2,2)
plt.title('Mileage Spread')
sns.boxplot(y=cars['car_driven'])
plt.ticklabel_format(useOffset=False, style='plain', axis=('y'))

plt.show()

print(cars.loc[cars['car_driven'] > 500000])
cars = cars[cars['car_driven'] <= 500000]
print(cars[cars['car_transmission'] == 118008.5011120378])
sns.boxplot(y='car_driven', x='car_transmission', data=cars)
plt.show()

cars = cars[cars['car_model_year'] >= 2000]
print(cars.dtypes)

plt.figure(figsize=(15,8))

plt.subplot(1,2,1)
plt.title('Mileage Distribution Plot')
sns.histplot(cars['car_driven'])
plt.ticklabel_format(useOffset=False, style='plain', axis=('x'))

plt.subplot(1,2,2)
plt.title('Mileage Spread')
sns.boxplot(y=cars['car_driven'])
plt.ticklabel_format(useOffset=False, style='plain', axis=('y'))

plt.show()

# encoded categorical variables and feature choosing

cars_encoded = pd.get_dummies(cars, columns=['car_brand', 'car_model', '
↳ 'car_transmission'])

# Split the data into features (X) and target (y)

```

```

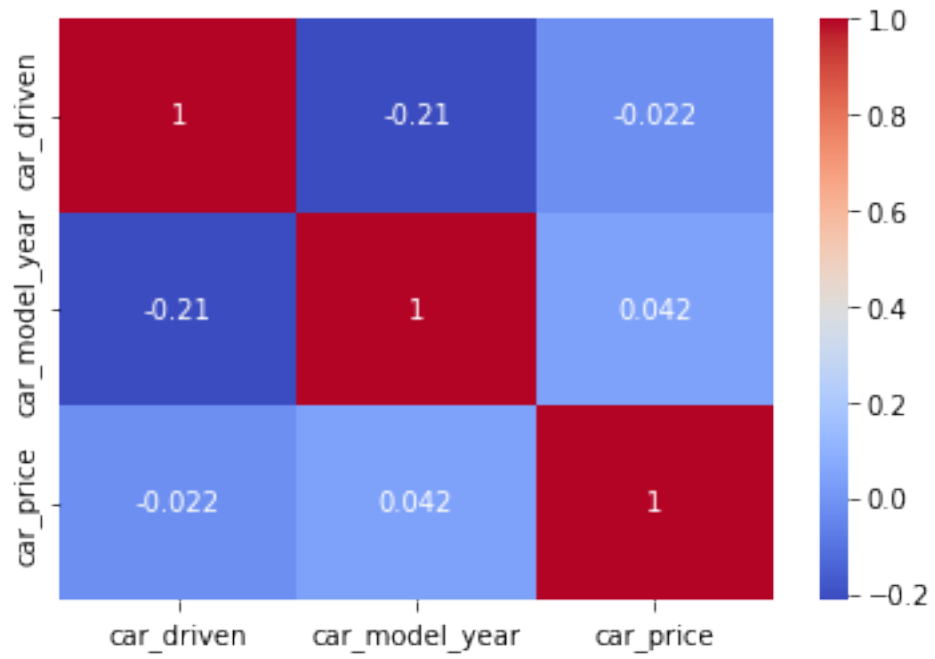
X = cars_encoded.drop('car_price', axis=1)
y = cars_encoded['car_price']

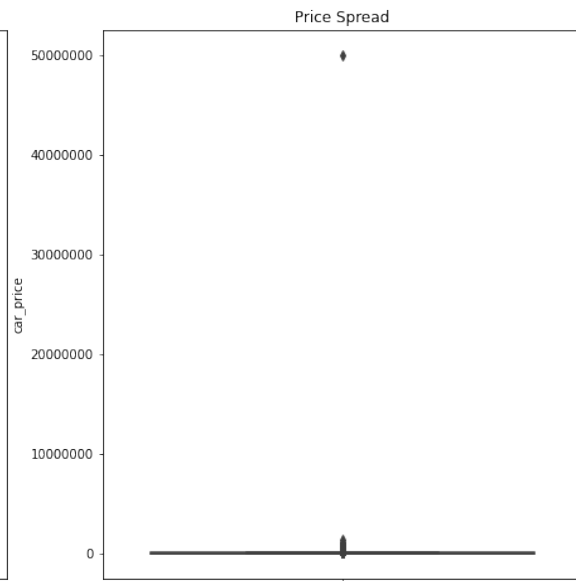
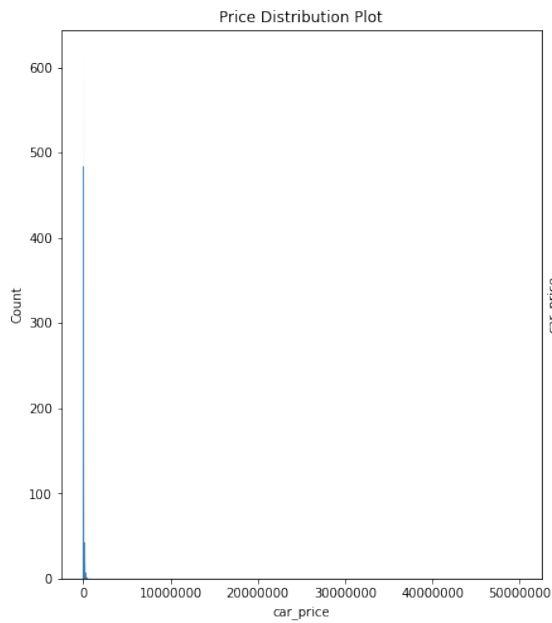
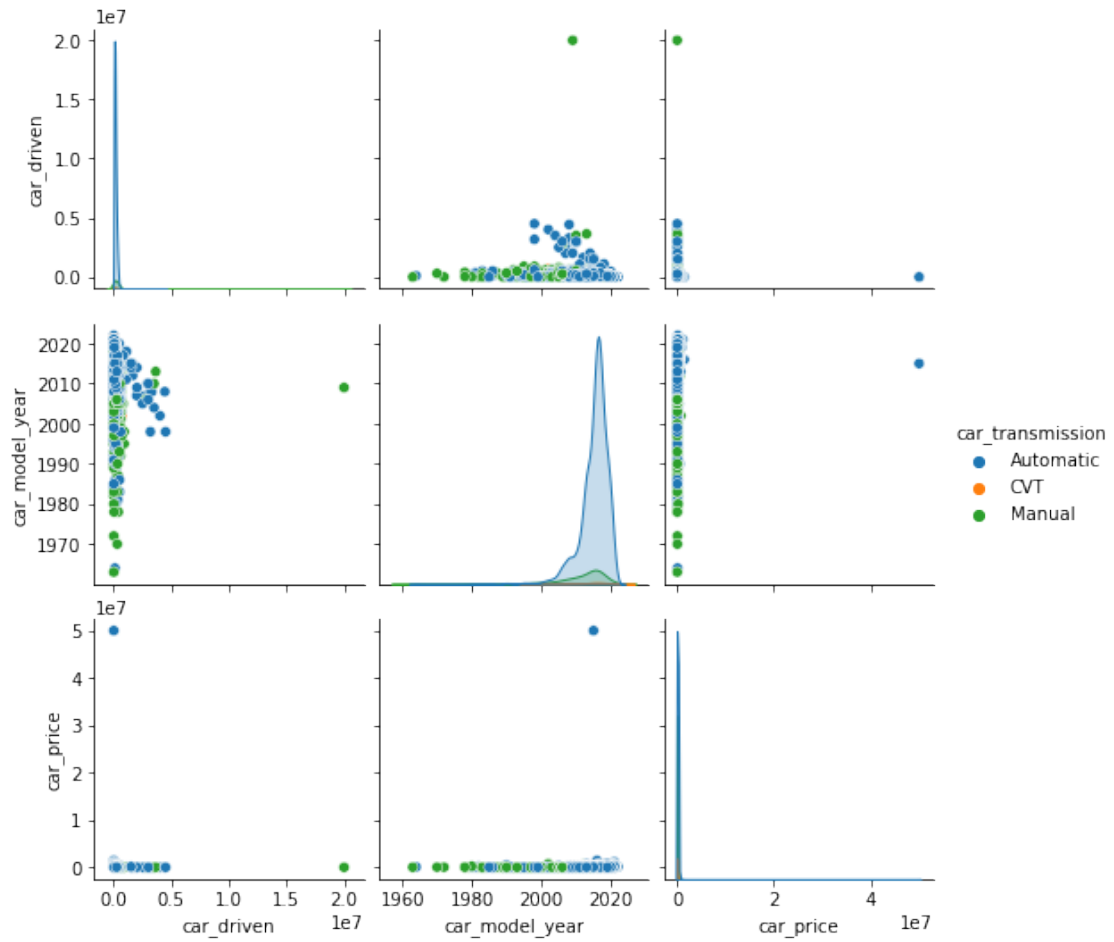
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
↳random_state = 42)

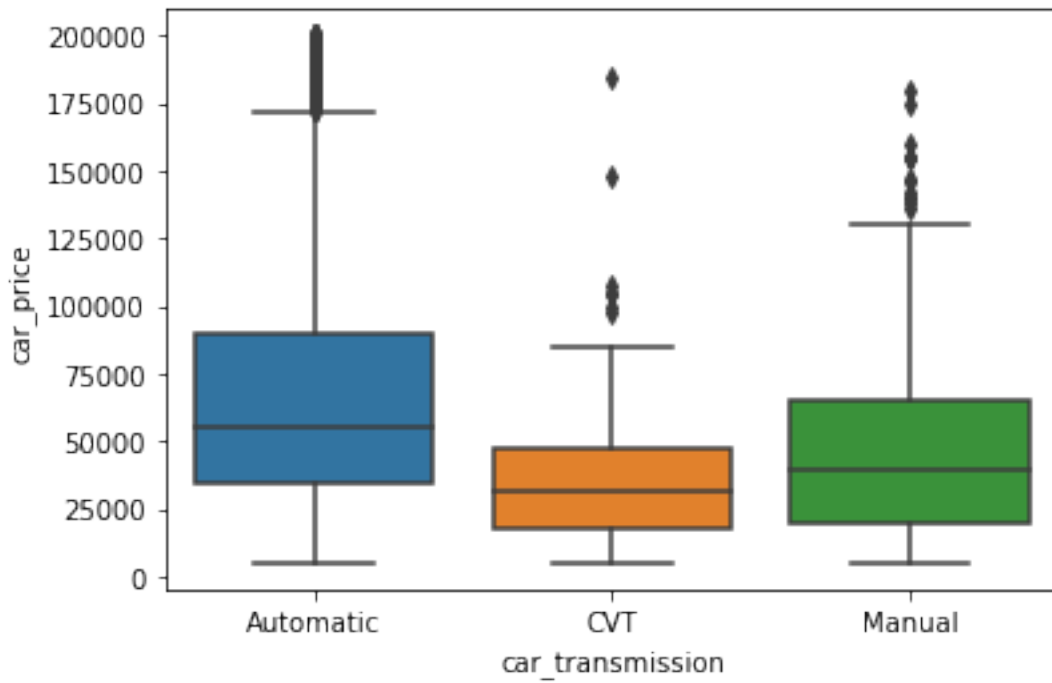
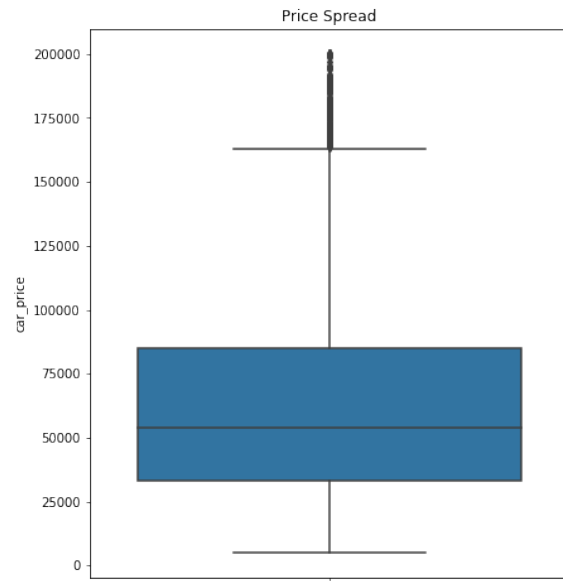
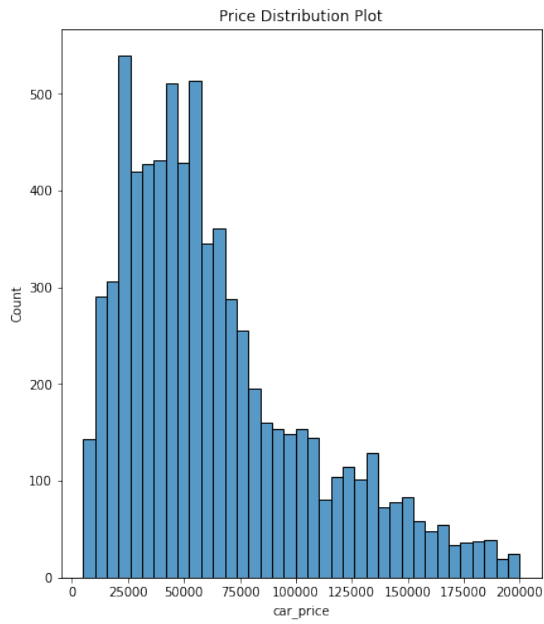
```

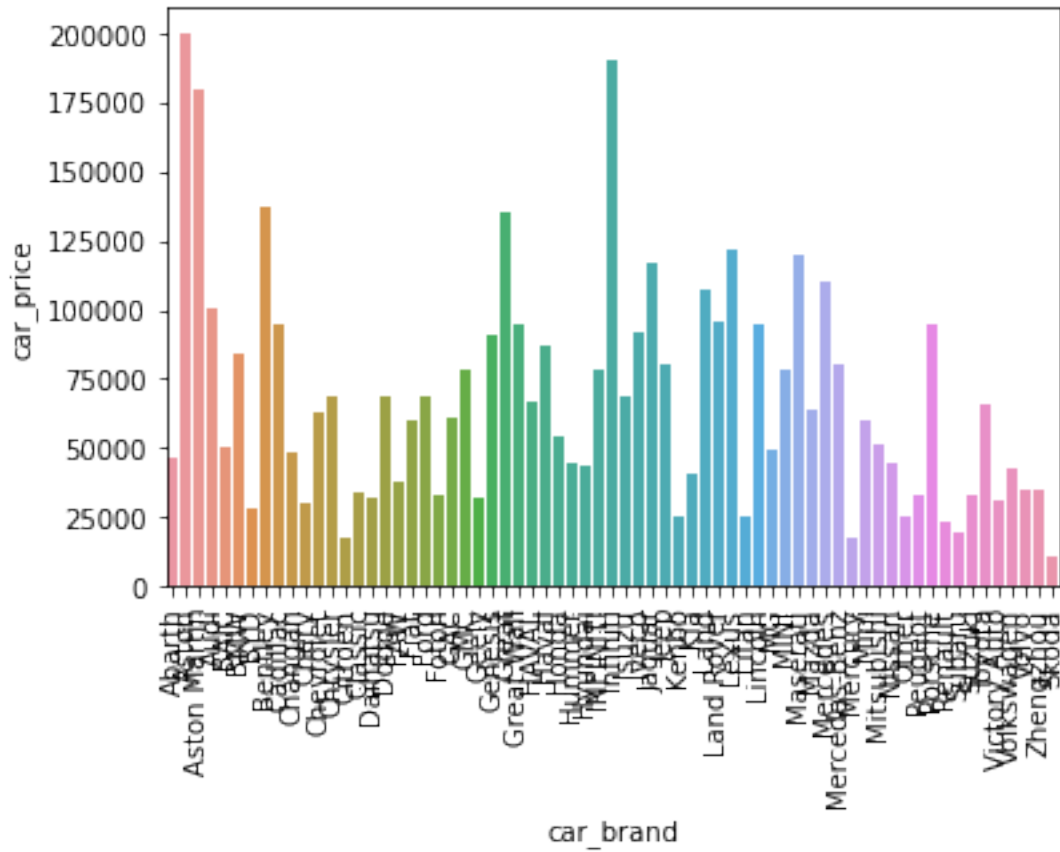
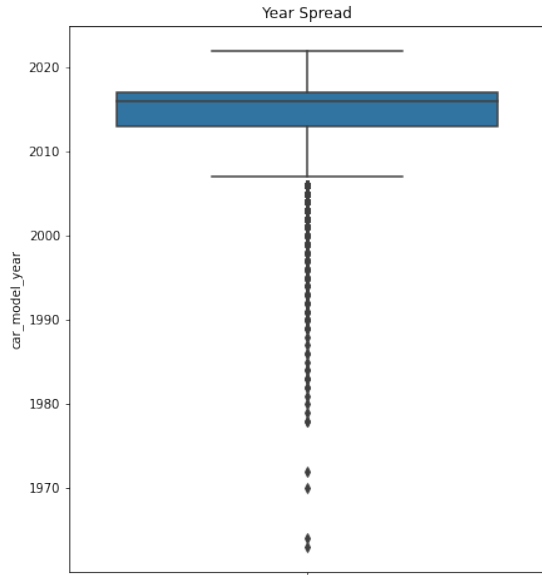
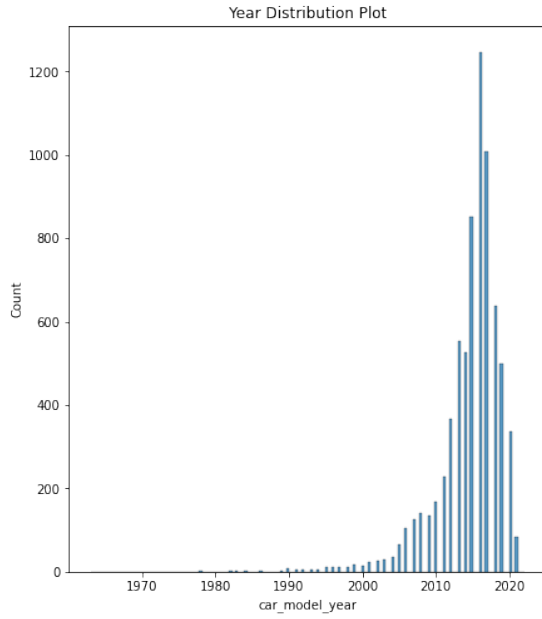
	car_brand	car_model	car_driven	car_transmission	car_model_year	\
0	Hyundai	Tucson	83491.0	Automatic	2018	
1	Chevrolet	Trailblazer	222000.0	Automatic	2009	
2	Great	Wall	0.0	Automatic	2022	
3	Ford	Fusion	178000.0	Automatic	2012	
4	Mitsubishi	Attrage	10500.0	Automatic	2020	
...	
7790	Kia	Sorento	257000.0	Manual	2006	
7791	Audi	A6	77000.0	Automatic	2015	
7792	Chevrolet	Camaro	150000.0	Automatic	2010	
7793	Nissan	Altima	18500.0	Automatic	2011	
7794	Cadillac	Other	256000.0	Automatic	2013	
	car_price					
0	64000.0					
1	20000.0					
2	135000.0					
3	23000.0					
4	32000.0					
...	...					
7790	15000.0					
7791	75000.0					
7792	53000.0					
7793	22000.0					
7794	40000.0					

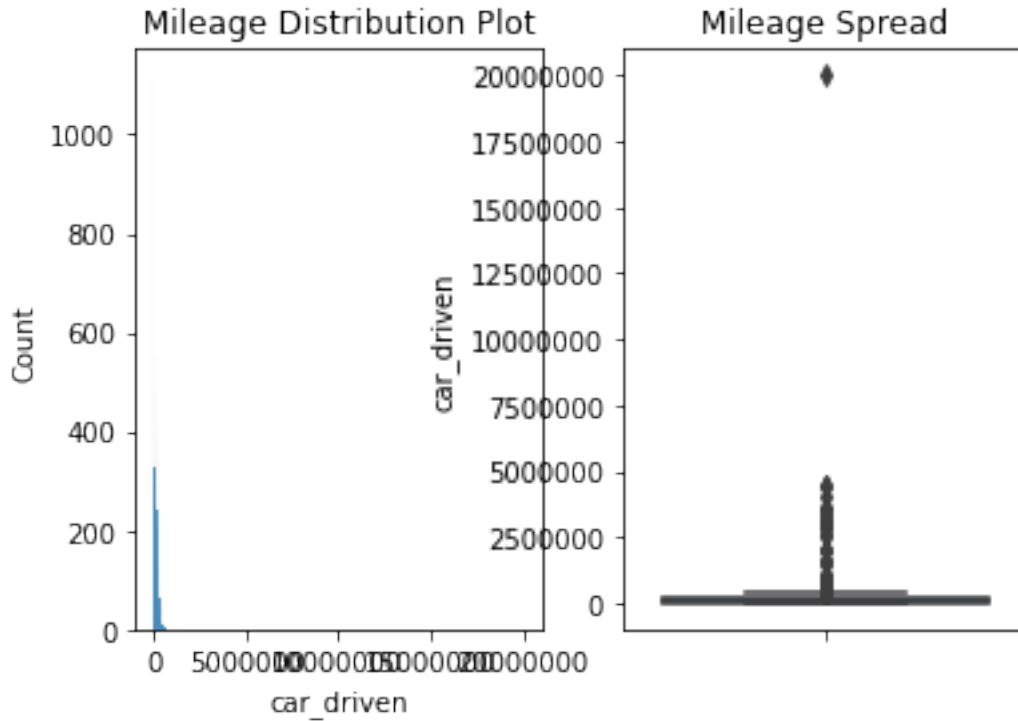
[7795 rows x 6 columns]











	car_brand	car_model	car_driven	car_transmission	car_model_year	\
18	Kia	Cerato	1640000.0	Automatic	2012	
345	Toyota	Camry	755000.0	Automatic	2018	
588	Hyundai	Sonata	625000.0	Automatic	2006	
1125	Suzuki	Swift	2680000.0	Automatic	2007	
1640	Toyota	Camry	692000.0	Manual	1998	
...	
7646	Honda	Accord	547000.0	Manual	2005	
7680	Nissan	Patrol	595000.0	Automatic	1998	
7702	Toyota	Camry	520000.0	Automatic	2009	
7721	Toyota	Land Cruiser	1500000.0	Automatic	2015	
7752	Toyota	Camry	550000.0	Manual	2010	
	car_price					
18	33000.0					
345	62500.0					
588	15000.0					
1125	14500.0					
1640	7000.0					
...	...					
7646	19000.0					
7680	20000.0					
7702	23000.0					
7721	150000.0					

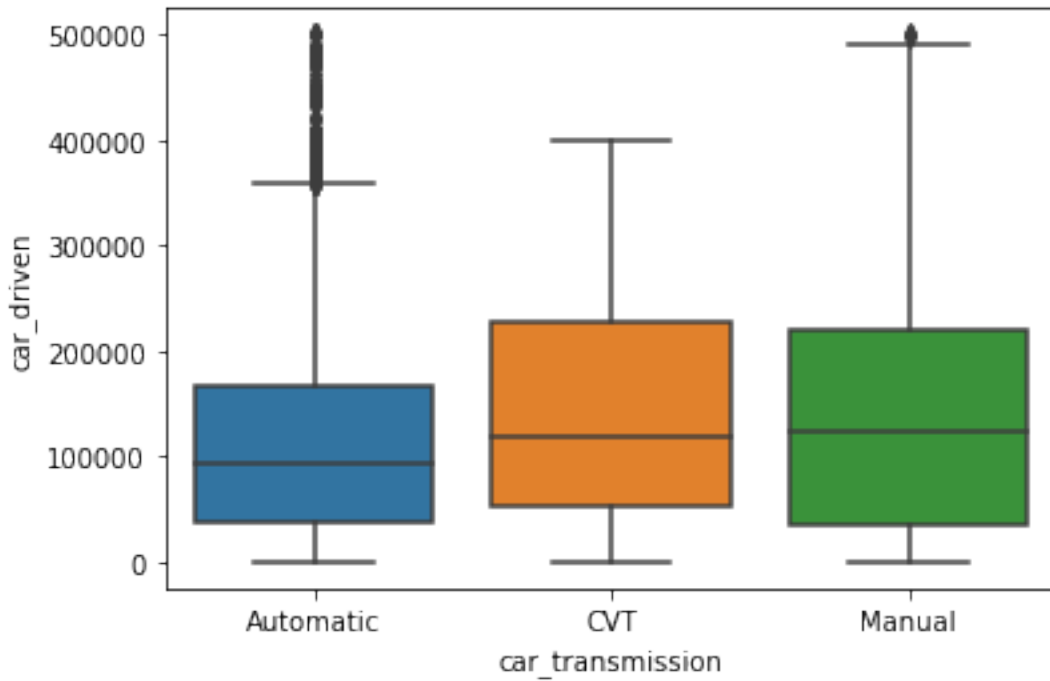
7752 20000.0

[78 rows x 6 columns]

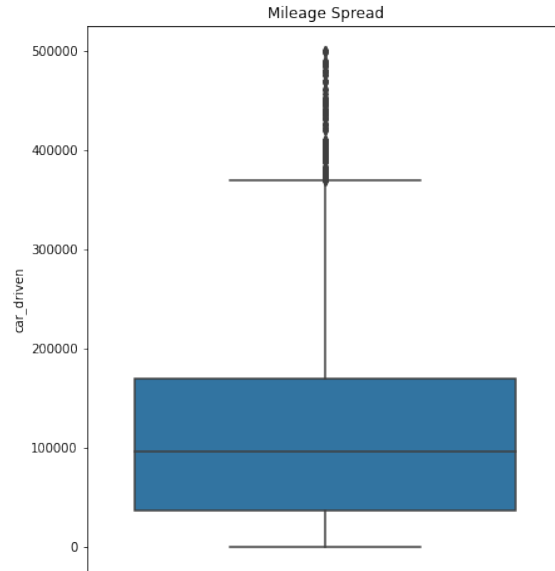
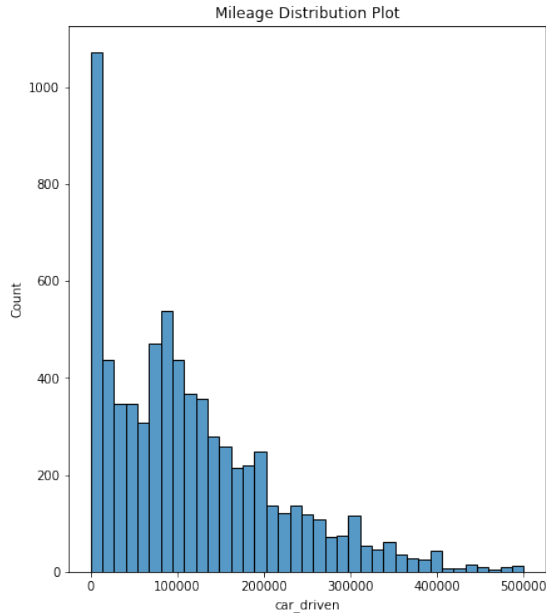
Empty DataFrame

Columns: [car_brand, car_model, car_driven, car_transmission, car_model_year, car_price]

Index: []



```
car_brand      object
car_model      object
car_driven     float64
car_transmission object
car_model_year  int64
car_price      float64
dtype: object
```



```
[ ]: print('Random Forest Regression')
      # Generate a baseline using DummyRegressor
      dummy_model = DummyRegressor(strategy='mean')
      dummy_model.fit(X_train, y_train)

      # Predict using the baseline model
      y_pred_baseline = dummy_model.predict(X_test)

      # Evaluate the baseline model
      mae_baseline = mean_absolute_error(y_test, y_pred_baseline)
      print(f'Baseline Mean Absolute Error: {mae_baseline}')
      mse_baseline = mean_squared_error(y_test, y_pred_baseline)
      print(f'Baseline Mean Squared Error: {mse_baseline}')
      print(f'Baseline Root Mean Squared Error: {mse_baseline**0.5}')

      print("Hyperparameter Optimization Step")
      def objective(trial):
          params = {
              'n_estimators': trial.suggest_int('n_estimators', 100, 1000),
              'max_depth': trial.suggest_int('max_depth', 1, 150),
              'min_samples_split': trial.suggest_int('min_samples_split', 2, 30),
              'min_samples_leaf': trial.suggest_int('min_samples_leaf', 1, 10),
              'max_features': trial.suggest_categorical('max_features', ['sqrt', '
      ↪ log2', None])
          }
          rf_model = RandomForestRegressor(random_state=42, **params)
```

```

rf_model.fit(X_train, y_train)
y_pred = rf_model.predict(X_test)
return mean_squared_error(y_test, y_pred)

study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=100)

rf_best_params = study.best_params
print(f'Best Hyperparameters: {rf_best_params}')
# rf_best_params = {'n_estimators': 869, 'max_depth': 133, 'min_samples_split': 7, 'min_samples_leaf': 1, 'max_features': 'sqrt'}

# Use the best model for predictions
best_rf_model = RandomForestRegressor(random_state=42, **rf_best_params)
best_rf_model.fit(X_train, y_train)
feature_importance = best_rf_model.feature_importances_

# Get the feature names
feature_names = X_train.columns

# Create a DataFrame for feature importance
feature_importance_df = pd.DataFrame({
    'Feature': feature_names,
    'Importance': feature_importance
})

# Sort the DataFrame by importance in descending order
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Display the feature importance DataFrame
print("Feature Importance:")
print(feature_importance_df.head(10))

y_pred = best_rf_model.predict(X_test)
y_pred_rf = y_pred
errors = abs(y_pred - y_test)
print('Mean Absolute Error:', round(np.mean(errors), 2))
mape = 100 * (errors / y_test)
# Calculate and display accuracy
accuracy = 100 - np.mean(mape)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
feature_importance = best_rf_model.feature_importances_

```

```
# print(f'Feature Importance: {feature_importance}')
print(f'Mean Squared Error: {mse}')
print(f'Root Mean Squared Error: {mse**0.5}')
print(f'R-squared: {r2}')
print('Accuracy:', round(accuracy, 2), '%.')
```

Random Forest Regression

[I 2023-12-12 17:24:05,358] A new study created in memory with name: no-name-9c395f7c-237a-44aa-9db0-38fd27bcb159

Baseline Mean Absolute Error: 32745.338787888144

Baseline Mean Squared Error: 1649521929.7686503

Baseline Root Mean Squared Error: 40614.30695910802

Hyperparameter Optimization Step

[I 2023-12-12 17:24:08,477] Trial 0 finished with value: 647726445.7002499 and parameters: {'n_estimators': 691, 'max_depth': 121, 'min_samples_split': 4, 'min_samples_leaf': 6, 'max_features': 'sqrt'}. Best is trial 0 with value: 647726445.7002499.

[I 2023-12-12 17:24:09,446] Trial 1 finished with value: 1343747525.856675 and parameters: {'n_estimators': 654, 'max_depth': 12, 'min_samples_split': 9, 'min_samples_leaf': 7, 'max_features': 'log2'}. Best is trial 0 with value: 647726445.7002499.

[I 2023-12-12 17:24:10,696] Trial 2 finished with value: 1315632930.8800197 and parameters: {'n_estimators': 827, 'max_depth': 49, 'min_samples_split': 30, 'min_samples_leaf': 7, 'max_features': 'log2'}. Best is trial 0 with value: 647726445.7002499.

[I 2023-12-12 17:24:11,176] Trial 3 finished with value: 811977958.0078212 and parameters: {'n_estimators': 124, 'max_depth': 101, 'min_samples_split': 19, 'min_samples_leaf': 9, 'max_features': 'sqrt'}. Best is trial 0 with value: 647726445.7002499.

[I 2023-12-12 17:24:11,354] Trial 4 finished with value: 1578015420.342894 and parameters: {'n_estimators': 178, 'max_depth': 1, 'min_samples_split': 9, 'min_samples_leaf': 5, 'max_features': 'sqrt'}. Best is trial 0 with value: 647726445.7002499.

[I 2023-12-12 17:24:40,594] Trial 5 finished with value: 334715380.52894765 and parameters: {'n_estimators': 942, 'max_depth': 65, 'min_samples_split': 16, 'min_samples_leaf': 3, 'max_features': None}. Best is trial 5 with value: 334715380.52894765.

[I 2023-12-12 17:24:58,663] Trial 6 finished with value: 481442651.1890297 and parameters: {'n_estimators': 709, 'max_depth': 85, 'min_samples_split': 18, 'min_samples_leaf': 9, 'max_features': None}. Best is trial 5 with value: 334715380.52894765.

[I 2023-12-12 17:25:04,465] Trial 7 finished with value: 293494966.68798095 and parameters: {'n_estimators': 508, 'max_depth': 61, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 7 with value:

293494966.68798095.

[I 2023-12-12 17:25:27,567] Trial 8 finished with value: 425810412.5457371 and parameters: {'n_estimators': 789, 'max_depth': 32, 'min_samples_split': 15, 'min_samples_leaf': 6, 'max_features': None}. Best is trial 7 with value: 293494966.68798095.

[I 2023-12-12 17:25:30,169] Trial 9 finished with value: 511651103.8895053 and parameters: {'n_estimators': 479, 'max_depth': 37, 'min_samples_split': 18, 'min_samples_leaf': 3, 'max_features': 'sqrt'}. Best is trial 7 with value: 293494966.68798095.

[I 2023-12-12 17:25:36,461] Trial 10 finished with value: 195946876.28859562 and parameters: {'n_estimators': 410, 'max_depth': 140, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:43,839] Trial 11 finished with value: 196326328.62762597 and parameters: {'n_estimators': 434, 'max_depth': 146, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:47,399] Trial 12 finished with value: 213232479.50763762 and parameters: {'n_estimators': 342, 'max_depth': 150, 'min_samples_split': 7, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:48,637] Trial 13 finished with value: 908159440.7022152 and parameters: {'n_estimators': 338, 'max_depth': 149, 'min_samples_split': 2, 'min_samples_leaf': 3, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:51,539] Trial 14 finished with value: 236936068.05811873 and parameters: {'n_estimators': 342, 'max_depth': 123, 'min_samples_split': 12, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:52,666] Trial 15 finished with value: 1071486931.9538132 and parameters: {'n_estimators': 439, 'max_depth': 126, 'min_samples_split': 24, 'min_samples_leaf': 4, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:53,754] Trial 16 finished with value: 706289497.5173566 and parameters: {'n_estimators': 247, 'max_depth': 100, 'min_samples_split': 6, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:56,237] Trial 17 finished with value: 704616652.0272862 and parameters: {'n_estimators': 572, 'max_depth': 136, 'min_samples_split': 11, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:25:57,394] Trial 18 finished with value: 1052561958.7243686 and parameters: {'n_estimators': 412, 'max_depth': 104, 'min_samples_split': 5, 'min_samples_leaf': 4, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:16,700] Trial 19 finished with value: 304096000.98131555 and parameters: {'n_estimators': 560, 'max_depth': 135, 'min_samples_split': 22, 'min_samples_leaf': 2, 'max_features': None}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:26:17,380] Trial 20 finished with value: 1053001627.9721396 and parameters: {'n_estimators': 267, 'max_depth': 89, 'min_samples_split': 13, 'min_samples_leaf': 4, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:21,067] Trial 21 finished with value: 212939175.77006018 and parameters: {'n_estimators': 378, 'max_depth': 145, 'min_samples_split': 7, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:27,417] Trial 22 finished with value: 196064542.08750382 and parameters: {'n_estimators': 421, 'max_depth': 138, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:30,362] Trial 23 finished with value: 683512630.2411408 and parameters: {'n_estimators': 611, 'max_depth': 109, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:33,299] Trial 24 finished with value: 201462862.69076768 and parameters: {'n_estimators': 257, 'max_depth': 115, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:33,976] Trial 25 finished with value: 1413504251.5993834 and parameters: {'n_estimators': 471, 'max_depth': 135, 'min_samples_split': 9, 'min_samples_leaf': 10, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:36,582] Trial 26 finished with value: 674574083.6745112 and parameters: {'n_estimators': 543, 'max_depth': 136, 'min_samples_split': 4, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:37,913] Trial 27 finished with value: 906117854.4916769 and parameters: {'n_estimators': 415, 'max_depth': 131, 'min_samples_split': 8, 'min_samples_leaf': 3, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:51,557] Trial 28 finished with value: 224082711.13472813 and parameters: {'n_estimators': 297, 'max_depth': 87, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:52,720] Trial 29 finished with value: 609309819.7727457 and parameters: {'n_estimators': 218, 'max_depth': 117, 'min_samples_split': 5, 'min_samples_leaf': 5, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:53,464] Trial 30 finished with value: 417388072.20297015 and parameters: {'n_estimators': 113, 'max_depth': 143, 'min_samples_split': 29, 'min_samples_leaf': 2, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:26:55,725] Trial 31 finished with value: 201497666.4259894 and parameters: {'n_estimators': 188, 'max_depth': 114, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:26:59,175] Trial 32 finished with value: 200447948.56451353 and parameters: {'n_estimators': 299, 'max_depth': 123, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:01,196] Trial 33 finished with value: 677310280.9162338 and parameters: {'n_estimators': 400, 'max_depth': 125, 'min_samples_split': 3, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:02,224] Trial 34 finished with value: 912671950.3761346 and parameters: {'n_estimators': 305, 'max_depth': 142, 'min_samples_split': 6, 'min_samples_leaf': 3, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:03,092] Trial 35 finished with value: 1307661826.1257915 and parameters: {'n_estimators': 498, 'max_depth': 127, 'min_samples_split': 11, 'min_samples_leaf': 7, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:10,286] Trial 36 finished with value: 211087789.40577805 and parameters: {'n_estimators': 647, 'max_depth': 96, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:13,360] Trial 37 finished with value: 388670046.73784924 and parameters: {'n_estimators': 447, 'max_depth': 141, 'min_samples_split': 9, 'min_samples_leaf': 2, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:16,783] Trial 38 finished with value: 906917090.7158121 and parameters: {'n_estimators': 970, 'max_depth': 120, 'min_samples_split': 6, 'min_samples_leaf': 3, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:36,145] Trial 39 finished with value: 463302334.52915746 and parameters: {'n_estimators': 719, 'max_depth': 69, 'min_samples_split': 3, 'min_samples_leaf': 8, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:39,360] Trial 40 finished with value: 250792419.1306514 and parameters: {'n_estimators': 374, 'max_depth': 77, 'min_samples_split': 7, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:42,479] Trial 41 finished with value: 201285984.7837914 and parameters: {'n_estimators': 284, 'max_depth': 113, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:44,897] Trial 42 finished with value: 201925778.28560516 and parameters: {'n_estimators': 170, 'max_depth': 108, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:27:46,403] Trial 43 finished with value: 689078985.9193519 and parameters: {'n_estimators': 307, 'max_depth': 131, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.
[I 2023-12-12 17:27:50,902] Trial 44 finished with value: 196402324.898236 and parameters: {'n_estimators': 357, 'max_depth': 150, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:27:57,217] Trial 45 finished with value: 196840988.0530377 and parameters: {'n_estimators': 510, 'max_depth': 147, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:00,694] Trial 46 finished with value: 464349707.5310907 and parameters: {'n_estimators': 516, 'max_depth': 146, 'min_samples_split': 2, 'min_samples_leaf': 3, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:22,140] Trial 47 finished with value: 289546721.29249877 and parameters: {'n_estimators': 598, 'max_depth': 150, 'min_samples_split': 8, 'min_samples_leaf': 2, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:22,875] Trial 48 finished with value: 1384486095.639725 and parameters: {'n_estimators': 469, 'max_depth': 6, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:26,212] Trial 49 finished with value: 739920623.246479 and parameters: {'n_estimators': 874, 'max_depth': 54, 'min_samples_split': 20, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:27,173] Trial 50 finished with value: 1252595457.2971034 and parameters: {'n_estimators': 527, 'max_depth': 140, 'min_samples_split': 26, 'min_samples_leaf': 6, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:31,330] Trial 51 finished with value: 196970662.3711337 and parameters: {'n_estimators': 345, 'max_depth': 130, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:35,869] Trial 52 finished with value: 196457849.4658131 and parameters: {'n_estimators': 360, 'max_depth': 149, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:39,988] Trial 53 finished with value: 207327975.23411897 and parameters: {'n_estimators': 432, 'max_depth': 150, 'min_samples_split': 6, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:41,709] Trial 54 finished with value: 686735507.7481896 and parameters: {'n_estimators': 374, 'max_depth': 139, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.
[I 2023-12-12 17:28:48,173] Trial 55 finished with value: 196331334.48840213 and parameters: {'n_estimators': 460, 'max_depth': 146, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:28:49,628] Trial 56 finished with value: 770184758.0075164 and parameters: {'n_estimators': 392, 'max_depth': 37, 'min_samples_split': 14, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:28:50,815] Trial 57 finished with value: 1057500784.9657 and parameters: {'n_estimators': 454, 'max_depth': 138, 'min_samples_split': 2, 'min_samples_leaf': 4, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:01,223] Trial 58 finished with value: 336625439.010241 and parameters: {'n_estimators': 332, 'max_depth': 134, 'min_samples_split': 16, 'min_samples_leaf': 3, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:04,776] Trial 59 finished with value: 227649843.85300234 and parameters: {'n_estimators': 432, 'max_depth': 144, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:06,275] Trial 60 finished with value: 765818104.8938036 and parameters: {'n_estimators': 369, 'max_depth': 19, 'min_samples_split': 7, 'min_samples_leaf': 5, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:12,263] Trial 61 finished with value: 196810095.22972837 and parameters: {'n_estimators': 496, 'max_depth': 147, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:20,531] Trial 62 finished with value: 196647110.52510384 and parameters: {'n_estimators': 559, 'max_depth': 150, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:23,231] Trial 63 finished with value: 677173529.7911136 and parameters: {'n_estimators': 569, 'max_depth': 129, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:27,415] Trial 64 finished with value: 201905442.42661455 and parameters: {'n_estimators': 413, 'max_depth': 150, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:29,776] Trial 65 finished with value: 674801267.3254457 and parameters: {'n_estimators': 481, 'max_depth': 142, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:35,173] Trial 66 finished with value: 203469006.21814546 and parameters: {'n_estimators': 543, 'max_depth': 134, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:39,465] Trial 67 finished with value: 197143040.42726707 and parameters: {'n_estimators': 348, 'max_depth': 144, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:29:42,275] Trial 68 finished with value: 683303802.2719265 and parameters: {'n_estimators': 604, 'max_depth': 137, 'min_samples_split': 4, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:45,644] Trial 69 finished with value: 198536859.03068975 and parameters: {'n_estimators': 239, 'max_depth': 139, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:29:54,790] Trial 70 finished with value: 445448570.2127462 and parameters: {'n_estimators': 324, 'max_depth': 145, 'min_samples_split': 6, 'min_samples_leaf': 7, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:00,789] Trial 71 finished with value: 196348084.515822 and parameters: {'n_estimators': 486, 'max_depth': 150, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:08,394] Trial 72 finished with value: 199523819.48172596 and parameters: {'n_estimators': 645, 'max_depth': 133, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:10,971] Trial 73 finished with value: 673963829.1111953 and parameters: {'n_estimators': 467, 'max_depth': 121, 'min_samples_split': 3, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:11,599] Trial 74 finished with value: 1406859404.3483052 and parameters: {'n_estimators': 434, 'max_depth': 146, 'min_samples_split': 2, 'min_samples_leaf': 10, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:15,987] Trial 75 finished with value: 201966914.13551953 and parameters: {'n_estimators': 400, 'max_depth': 150, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:20,567] Trial 76 finished with value: 196370889.24122992 and parameters: {'n_estimators': 365, 'max_depth': 128, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:23,479] Trial 77 finished with value: 371004668.2135528 and parameters: {'n_estimators': 370, 'max_depth': 126, 'min_samples_split': 4, 'min_samples_leaf': 2, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:24,073] Trial 78 finished with value: 1335372084.7702785 and parameters: {'n_estimators': 358, 'max_depth': 140, 'min_samples_split': 8, 'min_samples_leaf': 8, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:27,373] Trial 79 finished with value: 197138913.7257997 and parameters: {'n_estimators': 280, 'max_depth': 129, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:30:29,285] Trial 80 finished with value: 684891955.9956049 and parameters: {'n_estimators': 417, 'max_depth': 136, 'min_samples_split': 6, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:34,827] Trial 81 finished with value: 196701461.7909922 and parameters: {'n_estimators': 390, 'max_depth': 143, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:40,402] Trial 82 finished with value: 196663936.0529217 and parameters: {'n_estimators': 458, 'max_depth': 147, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:46,186] Trial 83 finished with value: 199074235.6941391 and parameters: {'n_estimators': 492, 'max_depth': 142, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:52,601] Trial 84 finished with value: 199329075.33869213 and parameters: {'n_estimators': 525, 'max_depth': 132, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:53,821] Trial 85 finished with value: 727338684.6540284 and parameters: {'n_estimators': 318, 'max_depth': 150, 'min_samples_split': 18, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:30:59,202] Trial 86 finished with value: 196218000.24722683 and parameters: {'n_estimators': 417, 'max_depth': 138, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:03,607] Trial 87 finished with value: 204306260.83423108 and parameters: {'n_estimators': 419, 'max_depth': 120, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:05,788] Trial 88 finished with value: 672464338.0398114 and parameters: {'n_estimators': 452, 'max_depth': 138, 'min_samples_split': 3, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:22,697] Trial 89 finished with value: 221313846.6021739 and parameters: {'n_estimators': 399, 'max_depth': 126, 'min_samples_split': 7, 'min_samples_leaf': 1, 'max_features': None}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:27,783] Trial 90 finished with value: 202035352.22851175 and parameters: {'n_estimators': 356, 'max_depth': 136, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'sqrt'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:37,120] Trial 91 finished with value: 196610902.90252313 and parameters: {'n_estimators': 580, 'max_depth': 146, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value:

195946876.28859562.

[I 2023-12-12 17:31:44,905] Trial 92 finished with value: 196145957.74070337 and parameters: {'n_estimators': 485, 'max_depth': 142, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:50,298] Trial 93 finished with value: 199097287.54993477 and parameters: {'n_estimators': 476, 'max_depth': 142, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:55,718] Trial 94 finished with value: 197003033.98066035 and parameters: {'n_estimators': 437, 'max_depth': 132, 'min_samples_split': 3, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:31:57,546] Trial 95 finished with value: 685320767.7834516 and parameters: {'n_estimators': 380, 'max_depth': 139, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:32:05,100] Trial 96 finished with value: 196965641.8653651 and parameters: {'n_estimators': 510, 'max_depth': 147, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:32:07,653] Trial 97 finished with value: 674831208.1464149 and parameters: {'n_estimators': 487, 'max_depth': 143, 'min_samples_split': 3, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:32:12,297] Trial 98 finished with value: 214815323.6373171 and parameters: {'n_estimators': 419, 'max_depth': 93, 'min_samples_split': 4, 'min_samples_leaf': 1, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

[I 2023-12-12 17:32:14,178] Trial 99 finished with value: 692540957.2982405 and parameters: {'n_estimators': 337, 'max_depth': 81, 'min_samples_split': 6, 'min_samples_leaf': 2, 'max_features': 'log2'}. Best is trial 10 with value: 195946876.28859562.

Best Hyperparameters: {'n_estimators': 410, 'max_depth': 140, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'log2'}

Feature Importance:

	Feature	Importance
1	car_model_year	0.221827
0	car_driven	0.185057
300	car_model_Land Cruiser	0.035441
43	car_brand_Lexus	0.034641
32	car_brand_Hyundai	0.022447
50	car_brand_Mercedes	0.021416
235	car_model_Figo	0.014856
326	car_model_Mustang	0.013660
102	car_model_Accent	0.013349
462	car_model_Yaris	0.013110

Mean Absolute Error: 8454.4
Mean Squared Error: 195946876.28859562
Root Mean Squared Error: 13998.102596016204
R-squared: 0.8808643164493225
Accuracy: 80.61 %.

```
[ ]: print("Linear Regression")
X2 = cars.drop(['car_price', 'car_transmission', 'car_model', 'car_brand'],
              axis=1)
y2 = cars['car_price']
# Create a Linear Regression model
linear_model = LinearRegression()

# Split the data into training and testing sets
X_train2, X_test2, y_train2, y_test2 = train_test_split(X2, y2, test_size=0.3)

# Train the Linear Regression model
linear_model.fit(X_train2, y_train2)

# Make predictions on the test set
y_pred2 = linear_model.predict(X_test2)

# Evaluate the model
mse = mean_squared_error(y_test2, y_pred2)
r2 = r2_score(y_test2, y_pred2)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

Linear Regression

Mean Squared Error: 1394215604.8679054
R-squared: 0.20886080400806872

```
[ ]: print("Gradient boosting Regression")

# Generate a baseline using DummyRegressor
dummy_model = DummyRegressor(strategy='mean')
dummy_model.fit(X_train, y_train)
y_pred_baseline = dummy_model.predict(X_test)

# Evaluate baseline model
mae_baseline = mean_absolute_error(y_test, y_pred_baseline)
print(f'Baseline Mean Absolute Error: {mae_baseline}')
mse_baseline = mean_squared_error(y_test, y_pred_baseline)
print(f'Baseline Mean Squared Error: {mse_baseline}')
print(f'Baseline Root Mean Squared Error: {mse_baseline**0.5}')

print("Hyperparameter Optimization Step")
```

```

# objective function for Optuna
def objective(trial):
    params = {
        'n_estimators': trial.suggest_int('n_estimators', 100, 1000),
        'max_depth': trial.suggest_int('max_depth', 1, 150),
        'learning_rate': trial.suggest_float('learning_rate', 0.01, 0.2),
        'subsample': trial.suggest_float('subsample', 0.5, 1.0),
        'colsample_bytree': trial.suggest_float('colsample_bytree', 0.5, 1.0),
    }
    xgb_model = xgb.XGBRegressor(random_state=42, **params)
    xgb_model.fit(X_train, y_train)
    y_pred = xgb_model.predict(X_test)
    return mean_squared_error(y_test, y_pred)

# Create an Optuna study and optimize the objective function
study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=100)

# Get the best hyperparameters
xgb_best_params = study.best_params
print(f'Best Hyperparameters: {xgb_best_params}')
# xgb_best_params = {'n_estimators': 888, 'max_depth': 12, 'learning_rate': 0.
↳059265067129644175, 'subsample': 0.6710882119982516, 'colsample_bytree': 0.
↳9147549947728586}

# Train the model with the best hyperparameters
best_xgb_model = xgb.XGBRegressor(random_state=42, **xgb_best_params)
best_xgb_model.fit(X_train, y_train)
y_pred = best_xgb_model.predict(X_test)
y_pred_xgb = y_pred

# Evaluate the best model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
print(f'Mean Absolute Error: {mae}')
print(f'Mean Squared Error: {mse}')
print(f'Root Mean Squared Error: {mse**0.5}')
print(f'R-squared: {r2}')

# Calculate accuracy metrics
errors = abs(y_pred - y_test)
mape = 100 * (errors / y_test)
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')

```

[I 2023-12-12 17:32:20,925] A new study created in memory with name: no-name-b64f83d9-c309-4642-bc8d-17f719049684

Gradient boosting Regression

Baseline Mean Absolute Error: 32745.338787888144
Baseline Mean Squared Error: 1649521929.7686503
Baseline Root Mean Squared Error: 40614.30695910802
Hyperparameter Optimization Step

[I 2023-12-12 17:32:32,919] Trial 0 finished with value: 204479863.83711797 and parameters: {'n_estimators': 630, 'max_depth': 58, 'learning_rate': 0.07376617711138349, 'subsample': 0.5486640157115119, 'colsample_bytree': 0.6326548431302376}. Best is trial 0 with value: 204479863.83711797.
[I 2023-12-12 17:32:47,716] Trial 1 finished with value: 230107527.58014816 and parameters: {'n_estimators': 329, 'max_depth': 70, 'learning_rate': 0.05368036923143994, 'subsample': 0.980823545806747, 'colsample_bytree': 0.5344162065211611}. Best is trial 0 with value: 204479863.83711797.
[I 2023-12-12 17:32:51,558] Trial 2 finished with value: 179072160.2740198 and parameters: {'n_estimators': 943, 'max_depth': 16, 'learning_rate': 0.13812619132003406, 'subsample': 0.712199452044775, 'colsample_bytree': 0.8470823132098337}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:05,264] Trial 3 finished with value: 200375711.06501725 and parameters: {'n_estimators': 663, 'max_depth': 88, 'learning_rate': 0.0386857716306806, 'subsample': 0.525851808786641, 'colsample_bytree': 0.5906967764942724}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:10,051] Trial 4 finished with value: 200064285.41673547 and parameters: {'n_estimators': 749, 'max_depth': 22, 'learning_rate': 0.17258611852951475, 'subsample': 0.8085028022683538, 'colsample_bytree': 0.7022907358440036}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:10,540] Trial 5 finished with value: 1018569570.337943 and parameters: {'n_estimators': 154, 'max_depth': 1, 'learning_rate': 0.04163266616550462, 'subsample': 0.5865059111499702, 'colsample_bytree': 0.8549522351694697}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:22,098] Trial 6 finished with value: 255529101.15232596 and parameters: {'n_estimators': 486, 'max_depth': 79, 'learning_rate': 0.1761934465327932, 'subsample': 0.630136386845763, 'colsample_bytree': 0.611069181414892}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:37,505] Trial 7 finished with value: 197100251.2073858 and parameters: {'n_estimators': 372, 'max_depth': 74, 'learning_rate': 0.02565821653053102, 'subsample': 0.7842638914061356, 'colsample_bytree': 0.7790333925878399}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:38,756] Trial 8 finished with value: 249539695.48918492 and parameters: {'n_estimators': 585, 'max_depth': 3, 'learning_rate': 0.07525834342686008, 'subsample': 0.9952231608308255, 'colsample_bytree': 0.8983957327278695}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:33:49,552] Trial 9 finished with value: 236792950.89391083 and parameters: {'n_estimators': 534, 'max_depth': 85, 'learning_rate': 0.15891045705279763, 'subsample': 0.5442977289452868, 'colsample_bytree': 0.5699202946488475}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:34:26,631] Trial 10 finished with value: 203032206.175274 and parameters: {'n_estimators': 997, 'max_depth': 138, 'learning_rate': 0.12456215535271017, 'subsample': 0.6991088127773158, 'colsample_bytree':

0.9737483195732508}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:34:37,075] Trial 11 finished with value: 184058189.45357895 and parameters: {'n_estimators': 962, 'max_depth': 40, 'learning_rate': 0.010733426531717516, 'subsample': 0.7722329154175709, 'colsample_bytree': 0.7760065346433914}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:34:48,264] Trial 12 finished with value: 180604264.17189604 and parameters: {'n_estimators': 995, 'max_depth': 42, 'learning_rate': 0.010646535314077252, 'subsample': 0.7095047335380091, 'colsample_bytree': 0.7790451997463047}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:34:58,744] Trial 13 finished with value: 193987515.8477752 and parameters: {'n_estimators': 829, 'max_depth': 36, 'learning_rate': 0.11419622311360825, 'subsample': 0.6888462596396938, 'colsample_bytree': 0.8470920512924425}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:35:22,132] Trial 14 finished with value: 229658536.76529786 and parameters: {'n_estimators': 858, 'max_depth': 111, 'learning_rate': 0.14368001230842067, 'subsample': 0.685800879114726, 'colsample_bytree': 0.7165085427358692}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:35:30,555] Trial 15 finished with value: 191478491.4913136 and parameters: {'n_estimators': 885, 'max_depth': 25, 'learning_rate': 0.1959086859197891, 'subsample': 0.843549443257874, 'colsample_bytree': 0.8136064567280747}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:35:44,598] Trial 16 finished with value: 197769449.16373786 and parameters: {'n_estimators': 744, 'max_depth': 52, 'learning_rate': 0.09926069148234976, 'subsample': 0.6264458522567324, 'colsample_bytree': 0.9287894714692493}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:35:49,661] Trial 17 finished with value: 187880683.9298918 and parameters: {'n_estimators': 929, 'max_depth': 20, 'learning_rate': 0.13469452846166485, 'subsample': 0.7344900257697683, 'colsample_bytree': 0.7253706591147919}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:36:04,741] Trial 18 finished with value: 201445708.0755163 and parameters: {'n_estimators': 775, 'max_depth': 45, 'learning_rate': 0.09696420903344917, 'subsample': 0.8360660274281746, 'colsample_bytree': 0.8775388981566055}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:36:36,284] Trial 19 finished with value: 192700762.3470516 and parameters: {'n_estimators': 996, 'max_depth': 101, 'learning_rate': 0.010842340640761424, 'subsample': 0.7311256022644517, 'colsample_bytree': 0.8100192154030418}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:36:36,985] Trial 20 finished with value: 199352916.37495083 and parameters: {'n_estimators': 147, 'max_depth': 13, 'learning_rate': 0.07802738971124473, 'subsample': 0.6435665357085874, 'colsample_bytree': 0.9995494173596471}. Best is trial 2 with value: 179072160.2740198.
[I 2023-12-12 17:36:45,077] Trial 21 finished with value: 176034285.11191586 and parameters: {'n_estimators': 936, 'max_depth': 35, 'learning_rate': 0.011751459616146267, 'subsample': 0.7367778714701413, 'colsample_bytree': 0.781915232394115}. Best is trial 21 with value: 176034285.11191586.
[I 2023-12-12 17:36:52,923] Trial 22 finished with value: 182007350.80913603 and parameters: {'n_estimators': 874, 'max_depth': 33, 'learning_rate': 0.029394884771824392, 'subsample': 0.745225643899293, 'colsample_bytree':

0.7500374902354099}. Best is trial 21 with value: 176034285.11191586.
[I 2023-12-12 17:37:11,377] Trial 23 finished with value: 194294093.76189548 and parameters: {'n_estimators': 808, 'max_depth': 60, 'learning_rate': 0.056705451704519905, 'subsample': 0.6637619515711256, 'colsample_bytree': 0.826457778596482}. Best is trial 21 with value: 176034285.11191586.
[I 2023-12-12 17:37:14,249] Trial 24 finished with value: 182206113.4478264 and parameters: {'n_estimators': 908, 'max_depth': 12, 'learning_rate': 0.018733684148665365, 'subsample': 0.7164320058389596, 'colsample_bytree': 0.6732279738128434}. Best is trial 21 with value: 176034285.11191586.
[I 2023-12-12 17:37:18,882] Trial 25 finished with value: 173088934.94701123 and parameters: {'n_estimators': 708, 'max_depth': 29, 'learning_rate': 0.02933704295848803, 'subsample': 0.6043503852812897, 'colsample_bytree': 0.7729749973754705}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:37:23,903] Trial 26 finished with value: 176855035.4463434 and parameters: {'n_estimators': 688, 'max_depth': 30, 'learning_rate': 0.03342763763977619, 'subsample': 0.5852586728577721, 'colsample_bytree': 0.6682687691082791}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:37:28,878] Trial 27 finished with value: 174040755.41350827 and parameters: {'n_estimators': 710, 'max_depth': 31, 'learning_rate': 0.030988710986717818, 'subsample': 0.5028988856126074, 'colsample_bytree': 0.6706826094703771}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:37:35,737] Trial 28 finished with value: 187998099.87280968 and parameters: {'n_estimators': 452, 'max_depth': 50, 'learning_rate': 0.04772520459755165, 'subsample': 0.5039497624349677, 'colsample_bytree': 0.7420433218866751}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:37:47,554] Trial 29 finished with value: 194207723.048829 and parameters: {'n_estimators': 698, 'max_depth': 57, 'learning_rate': 0.06162052558026434, 'subsample': 0.5008275109482093, 'colsample_bytree': 0.6507178064161404}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:37:59,910] Trial 30 finished with value: 190092334.92324397 and parameters: {'n_estimators': 607, 'max_depth': 63, 'learning_rate': 0.024379639127533326, 'subsample': 0.5722302703649008, 'colsample_bytree': 0.6940467621450253}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:38:05,351] Trial 31 finished with value: 183509266.17887378 and parameters: {'n_estimators': 681, 'max_depth': 32, 'learning_rate': 0.035860526863527524, 'subsample': 0.5913098741038858, 'colsample_bytree': 0.647759067400207}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:38:09,969] Trial 32 finished with value: 174280921.10175082 and parameters: {'n_estimators': 716, 'max_depth': 29, 'learning_rate': 0.03575599658443237, 'subsample': 0.5476444223766668, 'colsample_bytree': 0.6610744744879948}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:38:11,623] Trial 33 finished with value: 181145908.7134415 and parameters: {'n_estimators': 629, 'max_depth': 8, 'learning_rate': 0.04574560001271236, 'subsample': 0.5406214406753505, 'colsample_bytree': 0.6213588545968514}. Best is trial 25 with value: 173088934.94701123.
[I 2023-12-12 17:38:14,485] Trial 34 finished with value: 172710599.2393609 and parameters: {'n_estimators': 530, 'max_depth': 24, 'learning_rate': 0.020467685143818016, 'subsample': 0.5597295299045244, 'colsample_bytree':

0.6997618967547052}. Best is trial 34 with value: 172710599.2393609.
[I 2023-12-12 17:38:17,300] Trial 35 finished with value: 170571607.8692759 and parameters: {'n_estimators': 549, 'max_depth': 24, 'learning_rate': 0.060910345653761716, 'subsample': 0.5618172213028187, 'colsample_bytree': 0.5188266909178267}. Best is trial 35 with value: 170571607.8692759.
[I 2023-12-12 17:38:19,270] Trial 36 finished with value: 165764937.60187286 and parameters: {'n_estimators': 407, 'max_depth': 22, 'learning_rate': 0.06400417655928553, 'subsample': 0.5205569959331067, 'colsample_bytree': 0.5404031667426825}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:20,530] Trial 37 finished with value: 172069693.68318364 and parameters: {'n_estimators': 279, 'max_depth': 19, 'learning_rate': 0.05846785260580809, 'subsample': 0.568138043767049, 'colsample_bytree': 0.5032256014890338}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:21,740] Trial 38 finished with value: 169697502.6010387 and parameters: {'n_estimators': 283, 'max_depth': 16, 'learning_rate': 0.06457399175909617, 'subsample': 0.5615880615369441, 'colsample_bytree': 0.5131891519336426}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:22,350] Trial 39 finished with value: 789606129.2171105 and parameters: {'n_estimators': 229, 'max_depth': 1, 'learning_rate': 0.06621455915761508, 'subsample': 0.5315401972843705, 'colsample_bytree': 0.502417726495519}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:23,844] Trial 40 finished with value: 166440824.4781451 and parameters: {'n_estimators': 327, 'max_depth': 17, 'learning_rate': 0.08400035380135468, 'subsample': 0.563031738463942, 'colsample_bytree': 0.5492953155797445}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:25,116] Trial 41 finished with value: 168747624.64625633 and parameters: {'n_estimators': 314, 'max_depth': 17, 'learning_rate': 0.08298731326403626, 'subsample': 0.5772610262868618, 'colsample_bytree': 0.5448847425700243}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:26,207] Trial 42 finished with value: 177652697.1991753 and parameters: {'n_estimators': 380, 'max_depth': 8, 'learning_rate': 0.08273771483125131, 'subsample': 0.6114919086884965, 'colsample_bytree': 0.5538121002536343}. Best is trial 36 with value: 165764937.60187286.
[I 2023-12-12 17:38:27,819] Trial 43 finished with value: 163828546.53070742 and parameters: {'n_estimators': 416, 'max_depth': 17, 'learning_rate': 0.06868936602060988, 'subsample': 0.5254351696480434, 'colsample_bytree': 0.5348554247342359}. Best is trial 43 with value: 163828546.53070742.
[I 2023-12-12 17:38:29,268] Trial 44 finished with value: 162917639.05829144 and parameters: {'n_estimators': 400, 'max_depth': 15, 'learning_rate': 0.08550262266434973, 'subsample': 0.5334605068452769, 'colsample_bytree': 0.5417482083535005}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:30,468] Trial 45 finished with value: 170920686.81433877 and parameters: {'n_estimators': 421, 'max_depth': 8, 'learning_rate': 0.08729741241407245, 'subsample': 0.5239137329737872, 'colsample_bytree': 0.5447976567829562}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:39,624] Trial 46 finished with value: 213440493.6626385 and parameters: {'n_estimators': 356, 'max_depth': 138, 'learning_rate': 0.07266658264836981, 'subsample': 0.5195173850096391, 'colsample_bytree':

0.5818080821735724}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:40,536] Trial 47 finished with value: 175382873.40963146 and parameters: {'n_estimators': 209, 'max_depth': 16, 'learning_rate': 0.09138968031097913, 'subsample': 0.5338823217208967, 'colsample_bytree': 0.5974033774909914}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:41,744] Trial 48 finished with value: 207376940.94356093 and parameters: {'n_estimators': 482, 'max_depth': 5, 'learning_rate': 0.07348354665790113, 'subsample': 0.5810479407345112, 'colsample_bytree': 0.5632207842435264}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:46,463] Trial 49 finished with value: 211050896.65759456 and parameters: {'n_estimators': 307, 'max_depth': 43, 'learning_rate': 0.10810058658340045, 'subsample': 0.6088535606736128, 'colsample_bytree': 0.5386900263179599}. Best is trial 44 with value: 162917639.05829144.
[I 2023-12-12 17:38:47,811] Trial 50 finished with value: 162898863.92896238 and parameters: {'n_estimators': 406, 'max_depth': 13, 'learning_rate': 0.08250233370869499, 'subsample': 0.5476978675392475, 'colsample_bytree': 0.5300935892377621}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:38:56,733] Trial 51 finished with value: 206053578.2383488 and parameters: {'n_estimators': 416, 'max_depth': 147, 'learning_rate': 0.08557206488770251, 'subsample': 0.5496358162569547, 'colsample_bytree': 0.5246597849333642}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:38:58,216] Trial 52 finished with value: 175313028.72883818 and parameters: {'n_estimators': 333, 'max_depth': 20, 'learning_rate': 0.09387847659734658, 'subsample': 0.518670457783464, 'colsample_bytree': 0.5731926047805801}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:38:59,652] Trial 53 finished with value: 164754587.9675566 and parameters: {'n_estimators': 398, 'max_depth': 14, 'learning_rate': 0.08062180496657416, 'subsample': 0.5419310159120212, 'colsample_bytree': 0.5323635258140921}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:00,996] Trial 54 finished with value: 167154272.53298435 and parameters: {'n_estimators': 389, 'max_depth': 12, 'learning_rate': 0.0696038726307506, 'subsample': 0.5409521339021074, 'colsample_bytree': 0.5939501437356236}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:02,149] Trial 55 finished with value: 272883208.6719514 and parameters: {'n_estimators': 457, 'max_depth': 3, 'learning_rate': 0.07968259016422649, 'subsample': 0.5208878508910832, 'colsample_bytree': 0.5276255976162494}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:08,815] Trial 56 finished with value: 211309948.52989823 and parameters: {'n_estimators': 504, 'max_depth': 40, 'learning_rate': 0.09897400844443738, 'subsample': 0.5522121241945702, 'colsample_bytree': 0.6085454944385756}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:11,193] Trial 57 finished with value: 175192730.06047842 and parameters: {'n_estimators': 422, 'max_depth': 24, 'learning_rate': 0.06924000647751197, 'subsample': 0.5951392087799257, 'colsample_bytree': 0.5595078682481185}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:12,109] Trial 58 finished with value: 212384955.92934448 and parameters: {'n_estimators': 235, 'max_depth': 11, 'learning_rate': 0.051798363086414445, 'subsample': 0.620100837851487, 'colsample_bytree':

0.5771524426524319}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:15,969] Trial 59 finished with value: 193727819.35590786 and parameters: {'n_estimators': 344, 'max_depth': 38, 'learning_rate': 0.07748533203396114, 'subsample': 0.5302985992460675, 'colsample_bytree': 0.5390747994914775}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:26,418] Trial 60 finished with value: 219933707.10501355 and parameters: {'n_estimators': 455, 'max_depth': 70, 'learning_rate': 0.08876953459034959, 'subsample': 0.6375555964405566, 'colsample_bytree': 0.5279560314974134}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:27,712] Trial 61 finished with value: 168863776.08834577 and parameters: {'n_estimators': 359, 'max_depth': 13, 'learning_rate': 0.06973192348802836, 'subsample': 0.544045218655931, 'colsample_bytree': 0.5862047238589285}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:28,579] Trial 62 finished with value: 609495485.5372561 and parameters: {'n_estimators': 396, 'max_depth': 1, 'learning_rate': 0.07604242262359981, 'subsample': 0.5153967228248342, 'colsample_bytree': 0.5593943905496963}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:29,866] Trial 63 finished with value: 176638101.1774782 and parameters: {'n_estimators': 392, 'max_depth': 11, 'learning_rate': 0.05368074160653403, 'subsample': 0.5435437851331061, 'colsample_bytree': 0.5919115589898002}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:36,674] Trial 64 finished with value: 207615598.605675 and parameters: {'n_estimators': 267, 'max_depth': 95, 'learning_rate': 0.06692865627952267, 'subsample': 0.5767270665973939, 'colsample_bytree': 0.5142880960346682}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:39,774] Trial 65 finished with value: 183735538.01309478 and parameters: {'n_estimators': 499, 'max_depth': 26, 'learning_rate': 0.09131740877843846, 'subsample': 0.5112953050692487, 'colsample_bytree': 0.5512883163093542}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:42,313] Trial 66 finished with value: 183528928.99124333 and parameters: {'n_estimators': 572, 'max_depth': 20, 'learning_rate': 0.10173767325678015, 'subsample': 0.5016924066387225, 'colsample_bytree': 0.5298254882206347}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:43,507] Trial 67 finished with value: 174299539.3580011 and parameters: {'n_estimators': 428, 'max_depth': 8, 'learning_rate': 0.08099878469480838, 'subsample': 0.5329871064361662, 'colsample_bytree': 0.5674530405066542}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:45,246] Trial 68 finished with value: 167548551.2912306 and parameters: {'n_estimators': 386, 'max_depth': 16, 'learning_rate': 0.07271131721981394, 'subsample': 0.558923936283162, 'colsample_bytree': 0.6036539624571152}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:50,474] Trial 69 finished with value: 203727409.48070636 and parameters: {'n_estimators': 307, 'max_depth': 48, 'learning_rate': 0.06286912605622244, 'subsample': 0.5921520882921012, 'colsample_bytree': 0.6236846422879471}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:52,532] Trial 70 finished with value: 189288383.0437542 and parameters: {'n_estimators': 193, 'max_depth': 36, 'learning_rate': 0.08570700032887522, 'subsample': 0.5697870109012374, 'colsample_bytree':

0.539778301705887}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:53,936] Trial 71 finished with value: 166761428.17038292 and parameters: {'n_estimators': 380, 'max_depth': 15, 'learning_rate': 0.0729004957333407, 'subsample': 0.5563922547322576, 'colsample_bytree': 0.5941965053316921}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:57,056] Trial 72 finished with value: 178179350.46789402 and parameters: {'n_estimators': 468, 'max_depth': 27, 'learning_rate': 0.07791776112229024, 'subsample': 0.537972113382718, 'colsample_bytree': 0.5909141186813825}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:39:58,085] Trial 73 finished with value: 224691508.42581627 and parameters: {'n_estimators': 368, 'max_depth': 6, 'learning_rate': 0.057861768002674334, 'subsample': 0.5555916051025767, 'colsample_bytree': 0.573004923660717}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:00,455] Trial 74 finished with value: 180830493.00564682 and parameters: {'n_estimators': 518, 'max_depth': 21, 'learning_rate': 0.0952459555839364, 'subsample': 0.5000429561398103, 'colsample_bytree': 0.5050731997201652}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:01,045] Trial 75 finished with value: 239365696.09162584 and parameters: {'n_estimators': 101, 'max_depth': 14, 'learning_rate': 0.06827477056226283, 'subsample': 0.521558998260577, 'colsample_bytree': 0.5526139046041324}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:02,096] Trial 76 finished with value: 173392773.11592695 and parameters: {'n_estimators': 329, 'max_depth': 11, 'learning_rate': 0.08142050386536011, 'subsample': 0.5992159775115349, 'colsample_bytree': 0.5001697531381731}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:11,743] Trial 77 finished with value: 204424873.40421 and parameters: {'n_estimators': 433, 'max_depth': 112, 'learning_rate': 0.07336555735530935, 'subsample': 0.5822600234058937, 'colsample_bytree': 0.5216083676128166}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:13,683] Trial 78 finished with value: 164423690.2636961 and parameters: {'n_estimators': 395, 'max_depth': 22, 'learning_rate': 0.06280358896085023, 'subsample': 0.5695042638199586, 'colsample_bytree': 0.538342303587254}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:16,696] Trial 79 finished with value: 174877916.12075946 and parameters: {'n_estimators': 406, 'max_depth': 31, 'learning_rate': 0.06168880604280532, 'subsample': 0.5695222853944352, 'colsample_bytree': 0.5353128794093289}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:18,206] Trial 80 finished with value: 171841125.3107747 and parameters: {'n_estimators': 260, 'max_depth': 23, 'learning_rate': 0.053663449979827936, 'subsample': 0.554607743287777, 'colsample_bytree': 0.5166644473817816}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:19,696] Trial 81 finished with value: 166897293.2751273 and parameters: {'n_estimators': 362, 'max_depth': 18, 'learning_rate': 0.06555396922224849, 'subsample': 0.5362171493549865, 'colsample_bytree': 0.565342416274202}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:21,304] Trial 82 finished with value: 173382632.7487265 and parameters: {'n_estimators': 363, 'max_depth': 19, 'learning_rate': 0.08953431567332858, 'subsample': 0.5281151870152418, 'colsample_bytree':

0.5562313357509456}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:24,061] Trial 83 finished with value: 173504490.90120307 and parameters: {'n_estimators': 444, 'max_depth': 27, 'learning_rate': 0.06340974096589713, 'subsample': 0.5128566270987467, 'colsample_bytree': 0.5659820607998574}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:24,953] Trial 84 finished with value: 288925842.52462226 and parameters: {'n_estimators': 336, 'max_depth': 5, 'learning_rate': 0.04750921024958174, 'subsample': 0.5862941668717919, 'colsample_bytree': 0.5467364108887839}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:27,771] Trial 85 finished with value: 188376874.5642109 and parameters: {'n_estimators': 293, 'max_depth': 34, 'learning_rate': 0.08359292788104997, 'subsample': 0.5664981292100146, 'colsample_bytree': 0.5294765803368799}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:29,585] Trial 86 finished with value: 167447797.33747607 and parameters: {'n_estimators': 477, 'max_depth': 17, 'learning_rate': 0.07444355318033452, 'subsample': 0.540297710428185, 'colsample_bytree': 0.5790449656042025}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:31,676] Trial 87 finished with value: 163694041.75534335 and parameters: {'n_estimators': 411, 'max_depth': 23, 'learning_rate': 0.05726509932969867, 'subsample': 0.5480616709702739, 'colsample_bytree': 0.5384637372767708}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:33,643] Trial 88 finished with value: 168980075.64664716 and parameters: {'n_estimators': 404, 'max_depth': 22, 'learning_rate': 0.04155549846291431, 'subsample': 0.5507415327294322, 'colsample_bytree': 0.5133279353368488}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:34,875] Trial 89 finished with value: 171388384.1914216 and parameters: {'n_estimators': 318, 'max_depth': 15, 'learning_rate': 0.05748463161099249, 'subsample': 0.6020057943374633, 'colsample_bytree': 0.541848773066799}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:36,168] Trial 90 finished with value: 166959600.57242513 and parameters: {'n_estimators': 433, 'max_depth': 9, 'learning_rate': 0.08587582657947286, 'subsample': 0.5124388097683437, 'colsample_bytree': 0.5316776087857422}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:38,618] Trial 91 finished with value: 177476509.42338118 and parameters: {'n_estimators': 352, 'max_depth': 29, 'learning_rate': 0.06990720251808633, 'subsample': 0.5272131375646798, 'colsample_bytree': 0.5688404391831807}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:40,159] Trial 92 finished with value: 166142058.093121 and parameters: {'n_estimators': 379, 'max_depth': 18, 'learning_rate': 0.06589270156478014, 'subsample': 0.561084363822098, 'colsample_bytree': 0.5584126105529779}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:42,136] Trial 93 finished with value: 171262387.54487723 and parameters: {'n_estimators': 387, 'max_depth': 23, 'learning_rate': 0.08032854522080085, 'subsample': 0.5607558822224379, 'colsample_bytree': 0.5508127695289587}. Best is trial 50 with value: 162898863.92896238.
[I 2023-12-12 17:40:43,733] Trial 94 finished with value: 165883319.7380634 and parameters: {'n_estimators': 447, 'max_depth': 14, 'learning_rate': 0.06101378466200673, 'subsample': 0.5819444989521456, 'colsample_bytree':

0.5196966342069795}. Best is trial 50 with value: 162898863.92896238.
 [I 2023-12-12 17:40:44,772] Trial 95 finished with value: 314164734.8607198 and parameters: {'n_estimators': 454, 'max_depth': 3, 'learning_rate': 0.059674695374935395, 'subsample': 0.5770930255725314, 'colsample_bytree': 0.5206344912122307}. Best is trial 50 with value: 162898863.92896238.
 [I 2023-12-12 17:40:46,451] Trial 96 finished with value: 173698046.19088113 and parameters: {'n_estimators': 490, 'max_depth': 10, 'learning_rate': 0.05353835445581905, 'subsample': 0.5452192234283928, 'colsample_bytree': 0.5101801803298514}. Best is trial 50 with value: 162898863.92896238.
 [I 2023-12-12 17:40:49,155] Trial 97 finished with value: 173848979.9579455 and parameters: {'n_estimators': 421, 'max_depth': 28, 'learning_rate': 0.05054020934585571, 'subsample': 0.5914404249800073, 'colsample_bytree': 0.5403860624227373}. Best is trial 50 with value: 162898863.92896238.
 [I 2023-12-12 17:40:51,637] Trial 98 finished with value: 166616575.83444664 and parameters: {'n_estimators': 546, 'max_depth': 20, 'learning_rate': 0.06597292620173688, 'subsample': 0.5258487432429673, 'colsample_bytree': 0.5212864188288127}. Best is trial 50 with value: 162898863.92896238.
 [I 2023-12-12 17:40:52,680] Trial 99 finished with value: 198749775.88212192 and parameters: {'n_estimators': 407, 'max_depth': 6, 'learning_rate': 0.07672823885165489, 'subsample': 0.6119313588946045, 'colsample_bytree': 0.5093232704243466}. Best is trial 50 with value: 162898863.92896238.

Best Hyperparameters: {'n_estimators': 406, 'max_depth': 13, 'learning_rate': 0.08250233370869499, 'subsample': 0.5476978675392475, 'colsample_bytree': 0.5300935892377621}

Mean Absolute Error: 8167.862993379832
 Mean Squared Error: 162898863.92896238
 Root Mean Squared Error: 12763.183926002257
 R-squared: 0.9009575050575315
 Accuracy: 83.17 %.

```
[ ]: # Model Visualization and Analysis
from sklearn.model_selection import cross_val_score
# # Use cross_val_score for cross-validation
scores = cross_val_score(best_rf_model, X, y, cv=5,
    ↪scoring='neg_mean_squared_error')

# Display the mean squared error scores
print("RF Cross-Validation Mean Squared Error:", -scores.mean())

scores2 = cross_val_score(best_xgb_model, X, y, cv=5,
    ↪scoring='neg_mean_squared_error')

# Display the mean squared error scores
print("RF Cross-Validation Mean Squared Error:", -scores2.mean())

# Random Forest Visualization
```

```

plt.figure(figsize=(12, 8))

# Actual Prices
plt.scatter(X_test['car_driven'], y_test, label='Actual Prices', alpha=0.5)

# Random Forest Predictions
y_pred_rf = best_rf_model.predict(X_test)
plt.scatter(X_test['car_driven'], y_pred_rf, label='RF Predictions', alpha=0.5)

plt.xlabel('Distance Traveled (car_driven)')
plt.ylabel('Car Price')
plt.title('Random Forest: Actual vs. Predicted Prices over Distance Traveled')
plt.legend()
plt.show()

# XGBoost Visualization
plt.figure(figsize=(12, 8))

# Actual Prices
plt.scatter(X_test['car_driven'], y_test, label='Actual Prices', alpha=0.5)

# XGBoost Predictions
y_pred_xgb = best_xgb_model.predict(X_test)
plt.scatter(X_test['car_driven'], y_pred_xgb, label='XGB Predictions', alpha=0.
↳5)

plt.xlabel('Distance Traveled (car_driven)')
plt.ylabel('Car Price')
plt.title('XGBoost: Actual vs. Predicted Prices over Distance Traveled')
plt.legend()
plt.show()

# Random Forest Visualization by Top 10 Car Brands
top_brands_rf = feature_importance_df['Feature'].str.extract(r'car_brand_(.*)').
↳dropna()[0]

plt.figure(figsize=(15, 8))

for brand in top_brands_rf:
    brand_indices = X_test[X_test[f'car_brand_{brand}'] == 1].index

    if not brand_indices.empty: # Check if there are samples for the current
↳brand
        plt.scatter(X_test.loc[brand_indices, 'car_driven'], y_test.
↳loc[brand_indices], label=f'Actual Prices - {brand}', alpha=0.5)

        y_pred_rf_brand = best_rf_model.predict(X_test.loc[brand_indices])

```

```

plt.scatter(X_test.loc[brand_indices, 'car_driven'], y_pred_rf_brand,
↳label=f'RF Predictions - {brand}', alpha=0.5)

plt.xlabel('Distance Traveled (car_driven)')
plt.ylabel('Car Price')
plt.title('Random Forest: Actual vs. Predicted Prices by Top 10 Car Brands over
↳Distance Traveled')
plt.legend()
plt.show()

# XGBoost Visualization by Top 10 Car Brands
top_brands_xgb = feature_importance_df['Feature'].str.extract(r'car_brand_(.
↳*)').dropna()[0]

plt.figure(figsize=(15, 8))

for brand in top_brands_xgb:
    brand_indices = X_test[X_test[f'car_brand_{brand}'] == 1].index

    if not brand_indices.empty: # Check if there are samples for the current
↳brand
        plt.scatter(X_test.loc[brand_indices, 'car_driven'], y_test.
↳loc[brand_indices], label=f'Actual Prices - {brand}', alpha=0.5)

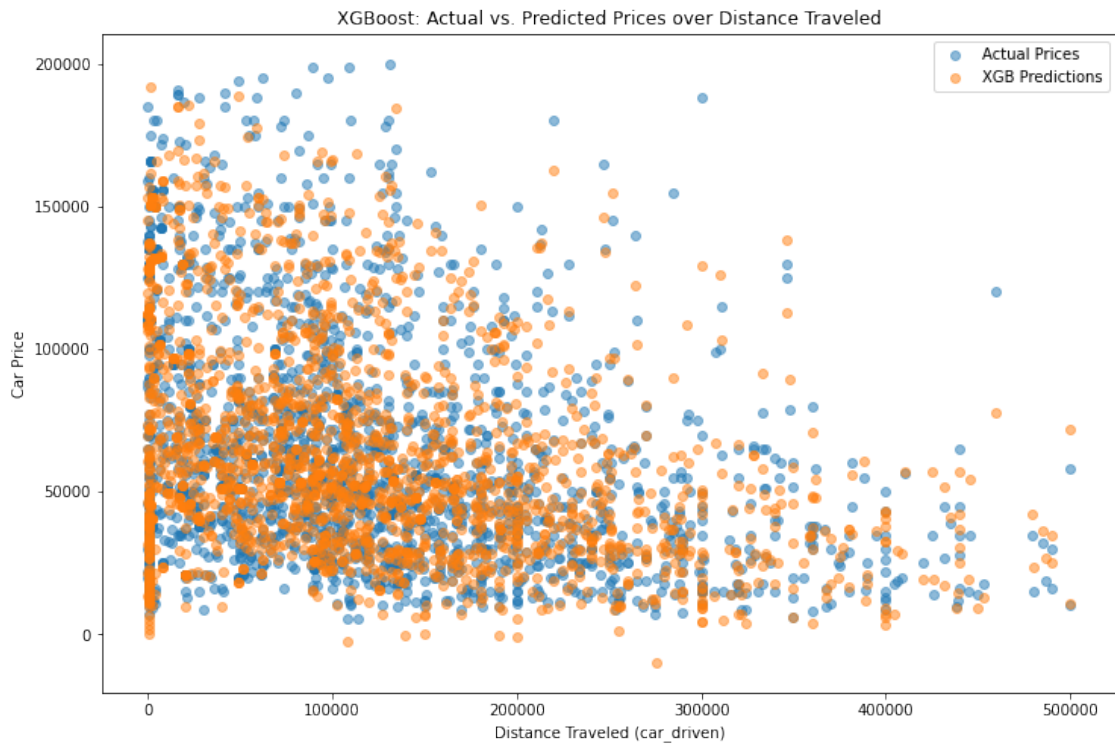
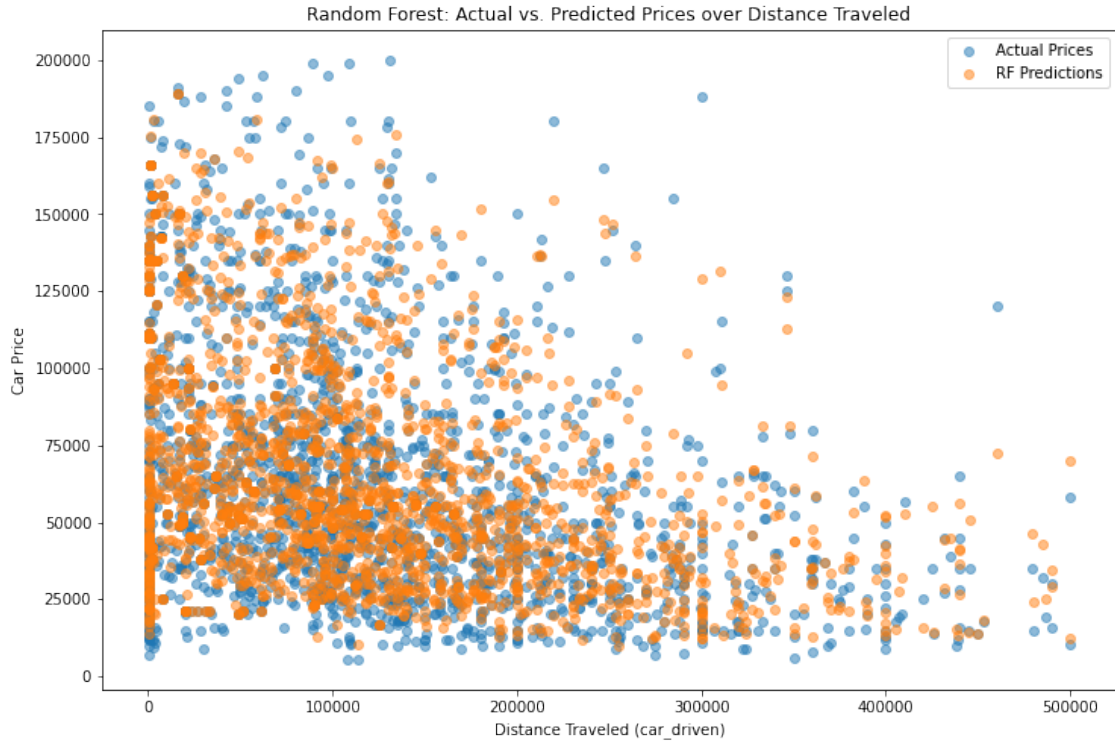
        y_pred_xgb_brand = best_xgb_model.predict(X_test.loc[brand_indices])
        plt.scatter(X_test.loc[brand_indices, 'car_driven'], y_pred_xgb_brand,
↳label=f'XGB Predictions - {brand}', alpha=0.5)

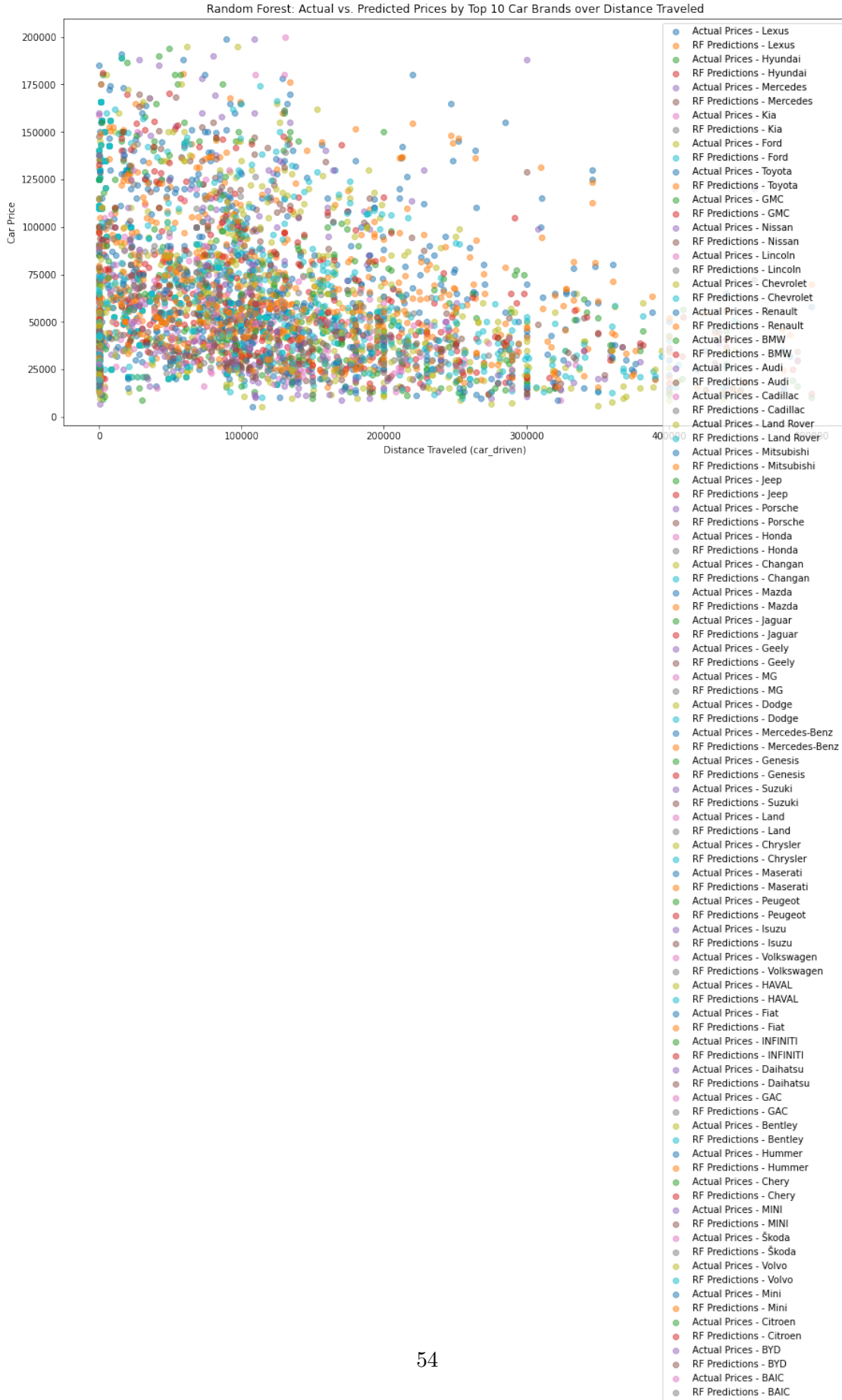
plt.xlabel('Distance Traveled (car_driven)')
plt.ylabel('Car Price')
plt.title('XGBoost: Actual vs. Predicted Prices by Top 10 Car Brands over
↳Distance Traveled')
plt.legend()
plt.show()

```

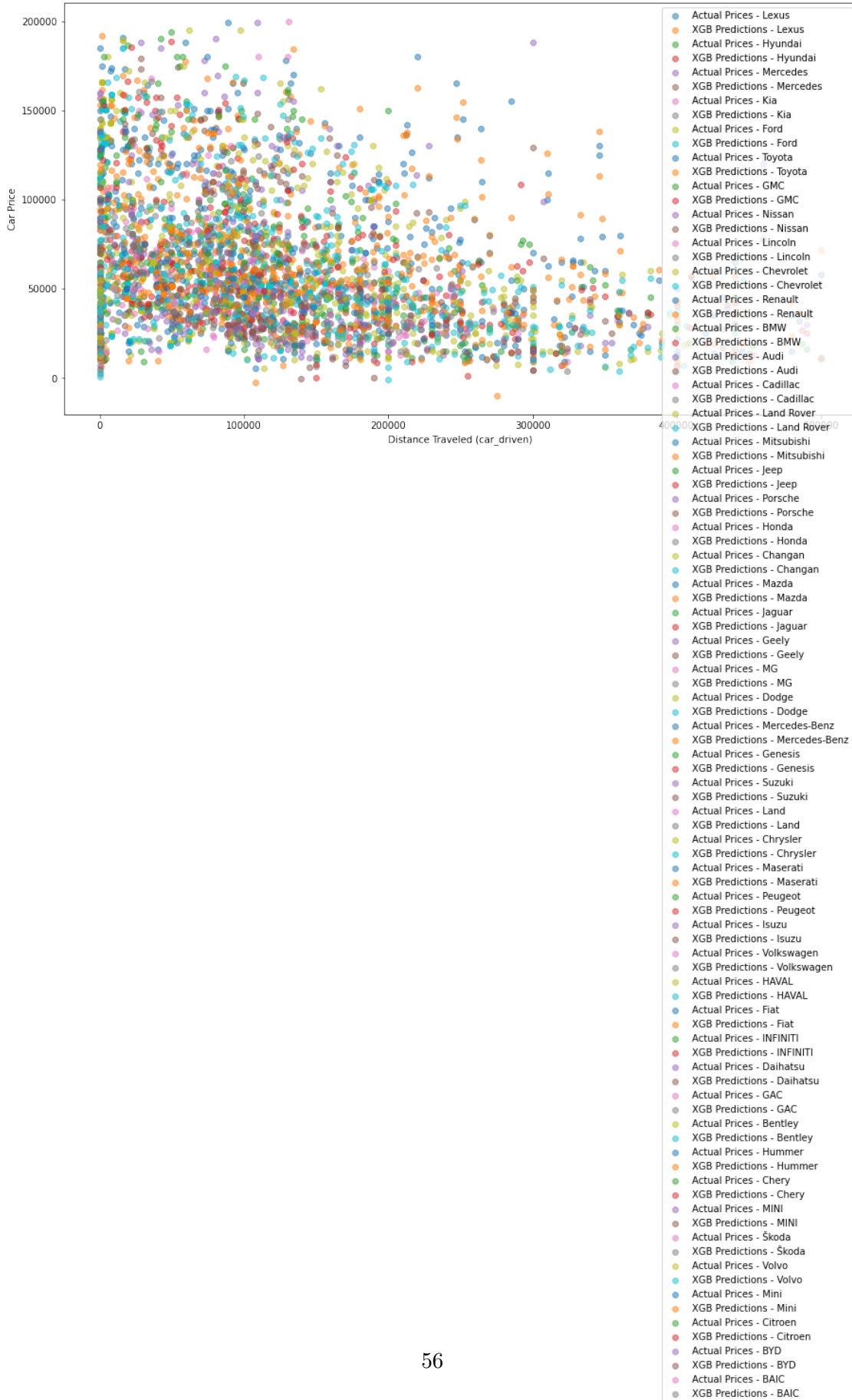
RF Cross-Validation Mean Squared Error: 238135352.76941156

RF Cross-Validation Mean Squared Error: 207643152.5300929





XGBoost: Actual vs. Predicted Prices by Top 10 Car Brands over Distance Traveled




```

[ ]: # Feature Engineering
# Depreciation rate of car based on mileage and accounting for inf values
cars['actual_depreciation_rate'] = cars['car_price'] / np.
↳where(cars['car_driven'] == 0, 1, cars['car_driven'])
# print(cars['actual_depreciation_rate'].describe())

# print(cars['car_brand'].value_counts())

X = cars[['car_brand', 'car_driven']]
y = cars['actual_depreciation_rate']

X_encoded = pd.get_dummies(X, columns=['car_brand'])
X_encoded = X_encoded.drop(['car_brand_Hummer', 'car_brand_Other'], axis=1)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.
↳3, random_state=42)
# print(X_test.columns)

# Random Forest model
rf_model = RandomForestRegressor(random_state=42, **rf_best_params)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

# XGBoost model
xgb_model = xgb.XGBRegressor(random_state=42, **xgb_best_params)
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)

# Evaluate models
mse_rf = mean_squared_error(y_test, y_pred_rf)
mse_xgb = mean_squared_error(y_test, y_pred_xgb)

print(f'RF Mean Squared Error: {mse_rf}')
print(f'XGB Mean Squared Error: {mse_xgb}')

# Compare model predictions with actual depreciation rates
results_rf = pd.DataFrame({'Actual_Price': y_test, 'Predicted_RF': y_pred_rf})
results_xgb = pd.DataFrame({'Actual_Price': y_test, 'Predicted_XGB':
↳y_pred_xgb})

# Outliers removal to make viewing easier
results_rf = results_rf[results_rf['Actual_Price'] < 5000]
results_xgb = results_xgb[results_xgb['Actual_Price'] < 5000]

```



```

# Visualize the results with a line of best fit
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.regplot(x='Actual_Price', y='Predicted_RF', data=results_rf,
            ↪line_kws={'color': 'red'})
plt.title('Random Forest Model: Actual_Price vs Predicted Depreciation Rate')
plt.xlabel('Actual Depreciation Rate')
plt.ylabel('Predicted Depreciation Rate (RF)')

plt.subplot(1, 2, 2)
sns.regplot(x='Actual_Price', y='Predicted_XGB', data=results_xgb,
            ↪line_kws={'color': 'red'})
plt.title('XGBoost Model: Actual_Price vs Predicted Depreciation Rate')
plt.xlabel('Actual Depreciation Rate')
plt.ylabel('Predicted Depreciation Rate (XGB)')

plt.tight_layout()
plt.show()

results_rf['Depreciation_Rate_RF'] = ((results_rf['Actual_Price'] -
            ↪results_rf['Predicted_RF']) / results_rf['Actual_Price']) * 100

# Extract encoded 'car_brand' columns
brand_columns = [col for col in X_encoded.columns if col.
            ↪startswith('car_brand_')]
results_rf['Car_Brand'] = X_test[brand_columns].idxmax(axis=1).apply(lambda x:
            ↪x.split('_')[-1])

# Group by Car Brand and calculate average depreciation rate
depreciation_by_brand_rf = results_rf.
            ↪groupby('Car_Brand')['Depreciation_Rate_RF'].mean().reset_index()

# Create a bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Car_Brand', y='Depreciation_Rate_RF',
            ↪data=depreciation_by_brand_rf)
plt.xticks(rotation=90)
plt.title('Average Depreciation Rate by Car Brand (Random Forest)')
plt.show()

# Calculate depreciation rate for XGBoost
results_xgb['Depreciation_Rate_XGB'] = ((results_xgb['Actual_Price'] -
            ↪results_xgb['Predicted_XGB']) / results_xgb['Actual_Price']) * 100

```

```

brand_columns_xgb = [col for col in X_encoded.columns if col.
    ↪startswith('car_brand_')]
results_xgb['Car_Brand'] = X_test[brand_columns_xgb].idxmax(axis=1).
    ↪apply(lambda x: x.split('_')[-1])

# Group by Car Brand and calculate average depreciation rate for XGB
depreciation_by_brand_xgb = results_xgb.
    ↪groupby('Car_Brand')['Depreciation_Rate_XGB'].mean().reset_index()

# Create a bar plot for XGBoost
plt.figure(figsize=(12, 6))
sns.barplot(x='Car_Brand', y='Depreciation_Rate_XGB',
    ↪data=depreciation_by_brand_xgb)
plt.xticks(rotation=90)
plt.title('Average Depreciation Rate (XGBoost) by Car Brand')
plt.show()

# Filter car brands with more than 50 cars
popular_brands = cars['car_brand'].value_counts()[cars['car_brand'].
    ↪value_counts() > 50].index

# Filter results for popular brands
popular_results_rf = results_rf[results_rf['Car_Brand'].isin(popular_brands)]
popular_results_xgb = results_xgb[results_xgb['Car_Brand'].isin(popular_brands)]

# Get the lowest 3 depreciation rates
lowest_depreciation_rf = popular_results_rf.
    ↪groupby('Car_Brand')['Depreciation_Rate_RF'].mean().nsmallest(3).
    ↪reset_index()
lowest_depreciation_xgb = popular_results_xgb.
    ↪groupby('Car_Brand')['Depreciation_Rate_XGB'].mean().nsmallest(3).
    ↪reset_index()

print("Lowest 3 Depreciation Rates (Random Forest):")
print(lowest_depreciation_rf)

print("\nLowest 3 Depreciation Rates (XGBoost):")
print(lowest_depreciation_xgb)

# Filter results for popular brands
popular_results_rf = results_rf[results_rf['Car_Brand'].isin(popular_brands)]
popular_results_xgb = results_xgb[results_xgb['Car_Brand'].isin(popular_brands)]

top_brands_rf = depreciation_by_brand_rf[depreciation_by_brand_rf['Car_Brand'].
    ↪isin(popular_brands)].nlargest(5, 'Depreciation_Rate_RF')

```

```

top_brands_xgb =
↳depreciation_by_brand_xgb[depreciation_by_brand_xgb['Car_Brand'].
↳isin(popular_brands)].nlargest(5, 'Depreciation_Rate_XGB')

low_brands_rf = depreciation_by_brand_rf[depreciation_by_brand_rf['Car_Brand'].
↳isin(popular_brands)].nsmallest(5, 'Depreciation_Rate_RF')
low_brands_xgb =
↳depreciation_by_brand_xgb[depreciation_by_brand_xgb['Car_Brand'].
↳isin(popular_brands)].nsmallest(5, 'Depreciation_Rate_XGB')

print("Top 5 Car Brands with Highest Depreciation Rates (Random Forest):")
print(top_brands_rf)

print("\nTop 5 Car Brands with Highest Depreciation Rates (XGBoost):")
print(top_brands_xgb)

plt.figure(figsize=(12, 6))
sns.barplot(x='Car_Brand', y='Depreciation_Rate_RF', data=top_brands_rf,
↳palette='viridis')
plt.title('Top 5 Car Brands with Highest Depreciation Rates (Random Forest)')
plt.xlabel('Car Brand')
plt.ylabel('Average Depreciation Rate')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(12, 6))
sns.barplot(x='Car_Brand', y='Depreciation_Rate_XGB', data=top_brands_xgb,
↳palette='viridis')
plt.title('Top 5 Car Brands with Highest Depreciation Rates (XGBoost)')
plt.xlabel('Car Brand')
plt.ylabel('Average Depreciation Rate')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(12, 6))
sns.barplot(x='Car_Brand', y='Depreciation_Rate_RF', data=low_brands_rf,
↳palette='viridis')
plt.title('Top 5 Car Brands with Lowest Depreciation Rates (Random Forest)')
plt.xlabel('Car Brand')
plt.ylabel('Average Depreciation Rate')
plt.xticks(rotation=45)
plt.show()

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

```

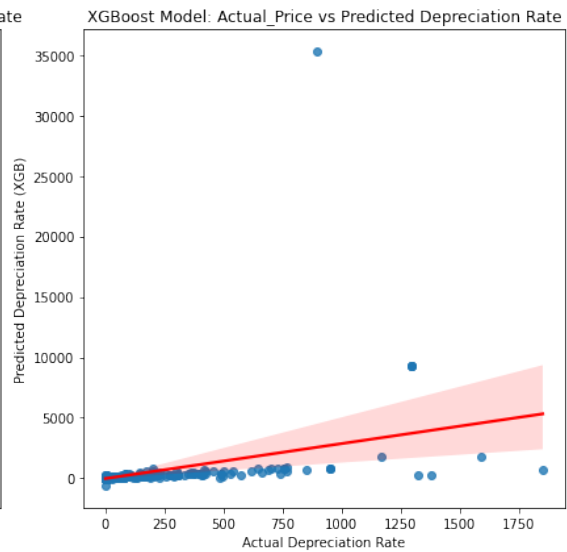
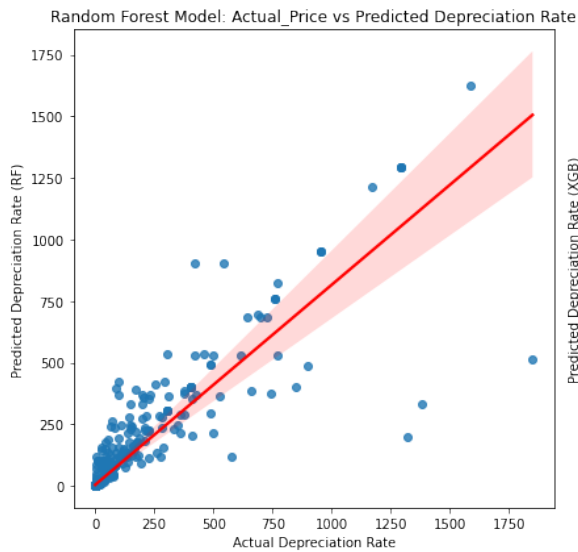
```

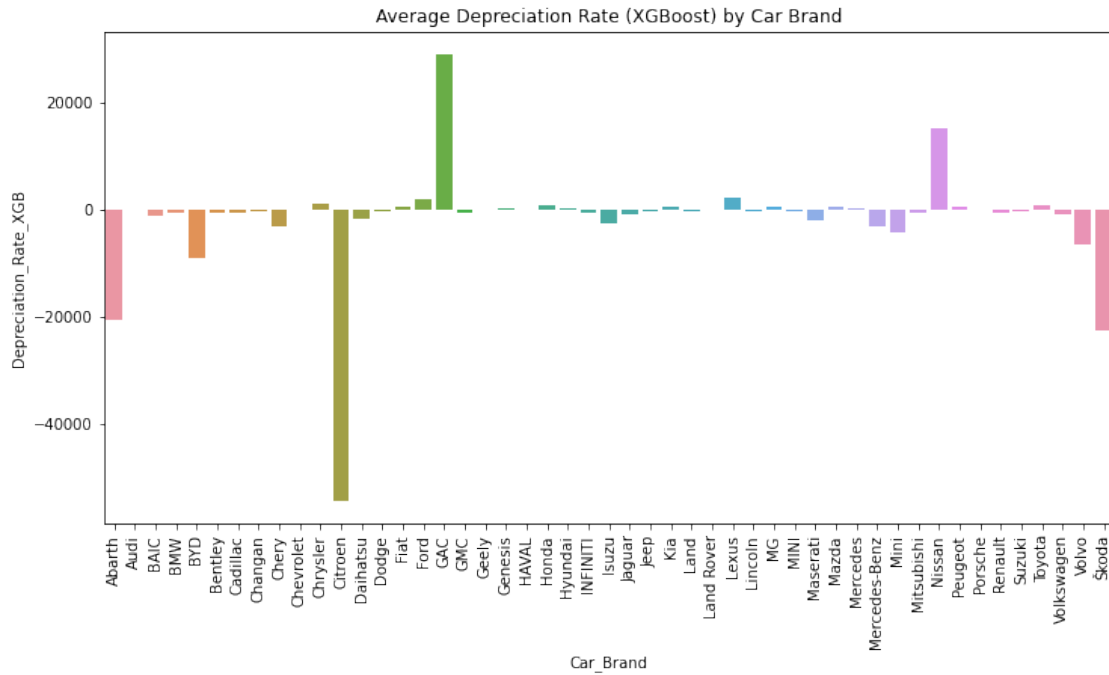
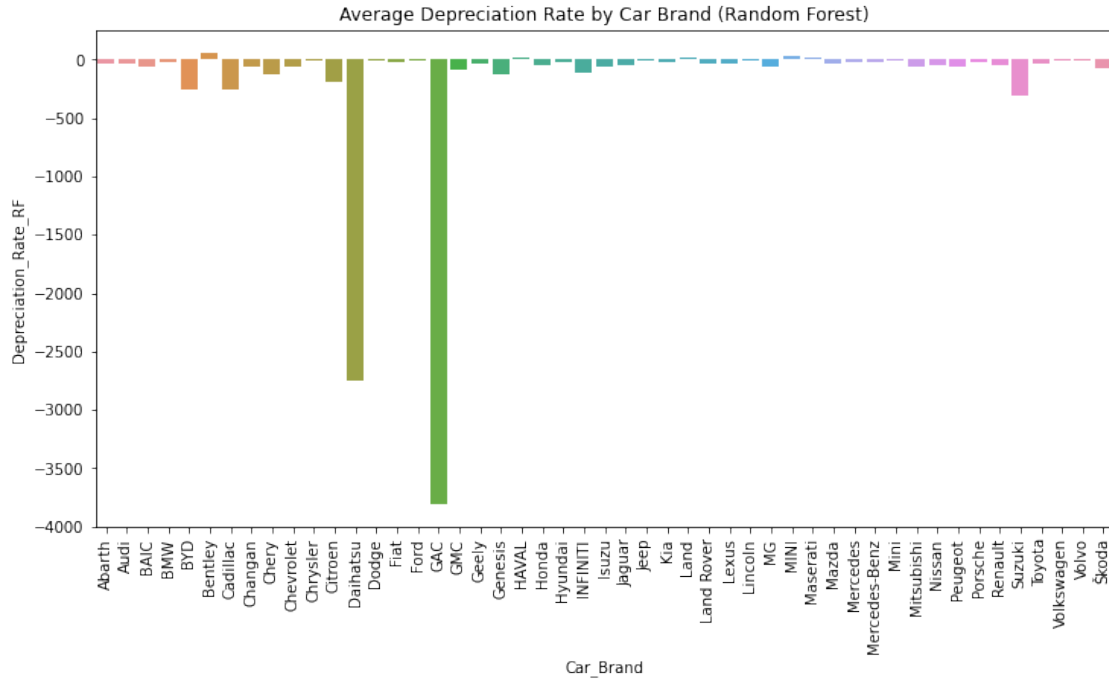
sns.barplot(x='Car_Brand', y='Depreciation_Rate_XGB', data=low_brands_xgb,
           palette='viridis')
plt.title('Top 5 Car Brands with Lowest Depreciation Rates (XGBoost)')
plt.xlabel('Car Brand')
plt.ylabel('Average Depreciation Rate')
plt.xticks(rotation=45)
plt.show()

```

RF Mean Squared Error: 3104.655806191558

XGB Mean Squared Error: 713289.2907992633





Lowest 3 Depreciation Rates (Random Forest):

Car_Brand	Depreciation_Rate_RF
0	GMC
	-88.152534

```

1 Chevrolet -66.994621
2 Mitsubishi -57.225495

```

Lowest 3 Depreciation Rates (XGBoost):

```

Car_Brand Depreciation_Rate_XGB
0 Renault -601.203239
1 GMC -598.801743
2 Mitsubishi -470.295595

```

Top 5 Car Brands with Highest Depreciation Rates (Random Forest):

```

Car_Brand Depreciation_Rate_RF
13 Dodge -6.205036
10 Chrysler -10.698451
31 Lincoln -12.328297
15 Ford -13.365128
26 Jeep -14.433586

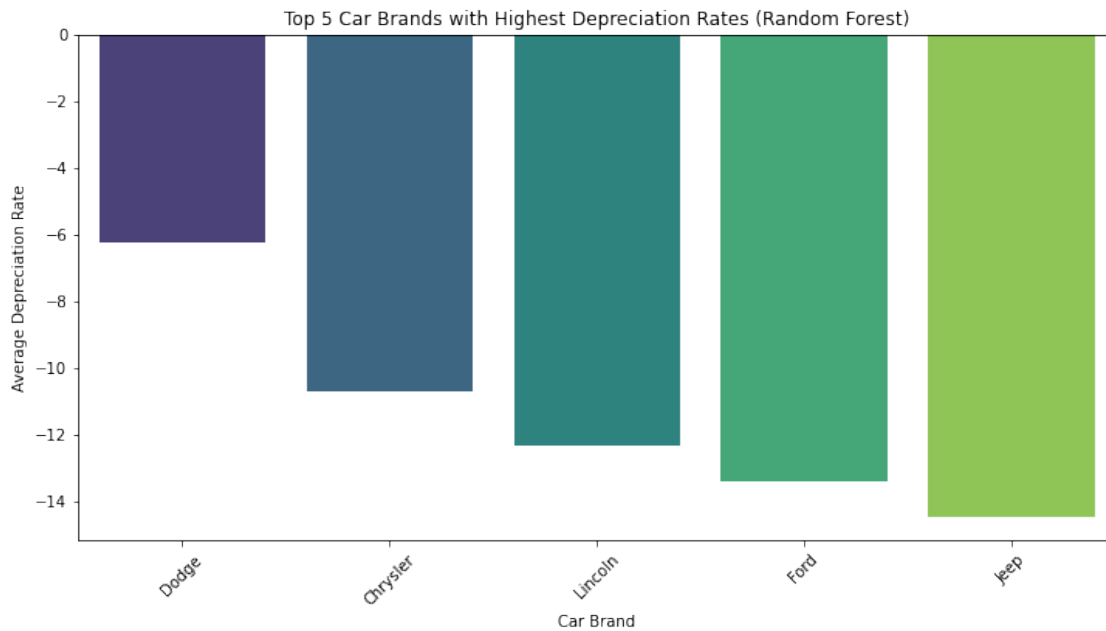
```

Top 5 Car Brands with Highest Depreciation Rates (XGBoost):

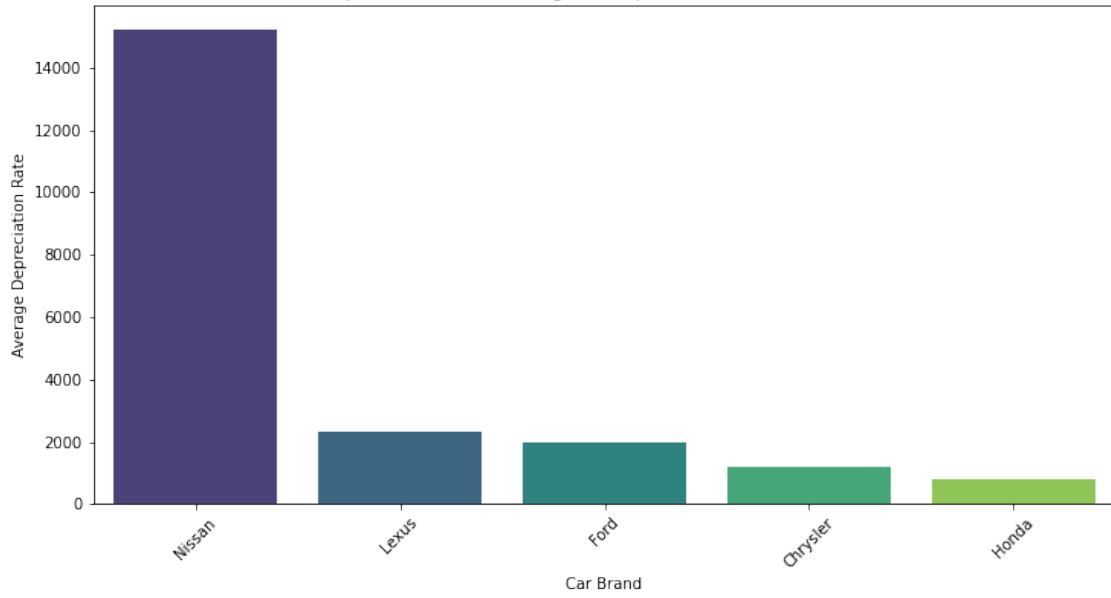
```

Car_Brand Depreciation_Rate_XGB
40 Nissan 15237.331452
30 Lexus 2329.814842
15 Ford 1982.935608
10 Chrysler 1177.632464
21 Honda 805.085869

```



Top 5 Car Brands with Highest Depreciation Rates (XGBoost)



Top 5 Car Brands with Lowest Depreciation Rates (Random Forest)

