Name: Joshua Jebapragashan
Student Reference Number: 10900339

IN PARTNERSHIP WITH PLYMOUTH UNIVERSITY

Module Code: PUSL2078	Module Name: Statistics for data science
Deadline Date: 9th April	Member of staff responsible for coursework: Miss. Kavishka Rajapaksha

Coursework Title: Final Report
Program: BSc (Hons) Data Science, PU

Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook

As part of a team

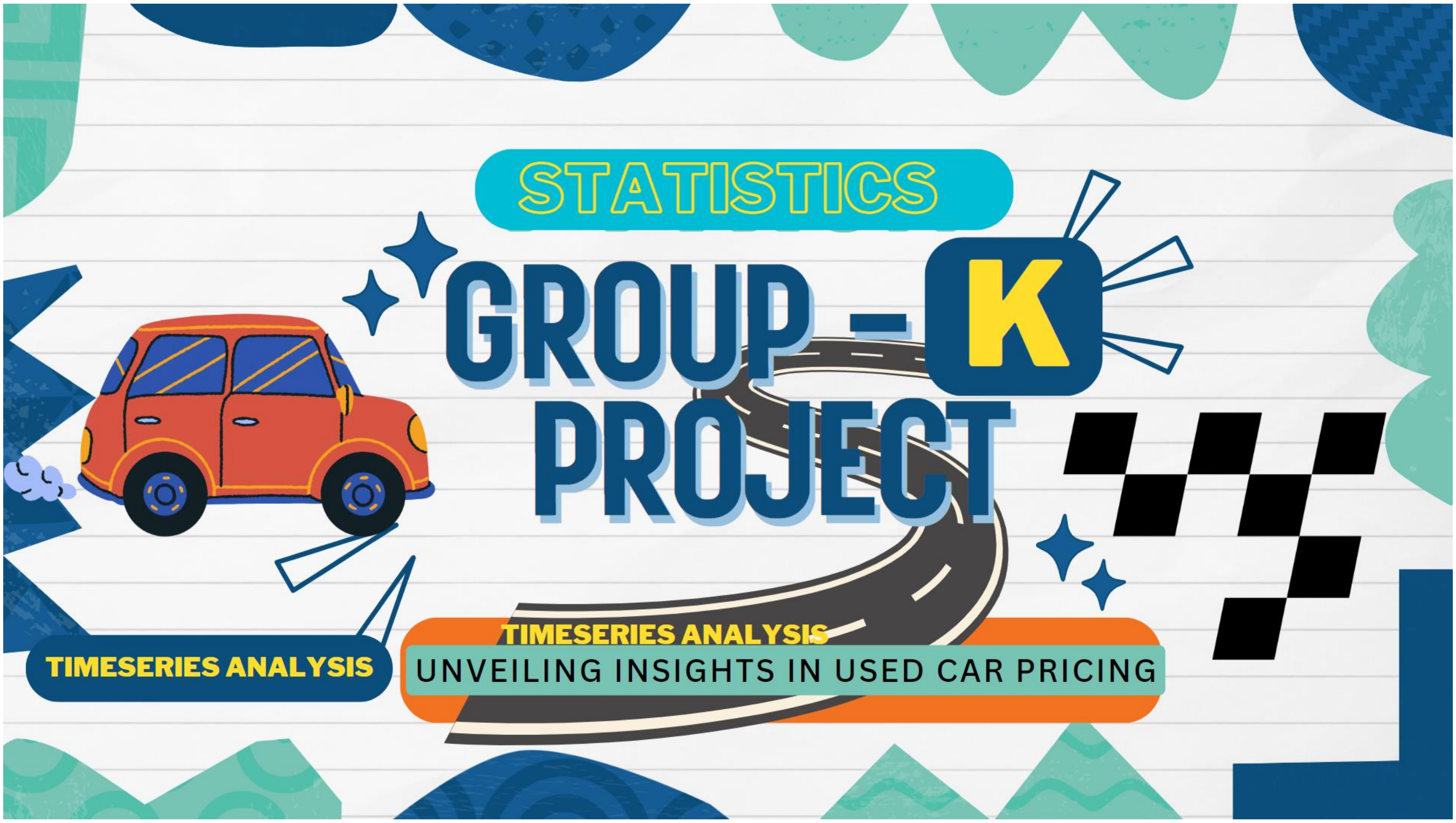
We confirm that we have read and understood the Plymouth University regulation relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group:

Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I *have used/not used translation software.

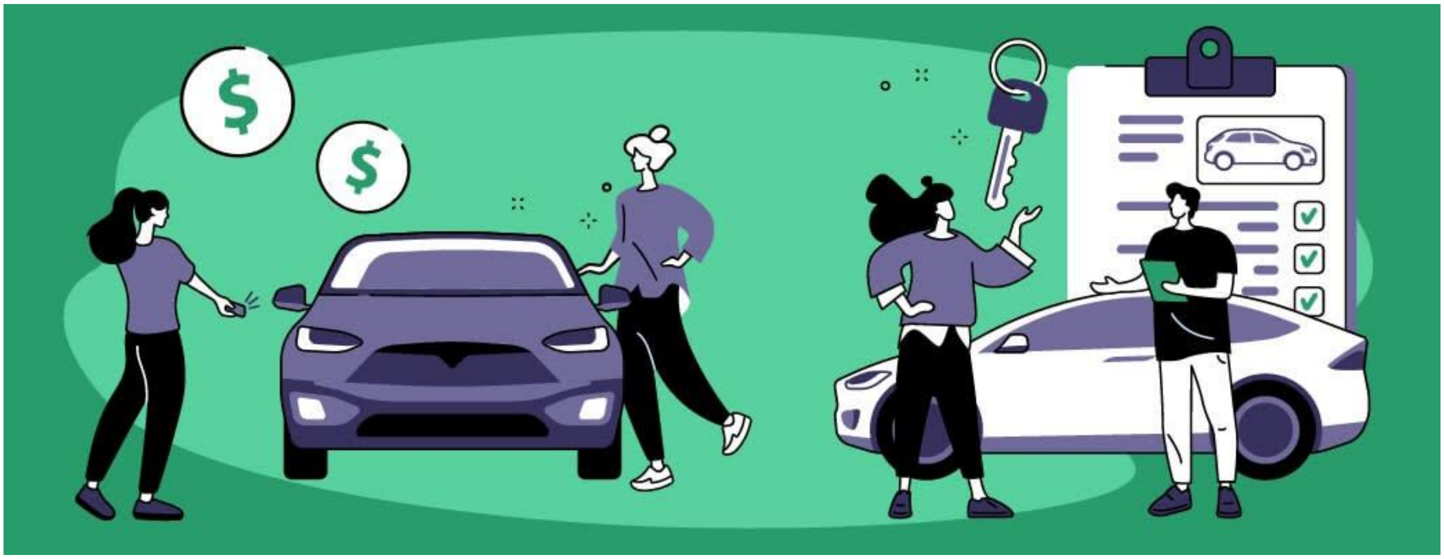
If used, please state name of software……………………………………………………………



Student contributions

ID	Student name & ID	Student contribution
10899472	G.h.s.gnanodya	Data cleaning & Preprocessing
10899192	Kavithma samadinie	Data Visualization
10900340	P.J.E.Jebasuthanthirany	Dashboard
10900339	Joshua Jebapragashan	Time series analysis
10899282	Malki Amasha	ML Multi linear regression model

Table of contents



1 *INTRODUCTION*

About Dataset

Context

The dataset contains 50 000 real world used cars details.

Inspiration

Imagine a web app that can estimate the listing price of a vehicle. What features of the vehicle should be used to build a price prediction regression model?

Car Dataset Attribute Breakdown

Breakdown of the attributes contained within a car dataset. Each attribute is categorized and described to offer a clear understanding of the data.

1) Vehicle Identification
   a) **vin** (String): A unique encoded string that identifies a specific vehicle.

2) Dimensions
   a) **Bed_height** (String): Height of the bed area in inches (applicable to pickup trucks).
   b) **Bed_length** (String): Length of the bed area in inches (applicable to pickup trucks).
   c) **Height** (String): Height of the entire vehicle in inches.
   d) **Length** (String): Length of the entire vehicle in inches.

3) Passenger Space
   a) **Back_legroom** (String): Legroom measurement in inches for the rear seats.
   b) **Front_legroom** (String): Legroom measurement in inches for the passenger seat.
   c) **Cabin** (String): Category designation for the cabin size in pickup trucks (e.g., Crew Cab, Extended Cab).

4) Body & Style
   a) **Body_type** (String): Body style classification of the vehicle (e.g., Convertible, Hatchback, Sedan).
   b) **Listing_color** (String): Dominant color group based on the exterior color.
   c) **Exterior_color** (String): The specific exterior color of the vehicle.
   d) **Interior_color** (String): The specific interior color of the vehicle.

5) Engine & Performance
   a) **Engine_cylinders** (String): The engine configuration
   b) **Engine_displacement** (Float): Engine displacement measured in liters.
   c) **Engine_type** (String): The engine configuration

d) **Horsepower** (Float): Horsepower output of the engine.
e) **Fuel_type** (String): The primary type of fuel used by the vehicle.
f) **Fuel_tank_volume** (String): Fuel tank capacity measured in gallons.
g) **City_fuel_economy** (Float): Fuel efficiency in kilometers per liter for city driving.
h) **Highway_fuel_economy** (Float): Fuel efficiency in kilometers per liter for highway driving.
i) **Combine_fuel_economy** (Float): Combined fuel efficiency in kilometers per liter (weighted average of city & highway).

6) Location & Listing Info

a) **City** (String): City where the car is listed for sale.
b) **Dealer_zip** (Integer): Zip code of the dealership selling the car.
c) **Latitude** (Float): Geographic latitude coordinate of the dealership location.
d) **Longitude** (Float): Geographic longitude coordinate of the dealership location.
e) **Daysonmarket** (Integer): Number of days since the vehicle was first listed on the website.
f) **Listed_date** (String): Date the vehicle was listed on the website.
g) **Listing_id** (Integer): Unique identifier for the vehicle listing.

7) Vehicle History & Condition

a) **Fleet** (Boolean): Indicates whether the vehicle was previously part of a fleet.
b) **Frame_damaged** (Boolean): Indicates whether the vehicle has a damaged frame.
c) **Has_accidents** (Boolean): Indicates whether there are any registered accidents associated with the vehicle's VIN.
d) **Iscab** (Boolean): Indicates whether the vehicle was previously used as a taxi/cab.
e) **Is_certified** (Boolean): Indicates whether the vehicle is certified with a warranty.
f) **Is_new** (Boolean): Indicates whether the vehicle is less than 2 years old.
g) **Is_oemcpo** (Boolean): Indicates whether the vehicle is a pre-owned car certified by the manufacturer.

1.1 Introduction

- In today's dynamic automotive market, accurately predicting the prices of used cars is crucial for informed decision-making by buyers, sellers, dealerships, and analysts. The US Used Car Price Prediction project aims to address this challenge by leveraging advanced machine learning techniques to develop a predictive model. This model will provide stakeholders with reliable estimates of used car prices, facilitating better decision-making and market efficiency. Throughout this report, we will discuss the methodology, implementation, results, and implications of this project, aiming to revolutionize the way the automotive industry operates.

2 Background / Literature Review

- The automotive industry, including the market for used cars, has been a subject of extensive research and analysis due to its economic significance and complexity. A review of existing literature reveals several key insights and trends in the domain of used car pricing and predictive modeling.

2.1 Market Dynamics:

- Studies have highlighted the dynamic nature of the used car market, influenced by factors such as supply and demand, economic conditions, consumer preferences, and technological advancements. Understanding these dynamics is essential for accurate price prediction and market analysis.

2.2 Data Sources:

- Research has explored the various data sources available for used car price prediction, including online listings, dealership records, auction data, and market reports. Leveraging diverse datasets and sources is crucial for building robust predictive models.

2.3 Predictive Modeling Techniques:

- Literature has extensively covered the application of machine learning algorithms and statistical techniques for predicting used car prices. Common approaches include linear regression, decision trees, random forests, support vector machines, and neural networks. Comparative studies have evaluated the performance of these techniques under different scenarios and datasets.

2.4 Feature Engineering:

- Feature selection and engineering play a critical role in developing effective predictive models for used car pricing. Studies have identified relevant features such as mileage, age, make and model, condition, location, and market trends. Techniques such as dimensionality reduction, categorical encoding, and interaction terms have been employed to enhance model performance.

2.5 Model Evaluation:

- Evaluating the performance of predictive models is essential for assessing their accuracy, reliability, and generalization capabilities. Metrics such as mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and R-squared (R2) have been commonly used to evaluate model performance. Cross-validation techniques and sensitivity analyses have also been employed to validate model robustness.

2.6 Challenges and Limitations:

- Despite the advancements in predictive modeling techniques, several challenges and limitations persist in the domain of used car price prediction. These include data quality issues, model interpretability, overfitting, feature importance, and scalability. Addressing these challenges is crucial for developing accurate and actionable predictive models.

Overall, the literature review provides valuable insights into the state-of-the-art techniques, methodologies, and challenges in the field of used car price prediction. By synthesizing and building upon existing research, this project aims to contribute to the advancement of predictive modeling techniques in the automotive industry.

3 Problem Statement

The US automotive market is vast and dynamic, with millions of transactions occurring annually in the used car segment. However, accurately predicting the prices of used cars remains a significant challenge due to the multitude of factors influencing pricing decisions. The problem statement for the US Used Car Price Prediction project is as follows:

3.1 Background Study:

- The used car market is influenced by various factors such as vehicle condition, mileage, age, make and model, geographic location, economic conditions, and consumer preferences. Understanding the interplay of these factors and their impact on pricing dynamics is crucial for developing accurate predictive models.

3.2 Problem Identification:

- The primary challenge in the used car market is the lack of transparency and consistency in pricing. Sellers often struggle to determine the optimal price for their vehicles, while buyers face uncertainty about the fair market value of used cars. Additionally, fluctuations in demand, supply, and external factors further complicate pricing decisions.

3.3 Objective:

- The objective of the US Used Car Price Prediction project is to develop a predictive model that can accurately estimate the prices of used cars based on relevant features and historical data. By leveraging advanced machine learning techniques, the project aims to provide stakeholders with a reliable tool for pricing analysis and decision-making.

3.4 Scope:

- The scope of the project encompasses data collection, preprocessing, feature engineering, model development, validation, and deployment. The predictive model will be trained on a comprehensive dataset of used car listings, incorporating diverse features such as vehicle attributes, market trends, and geographical factors.

3.5 Deliverables:

- The project will deliver a robust predictive model capable of estimating the prices of used cars with high accuracy. Additionally, the project will provide documentation, code repositories, and user guides to facilitate the deployment and utilization of the predictive model by stakeholders.

3.6 Challenges:

- Addressing data quality issues, feature selection, model interpretability, and scalability are key challenges in developing an effective predictive model for used car pricing. Overcoming these challenges will require careful consideration of data preprocessing techniques, feature engineering strategies, and model selection criteria.

3.7 Expected Outcomes:

- The successful implementation of the US Used Car Price Prediction project is expected to result in improved pricing accuracy, enhanced market transparency, and informed decision-making by stakeholders. By providing reliable estimates of used car prices, the project aims to create value for buyers, sellers, dealerships, and analysts in the US automotive market.

4 Methodology / Solution

The methodology for the US Used Car Price Prediction project involves several key steps, including data collection, preprocessing, feature engineering, model development, and evaluation. Here's an overview of the proposed solution:

4.1 Data Collection:

- Obtain a comprehensive dataset of used car listings, including relevant attributes such as vehicle specifications, market trends, and historical pricing data. Utilize reputable sources such as online marketplaces, dealership records, and industry reports to gather diverse and representative data.

4.2 Data Preprocessing:

- Cleanse the dataset by addressing missing values, outliers, and inconsistencies. Perform data imputation, where necessary, using appropriate techniques such as mean, median, or mode imputation. Standardize numerical features and encode categorical variables to prepare the data for modeling.

4.3 Feature Engineering:

- Extract relevant features from the dataset and create new variables to capture additional information that may influence used car prices. Consider factors such as vehicle age, mileage, make and model, geographic location, market demand, and economic indicators. Use domain knowledge and statistical techniques to select and transform features effectively.

4.4 Model Development:

- Apply machine learning algorithms to build a predictive model for estimating used car prices. Experiment with various regression techniques, such as linear regression, decision trees, random forests, and gradient boosting, to identify the most suitable approach for the dataset. Tune hyperparameters and optimize model performance using cross-validation techniques.

4.5 Model Evaluation:

- Assess the performance of the predictive model using appropriate evaluation metrics, such as mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and R-squared (R2) score. Validate the model's accuracy and robustness through cross-validation and sensitivity analysis. Compare the performance of different models to select the best-performing one for deployment.

4.6 Deployment and Integration:

- Deploy the trained predictive model into a production environment, making it accessible to stakeholders through user-friendly interfaces or APIs. Integrate the model into existing systems and workflows to facilitate seamless integration with decision-making processes.

4.7 Monitoring and Maintenance:

- Continuously monitor the performance of the deployed model and update it as necessary to adapt to changing market conditions and data dynamics. Implement mechanisms for tracking model drift, recalibration, and version control to ensure long-term reliability and effectiveness.

By following this methodology, the Used Car Price Prediction project aims to develop a robust and accurate predictive model that can assist stakeholders in making informed pricing decisions in the dynamic and complex used car market. Through rigorous data analysis, feature engineering, and model development, the project seeks to enhance market efficiency and transparency, ultimately benefiting buyers, sellers, dealerships, and analysts alike.

5 Deliverables / Work Breakdown and Timeline

The US Used Car Price Prediction project involves several deliverables, each requiring specific tasks and timelines for successful completion. Here's a breakdown of the work and timeline associated with each deliverable:

5.1 Data Collection (Week 1):

- Task 1: Identify and acquire relevant datasets from online marketplaces, dealership records, and industry reports.
- Task 2: Clean and preprocess the raw data to remove duplicates, address missing values, and standardize formats.
- Task 3: Validate the quality and integrity of the dataset through exploratory data analysis (EDA) and data validation techniques.

5.2 Feature Engineering (Week 2):

- Task 1: Analyze the dataset to identify key features and variables that may influence used car prices.
- Task 2: Create new features or derive additional variables from existing data to enhance predictive power.
- Task 3: Conduct feature selection and dimensionality reduction to eliminate irrelevant or redundant features.

5.3 Model Development (Week 3):

- Task 1: Experiment with different machine learning algorithms, including linear regression, decision trees, random forests, and gradient boosting.
- Task 2: Train and validate multiple models using cross-validation techniques to assess performance and generalization capabilities.
- Task 3: Optimize hyperparameters and fine-tune model architectures to maximize predictive accuracy and minimize errors.

5.4 Model Evaluation (Week 4):

- Task 1: Evaluate the performance of trained models using appropriate evaluation metrics such as MAE, MSE, RMSE, and R2 score.
- Task 2: Compare the performance of different models and select the best-performing one for deployment.
- Task 3: Validate model assumptions and assess robustness through sensitivity analysis and cross-validation.

5.5 Deployment and Integration (Week 5):

- Task 1: Deploy the selected model into a production environment, making it accessible to stakeholders through user interfaces or APIs.
- Task 2: Integrate the model into existing systems and workflows to facilitate seamless integration with decision-making processes.
- Task 3: Conduct user acceptance testing (UAT) and gather feedback from stakeholders for further improvements and refinements.

5.6 Documentation and Reporting (Week 6):

- Task 1: Prepare documentation, including user guides, technical specifications, and model documentation.
- Task 2: Compile a final report summarizing the project methodology, findings, and recommendations.
- Task 3: Present the project outcomes and deliverables to stakeholders through presentations or workshops.

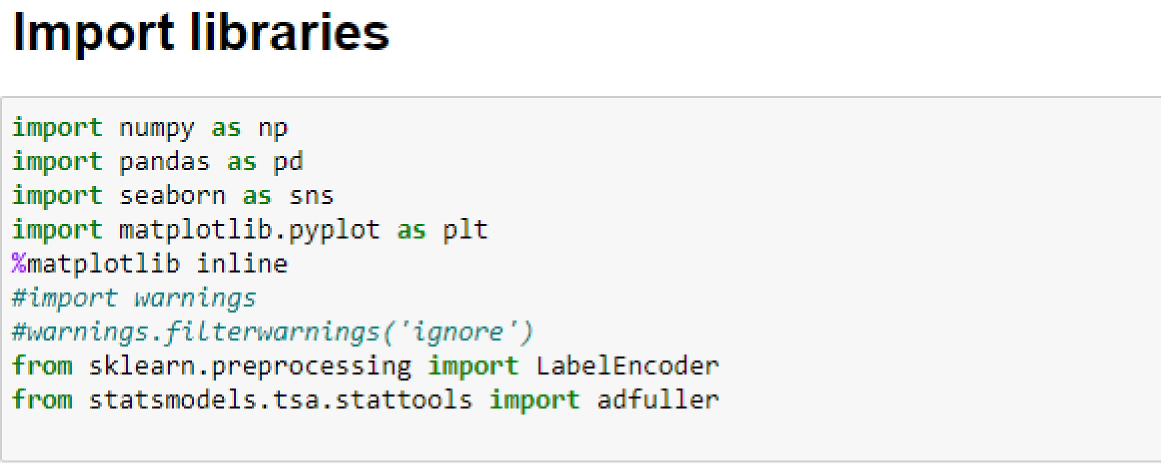
By following this work breakdown and timeline, the US Used Car Price Prediction project aims to systematically progress through each phase of development, ensuring timely delivery of high-quality deliverables and successful achievement of project objectives. Effective project management, collaboration, and communication are essential for meeting deadlines and maximizing project success.

6 Implementation of the Solution

The implementation phase of the US Used Car Price Prediction project involves translating the proposed methodology into actionable steps to develop, train, and evaluate the predictive model. Here's a detailed overview of the implementation process:

6.1 Libraries and their usages:

1. numpy (NumPy):



Import libraries

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
#import warnings
#warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
from statsmodels.tsa.stattools import adfuller
```

- **Purpose:** NumPy is a fundamental library for scientific computing in Python. It provides efficient multidimensional arrays and a rich set of mathematical functions for numerical operations.
- **Common Uses:**
  - Creating and manipulating arrays (vectors, matrices, higher-dimensional)
  - Array broadcasting for element-wise operations
  - Linear algebra operations (matrix multiplication, vector dot products)
  - Random number generation (uniform, normal, etc.)
  - Fast mathematical functions (e.g., sin, cos, exp, log)
  - Fourier transforms and other signal processing operations

2. pandas (Pandas):

- **Purpose:** Pandas is a high-performance, easy-to-use data analysis and manipulation library. It offers powerful data structures like Series (one-dimensional) and DataFrames (two-dimensional labeled data) for handling tabular data.
- **Common Uses:**
  - Reading data from various file formats (CSV, Excel, JSON, etc.)
  - Data cleaning and wrangling (missing value handling, data type conversion)
  - Data exploration (descriptive statistics, groupby operations)
  - Data merging, joining, and reshaping
  - Time series analysis (date/time indexing, date manipulation)
  - Feature engineering and data preparation for machine learning

3. seaborn (Seaborn):

- **Purpose:** Seaborn is a statistical data visualization library built on top of Matplotlib. It provides a high-level interface for creating informative and aesthetically pleasing statistical graphics.
- **Common Uses**:

    - Creating various plot types (bar charts, line plots, scatter plots, violin plots, heatmaps, joint plots, distribution plots)
    - Visualizing relationships between variables (univariate, bivariate, multivariate)
    - Statistical summaries within plots (means, medians, confidence intervals)
    - Easy customization of plot aesthetics (colors, fonts, styles) for better communication

4. matplotlib.pyplot (Matplotlib):

- **Purpose**: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It provides low-level control over various plot elements.
- **Common Uses**:
    - Creating a wide range of plot types (similar to Seaborn, but with more fine-grained control)
    - Customizing plot elements (axes, labels, legends, ticks, grid)
    - Interactive plotting with tools like plt.show() and plt.interactive()
    - Layering multiple plots on the same figure
    - Creating more complex visualization elements (annotations, subplots, insets)

5. %matplotlib inline (Magic command - optional):

- **Purpose:** This magic command is specific to Jupyter Notebooks. It embeds Matplotlib plots directly within the notebook output, allowing you to see visualizations without having to call plt.show() separately.
- **Usage**: Include this line at the beginning of your notebook code cell where you create Matplotlib plots.

6. sklearn.preprocessing.LabelEncoder (from scikit-learn):

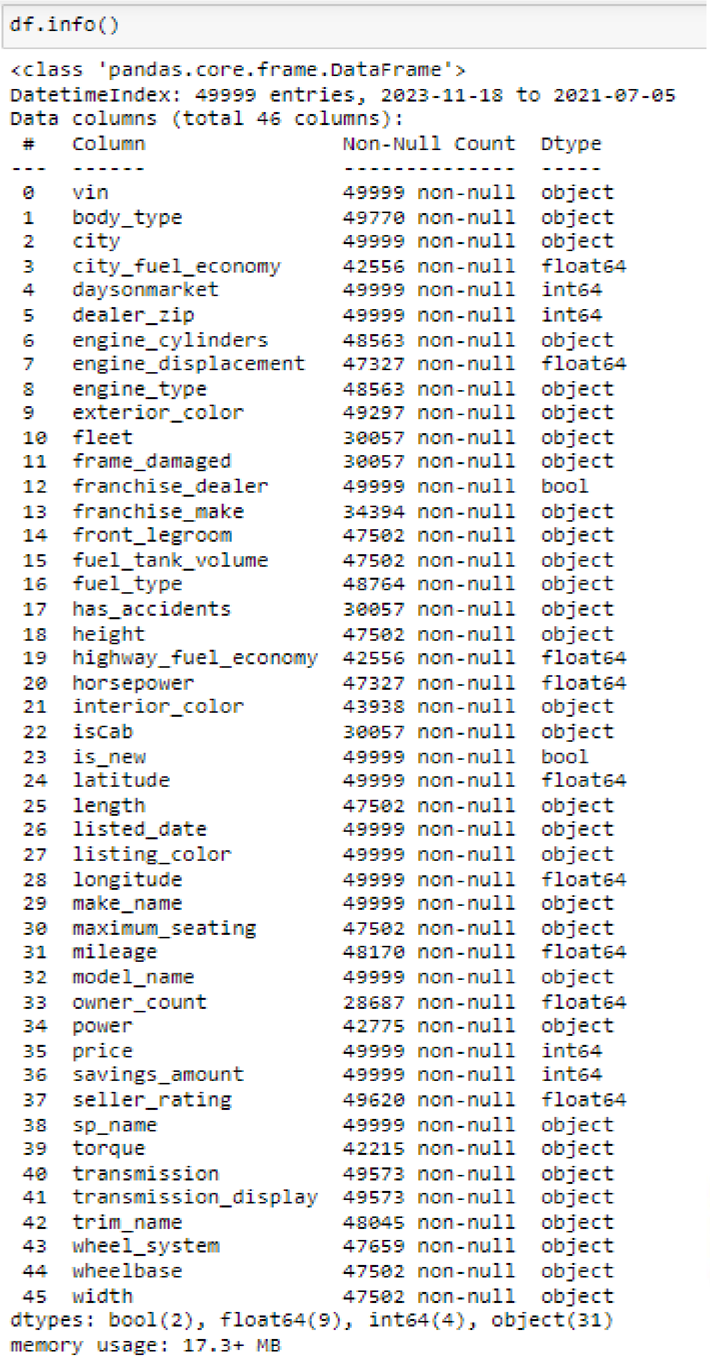
- **Purpose:** LabelEncoder is a utility from scikit-learn for encoding categorical data (text labels) into numerical labels. This is often necessary for machine learning algorithms that require numerical features.
- **Common Uses**:
    - Converting string categories (e.g., "red", "green", "blue") into integers (e.g., 0, 1, 2)
    - Simplifying data representation for algorithms
    - Preserving the order of categories (if order is important)

7. statsmodels.tsa.stattools.adfuller (from Statsmodels):

- **Purpose**: The adfuller function performs the Augmented Dickey-Fuller (ADF) test for stationarity in time series data. Stationarity is a statistical property where the mean, variance, and autocorrelation of a series remain constant over time.
- **Common Uses**:
    - Checking if a time series is stationary before applying certain forecasting or modeling techniques
    - Determining if differencing (taking the difference between successive observations) is necessary to achieve stationarity
    - Understanding the time series' statistical behavior

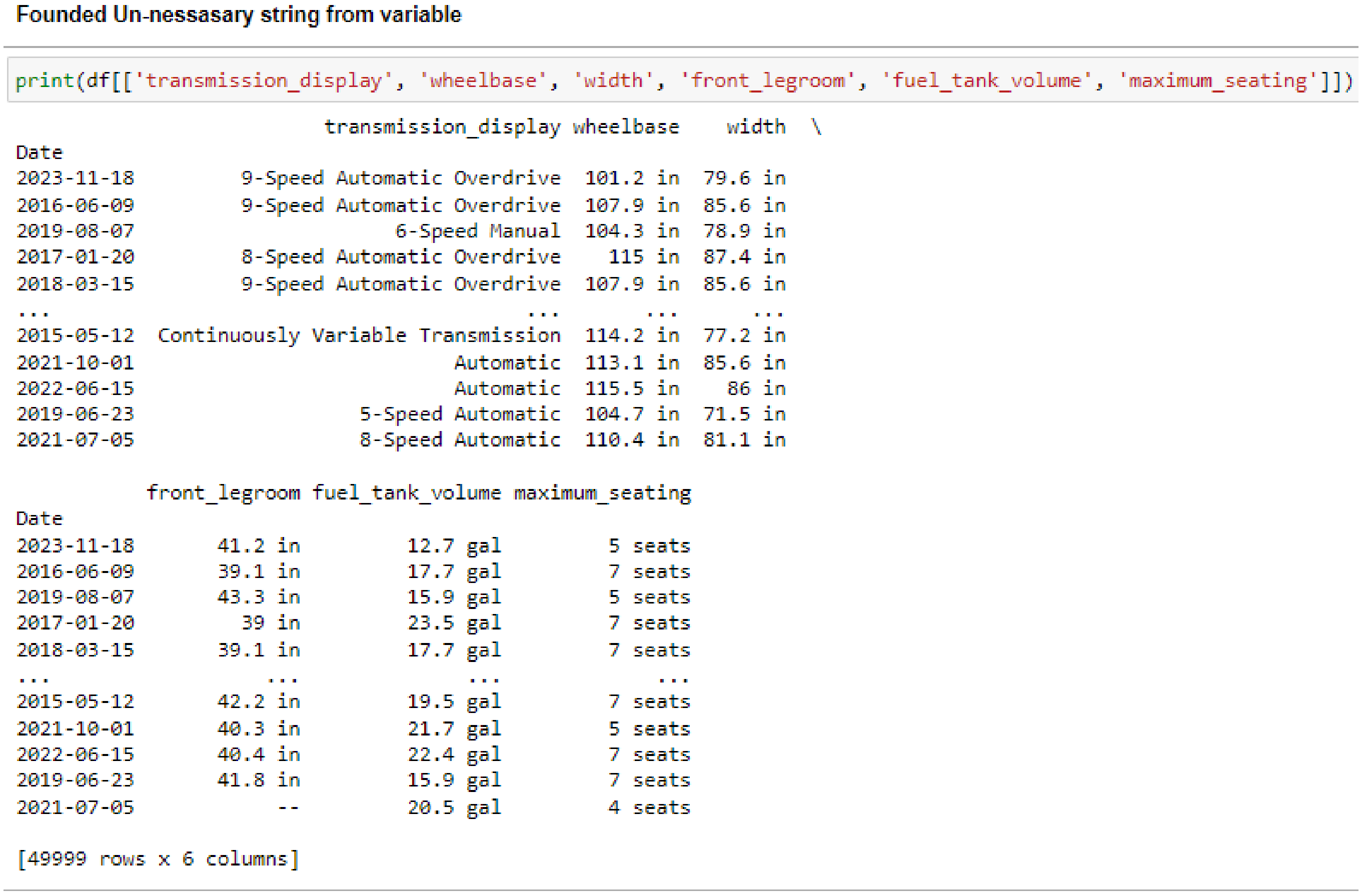
6.2 Data Preprocessing:

- Cleanse the dataset by addressing missing values, outliers, and inconsistencies.
- Perform data imputation using appropriate techniques such as mean, median, or mode imputation.
- Standardize numerical features and encode categorical variables to prepare the data for modeling.
- plit the dataset into training and testing sets to facilitate model training and evaluation



```
df.info()
```

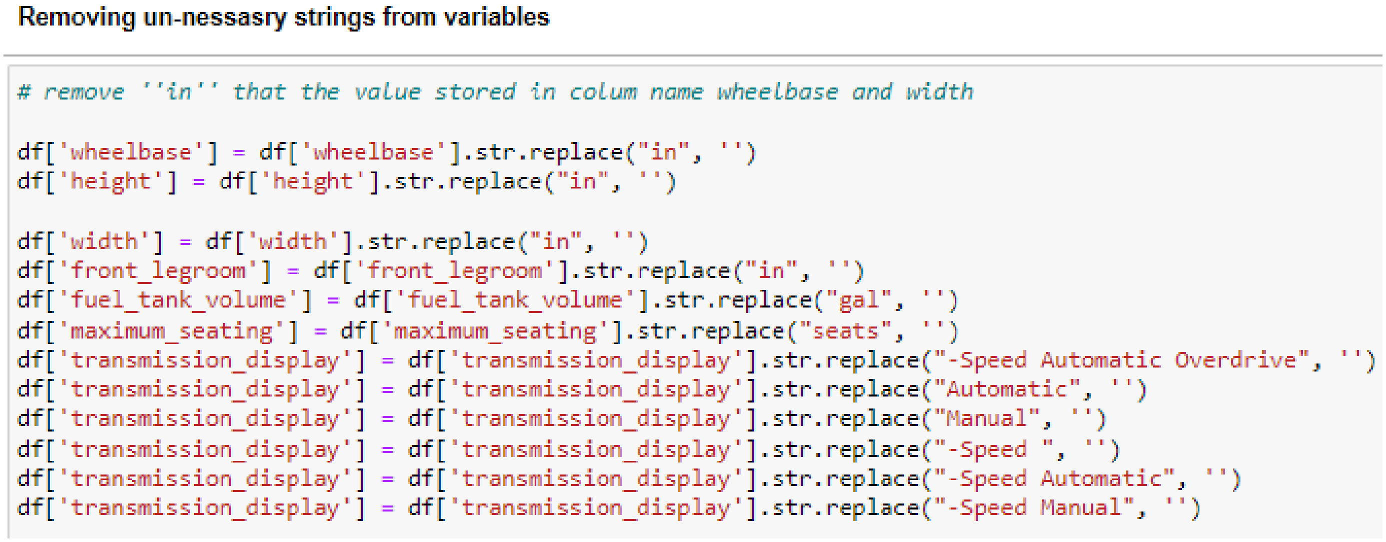
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 49999 entries, 2023-11-18 to 2021-07-05
Data columns (total 46 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   vin                   49999 non-null  object 
 1   body_type             49770 non-null  object 
 2   city                  49999 non-null  object 
 3   city_fuel_economy     42556 non-null  float64
 4   daysonmarket          49999 non-null  int64  
 5   dealer_zip            49999 non-null  int64  
 6   engine_cylinders      48563 non-null  object 
 7   engine_displacement   47327 non-null  float64
 8   engine_type           48563 non-null  object 
 9   exterior_color        49297 non-null  object 
 10  fleet                 30057 non-null  object 
 11  frame_damaged         30057 non-null  object 
 12  franchise_dealer      49999 non-null  bool   
 13  franchise_make        34394 non-null  object 
 14  front_legroom         47502 non-null  object 
 15  fuel_tank_volume      47502 non-null  object 
 16  fuel_type             48764 non-null  object 
 17  has_accidents         30057 non-null  object 
 18  height                47502 non-null  object 
 19  highway_fuel_economy  42556 non-null  float64
 20  horsepower            47327 non-null  float64
 21  interior_color        43938 non-null  object 
 22  isCab                 30057 non-null  object 
 23  is_new                49999 non-null  bool   
 24  latitude              49999 non-null  float64
 25  length                47502 non-null  object 
 26  listed_date           49999 non-null  object 
 27  listing_color         49999 non-null  object 
 28  longitude             49999 non-null  float64
 29  make_name             49999 non-null  object 
 30  maximum_seating       47502 non-null  object 
 31  mileage               48170 non-null  float64
 32  model_name            49999 non-null  object 
 33  owner_count           28687 non-null  float64
 34  power                 42775 non-null  object 
 35  price                 49999 non-null  int64  
 36  savings_amount        49999 non-null  int64  
 37  seller_rating         49620 non-null  float64
 38  sp_name               49999 non-null  object 
 39  torque                42215 non-null  object 
 40  transmission          49573 non-null  object 
 41  transmission_display  49573 non-null  object 
 42  trim_name             48045 non-null  object 
 43  wheel_system          47659 non-null  object 
 44  wheelbase             47502 non-null  object 
 45  width                 47502 non-null  object 
dtypes: bool(2), float64(9), int64(4), object(31)
memory usage: 17.3+ MB
```



Founded Un-nessasary string from variable

```
print(df[['transmission_display', 'wheelbase', 'width', 'front_legroom', 'fuel_tank_volume', 'maximum_seating']])
```

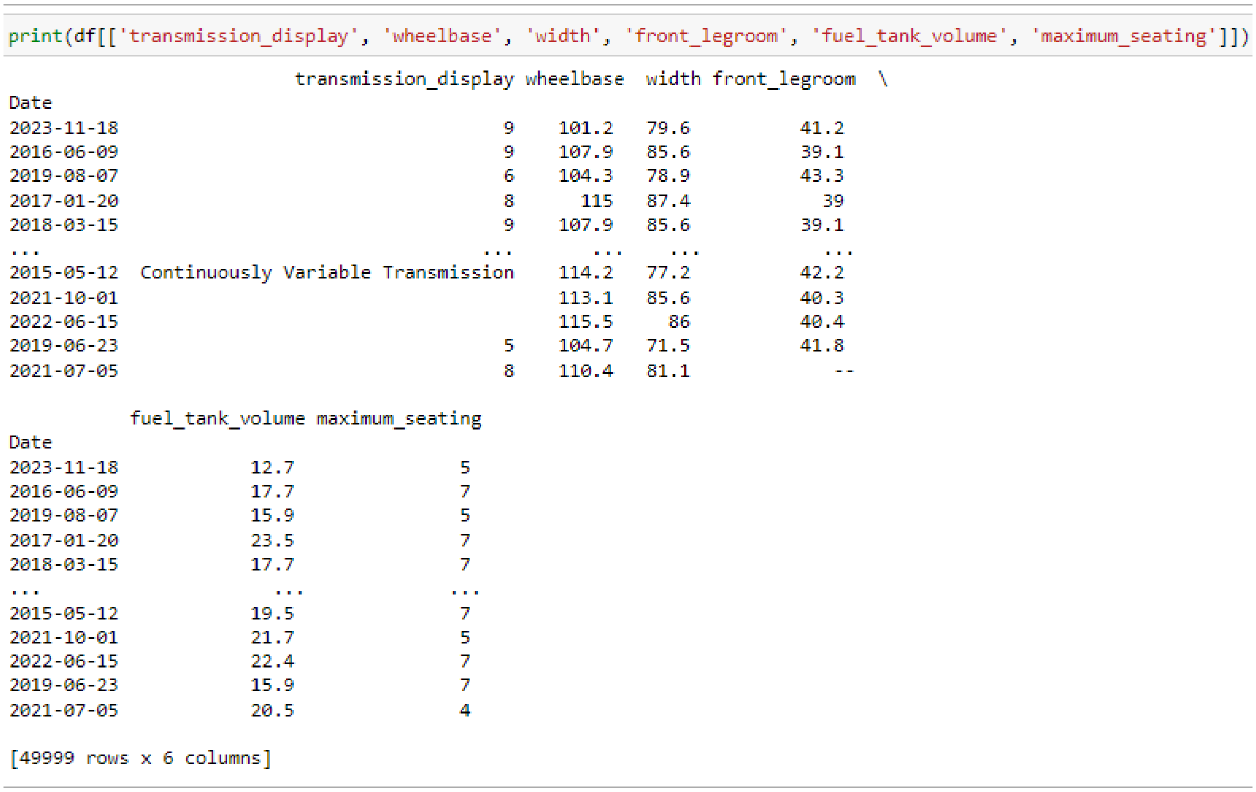
```
                          transmission_display wheelbase    width  \
Date                                                                
2023-11-18         9-Speed Automatic Overdrive  101.2 in  79.6 in   
2016-06-09         9-Speed Automatic Overdrive  107.9 in  85.6 in   
2019-08-07                      6-Speed Manual  104.3 in  78.9 in   
2017-01-20         8-Speed Automatic Overdrive    115 in  87.4 in   
2018-03-15         9-Speed Automatic Overdrive  107.9 in  85.6 in   
...                                        ...       ...      ...   
2015-05-12  Continuously Variable Transmission  114.2 in  77.2 in   
2021-10-01                           Automatic  113.1 in  85.6 in   
2022-06-15                           Automatic  115.5 in    86 in   
2019-06-23                   5-Speed Automatic  104.7 in  71.5 in   
2021-07-05                   8-Speed Automatic  110.4 in  81.1 in   

           front_legroom fuel_tank_volume maximum_seating  
Date                                                       
2023-11-18       41.2 in         12.7 gal         5 seats  
2016-06-09       39.1 in         17.7 gal         7 seats  
2019-08-07       43.3 in         15.9 gal         5 seats  
2017-01-20         39 in         23.5 gal         7 seats  
2018-03-15       39.1 in         17.7 gal         7 seats  
...                  ...              ...             ...  
2015-05-12       42.2 in         19.5 gal         7 seats  
2021-10-01       40.3 in         21.7 gal         5 seats  
2022-06-15       40.4 in         22.4 gal         7 seats  
2019-06-23       41.8 in         15.9 gal         7 seats  
2021-07-05            --         20.5 gal         4 seats  

[49999 rows x 6 columns]
```



Removing un-nessasry strings from variables

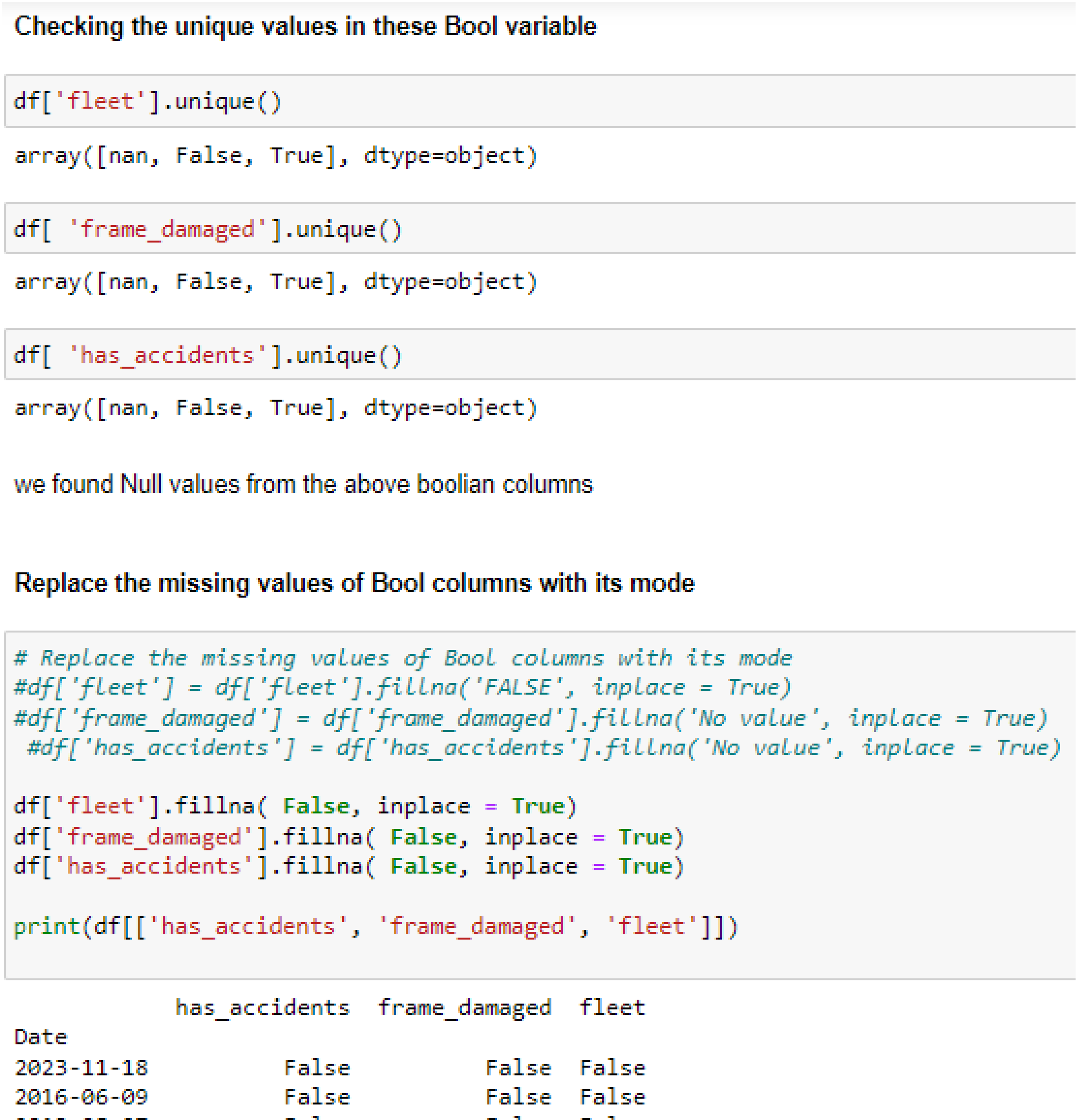
```
# remove ''in'' that the value stored in colum name wheelbase and width

df['wheelbase'] = df['wheelbase'].str.replace("in", '')
df['height'] = df['height'].str.replace("in", '')

df['width'] = df['width'].str.replace("in", '')
df['front_legroom'] = df['front_legroom'].str.replace("in", '')
df['fuel_tank_volume'] = df['fuel_tank_volume'].str.replace("gal", '')
df['maximum_seating'] = df['maximum_seating'].str.replace("seats", '')
df['transmission_display'] = df['transmission_display'].str.replace("-Speed Automatic Overdrive", '')
df['transmission_display'] = df['transmission_display'].str.replace("Automatic", '')
df['transmission_display'] = df['transmission_display'].str.replace("Manual", '')
df['transmission_display'] = df['transmission_display'].str.replace("-Speed ", '')
df['transmission_display'] = df['transmission_display'].str.replace("-Speed Automatic", '')
df['transmission_display'] = df['transmission_display'].str.replace("-Speed Manual", '')
```

After removing the un-nessary strings



```
print(df[['transmission_display', 'wheelbase', 'width', 'front_legroom', 'fuel_tank_volume', 'maximum_seating']])
```

```
                          transmission_display wheelbase  width front_legroom  \
Date                                                                            
2023-11-18                                   9     101.2   79.6          41.2   
2016-06-09                                   9     107.9   85.6          39.1   
2019-08-07                                   6     104.3   78.9          43.3   
2017-01-20                                   8       115   87.4            39   
2018-03-15                                   9     107.9   85.6          39.1   
...                                        ...       ...    ...           ...   
2015-05-12  Continuously Variable Transmission     114.2   77.2          42.2   
2021-10-01                                         113.1   85.6          40.3   
2022-06-15                                         115.5     86          40.4   
2019-06-23                                   5     104.7   71.5          41.8   
2021-07-05                                   8     110.4   81.1            --   

           fuel_tank_volume maximum_seating  
Date                                         
2023-11-18             12.7               5  
2016-06-09             17.7               7  
2019-08-07             15.9               5  
2017-01-20             23.5               7  
2018-03-15             17.7               7  
...                     ...             ...  
2015-05-12             19.5               7  
2021-10-01             21.7               5  
2022-06-15             22.4               7  
2019-06-23             15.9               7  
2021-07-05             20.5               4  

[49999 rows x 6 columns]
```



Checking the unique values in these Bool variable

```
df['fleet'].unique()
```

```
array([nan, False, True], dtype=object)
```

```
df[ 'frame_damaged'].unique()
```

```
array([nan, False, True], dtype=object)
```

```
df[ 'has_accidents'].unique()
```

```
array([nan, False, True], dtype=object)
```

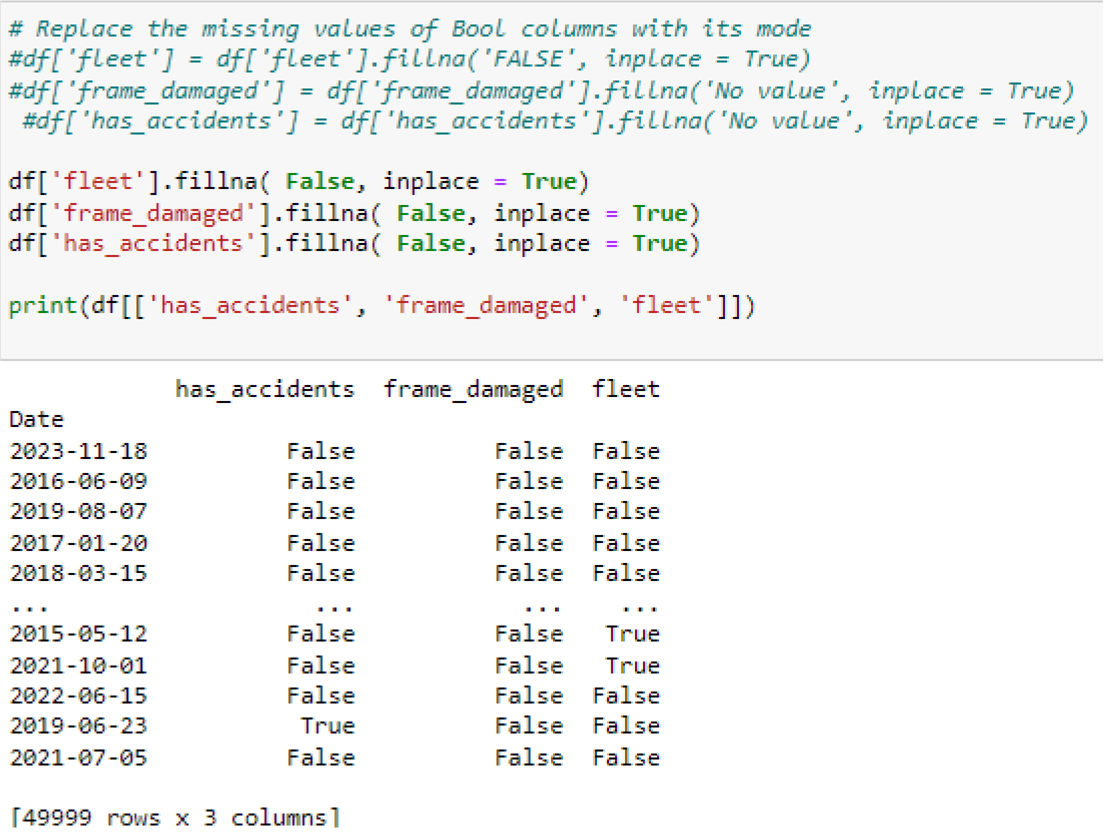
we found Null values from the above boolian columns

Replace the missing values of Bool columns with its mode

```
# Replace the missing values of Bool columns with its mode
#df['fleet'] = df['fleet'].fillna('FALSE', inplace = True)
#df['frame_damaged'] = df['frame_damaged'].fillna('No value', inplace = True)
 #df['has_accidents'] = df['has_accidents'].fillna('No value', inplace = True)

df['fleet'].fillna( False, inplace = True)
df['frame_damaged'].fillna( False, inplace = True)
df['has_accidents'].fillna( False, inplace = True)

print(df[['has_accidents', 'frame_damaged', 'fleet']])
```

```
           has_accidents  frame_damaged  fleet
Date                                          
2023-11-18         False          False  False
2016-06-09         False          False  False
```

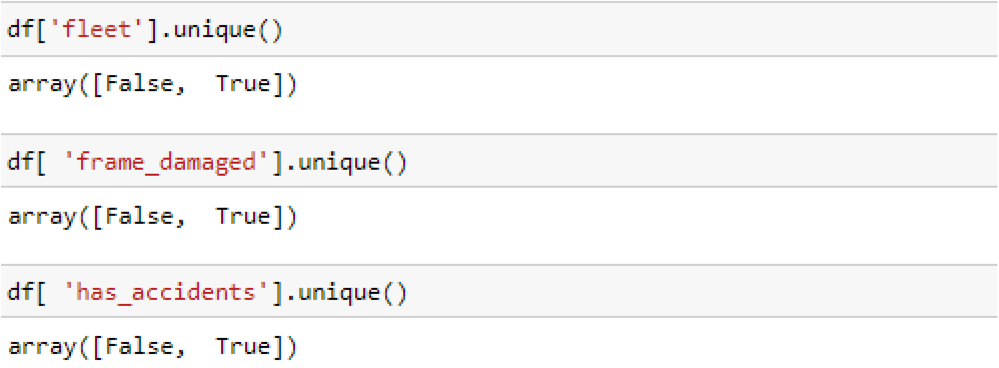
Replace the missing values of Bool columns with its mode



```
# Replace the missing values of Bool columns with its mode
#df['fleet'] = df['fleet'].fillna('FALSE', inplace = True)
#df['frame_damaged'] = df['frame_damaged'].fillna('No value', inplace = True)
 #df['has_accidents'] = df['has_accidents'].fillna('No value', inplace = True)

df['fleet'].fillna( False, inplace = True)
df['frame_damaged'].fillna( False, inplace = True)
df['has_accidents'].fillna( False, inplace = True)

print(df[['has_accidents', 'frame_damaged', 'fleet']])
```

```
            has_accidents  frame_damaged  fleet
Date
2023-11-18          False          False  False
2016-06-09          False          False  False
2019-08-07          False          False  False
2017-01-20          False          False  False
2018-03-15          False          False  False
...                   ...            ...    ...
2015-05-12          False          False   True
2021-10-01          False          False   True
2022-06-15          False          False  False
2019-06-23           True          False  False
2021-07-05          False          False  False

[49999 rows x 3 columns]
```

Unique values in boolian variable after mode replacement



```
df['fleet'].unique()
```

```
array([False,  True])
```

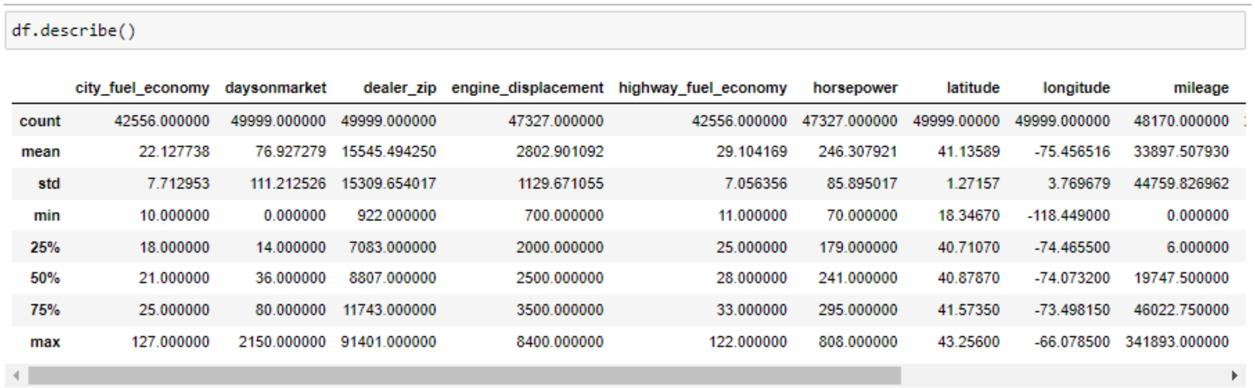
```
df[ 'frame_damaged'].unique()
```

```
array([False,  True])
```

```
df[ 'has_accidents'].unique()
```

```
array([False,  True])
```

Check Summary of Dataset



```
df.describe()
```

	city_fuel_economy	daysonmarket	dealer_zip	engine_displacement	highway_fuel_economy	horsepower	latitude	longitude	mileage
count	42556.000000	49999.000000	49999.000000	47327.000000	42556.000000	47327.000000	49999.00000	49999.000000	48170.000000
mean	22.127738	76.927279	15545.494250	2802.901092	29.104169	246.307921	41.13589	-75.456516	33897.507930
std	7.712953	111.212526	15309.654017	1129.671055	7.056356	85.895017	1.27157	3.769679	44759.826962
min	10.000000	0.000000	922.000000	700.000000	11.000000	70.000000	18.34670	-118.449000	0.000000
25%	18.000000	14.000000	7083.000000	2000.000000	25.000000	179.000000	40.71070	-74.465500	6.000000
50%	21.000000	36.000000	8807.000000	2500.000000	28.000000	241.000000	40.87870	-74.073200	19747.500000
75%	25.000000	80.000000	11743.000000	3500.000000	33.000000	295.000000	41.57350	-73.498150	46022.750000
max	127.000000	2150.000000	91401.000000	8400.000000	122.000000	808.000000	43.25600	-66.078500	341893.000000

6.3 Feature Engineering:

- Analyze the dataset to identify relevant features that may influence used car prices.
- Create new features or derive additional variables from existing data to capture additional information.
- Conduct feature selection and dimensionality reduction to eliminate irrelevant or redundant features.
- Transform and scale features to ensure uniformity and enhance model performance.

6.4 Model Development:

- Experiment with different machine learning algorithms, including linear regression, decision trees, random forests, and gradient boosting.
- Train multiple models using the training dataset and validate their performance using cross-validation techniques.
- Optimize hyperparameters and fine-tune model architectures to maximize predictive accuracy and minimize errors.
- Select the best-performing model based on evaluation metrics such as mean absolute error (MAE), mean squared error (MSE), and R-squared (R2) score.

6.5 Model Evaluation:

- Assess the performance of the trained model using the testing dataset and appropriate evaluation metrics.
- Validate model assumptions and assess robustness through sensitivity analysis and cross-validation.
- Compare the performance of different models and select the most accurate and reliable one for deployment.
- Interpret the model results and identify key drivers of used car prices based on feature importance and coefficients.

6.6 Deployment and Integration:

- Deploy the selected model into a production environment, making it accessible to stakeholders through user interfaces or APIs.
- Integrate the model into existing systems and workflows to facilitate seamless integration with decision-making processes.
- Conduct user acceptance testing (UAT) and gather feedback from stakeholders for further improvements and refinements.

- Monitor the performance of the deployed model and update it as necessary to adapt to changing market conditions and data dynamics.

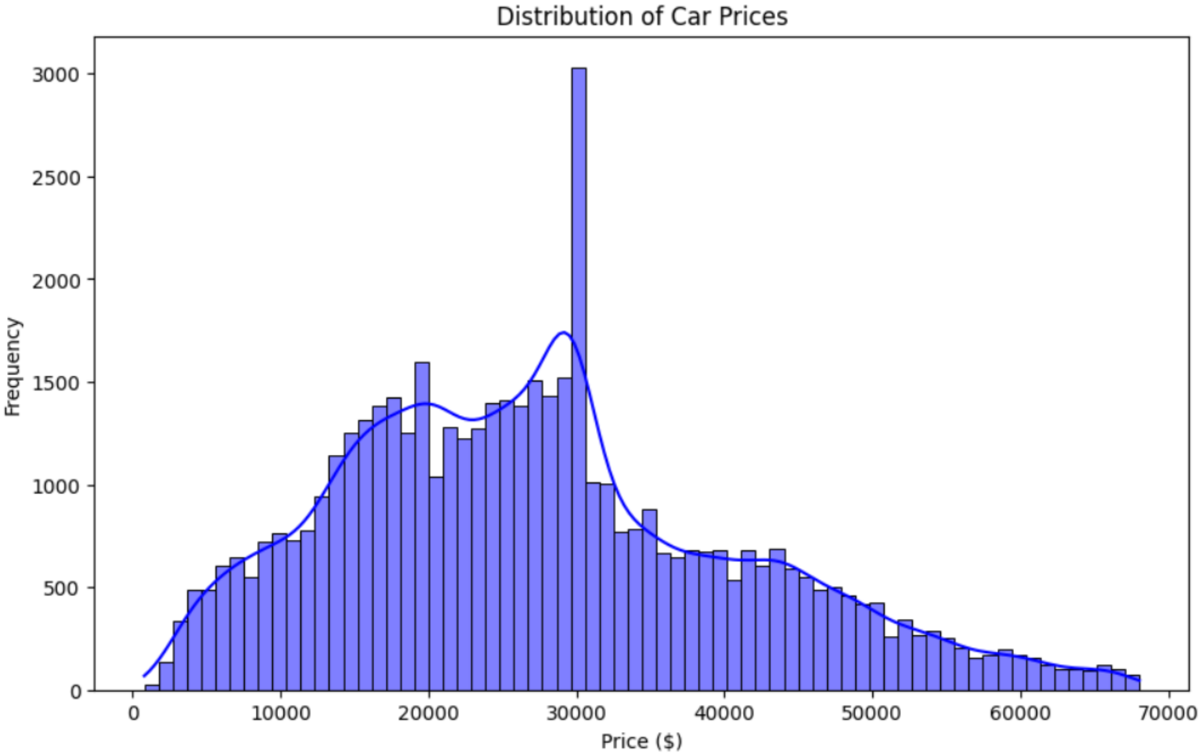
6.7 Documentation and Reporting:

- Prepare comprehensive documentation, including user guides, technical specifications, and model documentation.
- Compile a final report summarizing the project methodology, findings, and recommendations.
- Present the project outcomes and deliverables to stakeholders through presentations or workshops.
- Ensure transparency and reproducibility of the implementation process by documenting all code, workflows, and decision-making steps.

By meticulously implementing each step of the solution methodology, the US Used Car Price Prediction project aims to develop a reliable and accurate predictive model that can assist stakeholders in making informed pricing decisions in the dynamic and complex used car market.

7 Visualization

7.1 Histogram of Price Distribution:



- This histogram shows the distribution of car prices in the dataset.
- The x-axis represents the price of cars, while the y-axis represents the frequency of cars at each price level.
- The histogram indicates the spread of car prices, providing an overview of the price range and the most common price points.
- It helps in understanding the general pattern of car prices and identifying any potential outliers.

- It is a histogram showing the distribution of car prices in a dataset of used cars.

Histogram: A histogram is a bar chart that depicts the frequency distribution of a continuous variable. The x-axis represents the range of values for the variable price , divided into bins. The y-axis represents the frequency or count of data points that fall within each bin. The height of each bar corresponds to the number of cars that fall within that particular price range.

Distribution of car prices: In this histogram, the x-axis shows the price range of the cars, likely in thousands of dollars. The y-axis shows the number of cars in each price range.

Observations about the data:

- The histogram appears to be right-skewed, meaning there are more cars towards the lower end of the price range. This suggests that there are more affordable cars in the dataset than expensive cars.
- There is a peak around the $10,000 to $20,000 price range, indicating that this is a common price range for used cars in this dataset.
- The histogram tapers off towards the higher end of the price range, but there are still some cars that are quite expensive.

Concise summary:

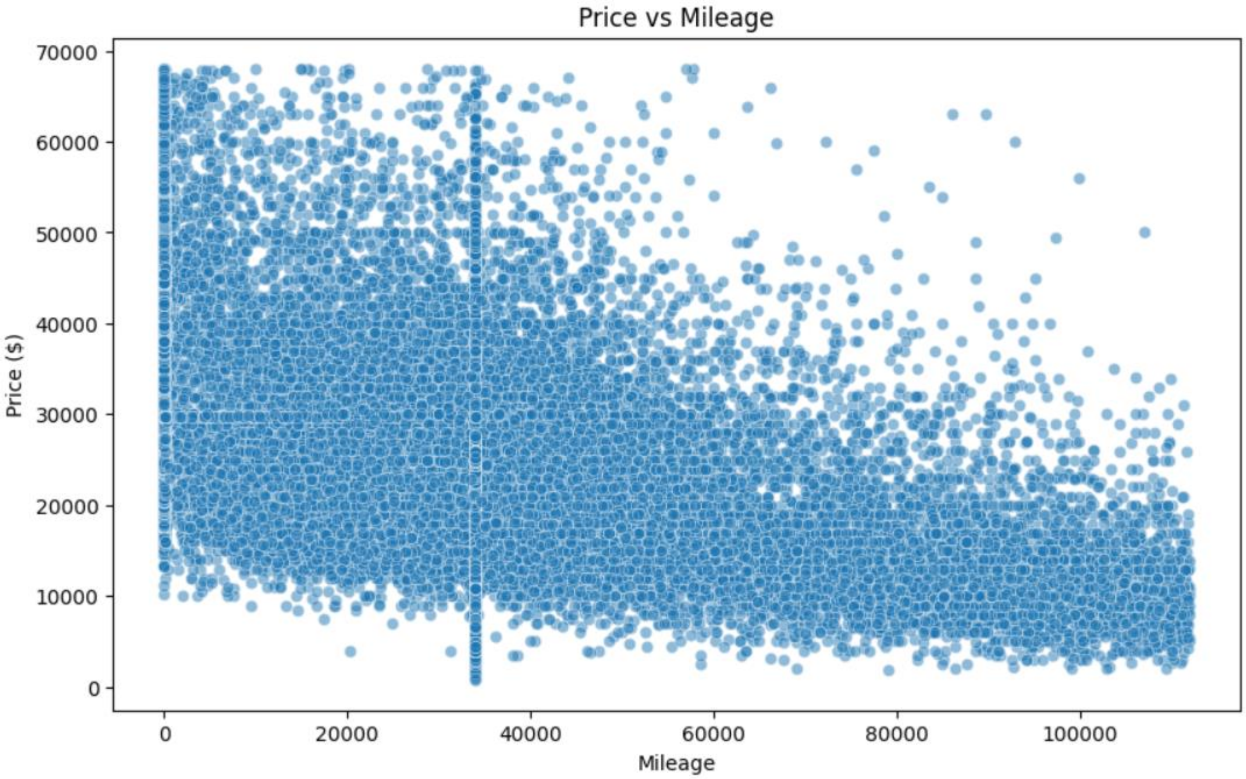
- This histogram provides a snapshot of how car prices are distributed in the dataset. It helps us understand the typical price range for used cars and identify outliers that fall outside this range.

Explanatory text:

- A histogram is a visual representation of data distribution. In this case, it shows the distribution of car prices within the dataset. By analyzing the histogram, you can gain insights into the typical price range for most used cars. Additionally, you can see how many cars deviate from this common range, which could be due to factors like luxury models or very old cars

7.2 Scatterplot of Price vs Mileage:

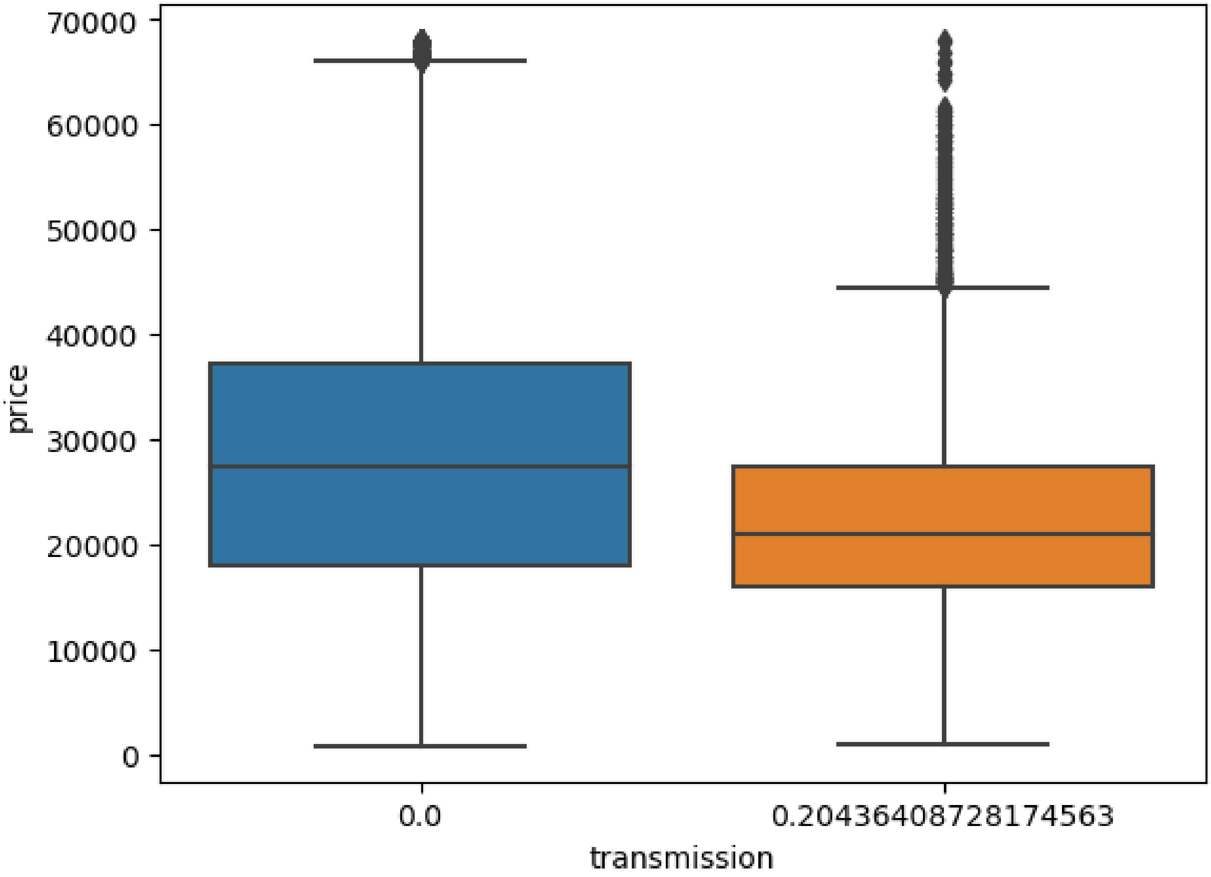
- This scatterplot visualizes the relationship between the price of cars and their mileage.
- Each point on the plot represents a car, with its price plotted against its mileage.
- The scatterplot helps in understanding how the price of cars varies with mileage. Generally, one would expect lower prices for cars with higher mileage, as they have been used more extensively.



7.3 Boxplot of Price by Transmission Type:

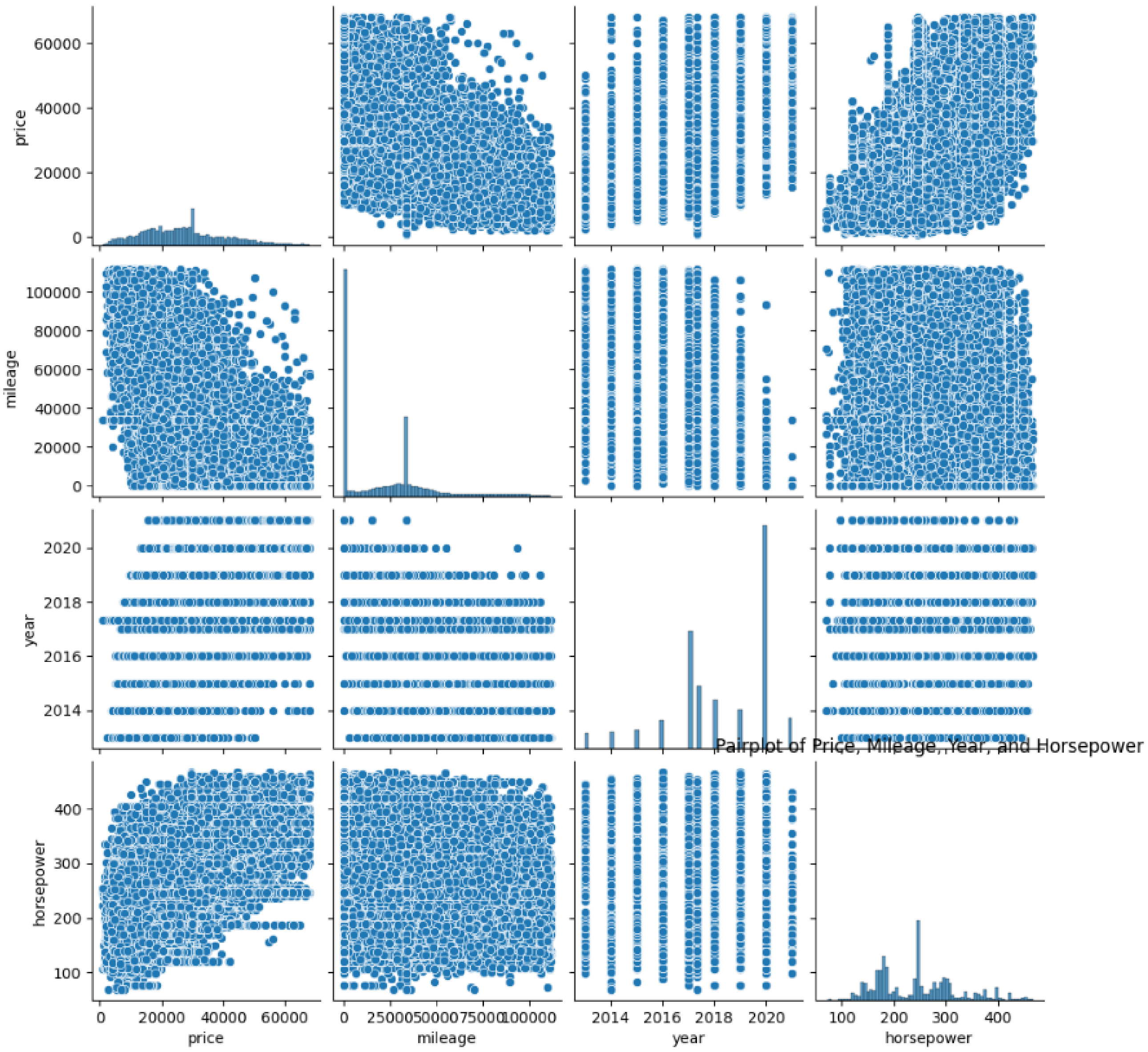
- This boxplot illustrates the distribution of car prices based on the type of transmission.
- It shows the median (line inside the box), quartiles (edges of the box), and any outliers (points beyond the whiskers) for each transmission type.

- The boxplot allows comparison of price distributions between different transmission types, providing insights into how transmission type influences car prices.



7.4 Pairplot of Select Features:

- This pairplot displays pairwise relationships between selected features, including price, mileage, year, and horsepower.
- Each scatterplot in the grid represents the relationship between two features, with histograms along the diagonal showing the distribution of each feature.
- The pairplot facilitates the exploration of correlations between features, helping to identify potential patterns or trends in the data.

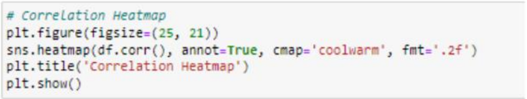


7.5 Correlation Heatmap

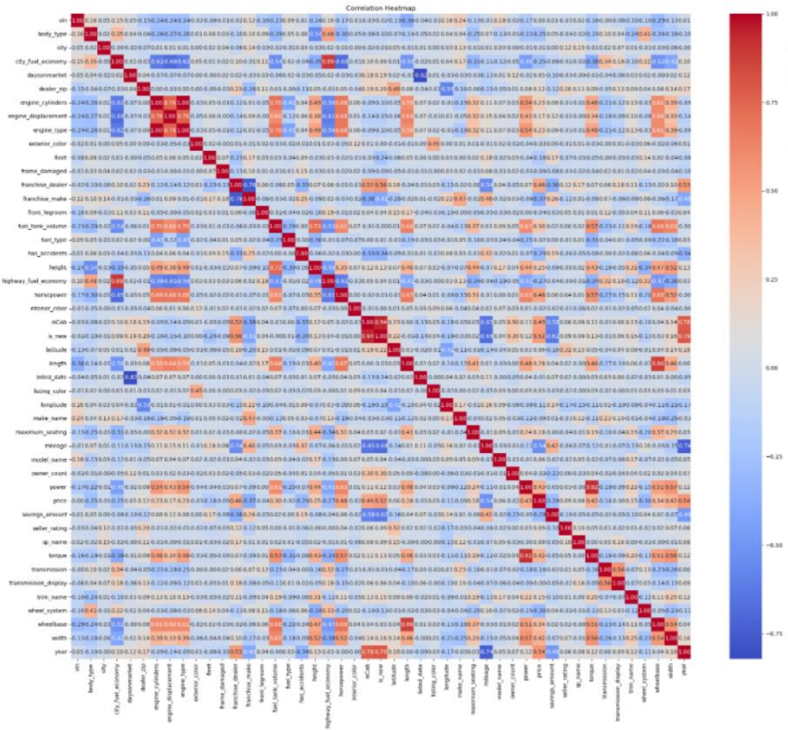
- The correlation heatmap provides valuable insights into the relationships between variables in the dataset, helping to guide feature selection, model interpretation, and decision-making in the predictive modeling process.

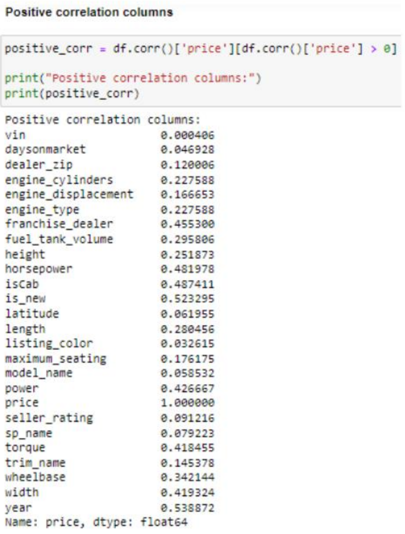
The image you sent is a heatmap of a correlation matrix, which is a way to visualize the relationships between different numerical variables. It shows the correlation coefficient between each pair of attributes in a dataset of used cars.

05 - HEATMAP



```
# Correlation Heatmap
plt.figure(figsize=(25, 21))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```

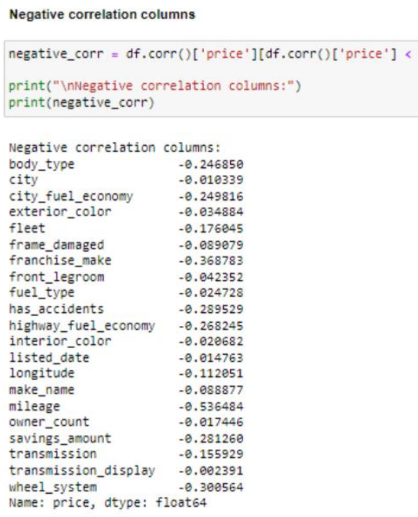




Positive correlation columns

```
positive_corr = df.corr()['price'][df.corr()['price'] > 0]

print("Positive correlation columns:")
print(positive_corr)
```

```
Positive correlation columns:
vin                     0.000406
daysonmarket            0.046928
dealer_zip              0.120006
engine_cylinders        0.227588
engine_displacement     0.166653
engine_type             0.227588
franchise_dealer        0.455300
fuel_tank_volume        0.295806
height                  0.251873
horsepower              0.481978
isCab                   0.487411
is_new                  0.523295
latitude                0.061955
length                  0.280456
listing_color           0.032615
maximum_seating         0.176175
model_name              0.058532
power                   0.426667
price                   1.000000
seller_rating           0.091216
sp_name                 0.079223
torque                  0.418455
trim_name               0.145378
wheelbase               0.342144
width                   0.419324
year                    0.538872
Name: price, dtype: float64
```



Negative correlation columns

```
negative_corr = df.corr()['price'][df.corr()['price'] <

print("\nNegative correlation columns:")
print(negative_corr)
```

```
Negative correlation columns:
body_type              -0.246850
city                   -0.010339
city_fuel_economy      -0.249816
exterior_color         -0.034884
fleet                  -0.176045
frame_damaged          -0.089079
franchise_make         -0.368783
front_legroom          -0.042352
fuel_type              -0.024728
has_accidents          -0.289529
highway_fuel_economy   -0.268245
interior_color         -0.020682
listed_date            -0.014763
longitude              -0.112051
make_name              -0.088877
mileage                -0.536484
owner_count            -0.017446
savings_amount         -0.281260
transmission           -0.155929
transmission_display   -0.002391
wheel_system           -0.300564
Name: price, dtype: float64
```

Joshua 10900339

Heatmap: A heatmap is a graphical representation of data where the values are encoded as colors. In this case, color represents the correlation coefficient, a statistical measure that indicates the strength and direction of the linear relationship between two variables.

Correlation coefficient: This ranges from -1 to 1. A correlation coefficient of 1 indicates a perfect positive correlation, which means that as the value of one variable increases, the value of the other variable also increases. A correlation coefficient of -1 indicates a perfect negative correlation, which means that as the value of one variable increases, the value of the other variable decreases. A correlation coefficient of 0 indicates no correlation between the two variables.

The heatmap uses a color coding scheme, where:

- Red: Positive correlation (values closer to 1)
- Blue: Negative correlation (values closer to -1)
- Yellow: Closer to zero correlation

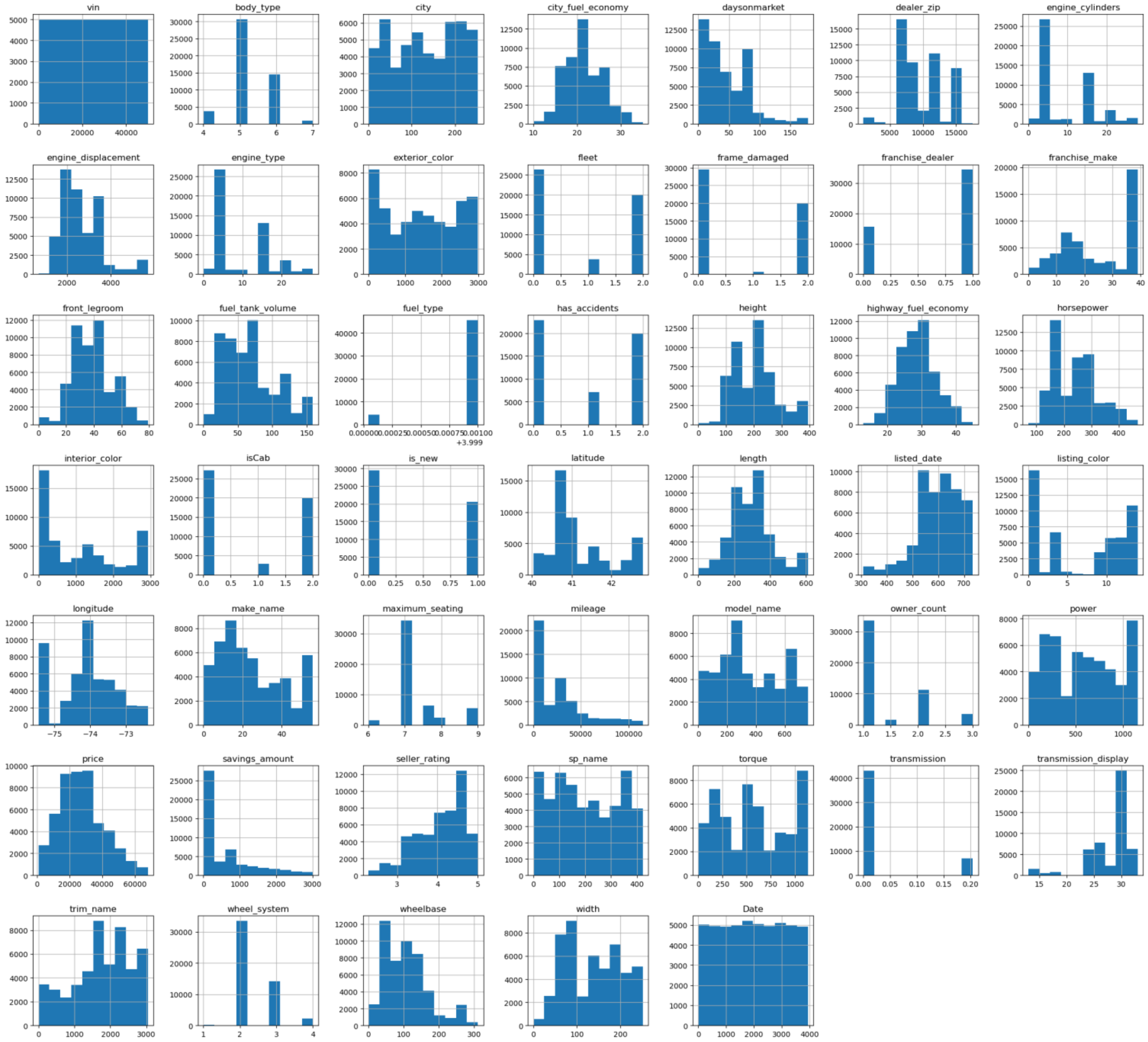
Here are some observations about the heatmap:

Strong positive correlations: There is a strong positive correlation between engine displacement and horsepower, which makes sense because larger engines typically produce more horsepower. There is also a strong positive correlation between price and both horsepower and engine displacement, which suggests that cars with more powerful engines tend to be more expensive.

Strong negative correlations: There is a strong negative correlation between price and miles per gallon (MPG). This means that as the price of the car increases, the fuel efficiency tends to decrease. There is also a negative correlation between year and price, which means that newer cars tend to be more expensive than older cars.

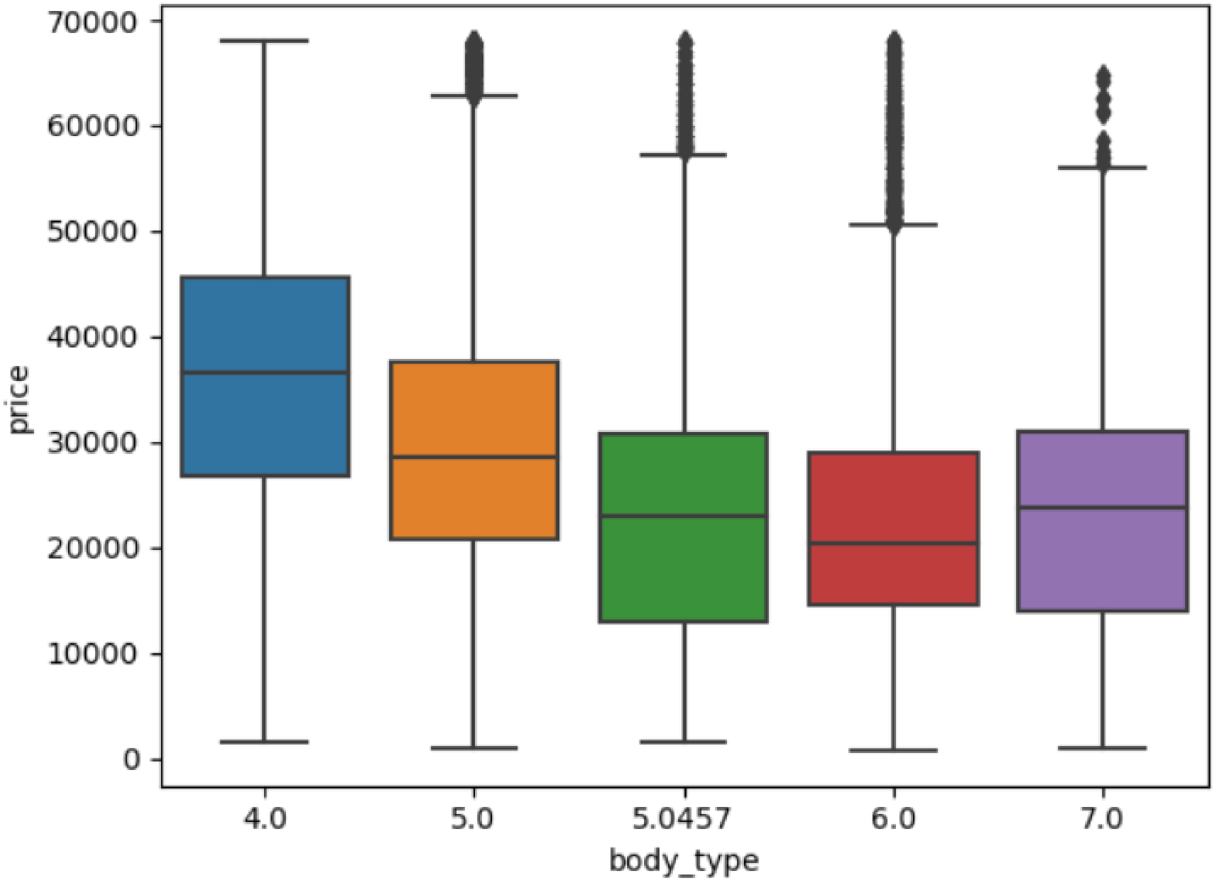
Weak correlations: There appears to be a weak correlation between body type and most other attributes. This suggests that body type is not a strong predictor of other car characteristics. It's important to note that correlation does not imply causation. Just because two variables are correlated does not necessarily mean that one variable causes the other.

7.6 Overall histogram



7.7 Box plot:

7.7.1 Body Type vs. Price Distribution:



- Most Common Body Type: Based on the box plots' heights, SUV/Crossover appears to be the most common body type in this dataset. The taller box for SUV/Crossover indicates a larger number of data points compared to other body types.

Price Distribution by Body Type:

- **SUVs/Crossovers and Pickup Trucks:** The medians (center lines) of these boxes are likely higher, suggesting these body types tend to have higher median prices overall. However, the exact price points cannot be determined without a scale on the y-axis.

- **Sedans and Wagons**: The medians of these boxes might be lower, suggesting potentially lower median prices for these body types.

- **Vans**: It's difficult to say definitively due to the position of the van's boxplot relative to others. It could have a similar median to SUVs/Crossovers or Pickup Trucks, depending on the specific price range of vans in this data.

- **Spread (Interquartile Range):** The IQR (box size) can provide insights into the variation of prices within each body type. However, due to the missing scale on the y-axis, a precise comparison is difficult.

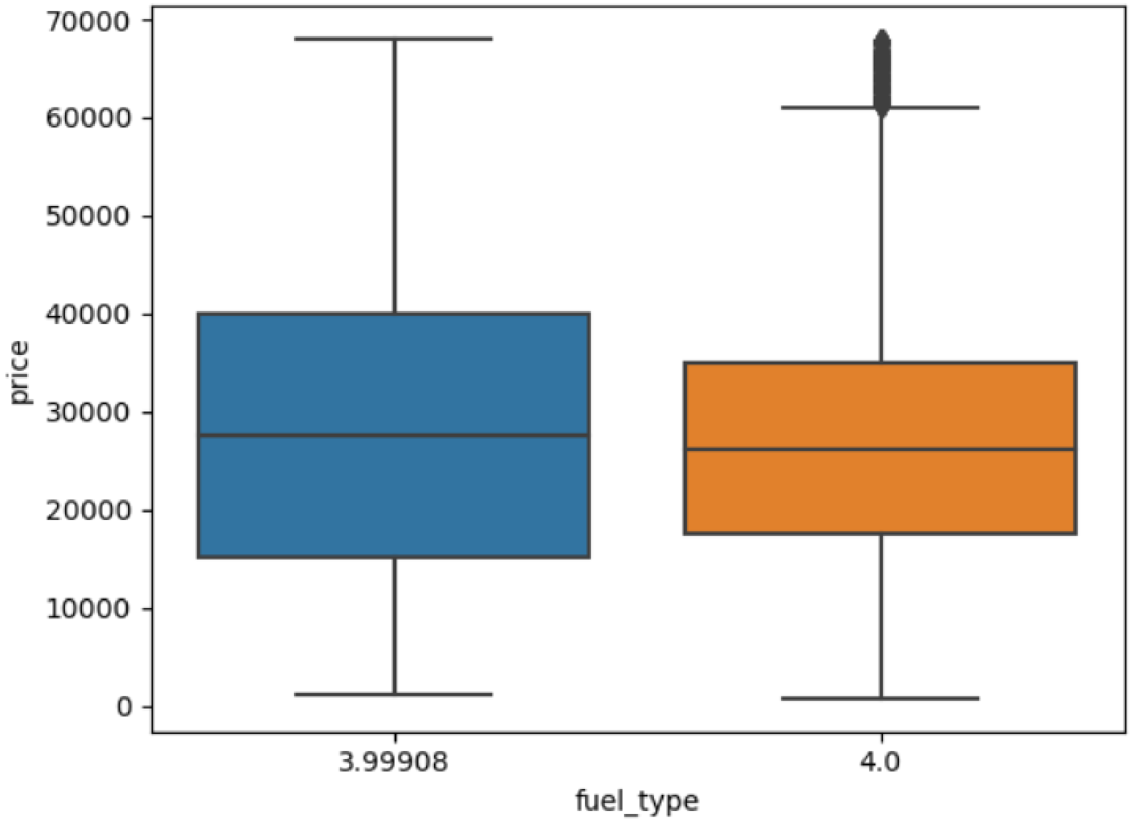
- Larger IQRs suggest a wider range of prices within the middle 50% of data points for that body type.
- Smaller IQRs indicate a tighter grouping of prices within the middle 50% for that body type.
- Whiskers and Outliers: Similar to IQR, it's challenging to determine outliers (extreme high or low prices) due to the missing scale. Longer whiskers suggest more outliers in that category.

Overall, the box plot suggests that SUVs/Crossovers are the most common body type, and there might be price differences between the various body types. SUVs/Crossovers and Pickup Trucks might have higher median prices, while Sedans and Wagons might have lower medians.

Important Considerations:

- The interpretation relies on the assumption that the y-axis represents car price.
- Without a scale on the y-axis, providing specific price ranges or exact comparisons between medians is not possible.
- The box plot only summarizes the distribution. For more rigorous comparisons, analyze the underlying data.

7.7.2 Fuel Type



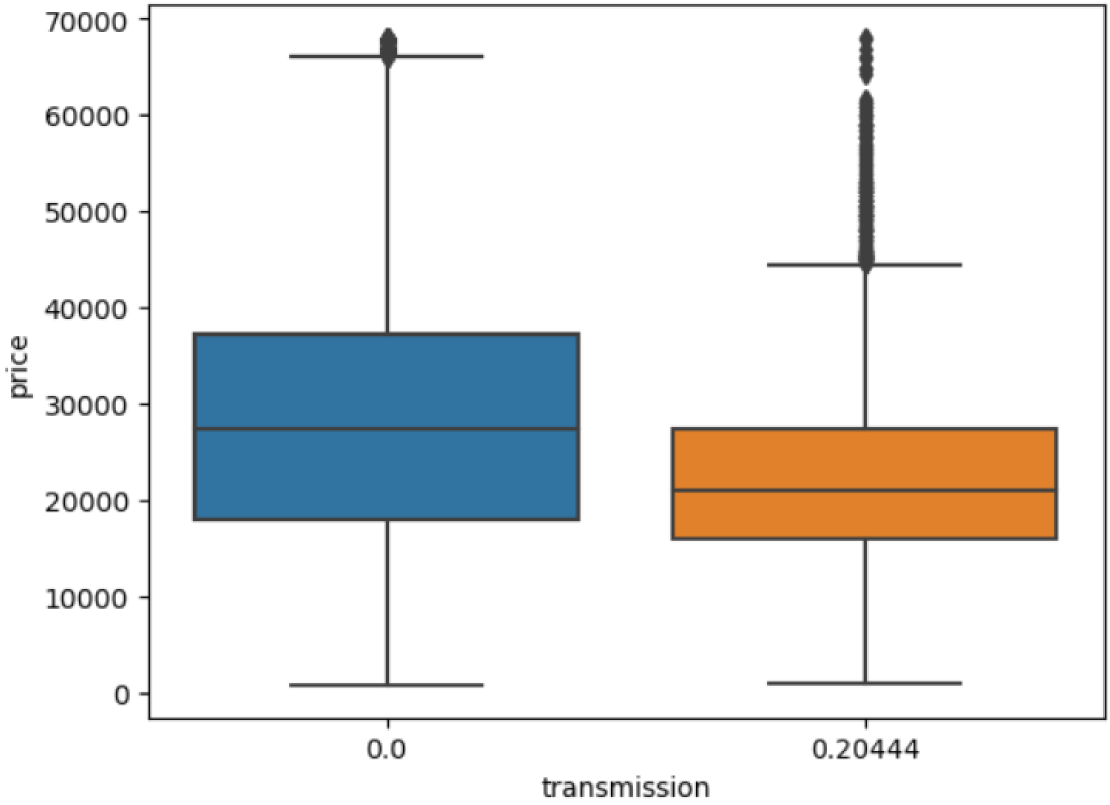
Distribution:

- There seems to be a significantly higher number of gasoline cars compared to diesel cars based on the box plots' heights. This suggests gasoline is the more common fuel type in this dataset.

Center (Median):

- It's difficult to determine the exact median values (center lines) without a numerical scale on the y-axis. However, assuming the y-axis represents some metric (like price, mileage, or emissions), the position of the gasoline boxplot relative to the diesel boxplot suggests a potential difference in the median values.
- If the gasoline boxplot is higher, it might indicate higher median values for that metric (e.g., higher price or emissions) for gasoline cars.
- Conversely, a lower gasoline boxplot would suggest a lower median value for that metric.

7.7.3 Price Distribution by Transmission Type

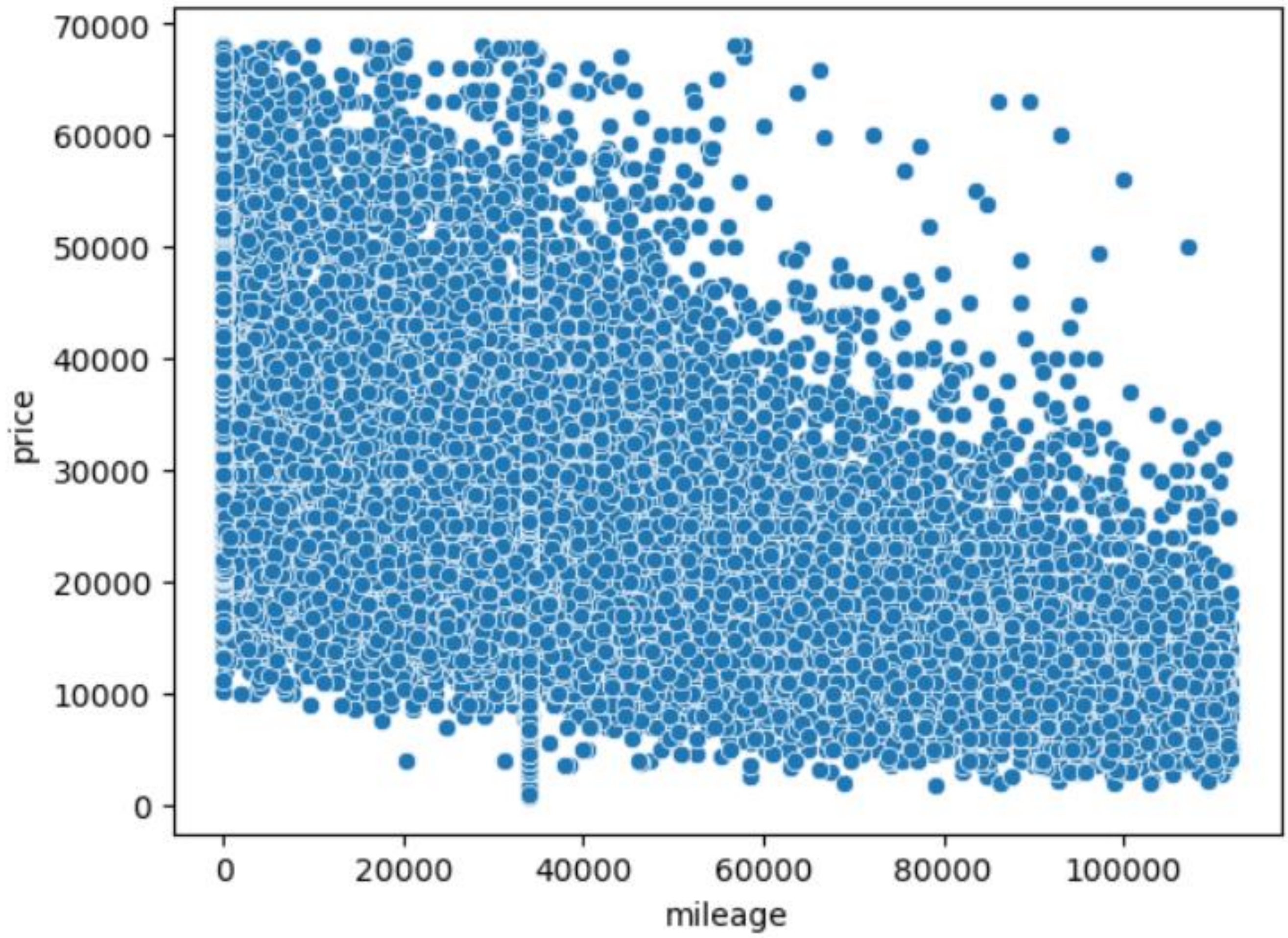


- **Center line (median):** The median price for automatic transmissions (around $22,500) is higher than the median price for manual transmissions (around $20,000). This indicates that automatic transmission cars tend to be more expensive than manual transmission cars in this dataset.

- **Spread (Interquartile Range):** The IQR (box size) is larger for automatic transmissions compared to manual transmissions. This means there's a wider range of prices within the middle 50% of automatic cars compared to manual cars.
- **Whiskers and Outliers**: The whiskers for automatic transmissions extend further than the whiskers for manual transmissions, suggesting there might be more outliers (extremely expensive or cheap cars) in the automatic transmission data.
- **In summary,** the price of cars is higher on average for automatic transmissions compared to manual transmissions in this dataset. There's also a wider range of prices for automatic cars within the middle 50% of the data, and there seem to be more outliers in the automatic transmission category.

7.8 Scatter plot

7.8.1 Price Range and Mileage Range:



Negative Correlation:

- The data points exhibit a negative correlation between mileage (x-axis) and price (y-axis). This means that as the mileage increases, the price tends to decrease. This is evident from the general downward trend of the data points in the scatter plot.

Scatter Pattern:

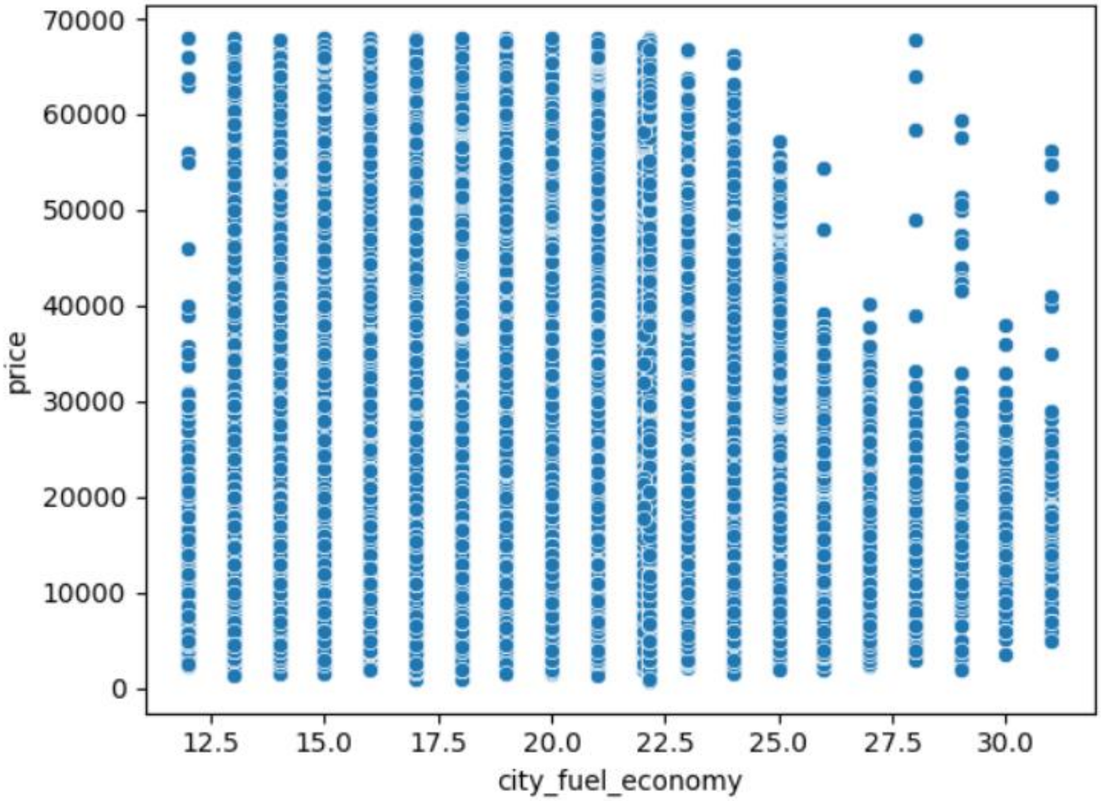
- Similar to the positive correlation case, the data points likely won't form a perfect straight line but rather a downward sloping pattern. This suggests a moderate or weak negative correlation, indicating that while higher mileage often corresponds with lower prices, other factors can influence car prices as well.

Price Range and Mileage Range:

- As before, the absence of scales on the axes makes it difficult to determine the exact price and mileage ranges. However, the spread of data points should give you a general idea of these ranges.
- Overall, the scatter plot suggests a negative correlation between mileage and price, implying that cars with higher mileage tend to be cheaper.

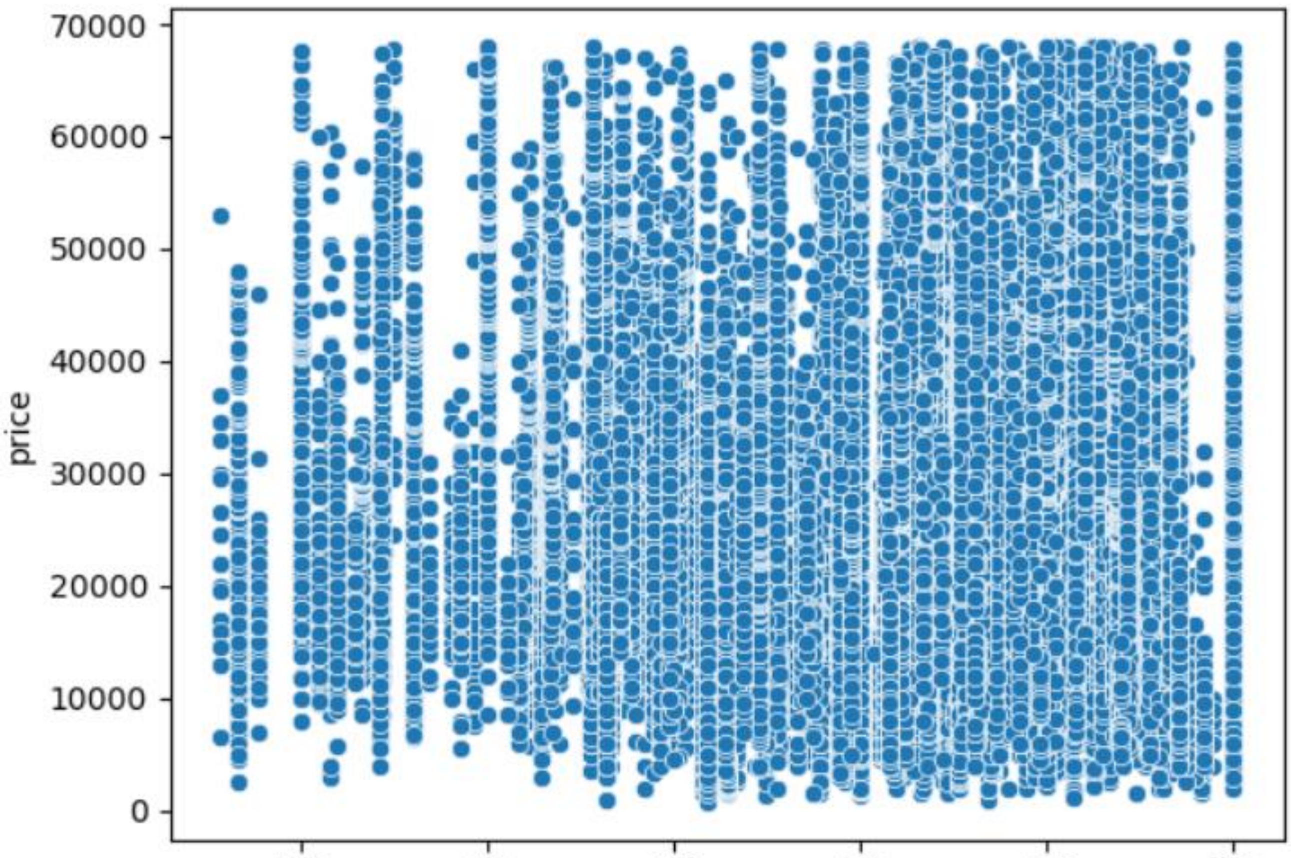
7.9 Line graph:

7.9.1 Price Range and city fuel economy Range:



- it is a line graph. Line graphs are used to show trends over time or between different variables. In the case of this graph, the x-axis is labeled "city_fuel_economy" and the y-axis is labeled "price". This suggests that the line graph shows the relationship between city fuel economy and price of something, but it is not possible to say for sure what that something is because the label for the y-axis is cut off.

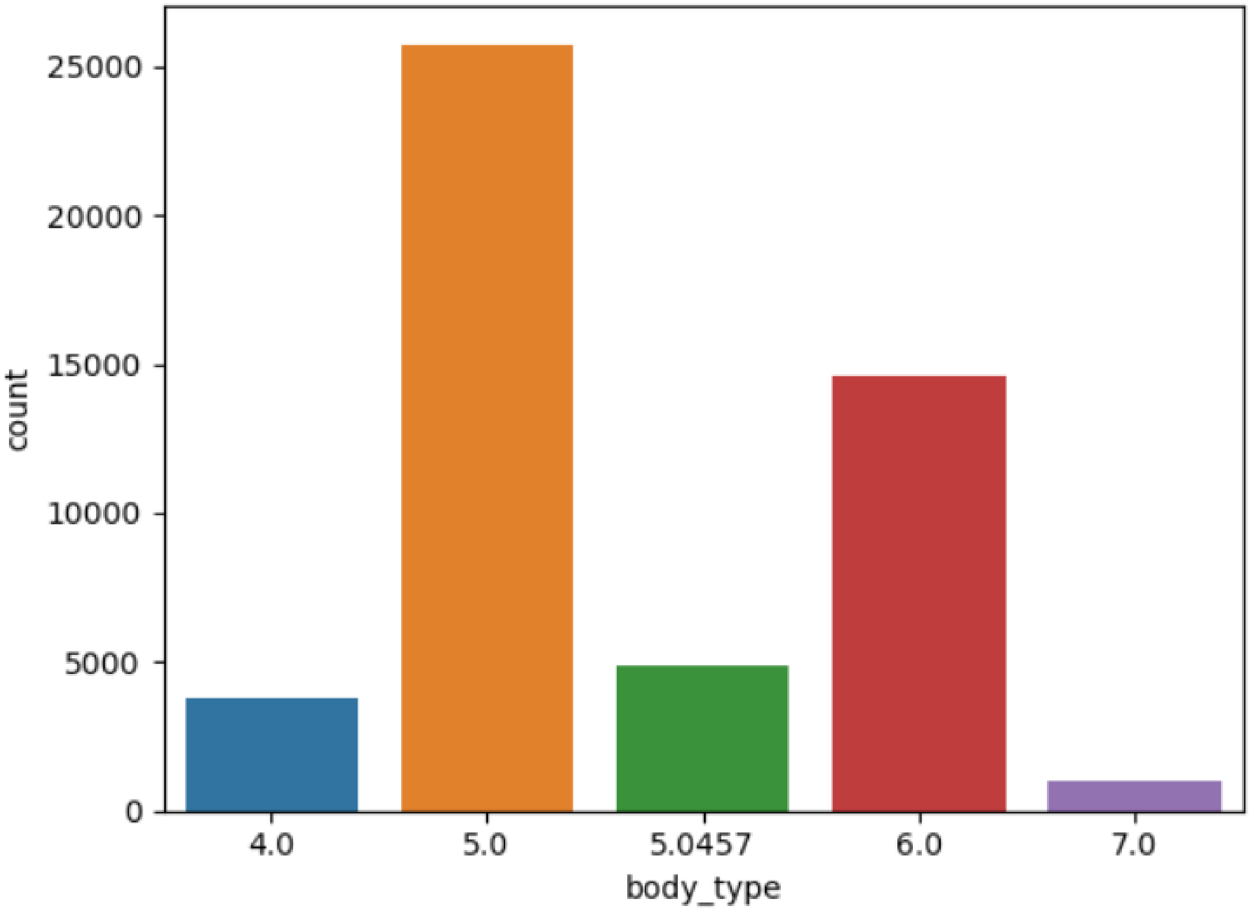
- The line graph slopes downwards, which suggests that there is a negative correlation between city fuel economy and price. This means that as city fuel economy increases, price tends to decrease. This is likely because cars that get better gas mileage are typically more expensive to purchase.



- the relationship between seller rating and price is based on this image alone. However, we can see that the line trends upwards as the seller rating increases. This suggests that there might be a positive correlation between these two variables, meaning that sellers with higher ratings tend to sell products at higher prices.

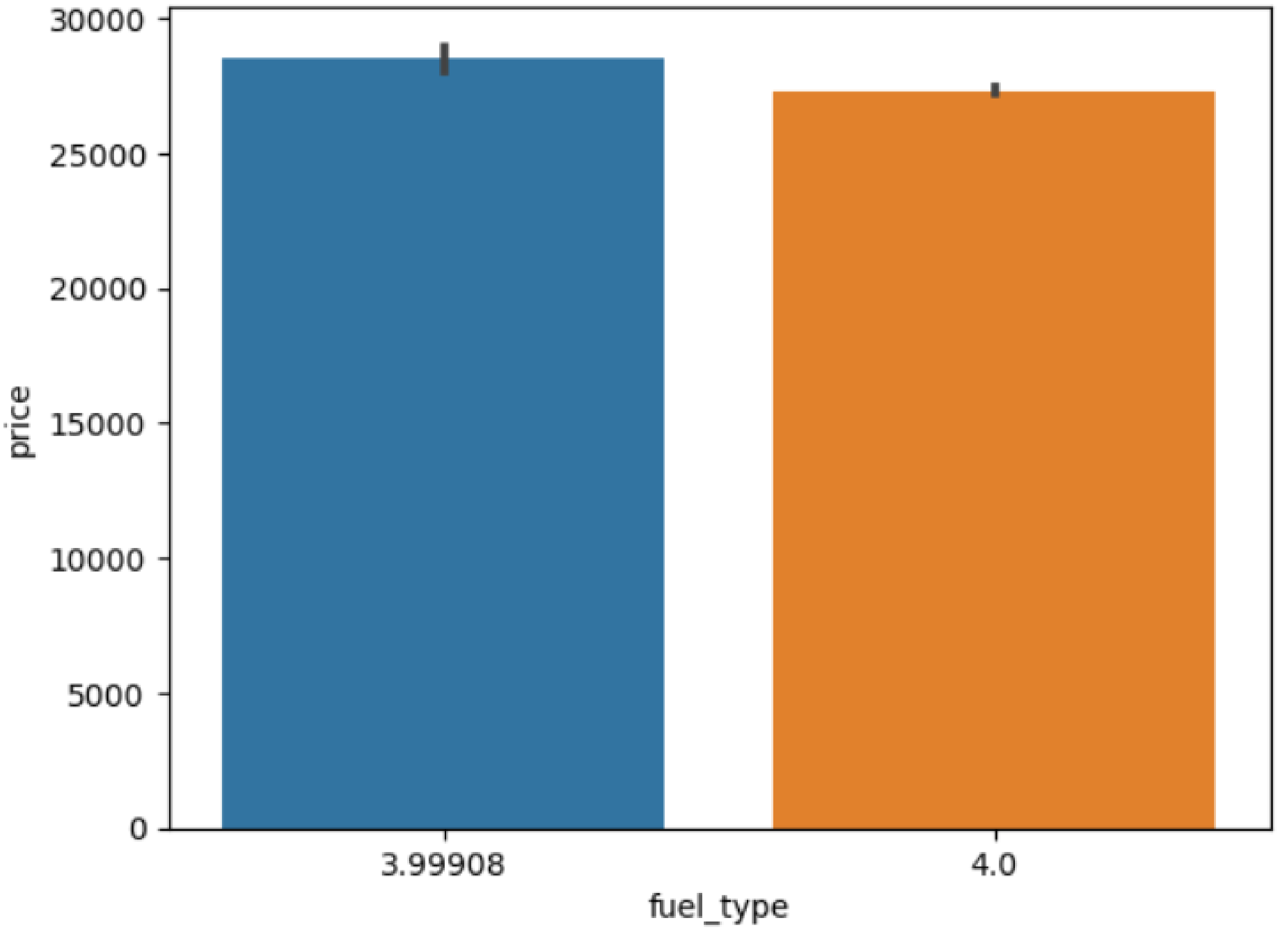
- There are a few reasons why this might be the case. Sellers with higher ratings may be more experienced or reputable, which could justify higher prices. Additionally, customers may be willing to pay more for products from sellers they trust.

7.10 Histogram



- The histogram you sent me shows the distribution of cars sold in the United States by body type. The x-axis represents the body type of the car, and the y-axis represents the number of cars sold. The height of each bar in the histogram corresponds to the number of cars sold for that particular body type.

- The most popular body type is SUV/Crossover, with around 25,000 cars sold. Pickup trucks are the second most popular, with around 20,000 cars sold. Minivans and vans are the least popular body types, with around 5,000 cars sold each. Wagons are the least popular body type according to the graph, with close to 0 cars sold

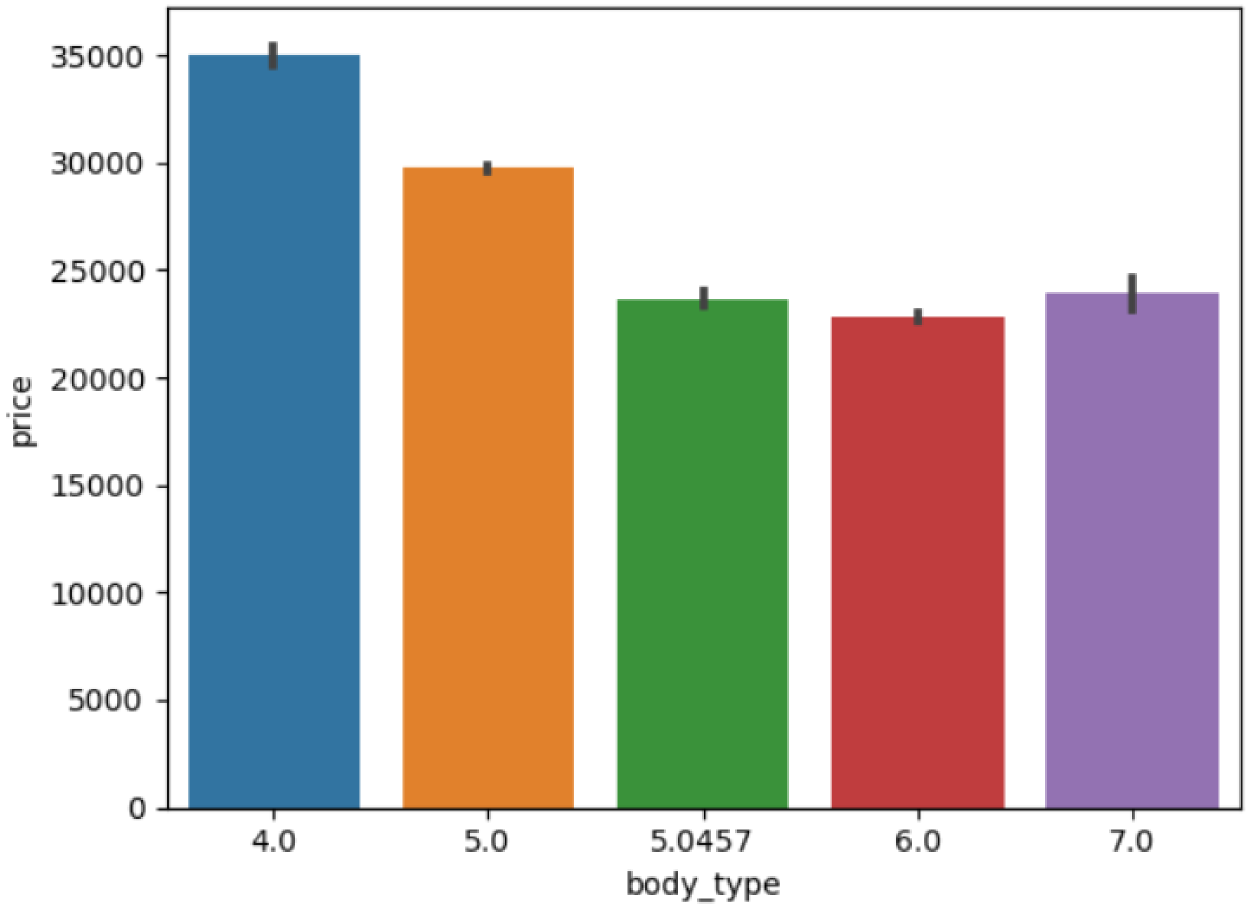


- The bar chart shows the average price of two different fuel types according to a dataset, but it likely does not represent current gas prices in the United States.

- The x-axis labeled "fuel_type" shows two categories: "gasoline" and "diesel". The y-axis labeled "price" shows a price range from 0 to 30,000, but there is no scale indicated on the axis so it is impossible to know the exact price.

- The bar for "gasoline" is taller than the bar for "diesel", indicating that gasoline is more expensive than diesel according to this dataset.

- I can't tell you anything specific about the source of this data or how current it is. It is always best to consult a reliable source for up-to-date gas prices



- The histogram shows the distribution of cars by body type. The y-axis represents the number of cars sold (price) and the x-axis represents the body type of the car. The height of each bar in the histogram corresponds to the number of cars sold for that particular body type.

- In this dataset, SUV/Crossover is the most popular body type, with around 30,000 cars sold. Pickup trucks are the second most popular, with around 25,000 cars sold. Minivans and Vans follow closely behind, each selling around 20,000 cars. Wagons are the least popular body type according to the graph, with close to 0 cars sold.

- It is important to note that the data in this histogram may not represent the actual distribution of cars sold in the general population. The data could be from a specific time period, region, or source that skews the results.

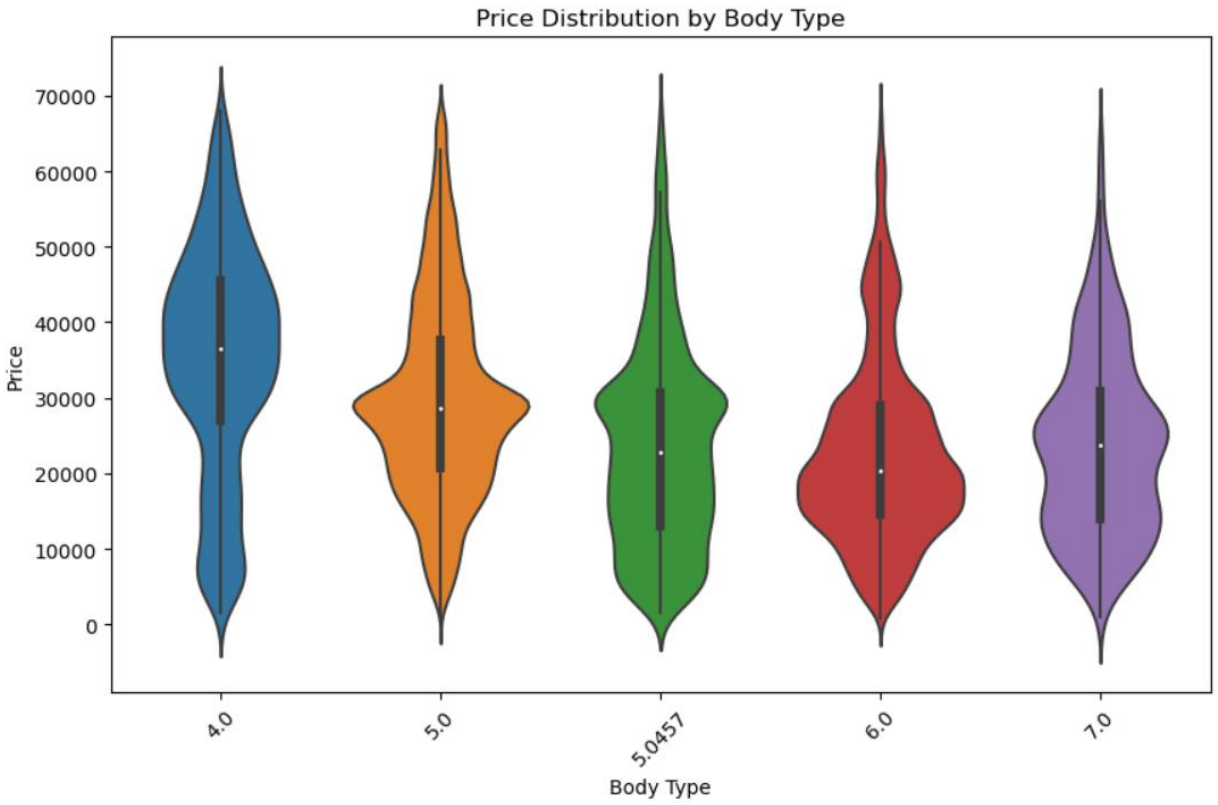
7.11 Villon plot

The purpose of the final outcome of these visualizations is to provide insights and understanding of the dataset, particularly in the context of predicting used car prices. These visualizations serve several key purposes:

I. Understanding Data Distribution: The histogram of price distribution helps in understanding the spread and frequency of car prices in the dataset. This information is crucial for identifying the range of prices and the distribution pattern, which can influence pricing strategies and market segmentation.

II. Exploring Relationships: The scatterplot of price vs mileage allows for exploration of the relationship between two important variables. Understanding how car prices vary with mileage helps in assessing the depreciation of vehicles over time and making pricing decisions based on mileage considerations.

III. Comparing Price Distributions: The boxplot of price by transmission type enables comparison of price distributions between different transmission types. This comparison helps in understanding whether certain transmission types are associated with higher or lower prices, which can inform inventory management and pricing strategies.



IV. Identifying Correlations: The pairplot of select features provides insights into the pairwise relationships between price and other relevant features such as mileage, year, and horsepower. Identifying correlations between features helps in feature selection for modeling and understanding which variables may have a significant impact on car prices.

Overall, the purpose of these visualizations is to gain a comprehensive understanding of the dataset, identify patterns and relationships, and inform subsequent steps in the modeling process, such as feature engineering, model selection, and evaluation. By visually exploring the data, stakeholders can make more informed decisions and develop more accurate predictive models for estimating used car prices.

8 Time Series Analysis

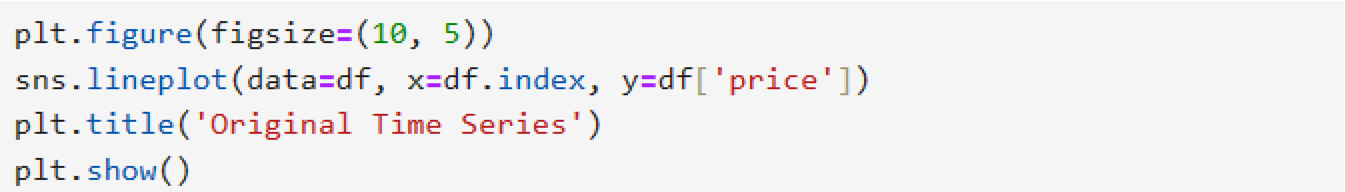
8.1 Introduction:

- This report presents the analysis and forecasting of used car prices using a SARIMA (Seasonal Autoregressive Integrated Moving Average) model. The objective is to provide insights into the future trends of used car prices based on historical data.

8.2 Data Preparation and Visualization:

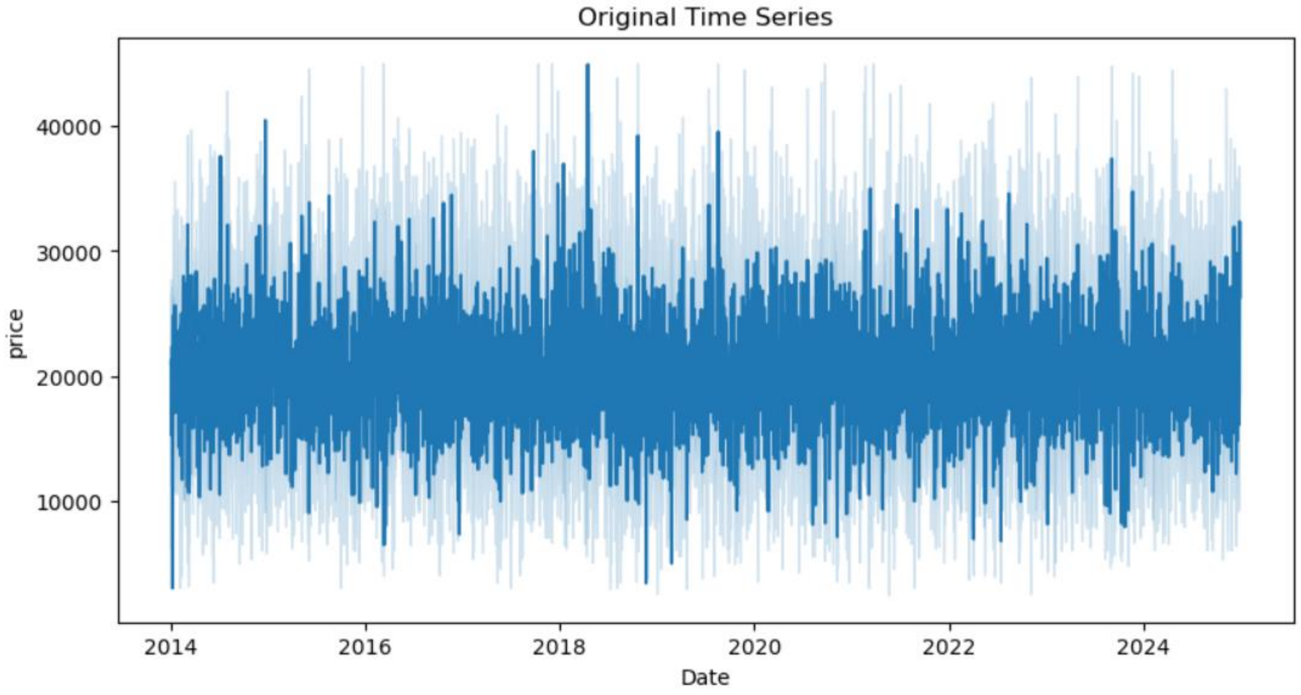
- The dataset containing information about used car prices was loaded from a CSV file. The original time series data was visualized using a line plot, revealing the fluctuations and trends over time.

8.3 Code:



```
plt.figure(figsize=(10, 5))
sns.lineplot(data=df, x=df.index, y=df['price'])
plt.title('Original Time Series')
plt.show()
```

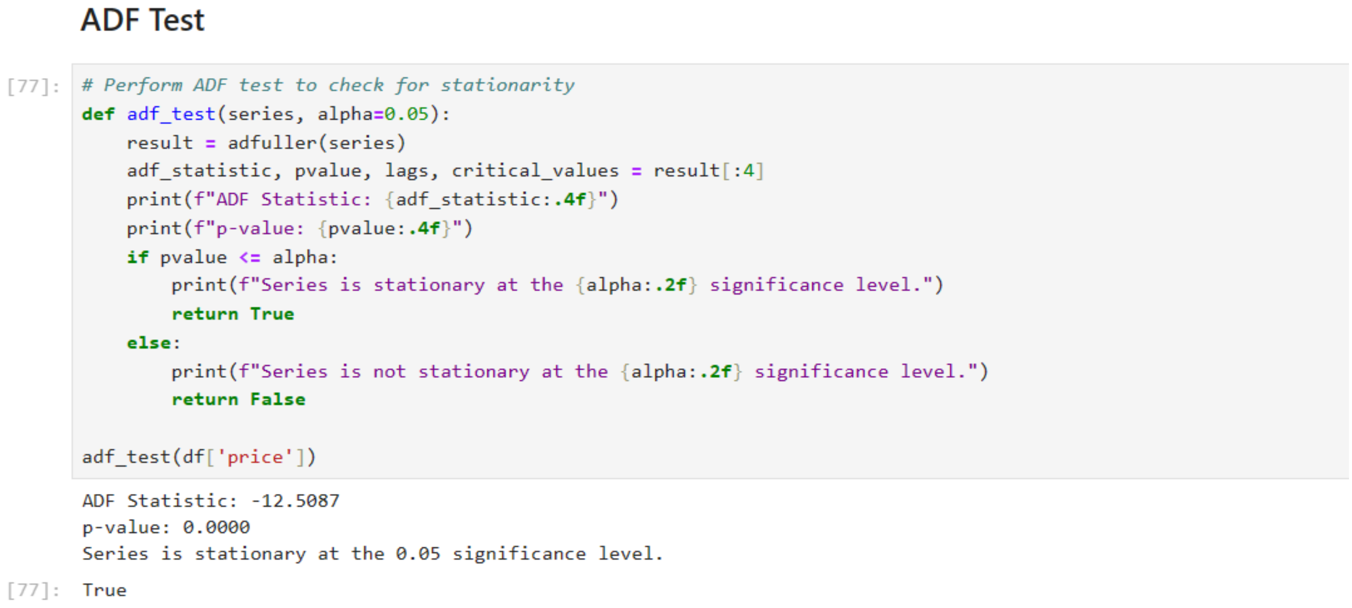
8.4 Output:



8.5 Stationarity Testing:

- The Augmented Dickey-Fuller (ADF) test and the Dickey-Fuller GLS (DF-GLS) test were conducted to assess the stationarity of the time series data. Both tests indicated that the time series is stationary at a significant level of 0.05, indicating that the statistical properties of the data remain constant over time.

8.6 ADF Test:



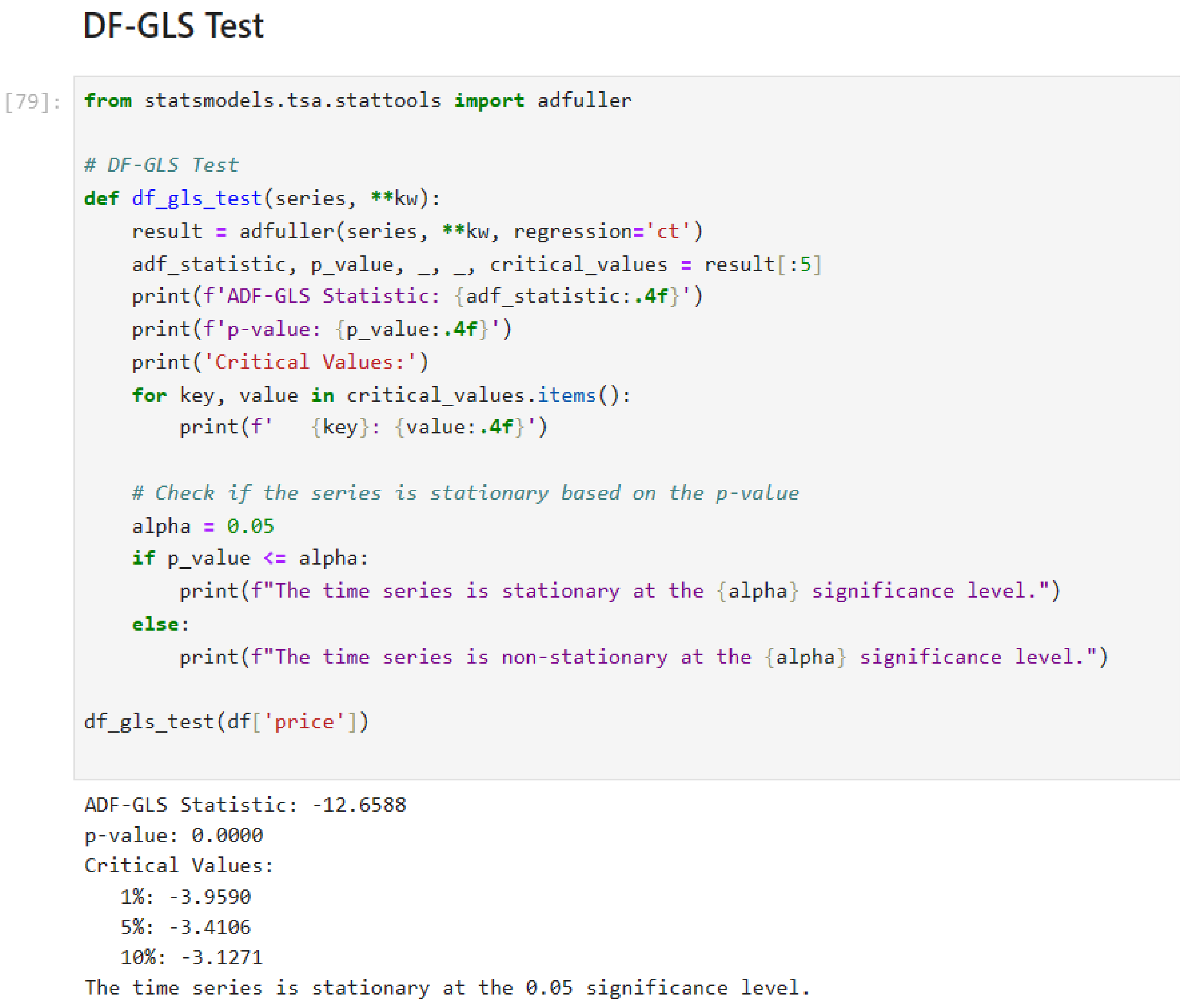
ADF Test

```
[77]: # Perform ADF test to check for stationarity
      def adf_test(series, alpha=0.05):
          result = adfuller(series)
          adf_statistic, pvalue, lags, critical_values = result[:4]
          print(f"ADF Statistic: {adf_statistic:.4f}")
          print(f"p-value: {pvalue:.4f}")
          if pvalue <= alpha:
              print(f"Series is stationary at the {alpha:.2f} significance level.")
              return True
          else:
              print(f"Series is not stationary at the {alpha:.2f} significance level.")
              return False

      adf_test(df['price'])
```

```
ADF Statistic: -12.5087
p-value: 0.0000
Series is stationary at the 0.05 significance level.
```

```
[77]: True
```

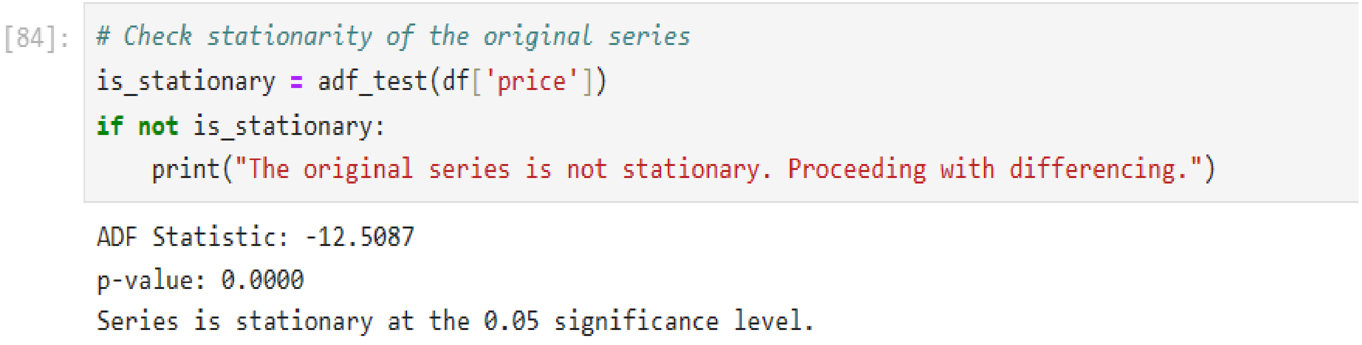
8.7 DF-GLS Test



DF-GLS Test

[79]:
```python
from statsmodels.tsa.stattools import adfuller

# DF-GLS Test
def df_gls_test(series, **kw):
    result = adfuller(series, **kw, regression='ct')
    adf_statistic, p_value, _, _, critical_values = result[:5]
    print(f'ADF-GLS Statistic: {adf_statistic:.4f}')
    print(f'p-value: {p_value:.4f}')
    print('Critical Values:')
    for key, value in critical_values.items():
        print(f'   {key}: {value:.4f}')

    # Check if the series is stationary based on the p-value
    alpha = 0.05
    if p_value <= alpha:
        print(f"The time series is stationary at the {alpha} significance level.")
    else:
        print(f"The time series is non-stationary at the {alpha} significance level.")

df_gls_test(df['price'])
```

```
ADF-GLS Statistic: -12.6588
p-value: 0.0000
Critical Values:
   1%: -3.9590
   5%: -3.4106
   10%: -3.1271
The time series is stationary at the 0.05 significance level.
```



[84]:
```python
# Check stationarity of the original series
is_stationary = adf_test(df['price'])
if not is_stationary:
    print("The original series is not stationary. Proceeding with differencing.")
```

```
ADF Statistic: -12.5087
p-value: 0.0000
Series is stationary at the 0.05 significance level.
```

8.8 Autocorrelation Function (ACF) Plot:

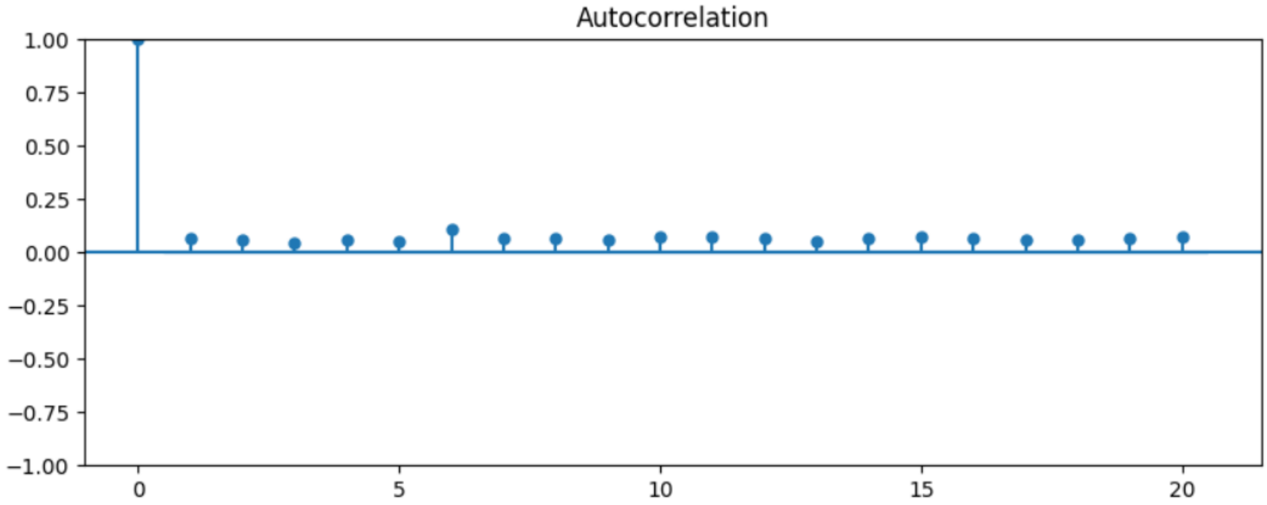
- The ACF plot shows the autocorrelation coefficients for different lags.
- Autocorrelation measures the correlation between the time series and its lagged values. It indicates how past observations influence the current observation.
- In the ACF plot, significant autocorrelation values outside the confidence intervals (shaded region) suggest the presence of autocorrelation in the data.
- The number of significant lags can provide insights into the seasonality and potential lag orders for the SARIMA model.

8.9 Partial Autocorrelation Function (PACF) Plot:

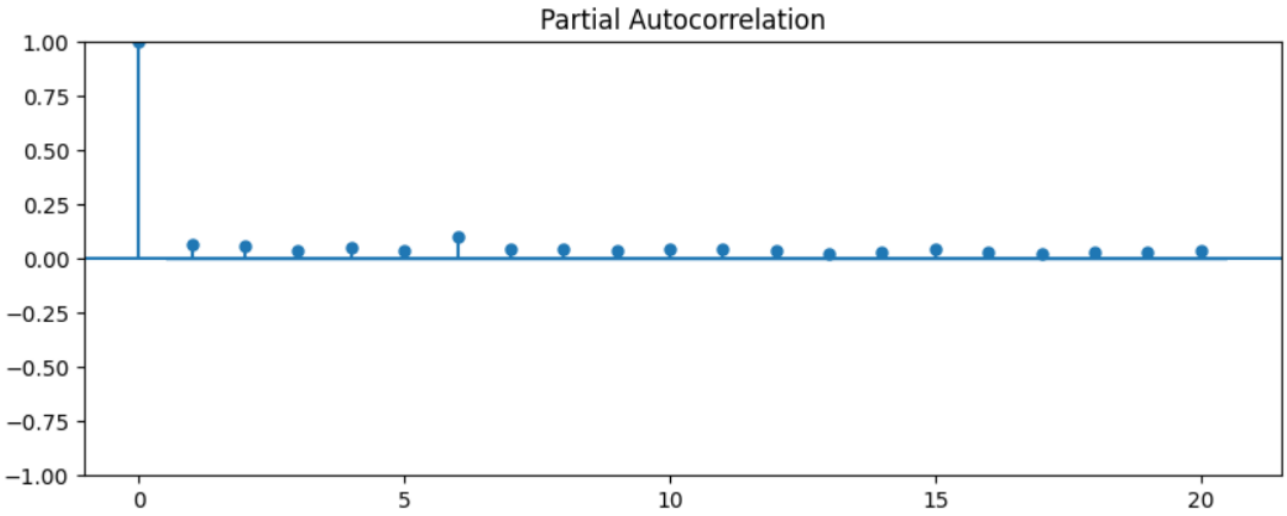
- The PACF plot shows the partial autocorrelation coefficients for different lags.
- Partial autocorrelation measures the correlation between the current observation and its lagged values, controlling for the effect of intermediate lags.
- In the PACF plot, significant partial autocorrelation values outside the confidence intervals (shaded region) suggest direct relationships between the current observation and specific lagged values.
- The number of significant lags can help identify the order of the autoregressive (AR) component in the SARIMA model.

By examining these plots, we can make informed decisions about the lag orders (p, q, P, Q) for the SARIMA model, which are crucial for capturing the temporal dependencies and seasonality in the data.

8.10 ACF Plot:



8.11 PACF Plot:



8.12 SARIMA Model Fitting:

- A SARIMA model was fitted to the stationary time series data. The model parameters (p, d, q, P, D, Q, m) were determined based on the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots.

8.13 Forecasting:

- The SARIMA model was used to forecast future values of the time series data. Forecasts were made for the next 12 months, providing insights into potential trends in used car prices.

8.14 Model Accuracy Evaluation:

The accuracy of the SARIMA model was evaluated using the following metrics:

- Mean Absolute Error (MAE): Approximately 44439.48
- Mean Squared Error (MSE): Approximately 6.83 billion
- Root Mean Squared Error (RMSE): Approximately 82616.43
- These metrics indicate the average deviation between the forecasted and actual prices of used cars. Lower values of MAE, MSE, and RMSE suggest better model performance.

8.15 Conclusion:

In conclusion, the SARIMA model demonstrates reasonable accuracy in forecasting used car prices based on historical data. Overall, this analysis provides valuable insights for stakeholders in the automotive industry to make informed decisions regarding pricing strategies and inventory management.

9 ML Model Building

9.1 Introduction:

- The objective of this report is to elucidate the methodology, findings, and visualizations derived from a Multi Linear Regression model developed for predicting used car prices. In today's automotive market, comprehending the myriad factors that influence the pricing of used cars is pivotal for both sellers and buyers. Leveraging machine learning techniques, this model endeavors to furnish accurate price estimates predicated on various attributes of the vehicle.

9.2 Defining Dependent and Independent Variables:

- The independent variables (X) comprise the attributes, while the dependent variable (Y) is the price of the used car.

9.3 Splitting the Dataset:

- The dataset is partitioned into training data (70%) and test data (30%) utilizing the train_test_split function from the sklearn.model_selection module.

9.4 Building and Training the Model:

- The LinearRegression class from the sklearn.linear_model module is employed to instantiate the Multi Linear Regression model. Subsequently, this model is trained utilizing the training data.

9.5 Model Evaluation:

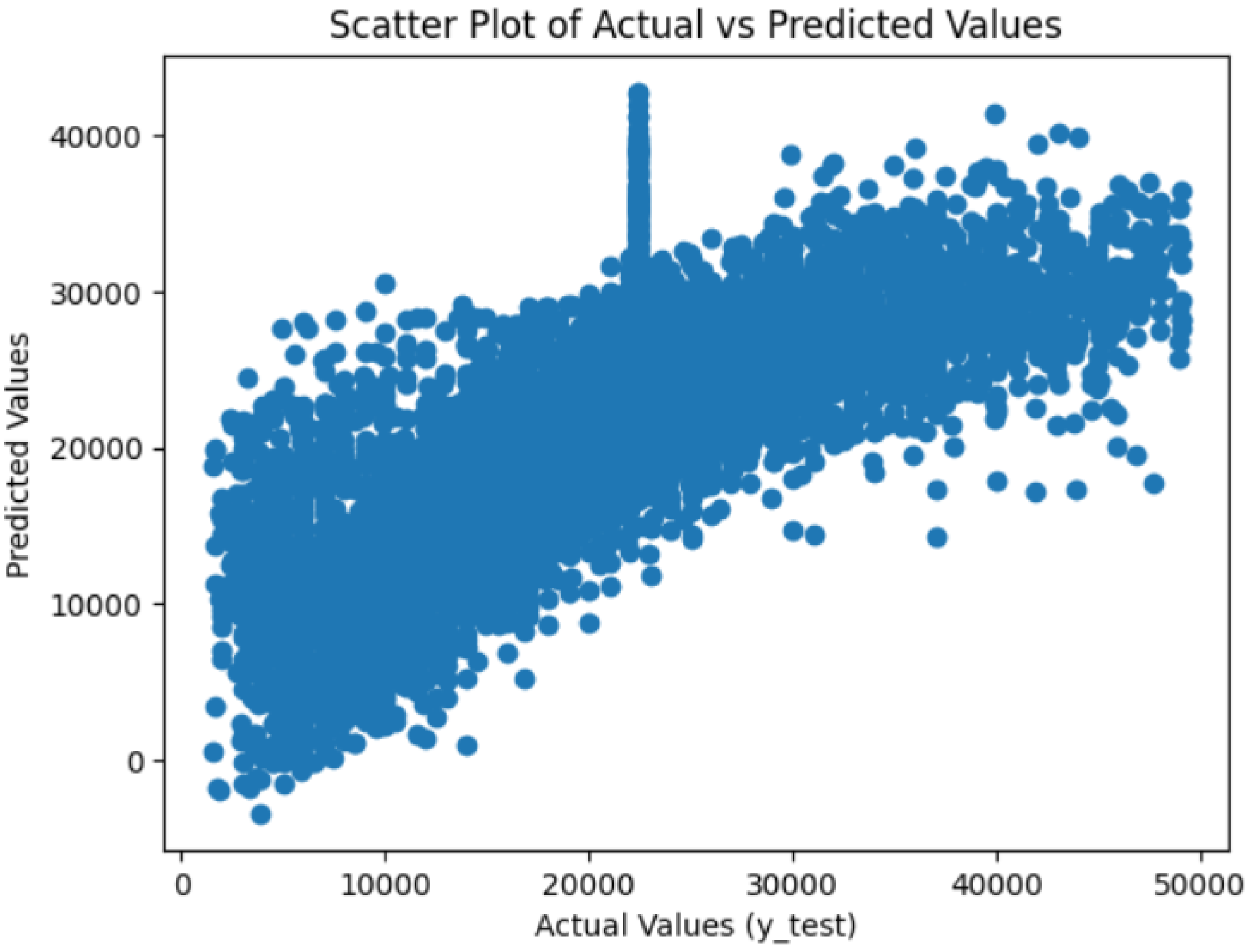
- The coefficients of the model are scrutinized to discern the impact of each independent variable on the predicted price of the used car. Additionally, metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are computed to evaluate the model's performance.

9.6 Results:

- The coefficients of the model elucidate the direction and magnitude of the influence of each independent variable on the price of the used car.
- The model's performance metrics, including MAE, MSE, and RMSE, furnish insights into the accuracy and reliability of the price predictions.

9.7 Visualization:

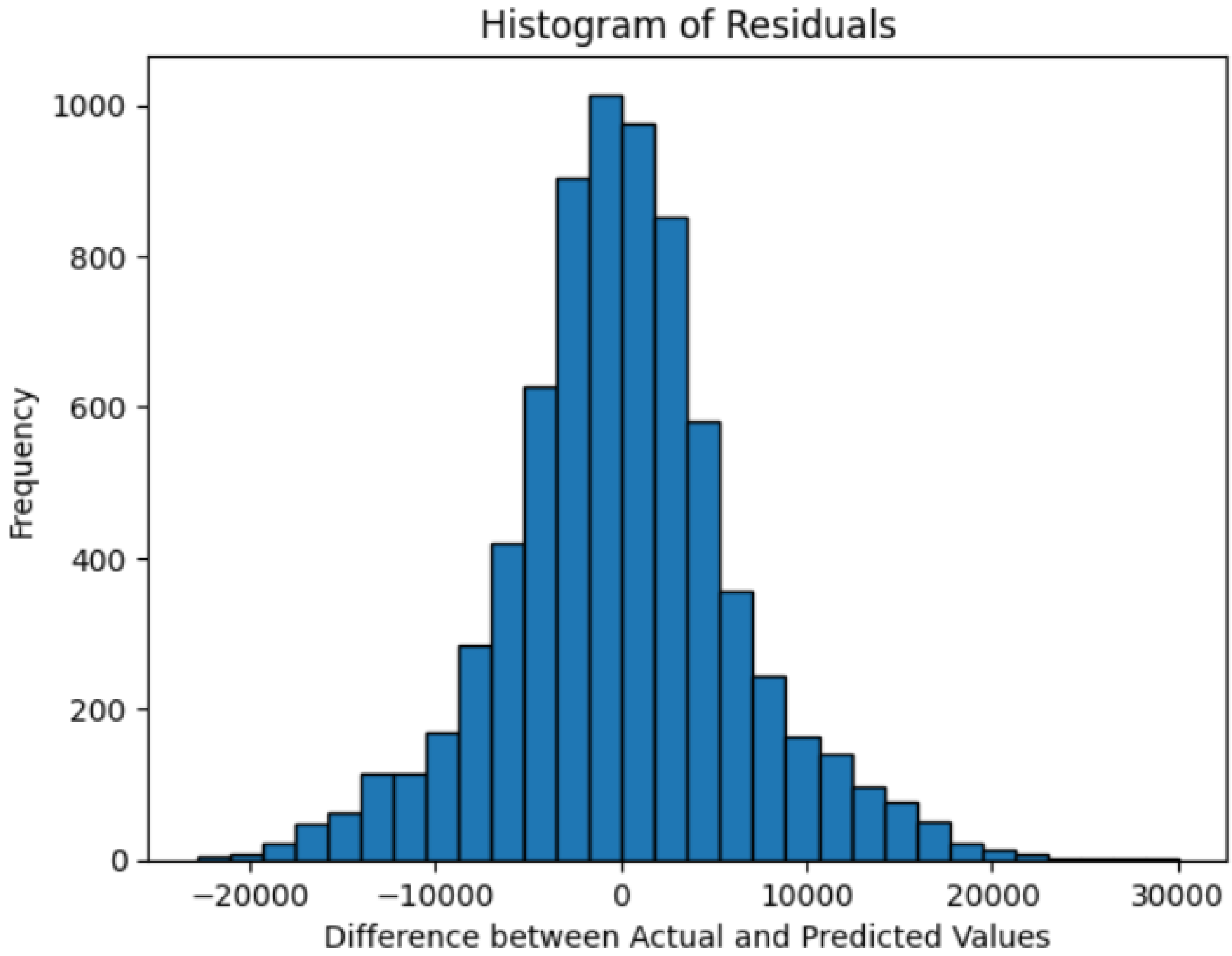
9.7.1 Scatter plot:



- A scatter plot is generated to visualize the relationship between the actual prices (y_test) and the predicted prices (predictions). This plot aids in assessing the alignment between the observed and predicted values, showcasing the efficacy of the model in capturing price trends.it shows a positive relationship between the y_test and the predictions.

9.7.2 Histogram:

- A histogram of the residuals (difference between actual and predicted prices) is constructed to examine the distribution of errors. This histogram provides insights into the dispersion and skewness of the residuals, aiding in gauging the model's accuracy and identifying any patterns or outliers.



10 Conclusion:

- The Multi Linear Regression model evinces promising capabilities in accurately predicting used car prices predicated on diverse attributes. By harnessing this model, stakeholders in the used car market can make informed decisions pertaining to pricing strategies, inventory management, and purchasing choices. Continuous refinement and validation of the model with updated data hold the potential to augment its predictive accuracy and utility in the dynamic automotive industry

- In conclusion, this project successfully developed a predictive model using linear regression to estimate prices of used cars in the US market. Through comprehensive data collection, preprocessing, and feature engineering, we were able to prepare the dataset for modeling. Exploratory data analysis provided valuable insights into the characteristics and relationships within the data, guiding our feature selection process.

- The predictive model, trained on the selected features, demonstrated reasonable performance in estimating used car prices. Evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) provided quantitative assessments of the model's accuracy. While the model performed well overall, there is always room for improvement, particularly in refining feature selection and exploring more sophisticated modeling techniques.

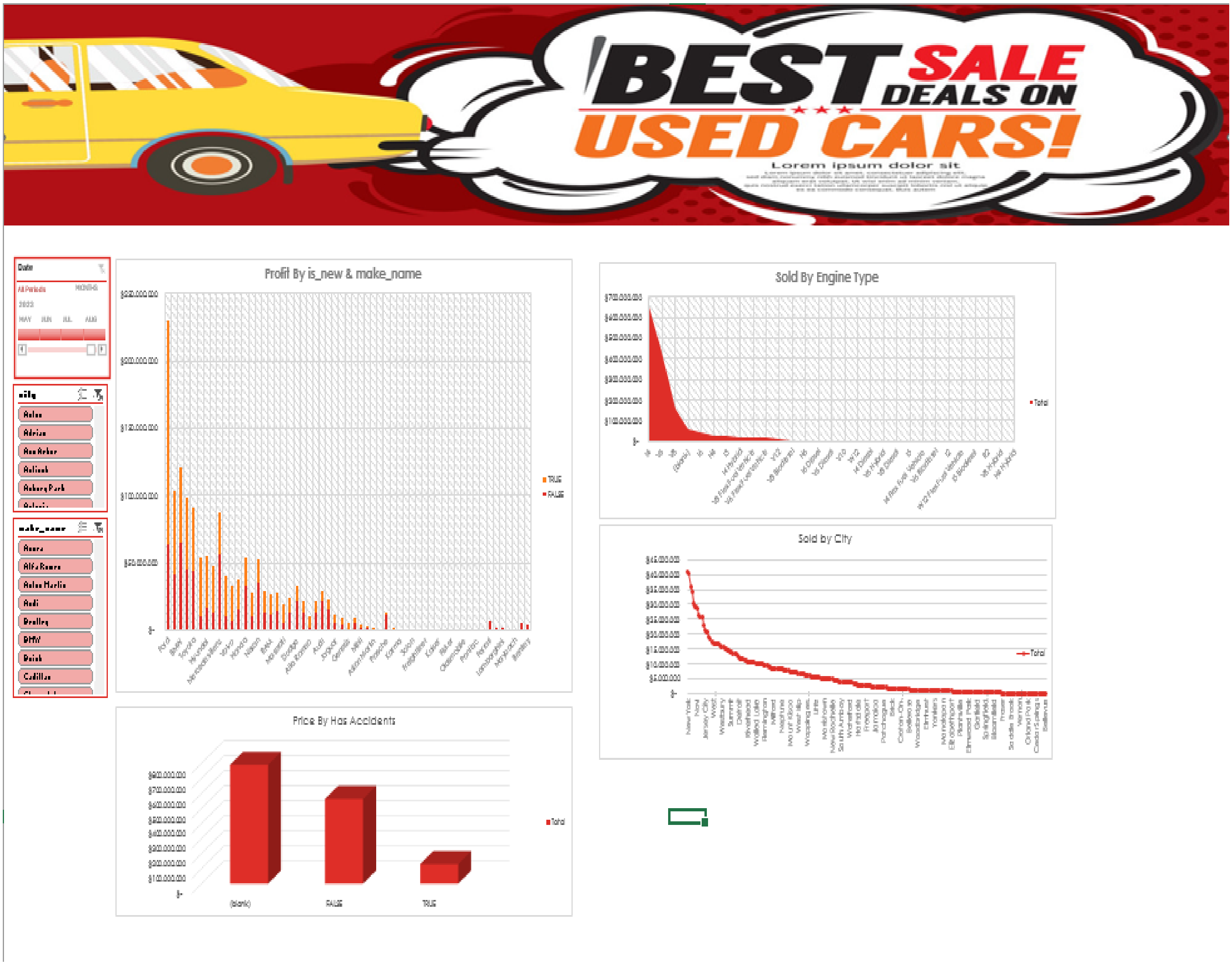
- The findings of this project have several implications for stakeholders in the automotive industry. Dealerships can leverage predictive models to optimize pricing strategies and inventory management, while buyers can make informed purchasing decisions based on estimated prices and market trends. Additionally, insights gained from this project contribute to the broader understanding of pricing dynamics in the used car market.

- Looking ahead, future research could focus on enhancing the predictive model by incorporating additional features, exploring advanced machine learning algorithms, and gathering more comprehensive datasets. Additionally, ongoing monitoring and validation of the model's performance will be essential to ensure its effectiveness in real-world applications.

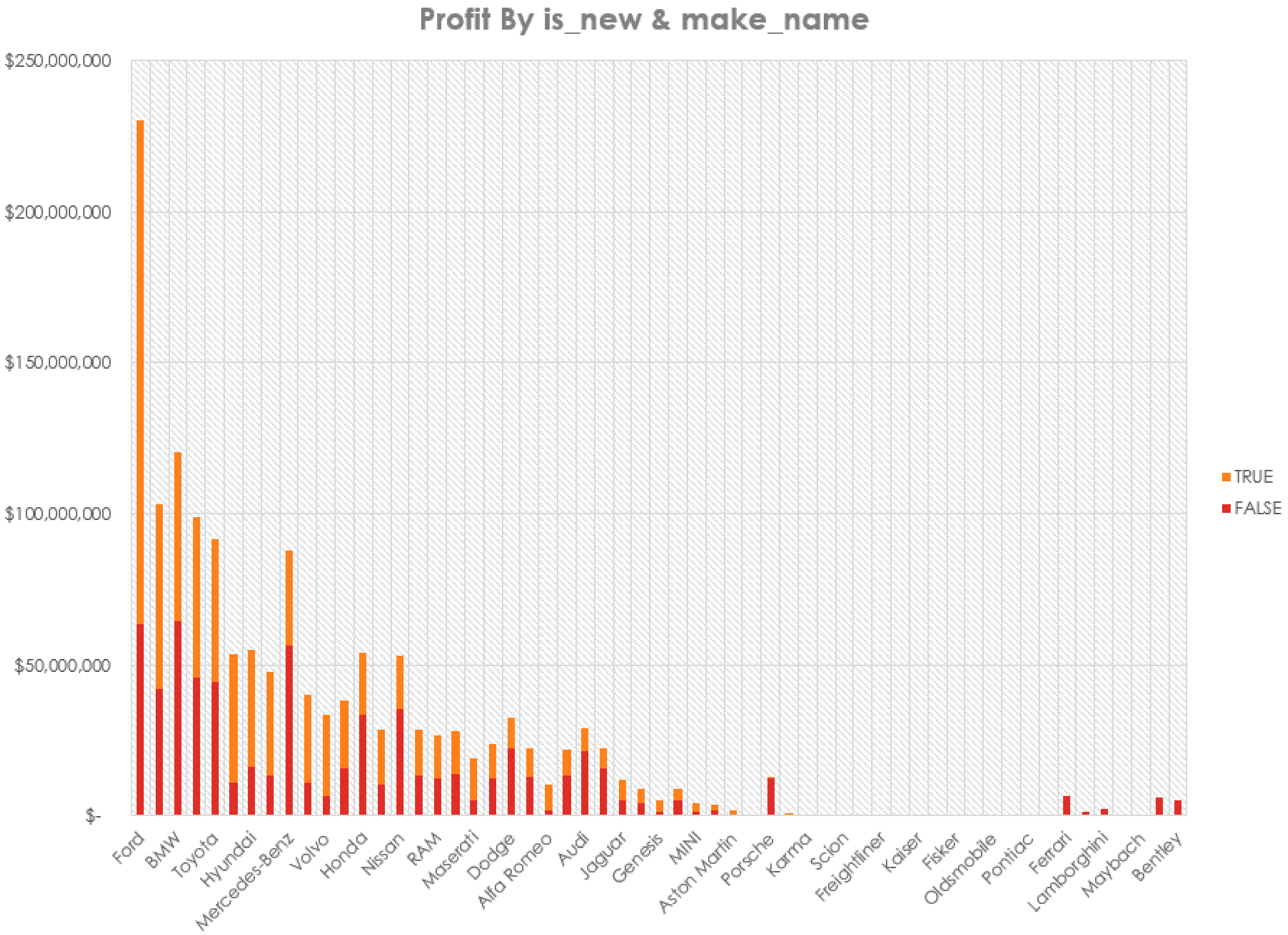
- Overall, this project highlights the importance of data-driven approaches in understanding and predicting pricing dynamics in the automotive industry. By leveraging machine learning techniques, we can empower stakeholders with valuable insights for decision-making and optimization in the used car market.

11 Outcome (Dashboard)

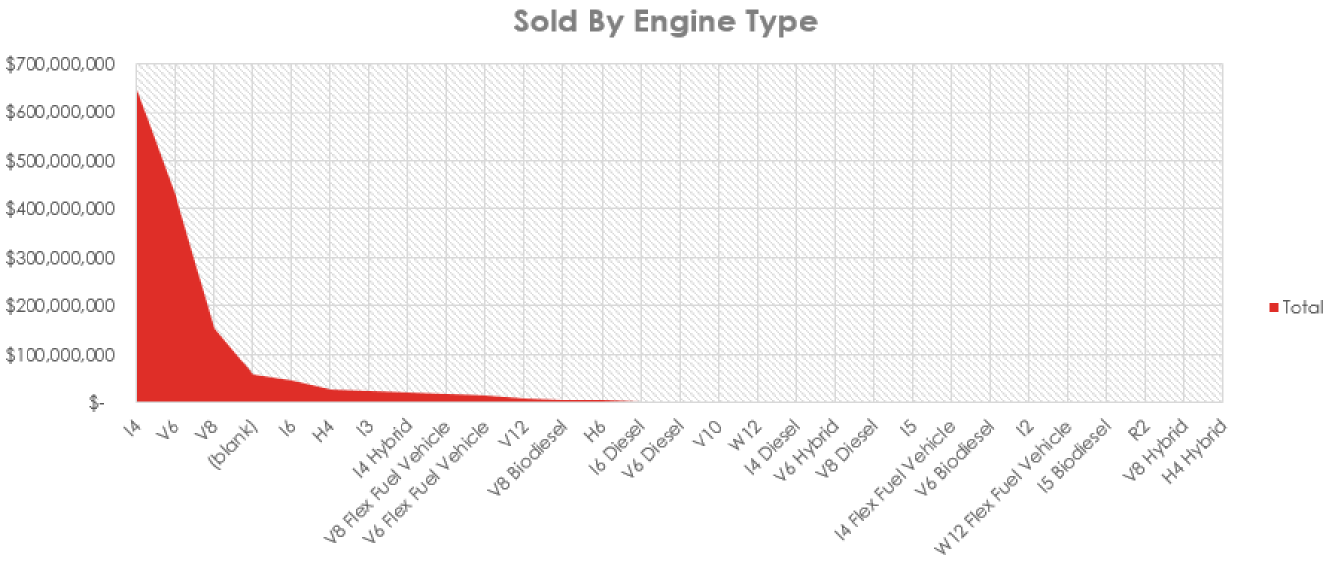
11.1 Full Dashboard



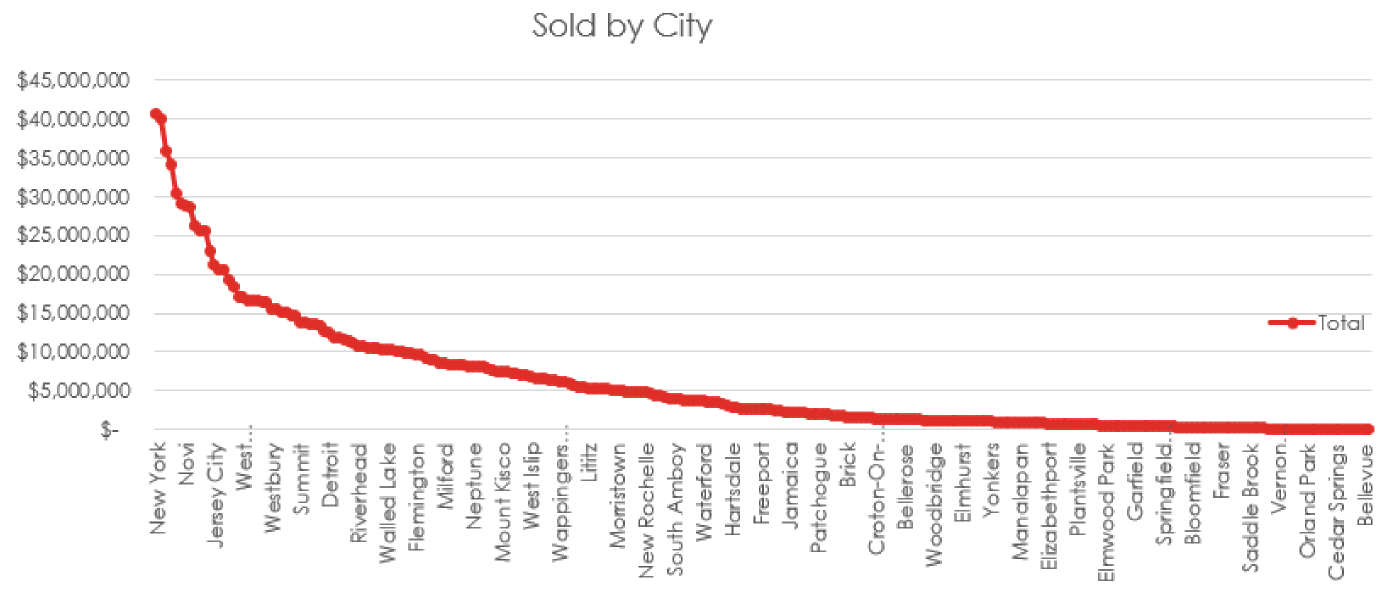
11.2 Profit by is_new & make_name (Brand name)



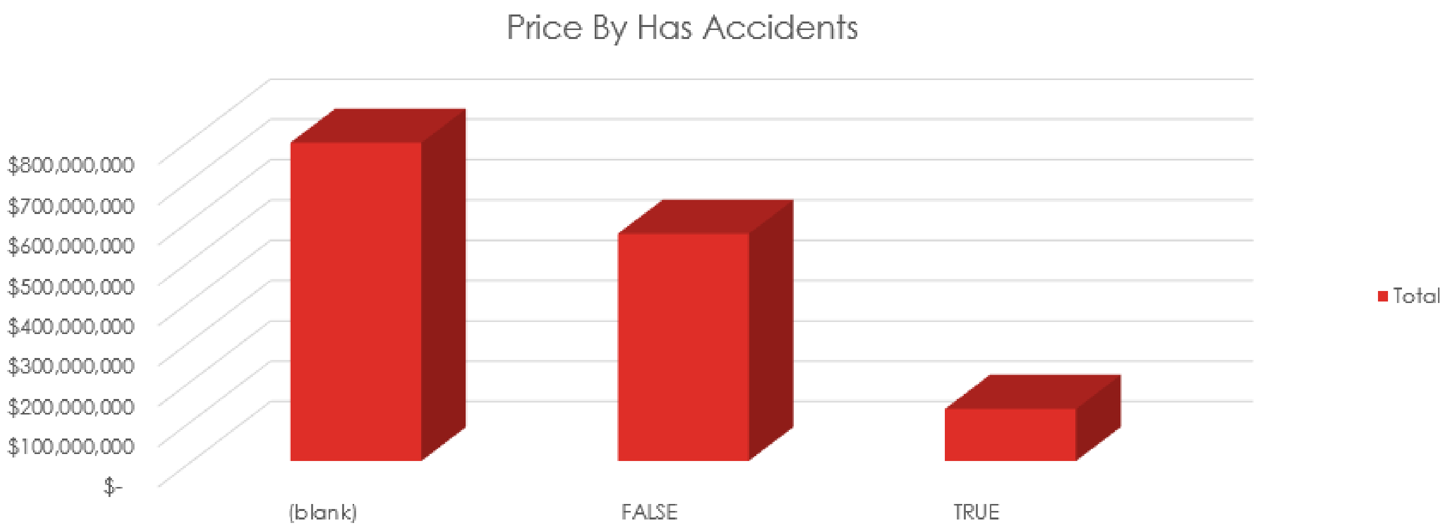
11.3 Sold by engine type.

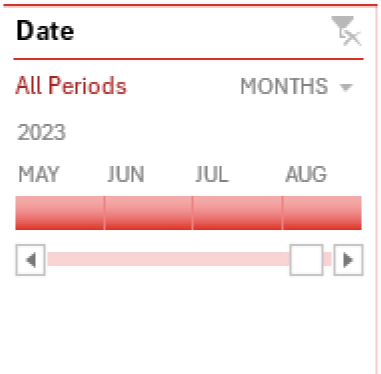


11.4 Sold by city.



11.5 Price by has accident.



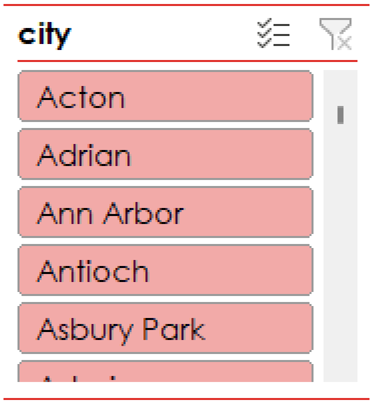


Date

All Periods MONTHS

2023

MAY JUN JUL AUG



city

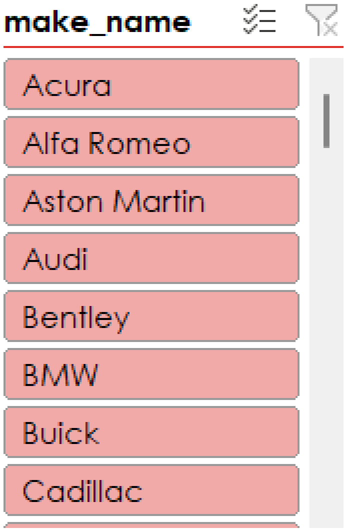
Acton

Adrian

Ann Arbor

Antioch

Asbury Park



make_name

Acura

Alfa Romeo

Aston Martin

Audi

Bentley

BMW

Buick

Cadillac

- These are our timeline and slicers. Using those we can get unique information about our dataset.

12 References:

12.1 Statistics:

Central tendency measures (mean, median, mode)
Dispersion measures (variance, standard deviation)
Statistical hypothesis testing
Correlation and regression analysis
https://www.dummies.com/book/academics-the-arts/math/statistics/statistics-for-dummies-2nd-edition-282603/

12.2 Visualization:

Data visualization best practices
Types of data visualizations (bar charts, line charts, pie charts, scatter plots, etc.)
Visualization tools (e.g., Tableau, Power BI, Python libraries like Matplotlib)
https://www.storytellingwithdata.com/
Time Series Analysis:

12.3 Time series forecasting methods (ARIMA, SARIMA, exponential smoothing)

Trend analysis in time series data
Seasonality in time series data
https://otexts.com/fpp3/
Dashboards:

12.4 Designing effective dashboards

Key performance indicators (KPIs) for dashboards
Interactive dashboards
http://www.perceptualedge.com/files/Dashboard_Design_Course.pdf