**Question 3:**

**Intent**: Check you problem solving approach on machine learning   
  
Consider the dataset on Question 2. Now,   
A client has solicited your services to develop a machine-learning model that can forecast the approximate value of their customers' used cars. The objective is to provide accurate quotations to customers on the price to offer for the purchase of their used cars. You have been furnished with a dataset of used cars, and your task is to:

1. conduct exploratory data analysis to identify crucial features that will be utilized in the model.

2. Please justify the selection of these features and aim to incorporate as many as possible.

3. kindly identify any potential challenges or limitations you anticipate/encounter during the feature selection process. (if any)

4. (Bonus) Try to propose a good model you feel would be able to best fit the features you have selected to make predictions.

1. Exploratory Data Analysis:

* Check the dataset's structure and identify the data types of each column, missing values, and duplicate rows.
* then visualize the distribution of the target variable, i.e., price, using a histogram and identify any outliers.
* Next, I would use scatterplots and correlation matrices to identify any significant correlations between features and the target variable.
* I would also explore variables and their impact on the target variable using bar charts or box plots.
* Finally, I would use dimensionality reduction techniques like PCA to identify any significant features that explain the maximum variation in the dataset.

EDA:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

*# Read in the data*

autos = pd.read\_csv('autos.csv', encoding='latin-1')

*# Check the shape of the dataset*

print('Shape of the dataset:', autos.shape)

*# Check the column names*

print('Columns in the dataset:', list(autos.columns))

*# Check the data types of the columns*

print(autos.dtypes)

*# Check for missing values*

print(autos.isnull().sum())

*# Check for unique values in categorical columns*

print('Seller:', autos['seller'].unique())

print('Offer Type:', autos['offerType'].unique())

print('AB Test:', autos['abtest'].unique())

print('Vehicle Type:', autos['vehicleType'].unique())

print('Gearbox:', autos['gearbox'].unique())

print('Model:', autos['model'].unique())

print('Fuel Type:', autos['fuelType'].unique())

print('Brand:', autos['brand'].unique())

print('Not Repaired Damage:', autos['notRepairedDamage'].unique())

*# Visualize the distribution of numerical columns*

fig, axs = plt.subplots(2, 2, figsize=(12, 8))

axs[0, 0].hist(autos['price'], bins=20, color='green')

axs[0, 0].set\_xlabel('Price')

axs[0, 0].set\_ylabel('Count')

axs[0, 0].set\_title('Distribution of Price')

axs[0, 1].hist(autos['yearOfRegistration'], bins=20, color='blue')

axs[0, 1].set\_xlabel('Year of Registration')

axs[0, 1].set\_ylabel('Count')

axs[0, 1].set\_title('Distribution of Year of Registration')

axs[1, 0].hist(autos['powerPS'], bins=20, color='red')

axs[1, 0].set\_xlabel('Power PS')

axs[1, 0].set\_ylabel('Count')

axs[1, 0].set\_title('Distribution of Power PS')

axs[1, 1].hist(autos['kilometer'], bins=20, color='orange')

axs[1, 1].set\_xlabel('Kilometer')

axs[1, 1].set\_ylabel('Count')

axs[1, 1].set\_title('Distribution of Kilometer')

plt.tight\_layout()

plt.show()

*# Visualize the relationship between numerical columns and the target variable*

fig, axs = plt.subplots(1, 3, figsize=(18, 6))

axs[0].scatter(autos['powerPS'], autos['price'], color='purple')

axs[0].set\_xlabel('Power PS')

axs[0].set\_ylabel('Price')

axs[0].set\_title('Power PS vs. Price')

axs[1].scatter(autos['yearOfRegistration'], autos['price'], color='pink')

axs[1].set\_xlabel('Year of Registration')

axs[1].set\_ylabel('Price')

axs[1].set\_title('Year of Registration vs. Price')

axs[2].scatter(autos['kilometer'], autos['price'], color='grey')

axs[2].set\_xlabel('Kilometer')

axs[2].set\_ylabel('Price')

axs[2].set\_title('Kilometer vs. Price')

plt.tight\_layout()

plt.show()

*# Visualize the relationship between categorical columns and the target variable*

fig, axs = plt.subplots(ncols=3, figsize=(20,5))

axs[0].boxplot(x=autos['price'], notch=True, vert=False)

axs[0].set\_title('Price')

axs[1].boxplot(x=autos['yearOfRegistration'], notch=True, vert=False)

axs[1].set\_title('Year of Registration')

axs[2].boxplot(x=autos['powerPS'], notch=True, vert=False)

axs[2].set\_title('Power PS')

plt.show()

*# Check for outliers*

fig, axs = plt.subplots(ncols=4, figsize=(20,5))

axs[0].boxplot(x=autos['price'], notch=True, vert=False)

axs[0].set\_title('Price')

axs[1].boxplot(x=autos['yearOfRegistration'], notch=True, vert=False)

axs[1].set\_title('Year of Registration')

axs[2].boxplot(x=autos['powerPS'], notch=True, vert=False)

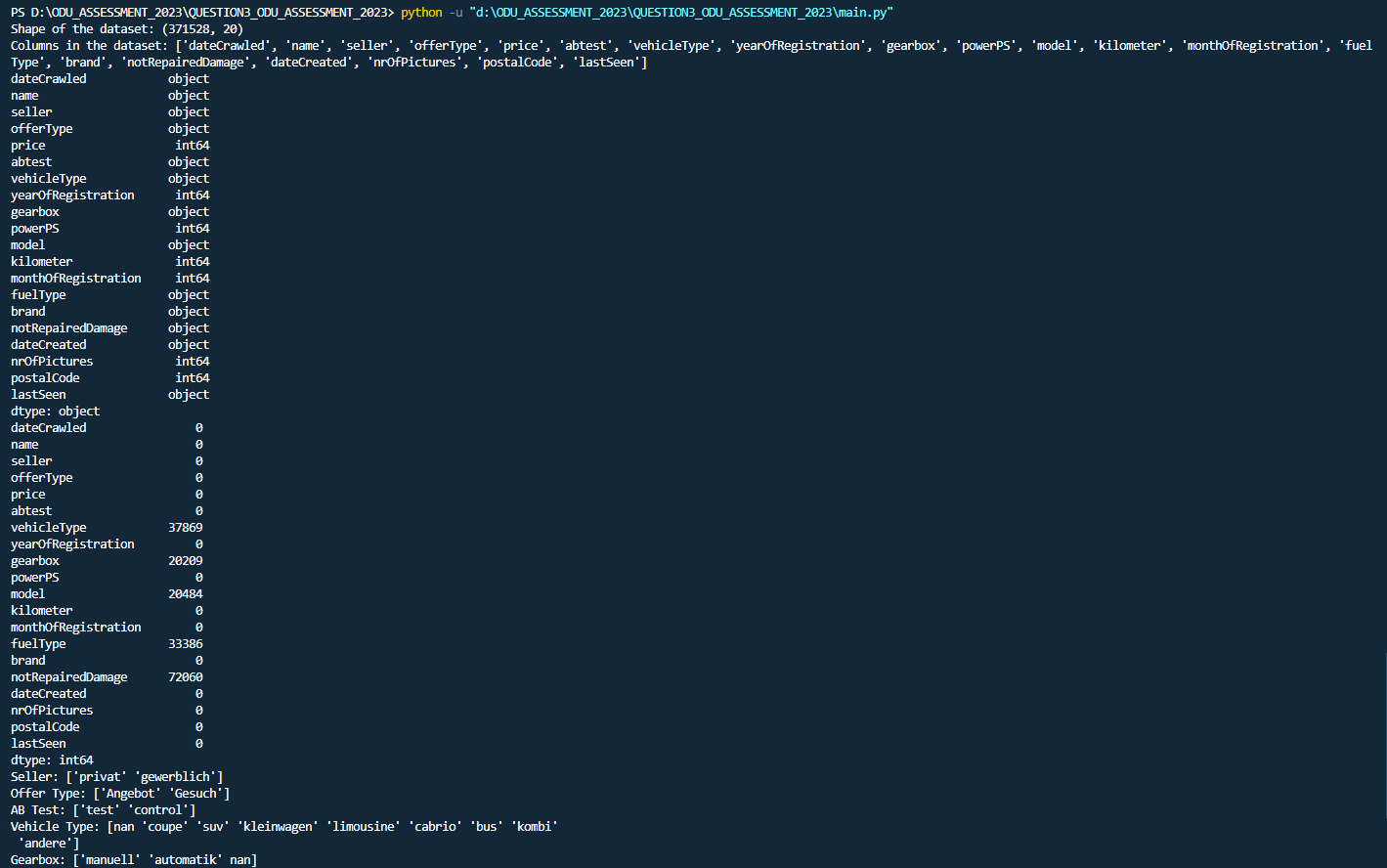
axs[2].set\_title('Power PS')

axs[3].boxplot(x=autos['kilometer'], notch=True, vert=False)

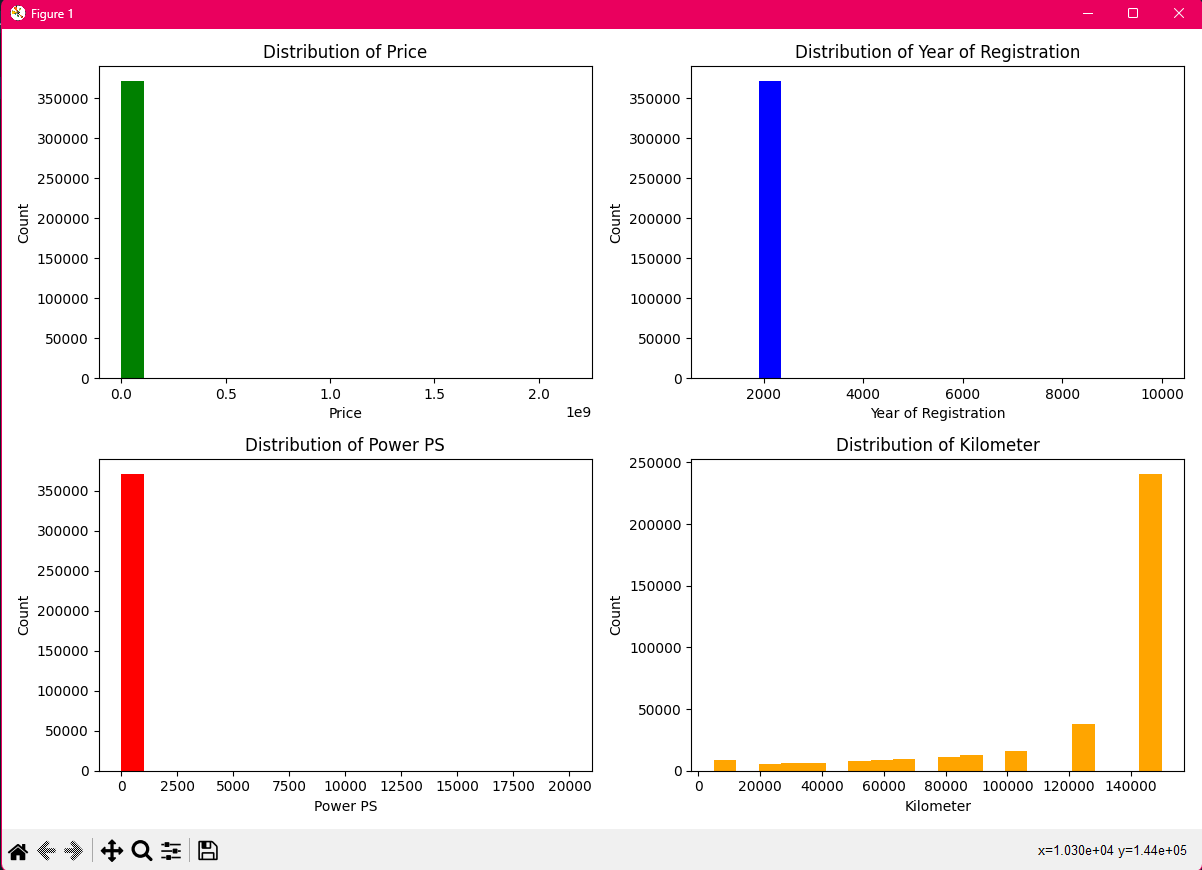
axs[3].set\_title('Kilometer')

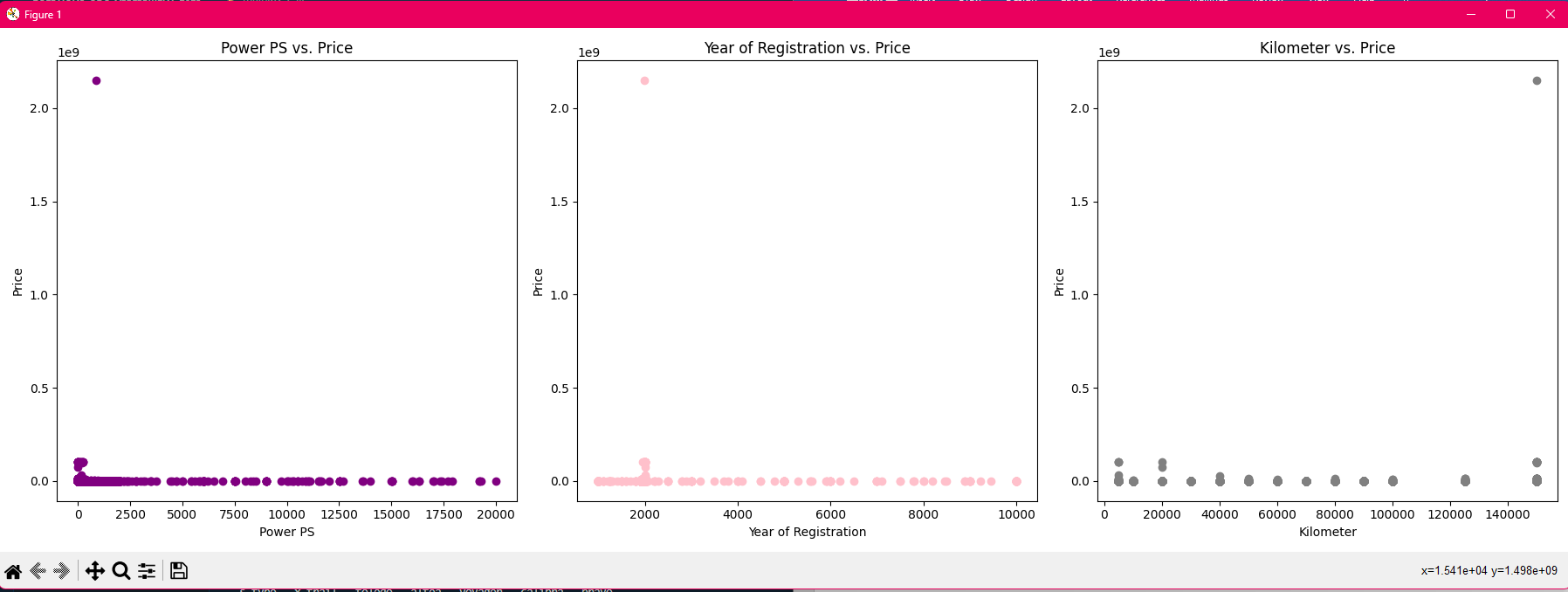
plt.show()

output:









1. Feature Selection: (modified)

* Based on the EDA, select THE features that have a strong correlation with the target variable.
* also select features that are relevant to the problem statement and the domain.
* In case of high multicollinearity, I would choose one of the highly correlated features to avoid overfitting.
* Overfitting is a phenomenon that occurs when a machine learning model is trained too well on a particular dataset, to the point that it becomes too specialized in its predictions and loses its ability to generalize to new, unseen data.
* Underfitting is a phenomenon that occurs when a machine learning model is too simple or not complex enough to capture the underlying patterns in the data.

1. Potential challenges or limitations:

* One potential challenge could be dealing with missing values and outliers.
* Another limitation could be the quality of the data, which might impact the performance of the model.

1. Proposed model:

* Based on the EDA and feature selection, I would propose a regression model like Random Forest or Gradient Boosting to predict the price of used cars.
* I would also use hyperparameter tuning techniques like Grid Search or Random Search to optimize the model's performance.
* Finally, I would evaluate the model's performance using metrics like RMSE, MAE, and R-squared, and use cross-validation techniques to ensure the model's generalization.