USED CAR PRICE PREDICTION PROJECT

WELCOME TO OUR TEAM

- Ahmed Hatem
- Mina Sami
- Mohamed Osama
- Saad Hannout

PROJECT OVERVIEW

The project focuses on predicting used car prices using machine learning techniques. The key objective is to develop a model that can accurately estimate the price of a used car based on various attributes such as the car's brand, model age, fuel type, engine specifications, and other related features.

Read More

www.reallygreatsite.com

Project Overview

- Objective: The main goal is to predict the price of used cars by leveraging machine learning models.
- Dataset: The dataset includes various features related to the car, such as brand, fuel_type, transmission, horsepower, cylinder_count, and more. It also includes the target variable, price.

- **The broader relevance of the project is linked to the growing demand for accurate and dynamic pricing models in the used car market. This project aims to help both sellers and buyers make more informed decisions by providing insights into the pricing structure of used cars.**

BENEFITS OF THE PROJECT

This project brings significant benefits to various stakeholders involved in the used car market

For Sellers

- **Competitive Pricing**
- **Data-Driven Decisions**

For Buyers

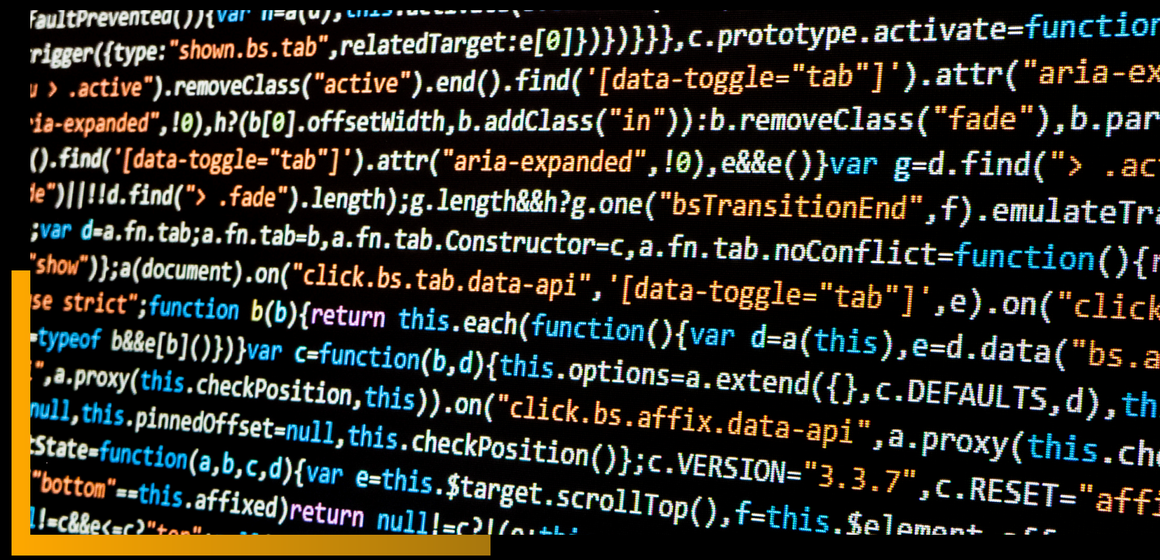
- **Informed Buying Decisions**
- **Better Bargaining Power**

Broader Market Insights

- **Trend Analysis**
- **Consumer Behavior**

www.reallygreatsite.com

TECHNICAL EXPLANATION OF THE CODE:



This section provides a comprehensive breakdown of the key steps and code used in the project, from data preprocessing and feature engineering to model training, evaluation, and visualization.

- Data preprocessing
- Model selection & Training
- Evaluation Metrics
- Visualization and Feature Correlation

DATA PREPROCESSING

- **Handling Missing Data:**

  Columns such as **fuel_type**, **accident**, and **clean_title** had missing values, which were filled with meaningful values. For example, missing **fuel_type** entries were replaced with **"Electricity"** and missing accident entries were filled with **"None reported."**

- **Encoding Categorical Variables:**

  - Categorical features like **brand**, **fuel_type**, and **transmission** were encoded numerically. Higher average prices for a brand were assigned higher numeric encodings to ensure that the model could interpret these variables meaningfully.

```
[103] df_train['fuel_type'].fillna('Electricity', inplace=True)
      df_test['fuel_type'].fillna('Electricity', inplace=True)

[104] df_train['accident'].fillna('None reported', inplace=True)
      df_test['accident'].fillna('None reported', inplace=True)

[105] df_train['clean_title'].fillna('No', inplace=True)
      df_test['clean_title'].fillna('No', inplace=True)

[106] df_train.head()
```

```
[111] def brand_encoder(brand):
          brand_mapping = {
              3: ['Bugatti', 'Lamborghini', 'Rolls-Royce', 'Bentley', 'McLaren', 'Ferrari', 'Aston
              2: ['Rivian', 'Porsche', 'Lucid', 'Maserati', 'Tesla', 'Maybach', 'Genesis', 'Land',
          }

          # Iterate through the mapping and return the  value
          for rating, brands in brand_mapping.items():
              if brand in brands:
                  return rating

          # Return 1 if the brand isn't found in the mapping
          return 1

      df_train['brand_encoded'] = df_train['brand'].apply(brand_encoder)
      df_test['brand_encoded'] = df_test['brand'].apply(brand_encoder)
```

```
[112] print('encoded_train' ,df_train['brand_encoded'].unique())
      print('encoded_test ' ,df_test ['brand_encoded'].unique())

      encoded_train [1 2 3]
      encoded_test  [2 1 3]
```

DATA PREPROCESSING

- **Feature Engineering:**

- Horsepower and Cylinder Count Extraction: Regular expressions were employed to extract numerical information such as **Horsepower** and **Cylinder Count** from text fields like **engine**.

```
pply extraction functions on train & test :
train['Horsepower'] = df_train['engine'].apply(extract_hp)
train['Displacement'] = df_train['engine'].apply(extract_displacement)
train['Engine Type'] = df_train['engine'].apply(extract_engine_type)
train['Cylinder Count'] = df_train['engine'].apply(extract_cylinder_count
train['Fuel Type'] = df_train['engine'].apply(extract_fuel_type)
```

```
import re

def extract_hp(engine):
    match = re.search(r'(\d+(\.\d+)?)HP', engine)
    return float(match.group(1)) if match else None

def extract_displacement(engine):
    match = re.search(r'(\d+\.\d+)L|(\d+\.\d+) Liter', engine)
    return float(match.group(1)) if match else None

def extract_engine_type(engine):
    match = re.search(r'(V\d+|I\d+|Flat \d+|Straight \d+)', engine)
    return match.group(1) if match else None

def extract_cylinder_count(engine):
    match = re.search(r'(\d+) Cylinder', engine)
    return int(match.group(1)) if match else None

def extract_fuel_type(engine):
    fuel_types = ['Gasoline', 'Diesel', 'Electric', 'Hybrid', 'Flex F
    for fuel in fuel_types:
        if fuel in engine:
            return fuel
    return None

def extract_displacement(engine):
    match = re.search(r'(\d+\.\d+)L|(\d+\.\d+) Liter', engine)
    if match:
        return float(match.group(1)) if match.group(1) else float(mat
    return None
```

- Model Age Calculation: The **model_age** feature was created by subtracting the car's **model_year** from the current year (2024). This allows the model to consider the car's age rather than just its manufacturing year.

```
[146] #  replace model_year column with the age of the car as (model_age) :-

      current_year = 2024

      df_train['model_age'] = current_year - df_train['model_year']
      df_test['model_age'] = current_year - df_test['model_year']

      df_train.drop('model_year', axis=1, inplace=True)
      df_test.drop('model_year', axis=1, inplace=True)
```

MODEL SELECTION AND TRAINING

Linear Regression

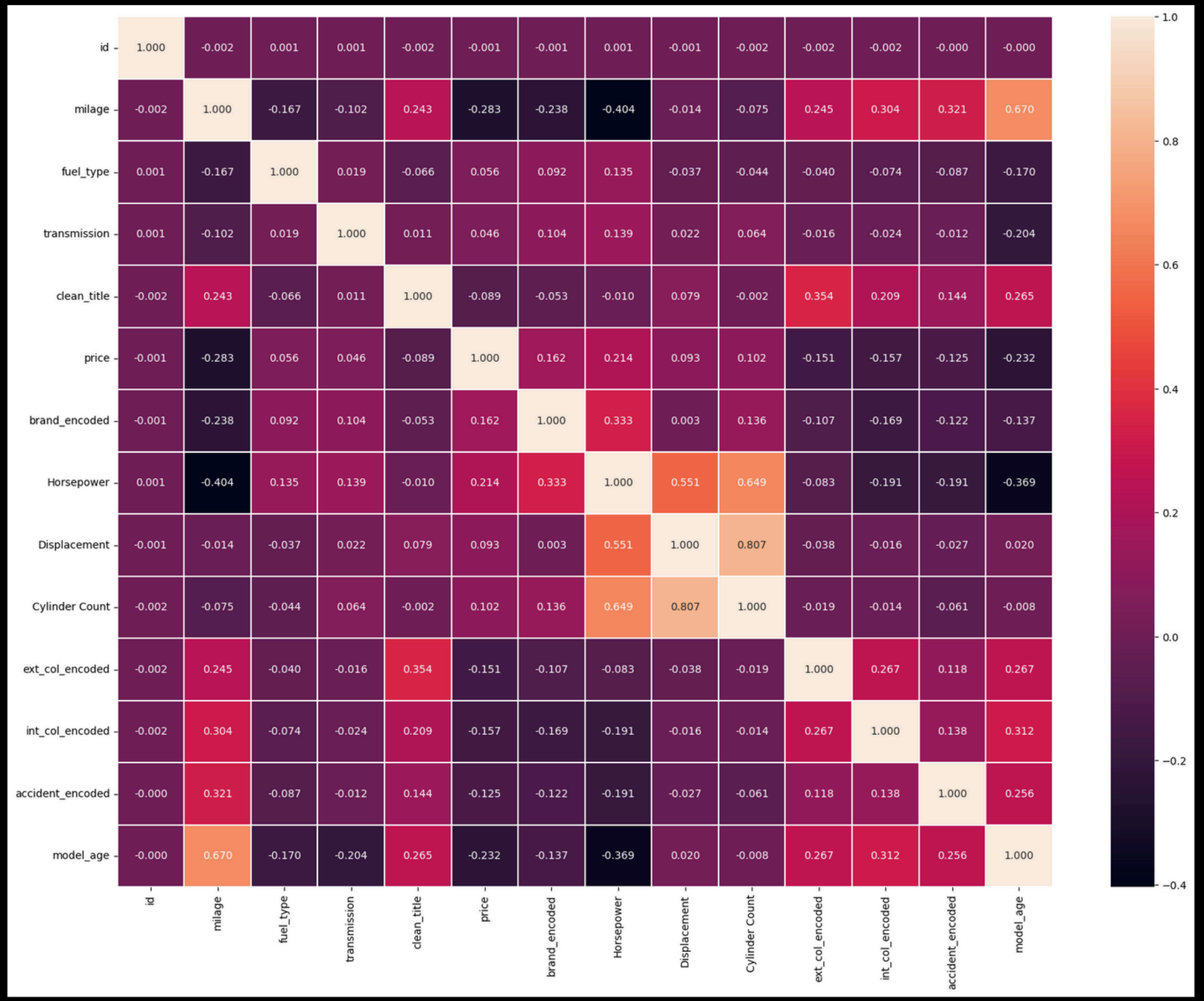
```
R-squared: 0.119536
Mean Squared Error (MSE): 4896381178.187447
Root Mean Squared Error (RMSE): 69974.146498
Mean Absolute Error (MAE): 23257.986219
```

XGBRegressor

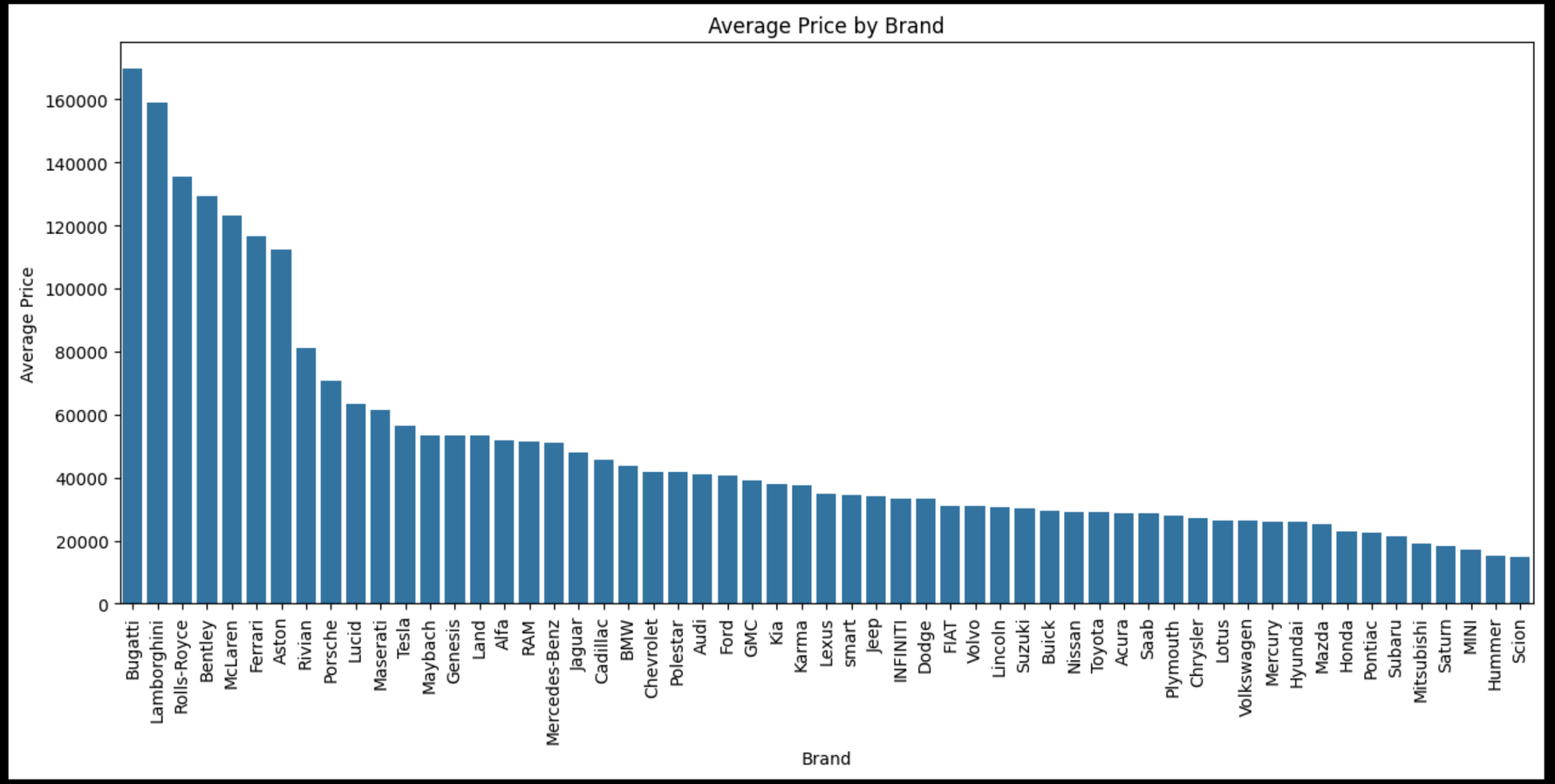
```
R-squared: 0.076129
Mean Squared Error (MSE): 5137773672.505373
Root Mean Squared Error (RMSE): 71678.264994
Mean Absolute Error (MAE): 20395.526138
```

Visualization and Feature Correlation

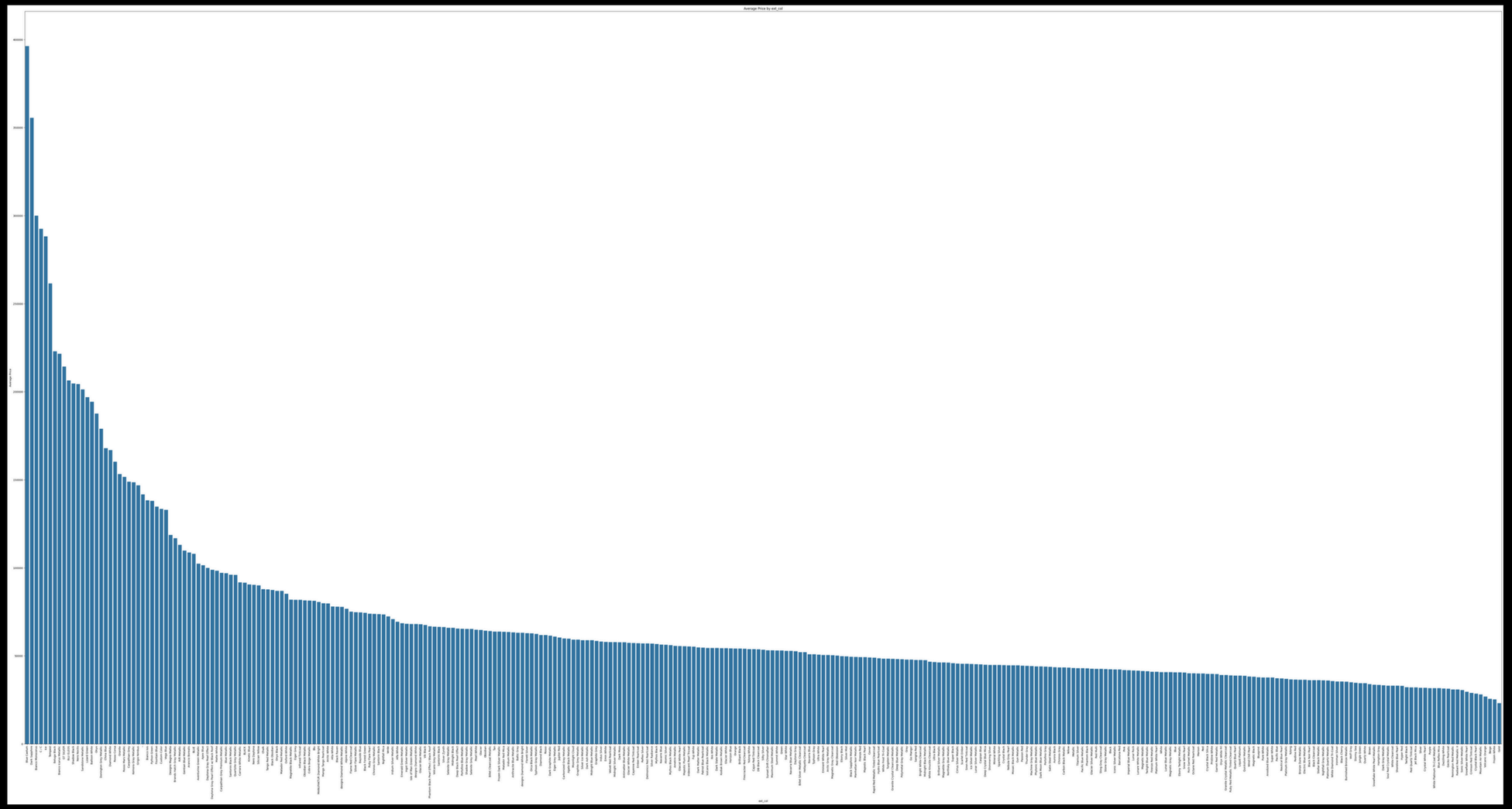
A heatmap was generated to analyze the correlation between numerical features. This helps identify highly correlated features that might cause multicollinearity issues in the model. Dropping highly correlated features, like **Displacement** (which correlated with **Cylinder Count**), helped improve model performance.



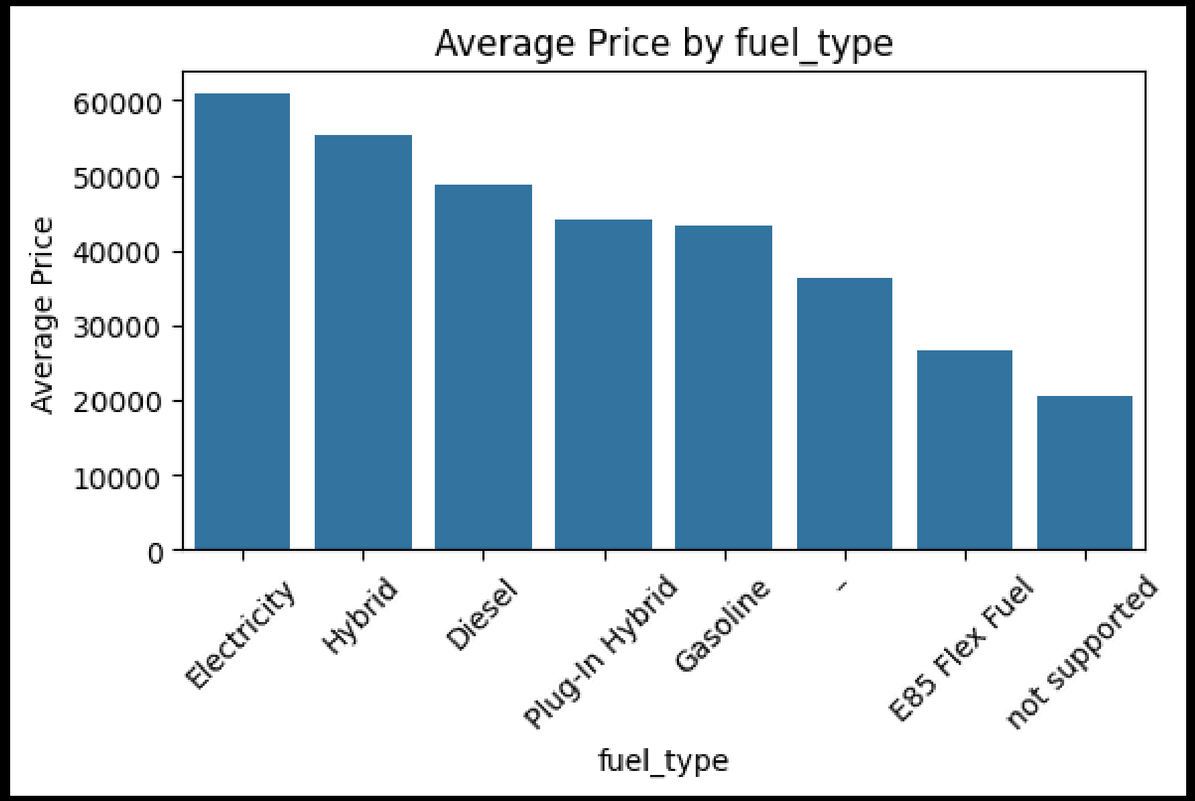
Visualization:

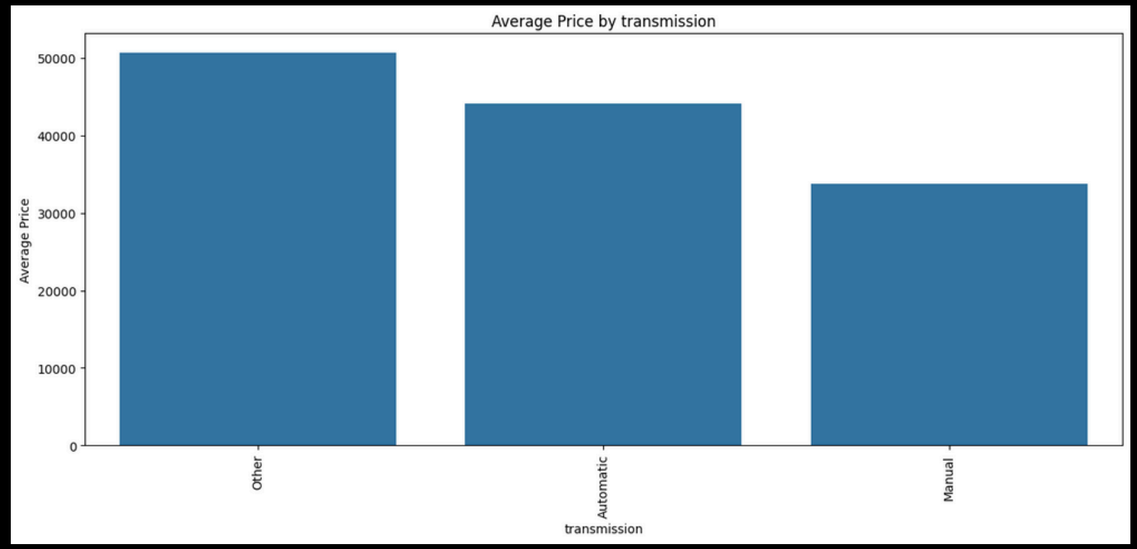


Visualization:

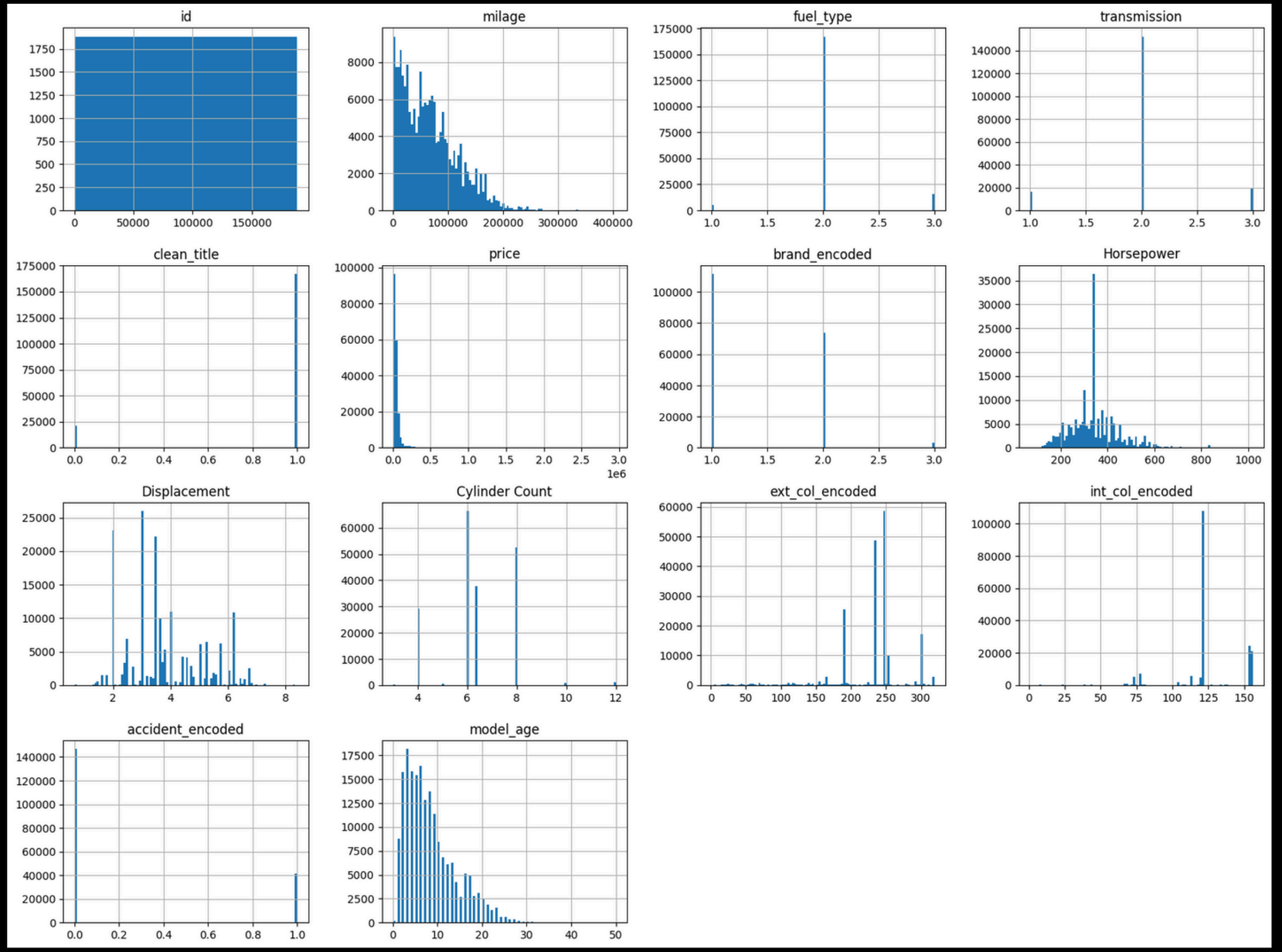


Visualization:





Visualization:



UsedP oCwer aBI Dres kPtoprices

brand
All

model_year
All

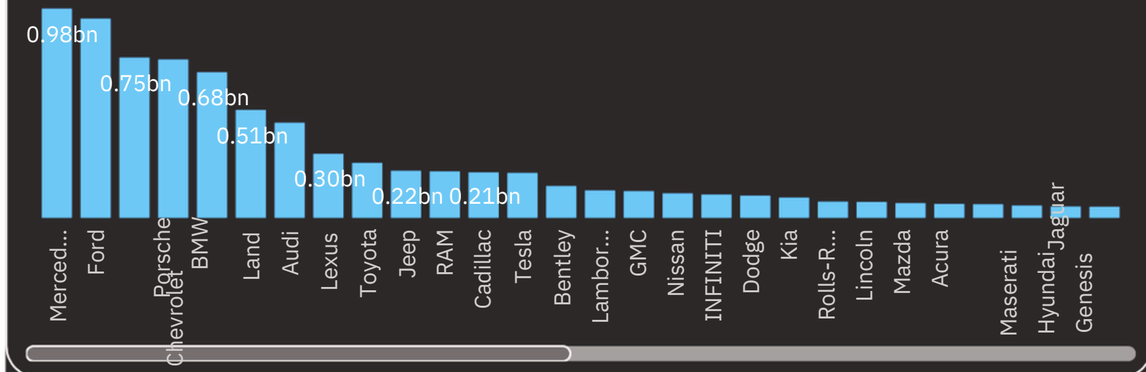
43.88K
Average of price

343.26
Average of Horsepower

8
Count of fuel_type

12388M
Sum of milage

Sum of price by brand



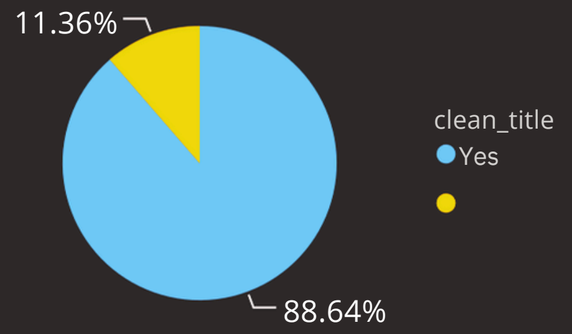
Average of price by fuel_type

Plug-In Hybrid
44150.97
Average of price

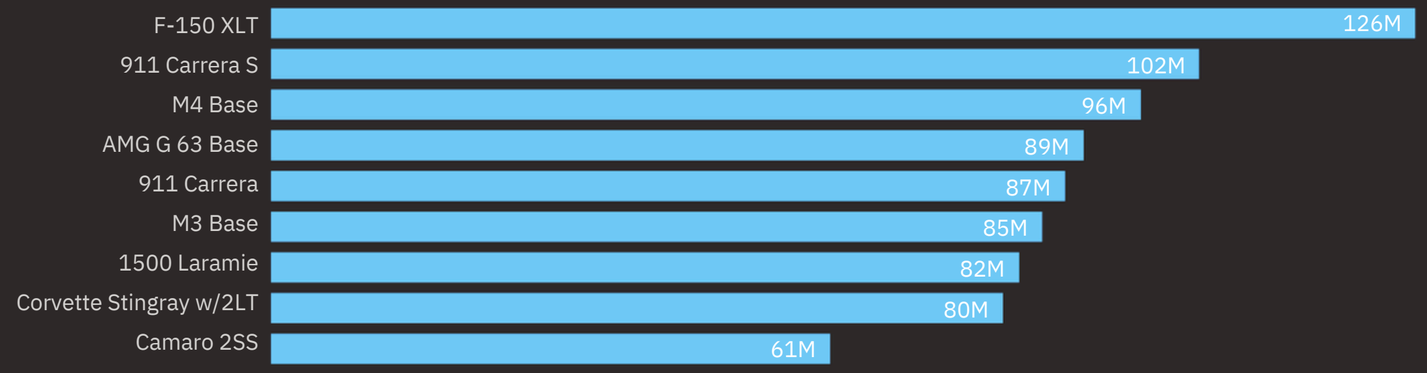
not supported
20692.73
Average of price

Hybrid
55473.45
Average of price

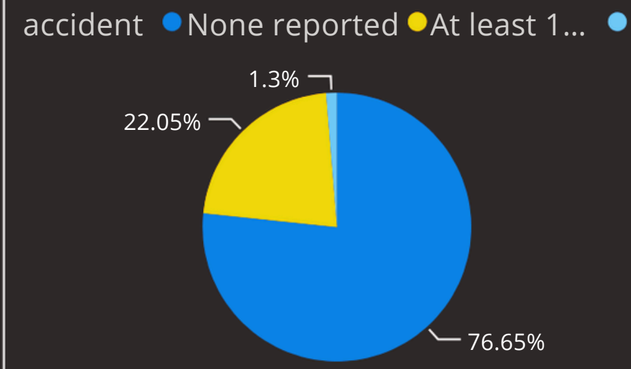
clean_title



Average of price by model



%GT Count of accident by accident



SUMMARY

This technical explanation outlines how the data was cleaned, processed, and transformed into features that could be used by machine learning models. We employed different models (**Linear Regression** and **XGBoost**) and evaluated their performance using standard regression metrics (RMSE, R^2, and MAE). Visualization techniques such as bar plots and correlation matrices were critical for understanding feature relationships and improving model accuracy.

HOW TO IMPROVE THE MODEL

1. Hyperparameter Tuning

- **Description**: Adjusting the hyperparameters of models like XGBoost can significantly improve performance. Techniques such as grid search or randomized search can be used to identify the optimal parameters (e.g., learning rate, depth of trees) for the model.

- **Benefit**: This can help improve model accuracy, reduce overfitting, and achieve better predictive performance.

HOW TO IMPROVE THE MODEL

Feature Selection:

- **Description**: Feature selection involves identifying and using only the most important features for the model. Methods such as recursive feature elimination (RFE) or feature importance rankings can be used to remove redundant or irrelevant features.

- **Benefit**: This can simplify the model, reduce overfitting, and increase interpretability while maintaining or improving accuracy.

THANK YOU

FOR YOUR ATTENTION