supervised-car-pricing-model

November 27, 2023

1 Project Name - Car Pricing Model

Project Type - EDA/Regression

Contribution - Individual

2 Project Summary -

The Chinese automobile company Geely Auto aimed to enter the US market and hired consultants to analyze the American car pricing dynamics through a data-driven regression model in order to optimally design and price their vehicles. Multiple regression techniques were applied, among which the linear regression model with the highest accuracy and generalizability established a quantifiable relationship between car features like horsepower, weight, engine-size and pricing. It enabled Geely to determine target specification ranges based on competitive analysis for given desired price points. The model provides data-backed guidance to strategically optimize their automobile configurations and dynamic pricing strategy in order to successfully penetrate the unfamiliar American car market. With continuous model re-training using latest market data, Geely can account for evolving pricing trends and adapt their pricing accordingly over time. Overall, the fitted linear model supplies actionable insights for data-informed decision making regarding optimal vehicle design and pricing to assist Geely's entry and viability in the US automobile industry.

3 GitHub Link -

https://github.com/shantanu0101/Car-Pricing-Model-Regression-Model-

4 Problem Statement

A Chinese automobile company Geely Auto aspires to enter the US market by setting up their manufacturing unit there and producing cars locally to give competition to their US and European counterparts.

They have contracted an automobile consulting company to understand the factors on which the pricing of cars depends. Specifically, they want to understand the factors affecting the pricing of cars in the American market, since those may be very different from the Chinese market. The company wants to know:

Which variables are significant in predicting the price of a car How well those variables describe the price of a car Based on various market surveys, the consulting firm has gathered a large data set of different types of cars across the America market.

5 General Guidelines : -

1. Well-structured, formatted, and commented code is required.
2. Exception Handling, Production Grade Code & Deployment Ready Code will be a plus. Those students will be awarded some additional credits.

   The additional credits will have advantages over other students during Star Student selection.

   ```
   [ Note: - Deployment Ready Code is defined as, the whole .ipynb notebook should be exe
             without a single error logged. ]
   ```
3. Each and every logic should have proper comments.
4. You may add as many number of charts you want. Make Sure for each and every chart the following format should be answered.

```
# Chart visualization code
```

- Why did you pick the specific chart?
- What is/are the insight(s) found from the chart?
- Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

[Hints : - Do the Vizualization in a structured way while following "UBM" Rule.

U - Univariate Analysis,

B - Bivariate Analysis (Numerical - Categorical, Numerical - Numerical, Categorical - Categorical)

M - Multivariate Analysis]

6. You may add more ml algorithms for model creation. Make sure for each and every algorithm, the following format should be answered.

- Explain the ML Model used and it's performance using Evaluation metric Score Chart.
- Cross- Validation & Hyperparameter Tuning
- Explain each evaluation metric's indication towards business and the business impact pf the ML model used.

6 *Let's Begin !*

6.1 *1. Know Your Data*

6.1.1 Import Libraries

[111]:
```
# Import Libraries
import math  # Import the math module for mathematical operations
import numpy as np  # Import NumPy for numerical operations
import pandas as pd  # Import Pandas for data manipulation
import seaborn as sns  # Import Seaborn for statistical data visualization
```

```
import matplotlib.pyplot as plt  # Import Matplotlib for plotting

from sklearn.preprocessing import StandardScaler  # Import StandardScaler for␣
 ↪standardization of features
from sklearn.preprocessing import MinMaxScaler  # Import MinMaxScaler for␣
 ↪scaling features to a range
from sklearn.metrics import mean_squared_error  # Import mean_squared_error for␣
 ↪calculating Mean Squared Error
from sklearn.metrics import mean_absolute_error  # Import mean_absolute_error␣
 ↪for calculating Mean Absolute Error
from sklearn.metrics import mean_absolute_percentage_error  # Import␣
 ↪mean_absolute_percentage_error for calculating MAPE
from sklearn.metrics import r2_score  # Import r2_score for calculating␣
 ↪R-squared

from scipy.stats import pointbiserialr  # Import pointbiserialr for␣
 ↪point-biserial correlation coefficient
from sklearn.model_selection import train_test_split  # Import train_test_split␣
 ↪for splitting data into train and test sets
from sklearn.model_selection import cross_val_score, cross_val_predict  #␣
 ↪Import cross-validation functions
from sklearn.model_selection import GridSearchCV  # Import GridSearchCV for␣
 ↪hyperparameter tuning
from sklearn.linear_model import LinearRegression  # Import LinearRegression␣
 ↪model
from sklearn.linear_model import Ridge  # Import Ridge Regression model
from sklearn.linear_model import Lasso  # Import Lasso Regression model
from sklearn.linear_model import ElasticNet  # Import ElasticNet model
```

6.1.2 Dataset Loading

[112]:
```
# Load Dataset
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[113]:
```
path = '/content/drive/MyDrive/CarPrice_project.csv'
cars_df = pd.read_csv(path)
```

6.1.3 Dataset First View

[114]:
```
# Dataset First Look
cars_df
```

[114]:
```
     car_ID  symboling                   CarName fueltype aspiration  \
0         1          3        alfa-romero giulia      gas        std
1         2          3       alfa-romero stelvio      gas        std
2         3          1  alfa-romero Quadrifoglio      gas        std
3         4          2               audi 100 ls      gas        std
4         5          2                audi 100ls      gas        std
..      …        …                       …      …        …
200     201         -1           volvo 145e (sw)      gas        std
201     202         -1               volvo 144ea      gas      turbo
202     203         -1               volvo 244dl      gas        std
203     204         -1                 volvo 246   diesel      turbo
204     205         -1               volvo 264gl      gas      turbo

    doornumber      carbody drivewheel enginelocation  wheelbase  …  \
0          two  convertible        rwd          front       88.6  …
1          two  convertible        rwd          front       88.6  …
2          two    hatchback        rwd          front       94.5  …
3         four        sedan        fwd          front       99.8  …
4         four        sedan        4wd          front       99.4  …
..         …          …        …            …        …  …
200       four        sedan        rwd          front      109.1  …
201       four        sedan        rwd          front      109.1  …
202       four        sedan        rwd          front      109.1  …
203       four        sedan        rwd          front      109.1  …
204       four        sedan        rwd          front      109.1  …

     enginesize  fuelsystem  boreratio  stroke compressionratio horsepower  \
0           130        mpfi       3.47    2.68              9.0        111
1           130        mpfi       3.47    2.68              9.0        111
2           152        mpfi       2.68    3.47              9.0        154
3           109        mpfi       3.19    3.40             10.0        102
4           136        mpfi       3.19    3.40              8.0        115
..          …         …        …     …               …        …
200         141        mpfi       3.78    3.15              9.5        114
201         141        mpfi       3.78    3.15              8.7        160
202         173        mpfi       3.58    2.87              8.8        134
203         145         idi       3.01    3.40             23.0        106
204         141        mpfi       3.78    3.15              9.5        114

     peakrpm citympg  highwaympg    price
0       5000      21          27  13495.0
1       5000      21          27  16500.0
```

```
2        5000      19           26  16500.0
3        5500      24           30  13950.0
4        5500      18           22  17450.0
..        …      …          …      …
200      5400      23           28  16845.0
201      5300      19           25  19045.0
202      5500      18           23  21485.0
203      4800      26           27  22470.0
204      5400      19           25  22625.0

[205 rows x 26 columns]
```

6.1.4 Dataset Rows & Columns count

[115]:
```python
# Dataset Rows & Columns count
cars_df.shape
```

[115]:
```
(205, 26)
```

6.1.5 Dataset Information

[116]:
```python
# Dataset Info
cars_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
 14  enginetype        205 non-null    object
 15  cylindernumber    205 non-null    object
 16  enginesize        205 non-null    int64
 17  fuelsystem        205 non-null    object
```

```
 18   boreratio         205 non-null    float64
 19   stroke            205 non-null    float64
 20   compressionratio  205 non-null    float64
 21   horsepower        205 non-null    int64
 22   peakrpm           205 non-null    int64
 23   citympg           205 non-null    int64
 24   highwaympg        205 non-null    int64
 25   price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

Duplicate Values

[117]:
```python
# Dataset Duplicate Value Count
cars_df.duplicated().sum()
```

[117]:
```
0
```

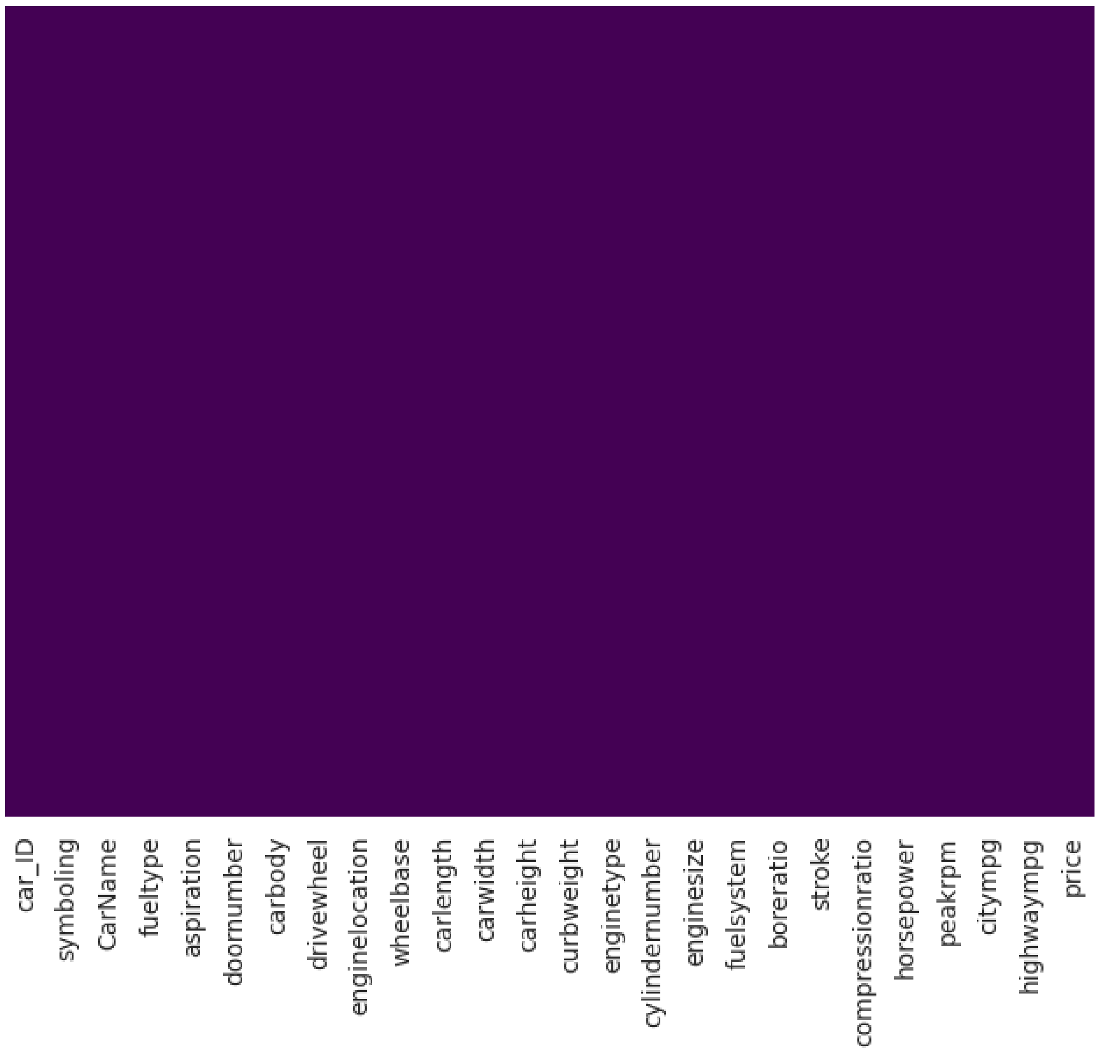
Missing Values/Null Values

[118]:
```python
# Missing Values/Null Values Count
cars_df.isnull().sum()
```

[118]:
```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
```

```
price                0
dtype: int64
```

[119]:
```
# Visualizing the missing values
plt.figure(figsize = (8,6), dpi = 100)
sns.heatmap(cars_df.isnull(), cmap = 'viridis', cbar = False, yticklabels =␣
 ↪False)
plt.show()
```



6.1.6 What did you know about your dataset?

The dataset comprises information on various attributes related to automobiles, encompassing both categorical and numerical features. Each entry in the dataset is associated with a unique car identification (car_ID). The features include assessments of risk (symboling), car name (CarName), fuel type (fueltype), aspiration type (aspiration), the number of doors (doornumber), car body style (carbody), drivetrain type (drivewheel), engine location (enginelocation), and dimensions such as wheelbase, car length, width, and height. Other essential characteristics encompass curb weight, engine type, cylinder count, engine size, fuel injection system type, bore ratio, stroke, compression ratio, horsepower, peak revolutions per minute, and fuel efficiency measured in miles per gallon for both city and highway driving. The dataset culminates in the target variable, 'Price,' representing the car's cost. With these diverse features, the dataset is well-structured for predictive modeling, aiming to establish relationships between the car attributes and their corresponding prices.

6.2 *2. Understanding Your Variables*

[120]:
```
# Dataset Columns
column_list = list(cars_df.columns)
column_list
```

[120]:
```
['car_ID',
 'symboling',
 'CarName',
 'fueltype',
 'aspiration',
 'doornumber',
 'carbody',
 'drivewheel',
 'enginelocation',
 'wheelbase',
 'carlength',
 'carwidth',
 'carheight',
 'curbweight',
 'enginetype',
 'cylindernumber',
 'enginesize',
 'fuelsystem',
 'boreratio',
 'stroke',
 'compressionratio',
 'horsepower',
 'peakrpm',
 'citympg',
 'highwaympg',
 'price']
```

[121]:
```
# Dataset Describe
cars_df.describe()
```

[121]:
```
           car_ID   symboling   wheelbase   carlength    carwidth   carheight  \
count  205.000000  205.000000  205.000000  205.000000  205.000000  205.000000
```

```
mean    103.000000    0.834146   98.756585  174.049268   65.907805   53.724878
std      59.322565    1.245307    6.021776   12.337289    2.145204    2.443522
min       1.000000   -2.000000   86.600000  141.100000   60.300000   47.800000
25%      52.000000    0.000000   94.500000  166.300000   64.100000   52.000000
50%     103.000000    1.000000   97.000000  173.200000   65.500000   54.100000
75%     154.000000    2.000000  102.400000  183.100000   66.900000   55.500000
max     205.000000    3.000000  120.900000  208.100000   72.300000   59.800000

        curbweight  enginesize   boreratio      stroke  compressionratio  \
count   205.000000  205.000000  205.000000  205.000000        205.000000
mean   2555.565854  126.907317    3.329756    3.255415         10.142537
std     520.680204   41.642693    0.270844    0.313597          3.972040
min    1488.000000   61.000000    2.540000    2.070000          7.000000
25%    2145.000000   97.000000    3.150000    3.110000          8.600000
50%    2414.000000  120.000000    3.310000    3.290000          9.000000
75%    2935.000000  141.000000    3.580000    3.410000          9.400000
max    4066.000000  326.000000    3.940000    4.170000         23.000000

       horsepower      peakrpm     citympg  highwaympg         price
count  205.000000   205.000000  205.000000  205.000000    205.000000
mean   104.117073  5125.121951   25.219512   30.751220  13276.710571
std     39.544167   476.985643    6.542142    6.886443   7988.852332
min     48.000000  4150.000000   13.000000   16.000000   5118.000000
25%     70.000000  4800.000000   19.000000   25.000000   7788.000000
50%     95.000000  5200.000000   24.000000   30.000000  10295.000000
75%    116.000000  5500.000000   30.000000   34.000000  16503.000000
max    288.000000  6600.000000   49.000000   54.000000  45400.000000
```

6.2.1 Variables Description

1. **car_ID:** Unique identifier for each car.
2. **symboling:** Risk rating associated with the car.
3. **CarName:** Name of the car.
4. **fueltype:** Type of fuel the car uses (e.g., gas or diesel).
5. **aspiration:** Type of aspiration (e.g., std or turbo).
6. **doornumber:** Number of doors on the car.
7. **carbody:** Body style of the car.
8. **drivewheel:** Type of drivetrain (e.g., front-wheel-drive, rear-wheel-drive, 9.or four-wheel-drive).
9. **enginelocation:** Location of the car engine (front or rear).
10. **wheelbase:** Distance between the centers of the front and rear wheels.
11. **carlength:** Length of the car.

12. **carwidth:** Width of the car.
13. **carheight:** Height of the car.
14. curbweight: Weight of the car without occupants or baggage.
15. **enginetype:** Type of engine.
16. **cylindernumber:** Number of cylinders in the engine.
17. **enginesize:** Size of the car's engine.
18. **fuelsystem:** Type of fuel injection system.
19. **boreratio:** Bore ratio of the engine.
20. **stroke:** Stroke or volume inside the engine.
21. **compressionratio:** Compression ratio of the engine.
22. **horsepower:** Horsepower of the car.
23. **peakrpm:** Peak revolutions per minute.
24. **citympg:** Miles per gallon in the city.
25. **highwaympg:** Miles per gallon on the highway.
26. **price:** Price of the car.

6.2.2 Check Unique Values for each variable.

[122]:
```python
# Check Unique Values for each variable.
for column in cars_df.columns:
    unique_values = cars_df[column].unique()
    print(f"Unique values in {column}:\n{unique_values}\n")
```

```
Unique values in car_ID:
[  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
 199 200 201 202 203 204 205]

Unique values in symboling:
[ 3  1  2  0 -1 -2]

Unique values in CarName:
```

```
['alfa-romero giulia' 'alfa-romero stelvio' 'alfa-romero Quadrifoglio'
 'audi 100 ls' 'audi 100ls' 'audi fox' 'audi 5000' 'audi 4000'
 'audi 5000s (diesel)' 'bmw 320i' 'bmw x1' 'bmw x3' 'bmw z4' 'bmw x4'
 'bmw x5' 'chevrolet impala' 'chevrolet monte carlo' 'chevrolet vega 2300'
 'dodge rampage' 'dodge challenger se' 'dodge d200' 'dodge monaco (sw)'
 'dodge colt hardtop' 'dodge colt (sw)' 'dodge coronet custom'
 'dodge dart custom' 'dodge coronet custom (sw)' 'honda civic'
 'honda civic cvcc' 'honda accord cvcc' 'honda accord lx'
 'honda civic 1500 gl' 'honda accord' 'honda civic 1300' 'honda prelude'
 'honda civic (auto)' 'isuzu MU-X' 'isuzu D-Max ' 'isuzu D-Max V-Cross'
 'jaguar xj' 'jaguar xf' 'jaguar xk' 'maxda rx3' 'maxda glc deluxe'
 'mazda rx2 coupe' 'mazda rx-4' 'mazda glc deluxe' 'mazda 626' 'mazda glc'
 'mazda rx-7 gs' 'mazda glc 4' 'mazda glc custom l' 'mazda glc custom'
 'buick electra 225 custom' 'buick century luxus (sw)' 'buick century'
 'buick skyhawk' 'buick opel isuzu deluxe' 'buick skylark'
 'buick century special' 'buick regal sport coupe (turbo)'
 'mercury cougar' 'mitsubishi mirage' 'mitsubishi lancer'
 'mitsubishi outlander' 'mitsubishi g4' 'mitsubishi mirage g4'
 'mitsubishi montero' 'mitsubishi pajero' 'Nissan versa' 'nissan gt-r'
 'nissan rogue' 'nissan latio' 'nissan titan' 'nissan leaf' 'nissan juke'
 'nissan note' 'nissan clipper' 'nissan nv200' 'nissan dayz' 'nissan fuga'
 'nissan otti' 'nissan teana' 'nissan kicks' 'peugeot 504' 'peugeot 304'
 'peugeot 504 (sw)' 'peugeot 604sl' 'peugeot 505s turbo diesel'
 'plymouth fury iii' 'plymouth cricket' 'plymouth satellite custom (sw)'
 'plymouth fury gran sedan' 'plymouth valiant' 'plymouth duster'
 'porsche macan' 'porcshce panamera' 'porsche cayenne' 'porsche boxter'
 'renault 12tl' 'renault 5 gtl' 'saab 99e' 'saab 99le' 'saab 99gle'
 'subaru' 'subaru dl' 'subaru brz' 'subaru baja' 'subaru r1' 'subaru r2'
 'subaru trezia' 'subaru tribeca' 'toyota corona mark ii' 'toyota corona'
 'toyota corolla 1200' 'toyota corona hardtop' 'toyota corolla 1600 (sw)'
 'toyota carina' 'toyota mark ii' 'toyota corolla'
 'toyota corolla liftback' 'toyota celica gt liftback'
 'toyota corolla tercel' 'toyota corona liftback' 'toyota starlet'
 'toyota tercel' 'toyota cressida' 'toyota celica gt' 'toyouta tercel'
 'vokswagen rabbit' 'volkswagen 1131 deluxe sedan' 'volkswagen model 111'
 'volkswagen type 3' 'volkswagen 411 (sw)' 'volkswagen super beetle'
 'volkswagen dasher' 'vw dasher' 'vw rabbit' 'volkswagen rabbit'
 'volkswagen rabbit custom' 'volvo 145e (sw)' 'volvo 144ea' 'volvo 244dl'
 'volvo 245' 'volvo 264gl' 'volvo diesel' 'volvo 246']

Unique values in fueltype:
['gas' 'diesel']

Unique values in aspiration:
['std' 'turbo']

Unique values in doornumber:
['two' 'four']
```

```
Unique values in carbody:
['convertible' 'hatchback' 'sedan' 'wagon' 'hardtop']

Unique values in drivewheel:
['rwd' 'fwd' '4wd']

Unique values in enginelocation:
['front' 'rear']

Unique values in wheelbase:
[ 88.6  94.5  99.8  99.4 105.8  99.5 101.2 103.5 110.   88.4  93.7 103.3
  95.9  86.6  96.5  94.3  96.  113.  102.   93.1  95.3  98.8 104.9 106.7
 115.6  96.6 120.9 112.  102.7  93.   96.3  95.1  97.2 100.4  91.3  99.2
 107.9 114.2 108.   89.5  98.4  96.1  99.1  93.3  97.   96.9  95.7 102.4
 102.9 104.5  97.3 104.3 109.1]

Unique values in carlength:
[168.8 171.2 176.6 177.3 192.7 178.2 176.8 189.  193.8 197.  141.1 155.9
 158.8 157.3 174.6 173.2 144.6 150.  163.4 157.1 167.5 175.4 169.1 170.7
 172.6 199.6 191.7 159.1 166.8 169.  177.8 175.  190.9 187.5 202.6 180.3
 208.1 199.2 178.4 173.  172.4 165.3 170.2 165.6 162.4 173.4 181.7 184.6
 178.5 186.7 198.9 167.3 168.9 175.7 181.5 186.6 156.9 157.9 172.  173.5
 173.6 158.7 169.7 166.3 168.7 176.2 175.6 183.5 187.8 171.7 159.3 165.7
 180.2 183.1 188.8]

Unique values in carwidth:
[64.1 65.5 66.2 66.4 66.3 71.4 67.9 64.8 66.9 70.9 60.3 63.6 63.8 64.6
 63.9 64.  65.2 62.5 66.  61.8 69.6 70.6 64.2 65.7 66.5 66.1 70.3 71.7
 70.5 72.  68.  64.4 65.4 68.4 68.3 65.  72.3 66.6 63.4 65.6 67.7 67.2
 68.9 68.8]

Unique values in carheight:
[48.8 52.4 54.3 53.1 55.7 55.9 52.  53.7 56.3 53.2 50.8 50.6 59.8 50.2
 52.6 54.5 58.3 53.3 54.1 51.  53.5 51.4 52.8 47.8 49.6 55.5 54.4 56.5
 58.7 54.9 56.7 55.4 54.8 49.4 51.6 54.7 55.1 56.1 49.7 56.  50.5 55.2
 52.5 53.  59.1 53.9 55.6 56.2 57.5]

Unique values in curbweight:
[2548 2823 2337 2824 2507 2844 2954 3086 3053 2395 2710 2765 3055 3230
 3380 3505 1488 1874 1909 1876 2128 1967 1989 2191 2535 2811 1713 1819
 1837 1940 1956 2010 2024 2236 2289 2304 2372 2465 2293 2734 4066 3950
 1890 1900 1905 1945 1950 2380 2385 2500 2410 2443 2425 2670 2700 3515
 3750 3495 3770 3740 3685 3900 3715 2910 1918 1944 2004 2145 2370 2328
 2833 2921 2926 2365 2405 2403 1889 2017 1938 1951 2028 1971 2037 2008
 2324 2302 3095 3296 3060 3071 3139 3020 3197 3430 3075 3252 3285 3485
 3130 2818 2778 2756 2800 3366 2579 2460 2658 2695 2707 2758 2808 2847
 2050 2120 2240 2190 2340 2510 2290 2455 2420 2650 1985 2040 2015 2280
```

```
 3110 2081 2109 2275 2094 2122 2140 2169 2204 2265 2300 2540 2536 2551
 2679 2714 2975 2326 2480 2414 2458 2976 3016 3131 3151 2261 2209 2264
 2212 2319 2254 2221 2661 2563 2912 3034 2935 3042 3045 3157 2952 3049
 3012 3217 3062]

Unique values in enginetype:
['dohc' 'ohcv' 'ohc' 'l' 'rotor' 'ohcf' 'dohcv']

Unique values in cylindernumber:
['four' 'six' 'five' 'three' 'twelve' 'two' 'eight']

Unique values in enginesize:
[130 152 109 136 131 108 164 209  61  90  98 122 156  92  79 110 111 119
 258 326  91  70  80 140 134 183 234 308 304  97 103 120 181 151 194 203
 132 121 146 171 161 141 173 145]

Unique values in fuelsystem:
['mpfi' '2bbl' 'mfi' '1bbl' 'spfi' '4bbl' 'idi' 'spdi']

Unique values in boreratio:
[3.47 2.68 3.19 3.13 3.5  3.31 3.62 2.91 3.03 2.97 3.34 3.6  2.92 3.15
 3.43 3.63 3.54 3.08 3.33 3.39 3.76 3.58 3.46 3.8  3.78 3.17 3.35 3.59
 2.99 3.7  3.61 3.94 3.74 2.54 3.05 3.27 3.24 3.01]

Unique values in stroke:
[2.68  3.47  3.4   2.8   3.19  3.39  3.03  3.11  3.23  3.46  3.9   3.41
 3.07  3.58  4.17  2.76  3.15  3.255 3.16  3.64  3.1   3.35  3.12  3.86
 3.29  3.27  3.52  2.19  3.21  2.9   2.07  2.36  2.64  3.08  3.5   3.54
 2.87 ]

Unique values in compressionratio:
[ 9.   10.    8.    8.5   8.3   7.    8.8   9.5   9.6   9.41  9.4   7.6
  9.2  10.1   9.1   8.1  11.5   8.6  22.7  22.   21.5   7.5  21.9   7.8
  8.4  21.    8.7   9.31  9.3   7.7  22.5  23.  ]

Unique values in horsepower:
[111 154 102 115 110 140 160 101 121 182  48  70  68  88 145  58  76  60
  86 100  78  90 176 262 135  84  64 120  72 123 155 184 175 116  69  55
  97 152 200  95 142 143 207 288  73  82  94  62  56 112  92 161 156  52
  85 114 162 134 106]

Unique values in peakrpm:
[5000 5500 5800 4250 5400 5100 4800 6000 4750 4650 4200 4350 4500 5200
 4150 5600 5900 5750 5250 4900 4400 6600 5300]

Unique values in citympg:
[21 19 24 18 17 16 23 20 15 47 38 37 31 49 30 27 25 13 26 36 22 14 45 28
 32 35 34 29 33]
```

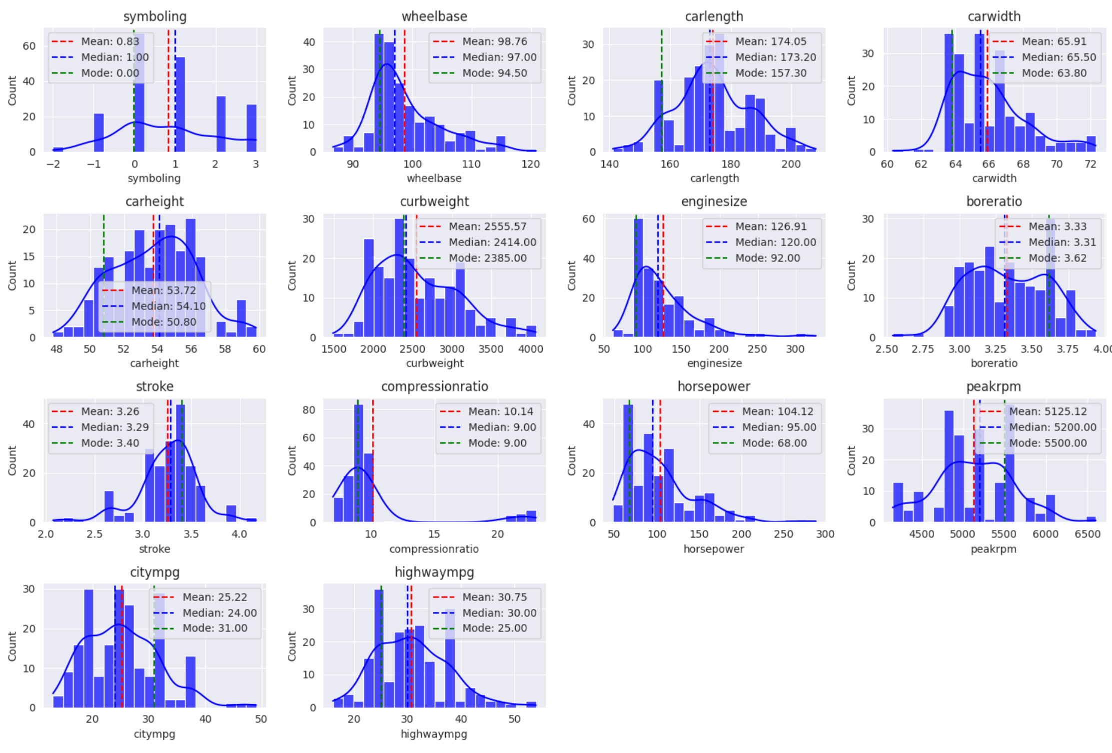
```
Unique values in highwaympg:
[27 26 30 22 25 20 29 28 53 43 41 38 24 54 42 34 33 31 19 17 23 32 39 18
 16 37 50 36 47 46]

Unique values in price:
[13495.     16500.     13950.     17450.     15250.     17710.     18920.
 23875.     17859.167  16430.     16925.     20970.     21105.     24565.
 30760.     41315.     36880.      5151.      6295.      6575.      5572.
  6377.      7957.      6229.      6692.      7609.      8558.      8921.
 12964.      6479.      6855.      5399.      6529.      7129.      7295.
  7895.      9095.      8845.     10295.     12945.     10345.      6785.
  8916.5    11048.     32250.     35550.     36000.      5195.      6095.
  6795.      6695.      7395.     10945.     11845.     13645.     15645.
  8495.     10595.     10245.     10795.     11245.     18280.     18344.
 25552.     28248.     28176.     31600.     34184.     35056.     40960.
 45400.     16503.      5389.      6189.      6669.      7689.      9959.
  8499.     12629.     14869.     14489.      6989.      8189.      9279.
  5499.      7099.      6649.      6849.      7349.      7299.      7799.
  7499.      7999.      8249.      8949.      9549.     13499.     14399.
 17199.     19699.     18399.     11900.     13200.     12440.     13860.
 15580.     16900.     16695.     17075.     16630.     17950.     18150.
 12764.     22018.     32528.     34028.     37028.     31400.5     9295.
  9895.     11850.     12170.     15040.     15510.     18620.      5118.
  7053.      7603.      7126.      7775.      9960.      9233.     11259.
  7463.     10198.      8013.     11694.      5348.      6338.      6488.
  6918.      7898.      8778.      6938.      7198.      7788.      7738.
  8358.      9258.      8058.      8238.      9298.      9538.      8449.
  9639.      9989.     11199.     11549.     17669.      8948.     10698.
  9988.     10898.     11248.     16558.     15998.     15690.     15750.
  7975.      7995.      8195.      9495.      9995.     11595.      9980.
 13295.     13845.     12290.     12940.     13415.     15985.     16515.
 18420.     18950.     16845.     19045.     21485.     22470.     22625.   ]
```

6.3 *3. Data Vizualization, Storytelling & Experimenting with charts : Understand the relationships between variables*

Chart - 1 Histogram (Checking Distribution of dataset Independent Variables)

[123]:
```python
numeric_columns = cars_df.select_dtypes(include=['number'])

# Drop 'car_ID' and 'symboling' columns
columns_to_plot = numeric_columns.drop(['car_ID', 'price'], axis=1)

# Create a histogram with KDE and lines for mean, median, and mode
plt.figure(figsize=(15, 10))
```

```
for i, column in enumerate(columns_to_plot.columns):
    plt.subplot(4, 4, i+1)
    sns.histplot(cars_df[column], kde=True, bins=20, alpha=0.7, color = 'blue')

    mean_val = cars_df[column].mean()
    median_val = cars_df[column].median()
    mode_val = cars_df[column].mode()[0]

    plt.axvline(mean_val, color='red', linestyle='--', label=f'Mean: {mean_val:.2f}')
    plt.axvline(median_val, color='blue', linestyle='--', label=f'Median: {median_val:.2f}')
    plt.axvline(mode_val, color='green', linestyle='--', label=f'Mode: {mode_val:.2f}')

    plt.title(column)
    plt.legend()

plt.tight_layout()
plt.show()
```



[124]:
```
cars_df.skew()
```

```
<ipython-input-124-6bac9c360b6d>:1: FutureWarning: The default value of
numeric_only in DataFrame.skew is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  cars_df.skew()
```

[124]:
```
car_ID              0.000000
symboling           0.211072
wheelbase           1.050214
carlength           0.155954
carwidth            0.904003
carheight           0.063123
curbweight          0.681398
enginesize          1.947655
boreratio           0.020156
stroke             -0.689705
compressionratio    2.610862
horsepower          1.405310
peakrpm             0.075159
citympg             0.663704
highwaympg          0.539997
price               1.777678
dtype: float64
```

1. Why did you pick the specific chart? This specific chart was picked up to check the if the numerical columns which are to be involved in the building the **Regression** model follows **Gaussian Distribution** or not.

In general, in a particualr dataset

Mean < **Median** < **Mode** - if the dataset follows **Left Skewed Distribution**

Mean > **Median** < **Mode** - if the dataset follows **Right Skewed Distribution**

Mean = **Median** = **Mode** - if the dataset follows **Symmetric Distribution**

2. What is/are the insight(s) found from the chart? The dataset exhibits a mix of symmetric and skewed distributions:

1. **Symmetric Distributions**: car_ID, carlength, carheight, boreratio, peakrpm, and stroke have skewness values close to zero, indicating relatively symmetric distributions.
2. **Right-Skewed Distributions**: Variables such as wheelbase, carwidth, curbweight, enginesize, compressionratio, horsepower, citympg, highwaympg, and price display right-skewed distributions, suggesting longer tails on the right.
3. **Left-Skewed Distribution**: The variable stroke has a negative skewness value, indicating a left-skewed distribution with a longer left tail.

The above skeweness in data needs to be treated because for specific machine learning algorithms such as linear regression works well if the data is symmetric i.e normally distributed. Hence for better model performance skewness should be reduced and we will see that in further stage of the project.

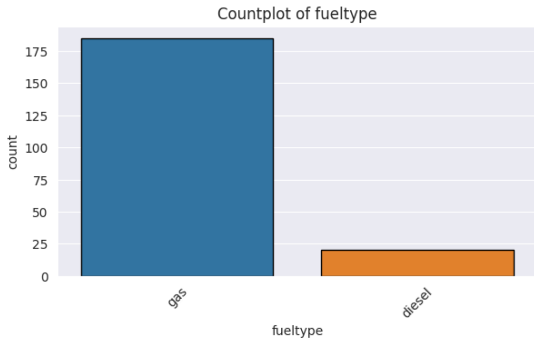
[124]:

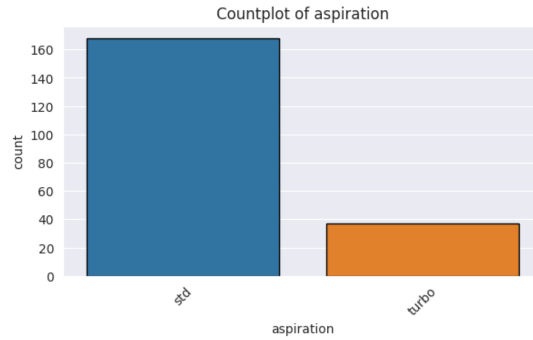
```

```

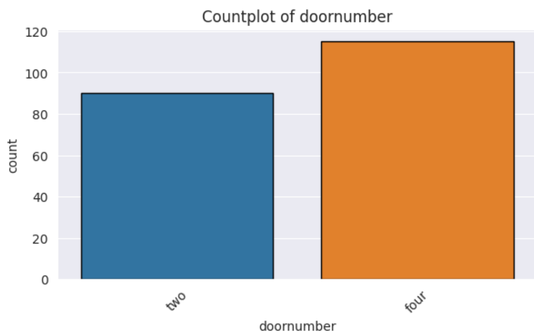
Chart - 2 Countplot (Checking Distribution of dataset Independent Variables)

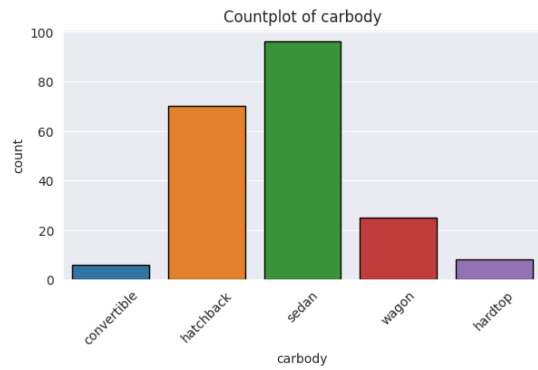
[125]:

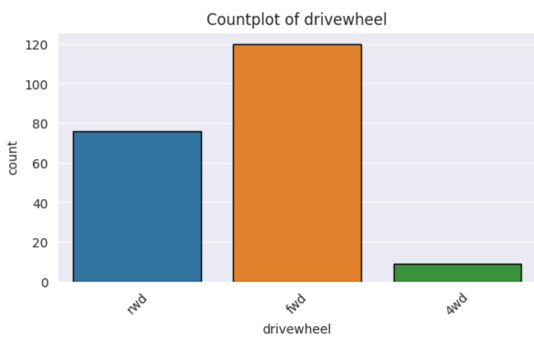
```
# Exclude 'CarName' from categorical variables
cat_vars = [var for var in cars_df.select_dtypes(include='object').columns if␣
 ↪var != 'CarName']

# Determine the number of rows and columns for subplots
num_rows = (len(cat_vars) + 1) // 2  # Ensuring an extra row if the number of␣
 ↪variables is odd
num_cols = 2  # Adjust this based on your preference

# Set up subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(15, 5 * num_rows))
fig.subplots_adjust(hspace=0.5)  # Adjust vertical spacing

# Flatten the axes array to simplify indexing
axes = axes.flatten()

# Loop through each categorical variable (excluding 'CarName') and create a␣
 ↪countplot
for i, var in enumerate(cat_vars):
    sns.countplot(x=var, data=cars_df, ax=axes[i], edgecolor='black')
    axes[i].set_title(f'Countplot of {var}')
    axes[i].tick_params(axis='x', rotation=45)  # Rotate x-axis labels for␣
 ↪better readability

# If the number of subplots is odd, remove the empty subplot
if len(cat_vars) % 2 != 0:
    fig.delaxes(axes[-1])

# Display the subplots
plt.show()
```

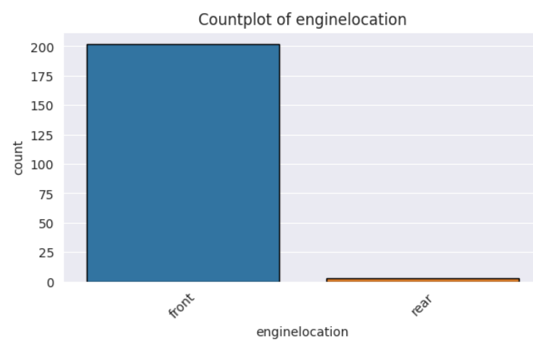

Countplot of fueltype
count
0
25
50
75
100
125
150
175
gas
diesel
fueltype

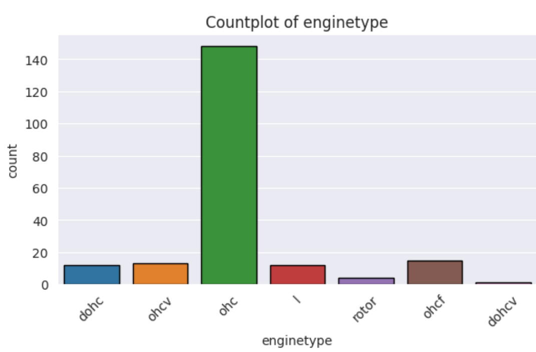

Countplot of aspiration
count
0
20
40
60
80
100
120
140
160
std
turbo
aspiration

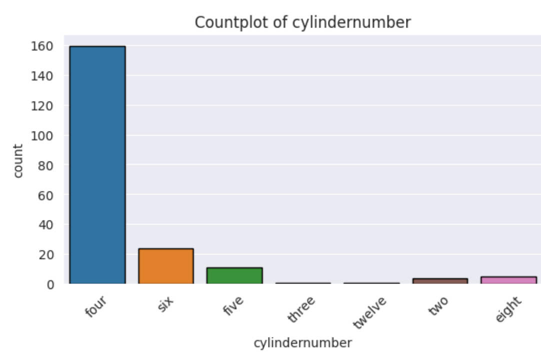

Countplot of doornumber
count
0
20
40
60
80
100
120
two
four
doornumber

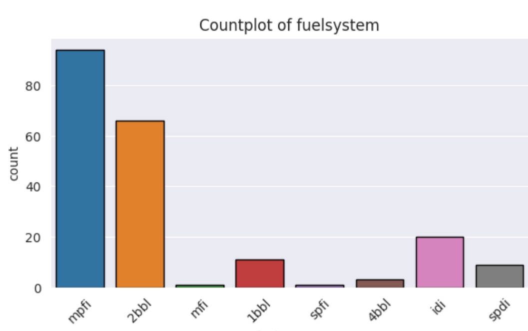

Countplot of carbody
count
0
20
40
60
80
100
convertible
hatchback
sedan
wagon
hardtop
carbody


Countplot of drivewheel
count
0
20
40
60
80
100
120
rwd
fwd
4wd
drivewheel


Countplot of enginelocation
count
0
25
50
75
100
125
150
175
200
front
rear
enginelocation


Countplot of enginetype
count
0
20
40
60
80
100
120
140
dohc
ohcv
ohc
l
rotor
ohcf
dohcv
enginetype


Countplot of cylindernumber
count
0
20
40
60
80
100
120
140
160
four
six
five
three
twelve
two
eight
cylindernumber


Countplot of fuelsystem
count
0
20
40
60
80
mpfi
2bbl
mfi
1bbl
spfi
4bbl
idi
spdi
fuelsystem

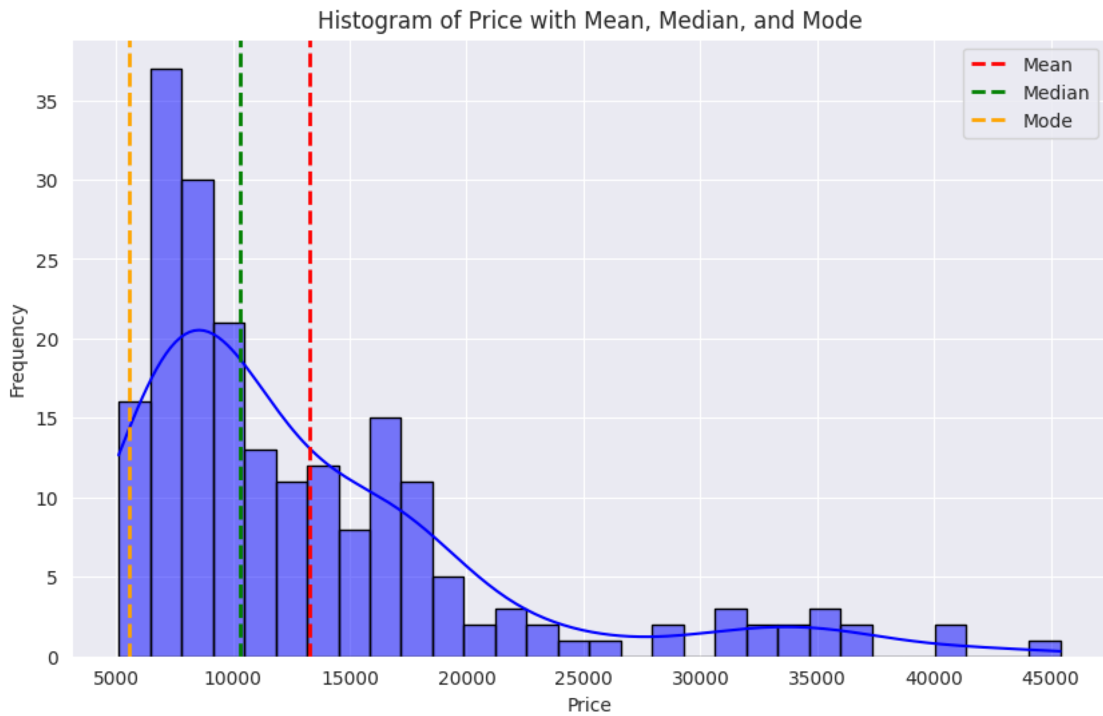
1. Why did you pick the specific chart? The countplot is a preferred choice for visualizing categorical data due to its simplicity and effectiveness. Designed specifically for categorical variables, it provides a clear representation of the frequency of each category through bar heights.

2. What is/are the insight(s) found from the chart? The data reveals that petrol is the most prevalent fueltype, followed by diesel, CNG, and hybrid. Natural aspiration dominates the aspiration category, followed by turbo. The majority of vehicles have four doors, followed by five and two. Hatchbacks are the most common carbody, followed by sedans and SUVs. Four-wheel drive is the most common drivewheel type, followed by rear-wheel drive and front-wheel drive. Front-engine placement is more common than rear-engine placement. The majority of engines are four-cylinder, followed by three-cylinder, six-cylinder, and five-cylinder engines. The most common number of cylinders is four, followed by three and five. MPFI is the most common fuel system, followed by SPFI and EFI. These insights provide a comprehensive understanding of the data's characteristics and enable predictions about future trends.

Chart - 3 Histogram (Checking Distribution of dataset Dependent Variable)

[126]:
```python
# Set up the figure size
plt.figure(figsize=(10, 6))

# Plotting the histogram for 'price' variable
sns.histplot(cars_df['price'], bins=30, kde=True, color='blue',␣
 ↪edgecolor='black')

# Plotting the mean line
plt.axvline(cars_df['price'].mean(), color='red', linestyle='dashed',␣
 ↪linewidth=2, label='Mean')

# Plotting the median line
plt.axvline(cars_df['price'].median(), color='green', linestyle='dashed',␣
 ↪linewidth=2, label='Median')

# Plotting the mode line
plt.axvline(cars_df['price'].mode().values[0], color='orange',␣
 ↪linestyle='dashed', linewidth=2, label='Mode')

# Adding labels and title
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.title('Histogram of Price with Mean, Median, and Mode')

# Adding a legend
plt.legend()
```

```
# Displaying the plot
plt.show()
```



1. Why did you pick the specific chart? This specific chart was picked up to check the if the numerical columns which are to be involved in the building the **Regression** model follows **Gaussian Distribution** or not.

In general, in a particualr dataset

Mean $<$ **Median** $<$ **Mode** - if the dataset follows **Left Skewed Distribution**

Mean $>$ **Median** $<$ **Mode** - if the dataset follows **Right Skewed Distribution**

Mean $=$ **Median** $=$ **Mode** - if the dataset follows **Symmetric Distribution**

2. What is/are the insight(s) found from the chart? When we talk particularly about the target variable i.e **'price'** even this has a skewed distribution i.e **right skewed** distribution. This needs to be treated as we know that a skewed dataset especially in case of regression models such as linear regression algorithms lowers the models performance.

Chart - 4 Boxplot (Identifying Outliers for dataset) Implemented a robust outlier detection technique, leveraging the Interquartile Range (IQR) method. This method ensures the identification of unexpected extreme values in the dataset tails.

Percentile Computation:

Calculated the first quartile (Q1, 25th percentile) and the third quartile (Q3, 75th percentile) to establish the data spread.

Bound Definition:

Defined lower and upper bounds using Q1, Q3, and the Interquartile Range (IQR):

Lower Bound: Q1 - 1.5 × IQR

Upper Bound: Q3 + 1.5 × IQR

Outlier Identification: Identified outliers as data points falling below the lower bound or above the upper bound.

[127]:
```python
# Count the number of outliers for each feature variable
outliers_count = {}

# Iterate through each column in the dataframe
for column in cars_df.columns:
    # Check if the data type of the column is numeric (integer or float)
    if cars_df[column].dtype in ['int64', 'float64']:
        # Calculate the first quartile (Q1)
        Q1 = cars_df[column].quantile(0.25)

        # Calculate the third quartile (Q3)
        Q3 = cars_df[column].quantile(0.75)

        # Calculate the Interquartile Range (IQR)
        IQR = Q3 - Q1

        # Calculate the lower and upper bounds to identify outliers
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        # Count the number of outliers for the current column
        outliers_count[column] = cars_df[(cars_df[column] < lower_bound) |
 ↪(cars_df[column] > upper_bound)].shape[0]

# Display the count of outliers for each feature variable
print("Number of outliers for each feature variable:")
outliers_count
```

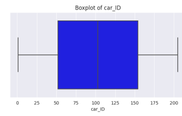
```
Number of outliers for each feature variable:
```

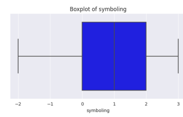
[127]:
```
{'car_ID': 0,
 'symboling': 0,
 'wheelbase': 3,
 'carlength': 1,
 'carwidth': 8,
 'carheight': 0,
```

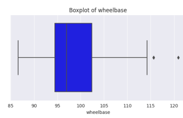
```
 'curbweight': 0,
 'enginesize': 10,
 'boreratio': 0,
 'stroke': 20,
 'compressionratio': 28,
 'horsepower': 6,
 'peakrpm': 2,
 'citympg': 2,
 'highwaympg': 3,
 'price': 15}
```

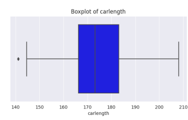
[128]:

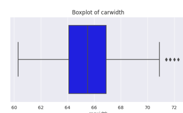
```
# Extracting numerical variables
numeric_variables = cars_df.select_dtypes(include=['float64', 'int64']).columns

# Removing 'price' column from the list of numerical variables
numeric_variables = numeric_variables.drop('price')

# Setting up subplots
fig, axes = plt.subplots(nrows=len(numeric_variables), ncols=1, figsize=(7, 5 *␣
 ↪len(numeric_variables)))
fig.subplots_adjust(hspace=0.5)

# Plotting box plots for each numeric variable
for i, variable in enumerate(numeric_variables):
    # Creating a box plot for the current numeric variable
    sns.boxplot(data=cars_df, x=variable, ax=axes[i], color = 'blue')

    # Setting the title of the subplot
    axes[i].set_title(f'Boxplot of {variable}')

# Displaying the subplots
plt.show()
```

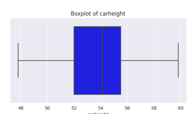

Boxplot of car_ID
0 25 50 75 100 125 150 175 200
car_ID

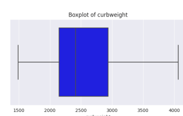

Boxplot of symboling
−2 −1 0 1 2 3
symboling

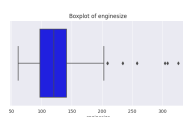

Boxplot of wheelbase
85 90 95 100 105 110 115 120
wheelbase

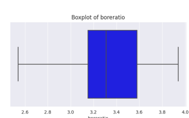

Boxplot of carlength
140 150 160 170 180 190 200 210
carlength

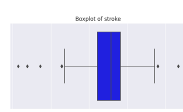

Boxplot of carwidth
60 62 64 66 68 70 72
carwidth

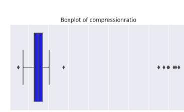

Boxplot of carheight
48 50 52 54 56 58 60
carheight

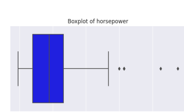

Boxplot of curbweight
1500 2000 2500 3000 3500 4000
curbweight

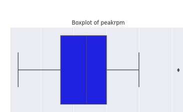

Boxplot of enginesize
50 100 150 200 250 300
enginesize

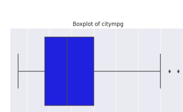

Boxplot of boreratio
2.6 2.8 3.0 3.2 3.4 3.6 3.8 4.0
boreratio

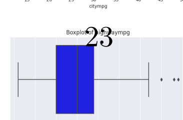

Boxplot of stroke
2.0 2.5 3.0 3.5 4.0
stroke


Boxplot of compressionratio
8 10 12 14 16 18 20 22
compressionratio


Boxplot of horsepower
50 100 150 200 250 300
horsepower


Boxplot of peakrpm
4500 5000 5500 6000 6500
peakrpm


Boxplot of citympg
15 20 25 30 35 40 45 50
citympg


15 20 25 30 35 40 45 50 55
highwaympg

[129]:
```
for column in cars_df.columns:
    if cars_df[column].dtype in ['int64', 'float64']:
        Q1 = cars_df[column].quantile(0.25)
        Q3 = cars_df[column].quantile(0.75)
        IQR = Q3 - Q1

        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        cars_df.loc[cars_df[column] < lower_bound, column] = lower_bound
        cars_df.loc[cars_df[column] > upper_bound, column] = upper_bound
```

1. Why did you pick the specific chart? The selection of this specific chart aimed to visually assess outliers within the independent numeric variables. A boxplot was utilized as the chosen visualization method for this examination.

2. What is/are the insight(s) found from the chart? From the visual analysis, it is evident that certain feature variables exhibit outliers, identified using the IQR approach. These outliers extend beyond the upper and lower bounds of the expected range. Subsequently, a robust data treatment strategy was implemented, replacing the identified outliers with their respective upper and lower bound values. This intervention ensures data integrity and aligns the feature variables with a more standardized and reliable distribution.

Chart - 5 Regplot (For Assessing the Correlation Between dependent and independent Variables)

[130]:
```
# Calculate the correlation matrix for all numeric variables with respect to
↪'price'
price_correlation = cars_df.corr()['price']

# Displaying the correlation matrix
price_correlation
```

```
<ipython-input-130-2787b5888632>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  price_correlation = cars_df.corr()['price']
```

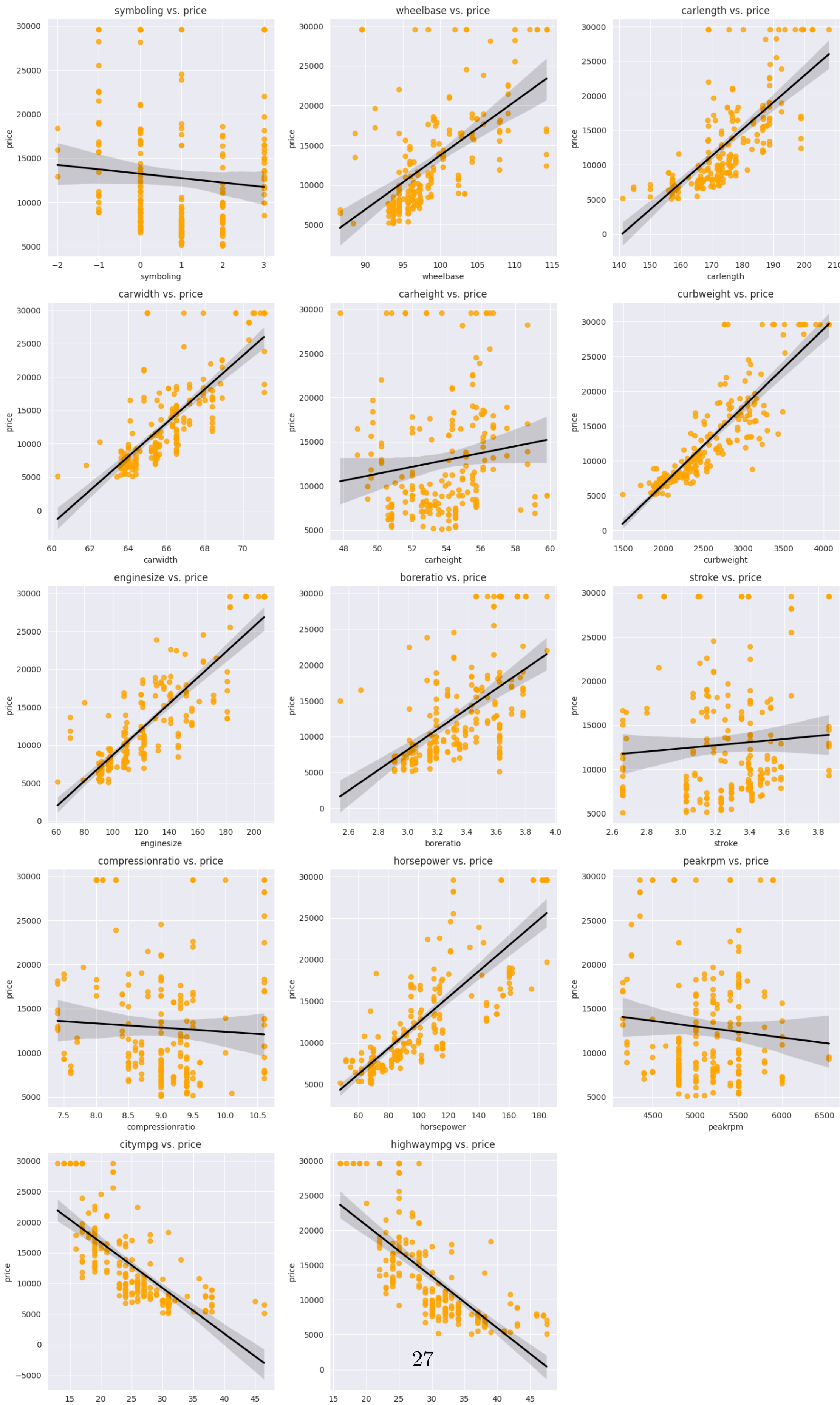
[130]:
```
car_ID           -0.089603
symboling        -0.092705
wheelbase         0.595909
carlength         0.712455
carwidth          0.783230
carheight         0.142033
```

```
curbweight          0.864597
enginesize          0.860063
boreratio           0.572685
stroke              0.073830
compressionratio   -0.056573
horsepower          0.821715
peakrpm            -0.088630
citympg            -0.718290
highwaympg         -0.733692
price               1.000000
Name: price, dtype: float64
```

[131]:

```python
# Extracting numerical variables
numeric_variables = cars_df.select_dtypes(include=['float64', 'int64']).columns

# Removing 'car_ID' and 'symboling' columns from the list of numerical variables
numeric_variables = numeric_variables.drop(['car_ID', 'price'])

# Defining dependent variable
dependent_variable = 'price'

# Setting up subplots
total_plots = len(numeric_variables)
num_cols = 3
num_rows = (total_plots - 1) // num_cols + 1
plt.figure(figsize=(15, 5 * num_rows))

# Plotting regression plots for each numeric variable
for i, column in enumerate(numeric_variables):
    plt.subplot(num_rows, num_cols, i+1)

    # Creating a regression plot for the current numeric variable against the␣
 ↪dependent variable
    sns.regplot(x=cars_df[column], y=cars_df[dependent_variable],␣
 ↪scatter_kws={"color": 'orange'}, line_kws={"color": "black"})

    # Setting the title of the subplot
    plt.title(f'{column} vs. {dependent_variable}')

    # Setting x-axis label
    plt.xlabel(column)

    # Setting y-axis label
    plt.ylabel(dependent_variable)

# Adjusting layout for better visualization
plt.tight_layout()
```

```
# Displaying the subplots
plt.show()
```


symboling vs. price
wheelbase vs. price
carlength vs. price
carwidth vs. price
carheight vs. price
curbweight vs. price
enginesize vs. price
boreratio vs. price
stroke vs. price
compressionratio vs. price
horsepower vs. price
peakrpm vs. price
citympg vs. price
highwaympg vs. price
price
symboling
wheelbase
carlength
carwidth
carheight
curbweight
enginesize
boreratio
stroke
compressionratio
horsepower
peakrpm
citympg
highwaympg

1. Why did you pick the specific chart? The application of **regplot** is fitting for our visualization needs as it not only facilitates the exploration of the linear relationship between independent and dependent variables but also incorporates a trendline. This feature enhances our understanding by visually representing the best-fit linear regression line, elucidating how variations in the independent variables correspond to changes in the dependent variable.

2. What is/are the insight(s) found from the chart? The following insights were found from the above visualization :

---

1. **Strong Positive Correlation**:

For variables like 'carwidth', 'curbweight', 'enginesize', and 'horsepower' that exhibit strong positive correlations with 'price', the regplot with a trendline will show a clear upward-sloping line. As these variables increase, the 'price' tends to increase, forming a distinct positive linear relationship captured by the trendline.

---

2. **Moderate Positive Correlation**:

Variables with moderate positive correlations such as 'wheelbase', 'carlength', and 'boreratio' will also result in a positive slope in the regplot with a trendline. The slope may not be as steep as in the case of strong positive correlations, indicating a moderately positive linear relationship represented by the trendline.

---

3. **Weak Positive Correlation**:

For variables like 'carheight', 'compressionratio', and 'peakrpm' with weak positive correlations, the regplot with a trendline may show a positive slope, but the relationship is not as pronounced. The points on the scatter plot may not form a clear linear trend, and the trendline captures the subtle positive relationship.

---

4. **Negative Correlation**:

Variables 'citympg' and 'highwaympg' with negative correlations will yield a regplot with a downward-sloping trendline. As these variables increase, the 'price' tends to decrease, showcasing a negative linear relationship captured by the trendline.

---

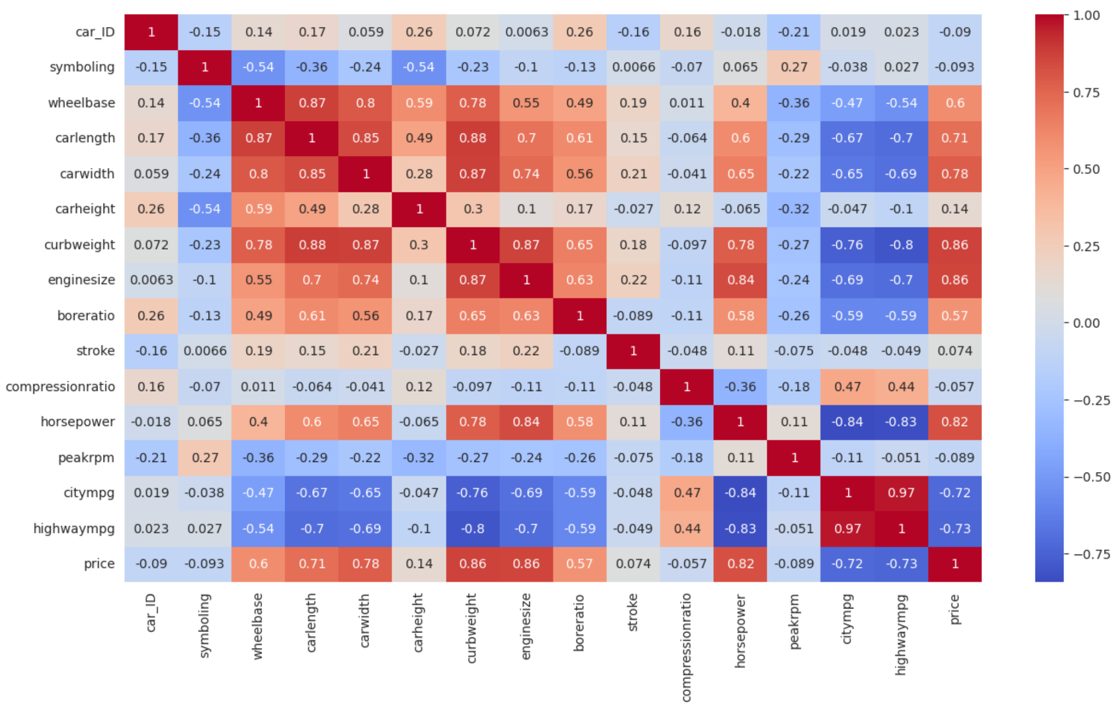
5. **Weak Negative Correlation**:

Variables 'symboling' and 'stroke' exhibit weak negative correlations. The regplot with a trendline may show a negative slope, but the relationship is not strongly evident. The trendline captures the subtle negative relationship, and the scatter of points may not form a distinct downward trend.

Chart - 6 Correlation HeatMap (Assessing Correlation amongst all the variables)

[132]:
```python
# Chart - 4 visualization code
## Correlation

# Setting up the figure size
plt.figure(figsize=(15, 8))

# Calculating the correlation matrix
correlation = cars_df.corr()

# Creating a heatmap to visualize the correlation matrix
sns.heatmap(correlation, annot=True, cmap='coolwarm')

# Displaying the plot
plt.show()
```

```
<ipython-input-132-6b03ad9b8594>:8: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation = cars_df.corr()
```



1. Why did you pick the specific chart? A correlation heatmap is used to visualize the strength and direction of relationships between two or more variables in a dataset. The heatmap displays correlation coefficients as color-coded values, allowing for easy identification of patterns and insights into the associations among variables.

2. What is/are the insight(s) found from the chart? The correlation matrix effectively visualizes the strength and direction of correlations between variables. The color scale employed in the matrix enhances this understanding, where dark red indicates a strong positive correlation, and dark blue represents a substantial negative correlation. The intensity of the color corresponds to the strength of the correlation, with darker shades signifying stronger associations. As the color transitions to lighter shades in either red or blue, the strength of the correlation diminishes. This visual representation provides a quick and insightful overview of the relationships within the dataset.

Focusing on the correlations between some notable variables:

Horsepower (hp) and **curb weight**: A strong positive correlation is observed, suggesting that as horsepower increases, so does curb weight. This is intuitive, as heavier vehicles typically require more powerful engines.

Horsepower (hp) and **city mpg**: A moderate negative correlation is seen, implying that as horsepower increases, city fuel efficiency tends to decrease. This is expected, as more powerful engines generally consume more fuel.

Wheelbase and **car length**: A strong positive correlation is evident, indicating that as wheelbase increases, car length also tends to increase. This is because wheelbase is the distance between the front and rear axles, and a larger wheelbase typically translates to a longer car.

Bore ratio and **stroke**: A moderate negative correlation is evident, suggesting that as bore ratio increases (meaning the cylinder is wider relative to its stroke), stroke tends to decrease (meaning the piston travels a shorter distance). This relationship is often observed in engine design, as bore and stroke are crucial factors in determining engine characteristics.

Compression ratio and **horsepower** (hp): A moderate positive correlation is seen, indicating that as compression ratio increases, horsepower also tends to increase. This is because a higher compression ratio allows for more efficient combustion, leading to higher power output.

6.4 *4. Feature Engineering & Data Pre-processing*

6.4.1 1. Handling Missing Values

[133]:
```
# Handling Missing Values & Missing Value Imputation
```

What all missing value imputation techniques have you used and why did you use those techniques? The dataset utilized for this project, namely cars_df, demonstrates a noteworthy attribute—complete absence of missing values. Given this, there is no need for imputation or handling of **missing data**, allowing us to seamlessly progress to the next step, which involves treating outliers in the dataset.

6.4.2 2. Handling Outliers

Having successfully identified outliers within the feature variables using the **IQR** approach, the ensuing step involves implementing outlier treatment for the following list of variables.

1. **wheelbase**
2. **carlength**
3. **carwidth**
4. **enginesize**
5. **stroke**
6. **compressionratio**
7. **horsepower**
8. **peakrpm**
9. **citympg**
10. **highwaympg**
11. **price**

[134]:
```
for column in cars_df.columns:
    if cars_df[column].dtype in ['int64', 'float64']:
        Q1 = cars_df[column].quantile(0.25)
        Q3 = cars_df[column].quantile(0.75)
        IQR = Q3 - Q1

        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        cars_df.loc[cars_df[column] < lower_bound, column] = lower_bound
        cars_df.loc[cars_df[column] > upper_bound, column] = upper_bound
```

What all outlier treatment techniques have you used and why did you use those techniques? Percentile Computation:

Calculate Q1 (25th percentile) and Q3 (75th percentile) of the dataset. Bound Definition:

Establish lower and upper bounds using the IQR:

Lower Bound: Q1 - 1.5 $\times$ IQR

Upper Bound: Q3 + 1.5 $\times$ IQR

Outlier Replacement:

Replace values that fall outside the upper or lower bounds with the respective boundary values.

The IQR methodology, being robust and distribution-free, furnishes a dependable means of identifying outliers. Temporary removal of these outliers unveils more stable structures and relationships in the data that may be obscured by extreme values. Importantly, no data is permanently lost, as outliers are reintroduced into the dataset post-analysis. This approach circumvents assumptions about the underlying distribution, allowing the analysis to unveil central tendencies that outliers may otherwise conceal.

6.4.3 3. Feature Manipulation

After having a look at the dataset, We can add three more features(variables) from the existing dataset. 1. **mileage** = (**0.6** * city miles per gallon) + (**0.4** * highway miles per gallon)

2. **car_area** = carlength * carwidth

3. **car_volume** = carlength * carwidth * carheight

[135]:
```
# Manipulate Features to minimize feature correlation and create new features
cars_df['mileage'] = 0.6*cars_df['citympg'] + 0.4*cars_df['highwaympg']
cars_df['car_area'] = cars_df['carlength']*cars_df['carwidth']
cars_df['car_volume'] = cars_df['carlength'] * cars_df['carwidth'] *␣
 ↪cars_df['carheight']
```

The dataset also has **CarName** column which contains the name of the car along with the company name. From this we can extract the company name in a seperate column.

[136]:
```
cars_df['company'] = cars_df['CarName'].str.split(" ", expand=True)[0]
```

Also when you pay closer attendtion to the dataset, you will find that some of the company names have been wrongly entered in the dataset with incorrect spellings due to which the they are being considered as seperate company which is not the case. Therefore we will be doing some modification in that by correcting the names of the company.

[137]:
```
cars_df['company'] = cars_df['company'].replace({'toyouta': 'Toyota','vw':
 ↪'Volkswagen','vokswagen':'Volkswagen','maxda':'Mazda','porcshce':'Porsche'})
cars_df['company'] = cars_df['company'].str.title()
```

Removing the irrelevant columns from the context of Model Implementation

From thw above dataset we will be removing two columns i.e Car_ID & CarName. The reason for dopping these two columns from the dataset is their relevance in building and implementing the machine learning model. Firslty talking about Car_ID though being a numeric column, it is just representing a particular id assigned to a particular and there's no actual relationship between that one could assess between it and price of a particular car.

Secondly talking about the CarName it is categorical variable which contains the names of the different cars. Here we have used this column indirectly by extracting the company name from this column and then later converting that to a Frequency Encoded column. Also car name direclty cannot used for building and implementing regression models as it contains text values. Also we will not be dropping the **CarName** column in this step because of we have some (**calculations/operations to perform before we actually drop it**).

1. **Car_ID**
2. **CarName**

[138]:
```
# Dropping the irrelevant columns
cars_df.drop(['car_ID' ],axis = 1, inplace = True)
```

6.4.4 4. Categorical Encoding

List of Variables which will have **Frequency Encoding** :-

1. company

List of Variables which will have **Label Encoding** :-

1. **doornumber**
2. **cylindernumber**

List of Variables which will have **One Hot Encoding** :-

1. **carbody**
2. **fueltype**
3. **aspiration**
4. **enginelocation**
5. **enginetype**
6. **fuelsystem**
7. **drivewheel**

What all categorical encoding techniques have you used & why did you use those techniques? In the above case I have used (**Frequency Encoding**), (**Label Encoding**) & (**One Hot Encoding**) as the variables were of such nature that the above three were most relevant options. For variable such as **carbody**, **fueltype**,**aspiration** etc. since they are **nominal** data that have no inherent **order/ranking** attached to them, so i went ahead One Hot Encoding. For variable such as **doornumber**, **cylindernumber** etc. Label Encoding since they were variables which can be represented in two numeric values. For the variable **company** name I have used the Frequency Encoding because it contains the 31 distinct names of the car company. So accordingly we will replace the names with the frequency of the respective car name.

[139]:
```
# Calculate the frequency of each category
frequency_map = cars_df['company'].value_counts(normalize=True).to_dict()

# Map the frequencies to the 'company_name' column
cars_df['company_name_frequency_encoded'] = cars_df['company'].
↪map(frequency_map)
```

[140]:
```
#Label Encoding
encoders_nums = {"doornumber":     {"four": 4, "two": 2},
                 "cylindernumber":{"four": 4, "six": 6, "five": 5, "eight": 8,
                                   "two": 2, "twelve": 12, "three":3 }
                 }
cars_df = cars_df.replace(encoders_nums)
```

Here in this step we are creating a copy of the updated dataset i.e cars_df till this step which will be used for EDA and answering some of the questions related to this project.

[141]:
```
# Created another copy of the above dataset for EDA as some additional columns␣
↪were added prior to this step
cars_df_copy2 = cars_df.copy()
```

[142]:
```
# Creating dummy variables for the below mentioned categorical variables
cars_df = pd.get_dummies(cars_df, columns=['carbody'], prefix=['body'],␣
↪drop_first=True)
cars_df = pd.get_dummies(cars_df, columns=['fueltype'], prefix=['fuel'],␣
↪drop_first = True)
cars_df = pd.get_dummies(cars_df, columns=['aspiration'], prefix=['asp'],␣
↪drop_first = True)
cars_df = pd.get_dummies(cars_df, columns=['enginelocation'], prefix=['eng'],␣
↪drop_first=True)
cars_df = pd.get_dummies(cars_df, columns=['enginetype'], prefix=['type'],␣
↪drop_first=True)
cars_df = pd.get_dummies(cars_df, columns=['fuelsystem'], prefix=['sys'],␣
↪drop_first=True)
cars_df = pd.get_dummies(cars_df, columns=['drivewheel'], prefix=['drive'],␣
↪drop_first=True)
```

6.4.5 5. Feature Selections

In this stage, We will be doing the feature selection from our dataset. For this project and problem statement we will be using the **Univariate Feature** Selection where in we will be using the Correlation approach (**Pearson & Point Biserial**) for selecting the best feature with respect to with respect to the dependent variable (**Price**).

Here we will be using the two different correlation approaches (**Pearson & Point Biserial**) for feature selection. The reason for choosing two different correlation approaches is that we have mixed data in our dataset i.e numeric (**continuous & discrete**) and categorically encoded data (**binary**). Therefore relevant correlation approaches were chosen considering the type of data involved.

Once we have have calculated the correlation value with respect to price, we will setup a threshhold value range say (**-0.3 to 0.3**) and will include only those features which will lie outside this range. The basic assumption behind this is that when we measure correlation of any variable with respect to dependent variable as the value is more towards the (+/-) 1, there appears to be a **strong linear relationship** with dependent variable and hence should include those variables.

For Numeric Data (**Continuous & Discrete**)

[143]:
```
# Extract only numeric columns (int and float)
numeric_columns = cars_df.select_dtypes(include=['int', 'float']).columns

# Calculate Pearson correlation for each numeric column with 'price'
```

```
correlation_pc = cars_df[numeric_columns].corrwith(cars_df['price'])

# Create a DataFrame to store variable names and their correlation values
correlation_df_pc = pd.DataFrame({'Variable': correlation_pc.index, 'Pearson␣
↪Correlation': correlation_pc.values})

# Filter variables based on correlation values outside the range (-0.3, 0.3)
selected_features_p_correlation = correlation_df_pc[(correlation_df_pc['Pearson␣
↪Correlation'] < -0.3) | (correlation_df_pc['Pearson Correlation'] > 0.3)]

#correlation_df
selected_features_p_correlation
```

[143]:

	Variable	Pearson Correlation
2	wheelbase	0.595909
3	carlength	0.712455
4	carwidth	0.783230
6	curbweight	0.864597
7	cylindernumber	0.677018
8	enginesize	0.860063
9	boreratio	0.572685
12	horsepower	0.821715
14	citympg	-0.718290
15	highwaympg	-0.733692
16	price	1.000000
17	mileage	-0.730484
18	car_area	0.762857
19	car_volume	0.650803
20	company_name_frequency_encoded	-0.341673

For Categorically Encoded Data (**Binary**)

[144]:

```
# Extract only uint8 columns
categorical_features = cars_df.select_dtypes(include='uint8').columns

# Calculate point-biserial correlation for each uint8 column with 'price'
correlation_values_pb = {}

for feature in categorical_features:
    correlation_pb, _ = pointbiserialr(cars_df[feature], cars_df['price'])
    if abs(correlation_pb) > 0.3:  # Check if correlation is outside the range␣
↪(-0.3, 0.3)
        correlation_values_pb[feature] = correlation_pb

selected_features_pb_correlation = pd.DataFrame(list(correlation_values_pb.
↪items()), columns=['Variable', 'Point Biserial Correlation'])
```

```
# Display the resulting correlation values
selected_features_pb_correlation
```

[144]:

```
    Variable  Point Biserial Correlation
0   eng_rear                    0.304551
1   type_ohc                   -0.338524
2  type_ohcv                    0.345786
3   sys_2bbl                   -0.550535
4   sys_mpfi                    0.545737
5  drive_fwd                   -0.636983
6  drive_rwd                    0.673377
```

[145]:

```
# Merging both the selected features into a single variable
selected_features = pd.concat([selected_features_p_correlation['Variable'],␣
 ↪selected_features_pb_correlation['Variable']], ignore_index=True).unique()
selected_features
```

[145]:

```
array(['wheelbase', 'carlength', 'carwidth', 'curbweight',
       'cylindernumber', 'enginesize', 'boreratio', 'horsepower',
       'citympg', 'highwaympg', 'price', 'mileage', 'car_area',
       'car_volume', 'company_name_frequency_encoded', 'eng_rear',
       'type_ohc', 'type_ohcv', 'sys_2bbl', 'sys_mpfi', 'drive_fwd',
       'drive_rwd'], dtype=object)
```

After doing the Feature Selection, here is the list of Variables which are selected for implementing different machine learning models.

Variable Names

1. wheelbase
2. carlength
3. carwidth
4. curbweight
5. cylindernumber
6. enginesize
7. boreratio
8. horsepower
9. citympg
10. highwaympg
11. mileage
12. car_area
13. eng_rear

14. type_ohc
15. type_ohcv
16. sys_2bbl
17. sys_mpfi
18. drive_fwd
19. drive_rwd
20. car_volume
21. company_name_frequency_encoded

6.4.6 6. Data Splitting

Once we have selected the best features according to the respective method applied, we need to now create a seperate dataset to store only those selected features with their values which will be then splitted into (**Training & Testing**) data.

So for the same, first we will merge the two dataset containing the selected features and then store it into another variable called **selected_features**.

Post this we will create a copy of original dataset with only those selected features.

[146]:
```
# Merging both the selected features into a single variable
selected_features = pd.concat([selected_features_p_correlation['Variable'],␣
↪selected_features_pb_correlation['Variable']], ignore_index=True).unique()

# Create a copy of the original dataset with only the selected features
selected_features_df = cars_df[selected_features].copy()

selected_features_df
```

[146]:
```
     wheelbase  carlength  carwidth  curbweight  cylindernumber  enginesize  \
0         88.6      168.8      64.1        2548               4         130
1         88.6      168.8      64.1        2548               4         130
2         94.5      171.2      65.5        2823               6         152
3         99.8      176.6      66.2        2337               4         109
4         99.4      176.6      66.4        2824               5         136
..         …          …        …           …              …           …
200      109.1      188.8      68.9        2952               4         141
201      109.1      188.8      68.8        3049               4         141
202      109.1      188.8      68.9        3012               6         173
203      109.1      188.8      68.9        3217               6         145
204      109.1      188.8      68.9        3062               4         141

     boreratio  horsepower  citympg  highwaympg  …  car_area  car_volume  \
0         3.47         111     21.0        27.0  …  10820.08  528019.904
1         3.47         111     21.0        27.0  …  10820.08  528019.904
```

```
2          2.68          154     19.0         26.0  …  11213.60  587592.640
3          3.19          102     24.0         30.0  …  11690.92  634816.956
4          3.19          115     18.0         22.0  …  11726.24  636734.832
..           …         …      …          …   …      …         …
200        3.78          114     23.0         28.0  …  13008.32  721961.760
201        3.78          160     19.0         25.0  …  12989.44  720913.920
202        3.58          134     18.0         23.0  …  13008.32  721961.760
203        3.01          106     26.0         27.0  …  13008.32  721961.760
204        3.78          114     19.0         25.0  …  13008.32  721961.760

     company_name_frequency_encoded  eng_rear  type_ohc  type_ohcv  sys_2bbl  \
0                          0.014634         0         0          0         0
1                          0.014634         0         0          0         0
2                          0.014634         0         0          1         0
3                          0.034146         0         1          0         0
4                          0.034146         0         1          0         0
..                              …       …       …        …       …
200                        0.053659         0         1          0         0
201                        0.053659         0         1          0         0
202                        0.053659         0         0          1         0
203                        0.053659         0         1          0         0
204                        0.053659         0         1          0         0

     sys_mpfi  drive_fwd  drive_rwd
0           1          0          1
1           1          0          1
2           1          0          1
3           1          1          0
4           1          0          0
..        …        …        …
200         1          0          1
201         1          0          1
202         1          0          1
203         0          0          1
204         1          0          1

[205 rows x 22 columns]
```

[147]:

```python
# Split your data to train and test. Choose Splitting ratio wisely.

# Create a dependent Variable
dependent_variable = 'price'

# Create an Independent Variable
independent_variable = list(set(selected_features_df.columns.tolist()) -
 {dependent_variable})
```

```
# Create variable x for storing independent variables values in it
x = selected_features_df[independent_variable].values

# Create variable y for storing dependent/target variable values in it
y = selected_features_df[dependent_variable].values

# Splitting the dataset into the Training set and Test set
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2,␣
↪random_state = 0)
```

[148]:
```
# Display the structure of the Training and Testing data
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(164, 21)
(164,)
(41, 21)
(41,)
```

###7. Data Transformation/Scaling

Do you think that your data needs to be transformed? If yes, which transformation have you used. Explain Why? Logarithmic transformation is often employed for skewed datasets due to several advantages it offers in statistical analyses.One primary benefit is the stabilization of variance, especially when dealing with heteroscedasticity, where the variability of the data fluctuates across different levels of independent variables. Additionally, skewed distributions, whether left-skewed or right-skewed, may violate the assumption of normality in statistical methods. Logarithmic transformation helps mitigate skewness, making the distribution more symmetric and aligning it closer to a normal distribution. Moreover, when relationships between variables are not linear, as assumed in linear regression, logarithmic transformation can be used to linearize these relationships and improve model fit.

[149]:
```
# Logarithmic transformation on training data
x_train_log = np.log1p(x_train)

# Logarithmic transformation on test data
x_test_log = np.log1p(x_test)
```

6.5 5. *EDA Process*

In this step after analysing the the dataset i.e cars_df, i have come up with certain questions related to project which will help us better understand the this project and also get a good grasp of how the price and other related variables behave with respect to each other and therefore will include bivariate anlaysis.

Q1. What are the top 10 cars by price factor?

[150]:
```python
# Extracting the relevant columns
columns_of_interest1 = ['CarName', 'price']
cars_data1 = cars_df_copy2[columns_of_interest1]

# Sorting the DataFrame by price in descending order
top_10_cars = cars_data1.sort_values(by='price', ascending=False).head(10)
top_10_cars
```

[150]:

	CarName	price
17	bmw x3	29575.5
74	buick regal sport coupe (turbo)	29575.5
48	jaguar xf	29575.5
129	porsche cayenne	29575.5
16	bmw x5	29575.5
15	bmw x4	29575.5
47	jaguar xj	29575.5
126	porcshce panamera	29575.5
73	buick century special	29575.5
72	buick skylark	29575.5

Q2. What are the lowest 10 cars by price factor?

[151]:
```python
# Extracting the relevant columns
columns_of_interest2 = ['CarName', 'price']
cars_data2 = cars_df_copy2[columns_of_interest2]

# Sorting the DataFrame by price in ascending order
lowest_10_cars = cars_data2.sort_values(by='price', ascending=True).head(10)

# Displaying the lowest 10 priced cars
lowest_10_cars
```

[151]:

	CarName	price
138	subaru	5118.0
18	chevrolet impala	5151.0
50	maxda rx3	5195.0
150	toyota corona mark ii	5348.0
76	mitsubishi mirage	5389.0
32	honda civic	5399.0
89	Nissan versa	5499.0
118	plymouth fury iii	5572.0
21	dodge rampage	5572.0
51	maxda glc deluxe	6095.0

Q3. What are the top 10 cars by car volume factor ?

[152]:
```python
# Extracting the relevant columns
columns_of_interest3 = ['CarName', 'car_volume']
cars_data3 = cars_df_copy2[columns_of_interest3]

# Sorting the DataFrame by price in ascending order
top_10_cars_in_volume_terms= cars_data3.sort_values(by='car_volume',␣
↪ascending=False).head(10)

# Displaying the highest 10 cars in terms of car volume
top_10_cars_in_volume_terms
```

[152]:

	CarName	car_volume
73	buick century special	838928.097
71	buick opel isuzu deluxe	813874.590
70	buick skyhawk	810993.618
114	peugeot 505s turbo diesel	798599.412
110	peugeot 504	798599.412
109	peugeot 504 (sw)	798599.412
68	buick century luxus (sw)	787769.849
17	bmw x3	786358.990
74	buick regal sport coupe (turbo)	784636.848
113	peugeot 504	771389.892

Q4. What are the lowest 10 cars by car volume factor ?

[153]:
```python
# Extracting the relevant columns
columns_of_interest4 = ['CarName', 'car_volume']
cars_data4 = cars_df_copy2[columns_of_interest4]

# Sorting the DataFrame by car_volume in ascending order
lowest_10_cars_in_volume_terms = cars_data4.sort_values(by='car_volume',␣
↪ascending=True).head(10)

# Displaying the lowest 10 cars in terms of car volume
lowest_10_cars_in_volume_terms
```

[153]:

	CarName	car_volume
18	chevrolet impala	452643.156
30	honda civic	469388.952
31	honda civic cvcc	469388.952
32	honda civic	504960.000
34	honda civic cvcc	504960.000
33	honda accord cvcc	504960.000
120	plymouth fury iii	507808.444
24	dodge monaco (sw)	507808.444
25	dodge colt hardtop	507808.444
26	dodge colt (sw)	507808.444

Q5. What are the top 10 cars by car area factor ?

[154]:
```python
# Extracting the relevant columns
columns_of_interest5 = ['CarName', 'car_area']
cars_data5 = cars_df_copy2[columns_of_interest5]

# Sorting the DataFrame by car_area in descending order
top_10_cars_in_area_terms = cars_data5.sort_values(by='car_area',␣
↪ascending=False).head(10)

# Displaying the highest 10 cars in terms of car area
top_10_cars_in_area_terms
```

[154]:

	CarName	car_area
73	buick century special	14795.91
70	buick skyhawk	14404.86
71	buick opel isuzu deluxe	14404.86
74	buick regal sport coupe (turbo)	14163.12
17	bmw x3	13967.30
48	jaguar xf	13892.16
47	jaguar xj	13892.16
6	audi 100ls	13700.97
7	audi 5000	13700.97
8	audi 4000	13700.97

Q6. What are the lowest 10 car by car area factor ?

[155]:
```python
# Extracting the relevant columns
columns_of_interest6 = ['CarName', 'car_area']
cars_data6 = cars_df_copy2[columns_of_interest6]

# Sorting the DataFrame by car_area in ascending order
lowest_10_cars_in_area_terms = cars_data6.sort_values(by='car_area',␣
↪ascending=True).head(10)

# Displaying the lowest 10 cars in terms of car area
lowest_10_cars_in_area_terms
```

[155]:

	CarName	car_area
18	chevrolet impala	8508.33
31	honda civic cvcc	9239.94
30	honda civic	9239.94
34	honda civic cvcc	9600.00
33	honda accord cvcc	9600.00
32	honda civic	9600.00
45	isuzu D-Max V-Cross	9915.24
44	isuzu D-Max	9915.24
19	chevrolet monte carlo	9915.24

```
138                  subaru   9947.46
```

Q7. What is the average car volume against each car company ?

[156]:
```
# Extracting relevant columns
columns_of_interest7 = ['company', 'car_volume']

# Creating a DataFrame with the relevant columns
average_car_volume_by_company = cars_df_copy2[columns_of_interest7]

# Grouping by 'company' and calculating the average volume
average_car_volume_by_company = average_car_volume_by_company.
 ↪groupby('company')['car_volume'].mean().reset_index().sort_values(by =␣
 ↪'car_volume', ascending = True)

# Renaming the columns for clarity
average_car_volume_by_company.columns = ['company', 'average car volume']

# Displaying the DataFrame
average_car_volume_by_company
```

[156]:

	company	average car volume
4	Chevrolet	497806.332000
5	Dodge	534625.439111
7	Isuzu	543500.324500
0	Alfa-Romero	547877.482667
6	Honda	551744.387462
14	Plymouth	551834.287429
11	Mitsubishi	556000.344615
15	Porsche	581886.421800
18	Subaru	589594.428333
12	Nissan	597112.215778
9	Mazda	598388.211647
19	Toyota	601358.206250
20	Volkswagen	625175.615500
16	Renault	630440.820000
10	Mercury	664789.760000
2	Bmw	673741.664000
1	Audi	688157.514000
17	Saab	696139.290000
8	Jaguar	704646.084000
21	Volvo	721492.677818
13	Peugeot	747793.413091
3	Buick	770478.162750

Q8. What is the average car area against each car company ?

[157]:
```python
# Extracting relevant columns for car_area
columns_of_interest8 = ['company', 'car_area']

# Creating a DataFrame with the relevant columns for car_area
average_car_area_by_company = cars_df_copy2[columns_of_interest8]

# Grouping by 'company' and calculating the average car area
average_car_area_by_company = average_car_area_by_company.
↪groupby('company')['car_area'].mean().reset_index().sort_values(by␣
↪='car_area', ascending = True)

# Renaming the columns for clarity
average_car_area_by_company.columns = ['company', 'average car area']

# Displaying the DataFrame for car_area
average_car_area_by_company
```

[157]:

	company	average car area
4	Chevrolet	9507.750000
5	Dodge	10334.722222
6	Honda	10354.879231
7	Isuzu	10408.315000
14	Plymouth	10602.431429
0	Alfa-Romero	10951.253333
11	Mitsubishi	10969.524615
18	Subaru	10972.526667
12	Nissan	11137.121111
19	Toyota	11199.268125
9	Mazda	11208.834118
20	Volkswagen	11326.371667
15	Porsche	11392.728000
16	Renault	11922.315000
10	Mercury	12131.200000
2	Bmw	12280.135000
17	Saab	12408.900000
1	Audi	12624.977143
21	Volvo	12831.534545
13	Peugeot	13072.030000
8	Jaguar	13772.780000
3	Buick	13812.711250

Q9. What is the average mileage across each car company ?

[158]:
```python
# Extracting relevant columns for mileage and company
columns_of_interest9 = ['company', 'mileage']
average_mileage_by_company = cars_df_copy2[columns_of_interest9]
```

```
# Grouping by 'company' and calculating the average mileage
average_mileage_by_company = average_mileage_by_company.
 ↪groupby('company')['mileage'].mean().reset_index().sort_values(by =␣
 ↪'mileage', ascending = True)

# Renaming the columns for clarity
average_mileage_by_company.columns = ['company', 'average mileage']

# Displaying the DataFrame
average_mileage_by_company
```

[158]:

	company	average mileage
8	Jaguar	15.933333
3	Buick	19.500000
15	Porsche	20.840000
1	Audi	20.971429
10	Mercury	21.000000
2	Bmw	21.775000
0	Alfa-Romero	22.866667
21	Volvo	23.036364
17	Saab	23.133333
13	Peugeot	24.127273
16	Renault	26.200000
11	Mitsubishi	27.415385
18	Subaru	28.100000
9	Mazda	28.200000
12	Nissan	29.322222
19	Toyota	29.662500
5	Dodge	30.444444
14	Plymouth	30.542857
20	Volkswagen	31.116667
6	Honda	32.100000
7	Isuzu	33.000000
4	Chevrolet	42.300000

Q10. What is the average mileage of against different fuel types i.e (gas vs diseal ?)

[159]:

```
# Extracting relevant columns
columns_of_interest10 = ['fueltype', 'mileage']
average_mileage_by_fueltype = cars_df_copy2[columns_of_interest10]

# Grouping by 'fueltype' and calculating the average mileage
average_mileage_by_fueltype = average_mileage_by_fueltype.
 ↪groupby('fueltype')['mileage'].mean().reset_index()

# Renaming the columns for clarity
average_mileage_by_fueltype.columns = ['fueltype', 'average mileage']
```

```python
# Displaying the DataFrame
average_mileage_by_fueltype
```

[159]:

	fueltype	average mileage
0	diesel	32.030000
1	gas	26.894054

Q11. What is the average price of cars against different risk rating ?

[160]:
```python
# Group by 'symboling' and calculate the average price for each group
average_price_against_safety_ratings = pd.DataFrame(cars_df_copy2.
    ↪groupby('symboling')['price'].mean()).reset_index().sort_values(by='price',␣
    ↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_safety_ratings = average_price_against_safety_ratings.
    ↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_safety_ratings
```

[160]:

	symboling	average price
3	1	9711.064815
4	2	10109.281250
2	0	13670.151746
0	-2	15781.666667
5	3	16468.037037
1	-1	17029.181818

Q12. What is the average price of cars for each car company ?

[161]:
```python
# Group by 'company' and calculate the average price for each group
average_price_against_company = pd.DataFrame(cars_df_copy2.
    ↪groupby('company')['price'].mean()).reset_index().sort_values(by='price',␣
    ↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_company = average_price_against_company.
    ↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_company
```

[161]:

	company	average price
4	Chevrolet	6007.000000
5	Dodge	7875.444444

```
14      Plymouth    7963.428571
6          Honda    8184.692308
18        Subaru    8541.250000
7          Isuzu    8916.500000
11    Mitsubishi    9239.769231
16       Renault    9595.000000
19        Toyota    9885.812500
20    Volkswagen   10077.500000
12        Nissan   10415.666667
9          Mazda   10652.882353
17          Saab   15223.333333
13       Peugeot   15489.090909
0    Alfa-Romero   15498.333333
10       Mercury   16503.000000
1           Audi   17859.166714
21         Volvo   18063.181818
2            Bmw   23590.187500
15       Porsche   28064.000000
3          Buick   28731.687500
8         Jaguar   29575.500000
```

Q13. What is the average price of cars against different car body types ?

[162]:
```python
# Group by 'carbody' and calculate the average price for each group
average_price_against_body_type = pd.DataFrame(cars_df_copy2.
↪groupby('carbody')['price'].mean()).reset_index().sort_values(by='price',␣
↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_body_type = average_price_against_body_type.
↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_body_type
```

[162]:

	carbody	average price
2	hatchback	10350.580957
4	wagon	12371.960000
3	sedan	13788.859375
1	hardtop	19304.812500
0	convertible	19735.000000

Q14. What is the average car price against different engine types of the car ?

[163]:
```python
# Group by 'enginetype' and calculate the average price for each group
```

```
average_price_against_engine_type = pd.DataFrame(cars_df_copy2.
 ↪groupby('enginetype')['price'].mean()).reset_index().sort_values(by='price',␣
 ↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_engine_type = average_price_against_engine_type.
 ↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_engine_type
```

[163]:

	enginetype	average price
3	ohc	11423.690318
4	ohcf	12748.100000
6	rotor	13020.000000
2	l	14627.583333
0	dohc	17395.666667
5	ohcv	21735.115385
1	dohcv	29575.500000

Q15. What is the average car price against different fuel systems of cars ?

[164]:
```
# Group by 'fuelsystem' and calculate the average price for each group
average_price_against_fuel_system = pd.DataFrame(cars_df_copy2.
 ↪groupby('fuelsystem')['price'].mean()).reset_index().sort_values(by='price',␣
 ↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_fuel_system = average_price_against_fuel_system.
 ↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_fuel_system
```

[164]:

	fuelsystem	average price
1	2bbl	7478.151515
0	1bbl	7555.545455
6	spdi	10990.444444
7	spfi	11048.000000
2	4bbl	12145.000000
4	mfi	12964.000000
3	idi	15736.925000
5	mpfi	16804.789011

Q16. What is the average price of cars against fuel type i.e (gas vs diesel) ?

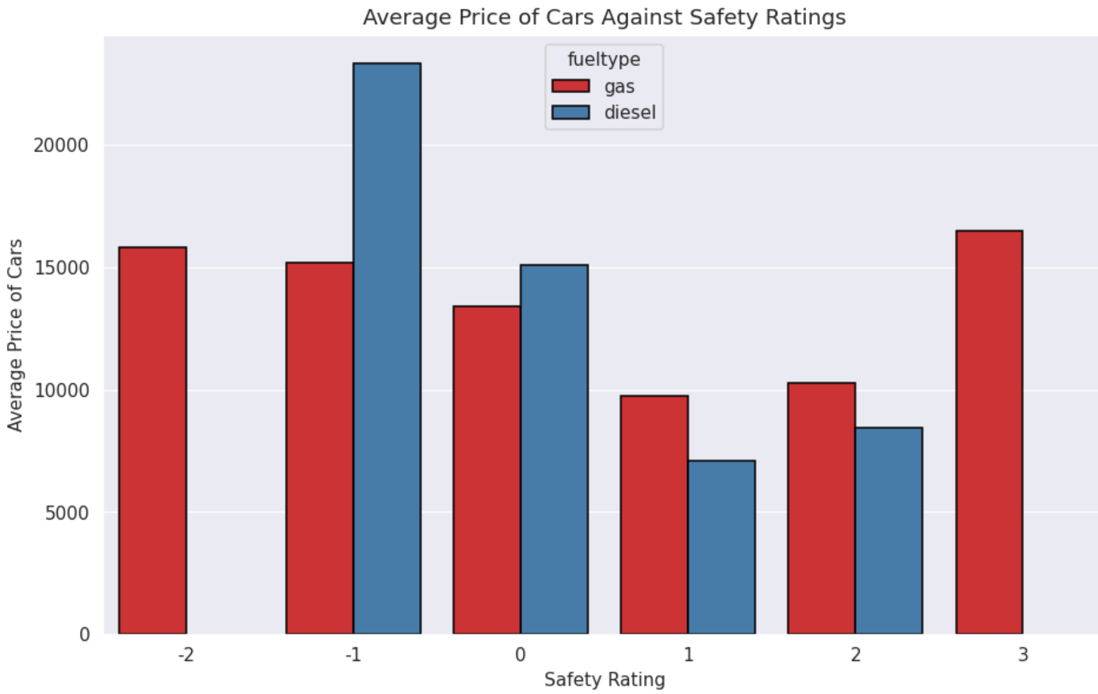
[165]:
```python
# Group by 'fueltype' and calculate the average price for each group
average_price_against_fuel_type = pd.DataFrame(cars_df_copy2.
 ↪groupby('fueltype')['price'].mean()).reset_index().sort_values(by='price',␣
 ↪ascending=True)

# Rename the 'price' column to 'average price'
average_price_against_fuel_type = average_price_against_fuel_type.
 ↪rename(columns={'price': 'average price'})

# Print or use the DataFrame
average_price_against_fuel_type
```

[165]:

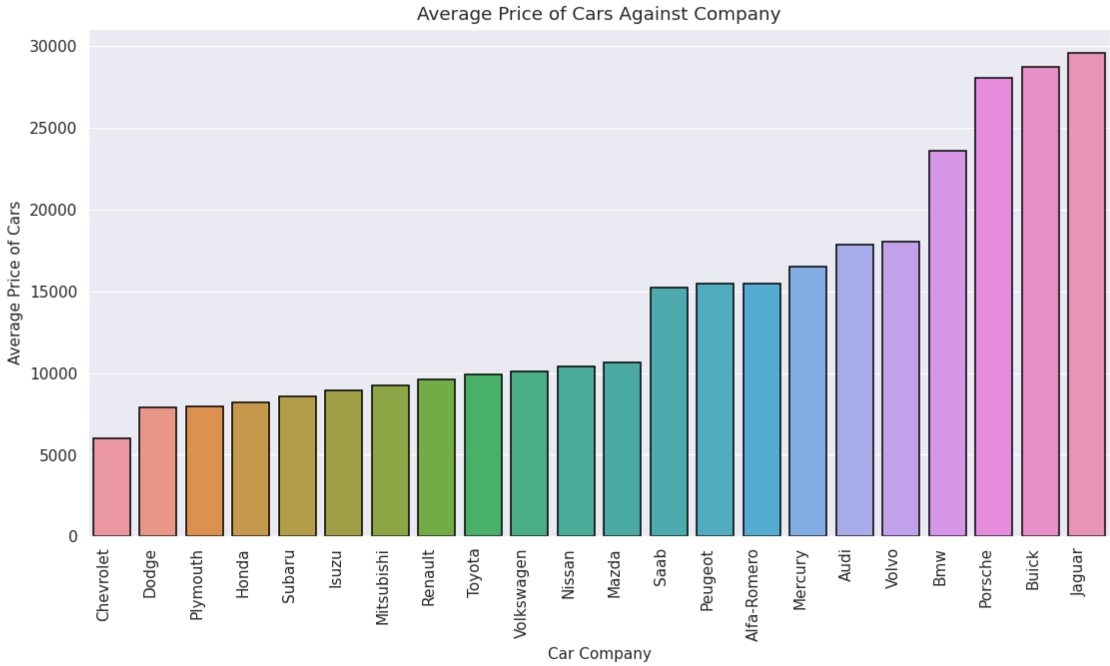
	fueltype	average price
1	gas	12517.190092
0	diesel	15736.925000

6.5.1 Visualizing The Above Data.

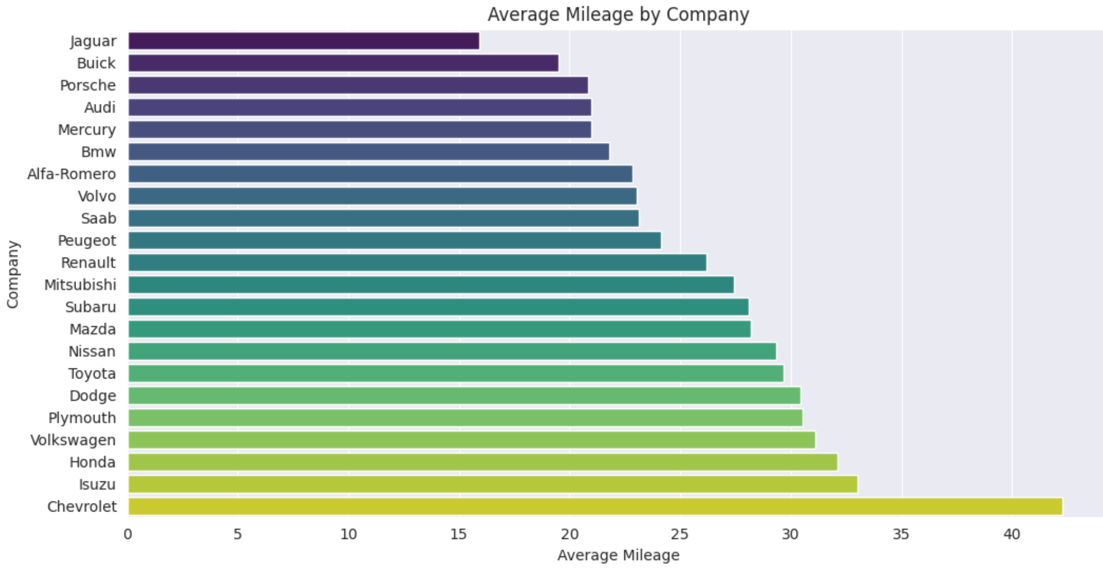
[166]:
```python
# Plotting
plt.figure(figsize=(10, 6), dpi=110)  # Setting the figure size & dpi
sns.set_style('darkgrid')  # Setting the plot style to darkgrid using Seaborn

# Creating a bar plot using Seaborn
# x-axis: 'symboling', y-axis: 'price', data source: 'cars_df_copy2'
# Additional parameters:
#   - edgecolor: Black border for bars
#   - hue: 'fueltype', differentiating bars by fuel type
#   - errorbar: None, no error bars displayed
#   - palette: 'Set1', color palette for the plot
sns.barplot(x='symboling', y='price', data=cars_df_copy2, edgecolor='black',␣
 ↪hue='fueltype', errorbar=None, palette='Set1')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price of Cars Against Safety Ratings')
plt.xlabel('Safety Rating')
plt.ylabel('Average Price of Cars')

# Displaying the plot
plt.show()
```



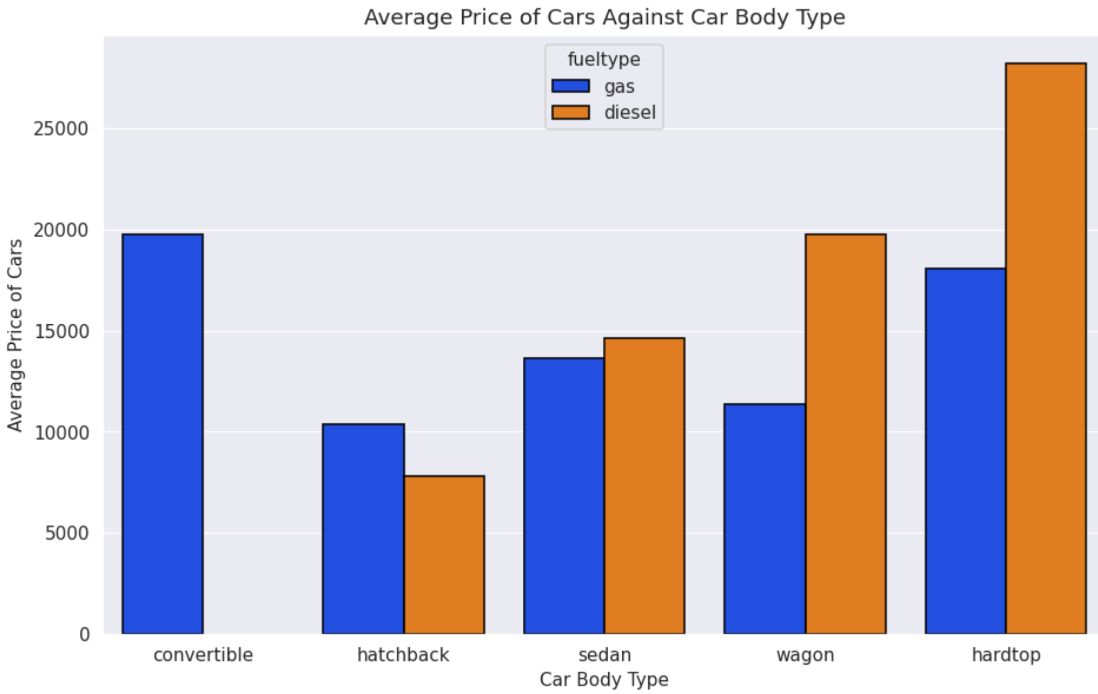
[167]:
```python
# Plotting
plt.figure(figsize=(12, 6), dpi=110)  # Setting the figure size & dpi
sns.set_style('darkgrid')  # Setting the plot style to darkgrid using Seaborn

# Creating a bar plot using Seaborn
# x-axis: 'company', y-axis: 'price', data source:␣
↪'average_price_against_company'
# Additional parameters:
#   - edgecolor: Black border for bars
sns.barplot(x='company', y='average price', data=average_price_against_company,␣
↪edgecolor='black')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price of Cars Against Company')
plt.xlabel('Car Company')
plt.ylabel('Average Price of Cars')

# Rotating x-axis labels for better visibility
plt.xticks(rotation=90, ha='right')

# Displaying the plot
plt.show()
```



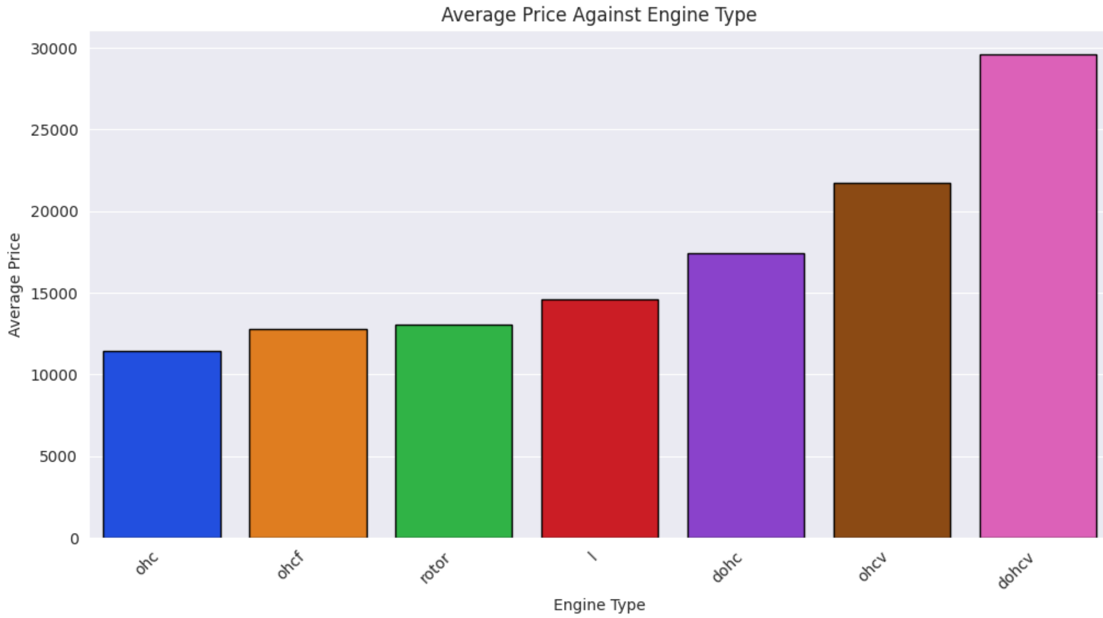
[168]:
```
# Plotting
plt.figure(figsize=(12, 6))  # Setting the figure size
# Creating a bar plot using Seaborn
# x-axis: 'average mileage', y-axis: 'company', data source:␣
↪'average_mileage_by_company'
# Additional parameter:
#   - palette: 'viridis', color palette for the plot
sns.barplot(x='average mileage', y='company', data=average_mileage_by_company,␣
↪palette='viridis')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Mileage by Company')
plt.xlabel('Average Mileage')
plt.ylabel('Company')

# Displaying the plot
plt.show()
```



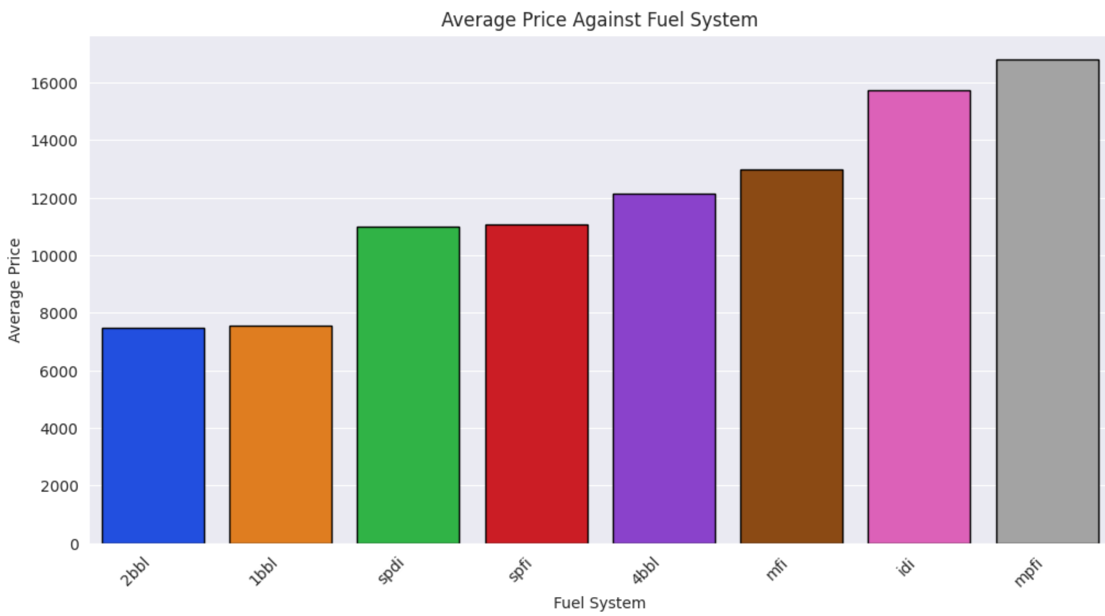
[169]:
```python
# Plotting
plt.figure(figsize=(10, 6), dpi=110)  # Setting the figure size
sns.set_style('darkgrid')  # Setting the plot style to darkgrid using Seaborn

# Creating a bar plot using Seaborn
# x-axis: 'carbody', y-axis: 'price', data source: 'cars_df_copy1'
# Additional parameters:
#   - hue: 'fueltype', differentiating bars by fuel type
#   - errorbar: None, no error bars displayed
#   - edgecolor: Black border for bars
sns.barplot(x='carbody', y='price', data=cars_df_copy2, hue='fueltype',␣
 ↪errorbar=None, edgecolor='black', palette='bright')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price of Cars Against Car Body Type')
plt.xlabel('Car Body Type')
plt.ylabel('Average Price of Cars')

# Displaying the plot
plt.show()
```



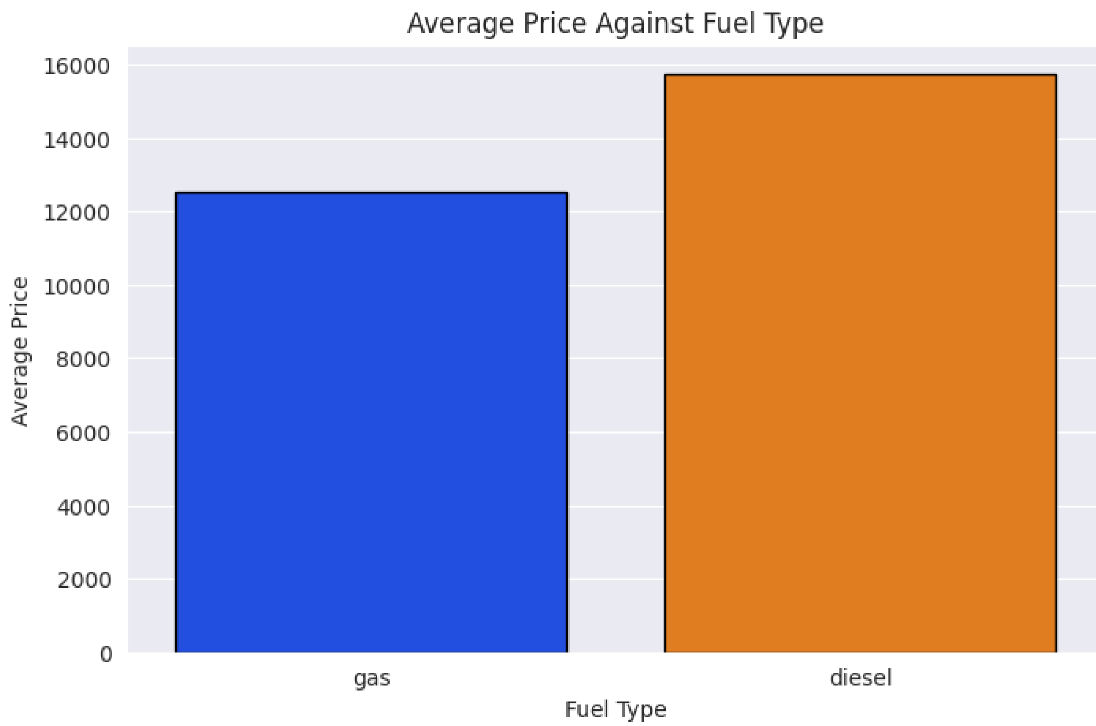
[170]:
```python
# Plotting
plt.figure(figsize=(12, 6))  # Setting the figure size
# Creating a bar plot using Seaborn
# x-axis: 'enginetype', y-axis: 'average price', data source:␣
↪'average_price_against_engine_type'
# Additional parameters:
#   - edgecolor: Black border for bars
#   - palette: 'bright', bright color palette for the plot
sns.barplot(x='enginetype', y='average price',␣
↪data=average_price_against_engine_type, edgecolor='black', palette='bright')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price Against Engine Type')
plt.xlabel('Engine Type')
plt.ylabel('Average Price')

# Rotating x-axis labels for better visibility
plt.xticks(rotation=45, ha='right')

# Displaying the plot
plt.show()
```



[171]:
```
# Plotting
plt.figure(figsize=(12, 6))  # Setting the figure size
# Creating a bar plot using Seaborn
# x-axis: 'fuelsystem', y-axis: 'average price', data source:␣
↪'average_price_against_fuel_system'
# Additional parameters:
#   - edgecolor: Black border for bars
#   - palette: 'bright', bright color palette for the plot
sns.barplot(x='fuelsystem', y='average price',␣
↪data=average_price_against_fuel_system, edgecolor='black', palette='bright')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price Against Fuel System')
plt.xlabel('Fuel System')
plt.ylabel('Average Price')

# Rotating x-axis labels for better visibility
plt.xticks(rotation=45, ha='right')

# Displaying the plot
plt.show()
```



[172]:
```
# Plotting
plt.figure(figsize=(8, 5))  # Setting the figure size
# Creating a bar plot using Seaborn
# x-axis: 'fueltype', y-axis: 'average price', data source:
 ↪'average_price_against_fuel_type'
# Additional parameters:
#   - edgecolor: Black border for bars
#   - palette: 'bright', bright color palette for the plot
sns.barplot(x='fueltype', y='average price',
 ↪data=average_price_against_fuel_type, edgecolor='black', palette='bright')

# Setting plot title, x-axis label, and y-axis label
plt.title('Average Price Against Fuel Type')
plt.xlabel('Fuel Type')
plt.ylabel('Average Price')

# Displaying the plot
plt.show()
```



6.6 *6. ML Model Implementation*

6.6.1 ML Model 1 - Linear Regression

[173]:
```
# Create a object to run Linear Regression
regressor = LinearRegression()

# Fit into the Algorithm
regressor.fit(x_train,y_train)
```

[173]:
```
LinearRegression()
```

[174]:
```
# Calculate and show the Coefficients of Independent variables
regressor.coef_
```

[174]:
```
array([-2.96366573e+03,  1.04938440e+03, -2.89222273e+01, -1.41493439e+03,
        2.01363127e+01,  8.88570310e+02, -1.08973681e+02,  5.25129440e+02,
        3.52717515e+00, -1.08474287e+04, -8.15859597e+02, -4.66495226e+02,
        2.15892426e+01,  9.11549531e+01, -2.63051185e+03, -1.21164820e+02,
        2.46858718e+01,  2.93855733e-03, -1.47523782e+03,  9.55096440e+03,
        1.79489569e+03])
```

[175]:
```
# Calculate and display the Intercept Value
regressor.intercept_
```

[175]: 203808.83025884215

[176]:
```
# Calculate and display the Predicted values for Training dataset
y_predict_train = regressor.predict(x_train)

# Calculate and display the Predicted value for the Test dataset
y_predict_test = regressor.predict(x_test)
```

[177]:
```
# Display the Predicted values for Training Data
y_predict_train
```

[177]:
```
array([16183.6130289 , 28069.53838862, 15209.20296173, 10646.00170657,
        7880.72751545,  7139.19172011, 14625.43118881, 10832.65106574,
       10289.83059088, 22254.83151042,  6044.84521341,  8802.73580508,
       10107.5576675 , 19316.7679723 , 18785.15704832,  9971.03356385,
       18695.26299279,  8790.57676719,  7061.59386682,  6395.00436652,
       13913.85107723,  7604.73159651, 10258.27736287,  9354.5708033 ,
       14733.3127523 ,  9856.18023438,  6414.53137363, 10064.64102861,
        6313.78114488,  6621.93675189,  8722.47882443,  6663.32029003,
        6034.67829072,  6509.48540278, 17214.06817249,  6246.77961549,
       16945.94337603,  7179.84570626, 17793.4458281 , 19638.50153521,
       20740.62200205,  9884.51372636, 17020.07353927, 10067.61855647,
       29523.76809782, 14179.22235905, 14950.39695877, 18491.46150522,
        6393.84108988, 10258.27736287, 17209.7273977 , 10107.5576675 ,
        9372.37302937, 17190.37315623,  9775.44985511, 12396.20026444,
       17895.89808234, 29543.69590315, 16969.46450804,  8711.89729899,
        5029.70606961,  6347.28695267, 11487.98399395,  6267.92786794,
        6481.28505975,  6009.23389441, 19138.77390472,  6197.27312031,
       13934.65931811,  6176.2361125 , 17214.06817249,  6402.37566697,
       10311.18499011,  8311.1828302 , 11038.60763469, 24049.01996631,
       32512.08631998, 28252.11088357,  9715.09322841, 18521.76288136,
       17716.89392304,  6213.84829943,  9668.19981142,  9365.87575679,
       14278.68281474,  6444.77604354, 23456.91137593,  7309.19426991,
        9135.80709983,  9960.05000222, 21293.1930906 , 20969.30489244,
       25086.50759671, 14643.06706455,  6970.75116071,  6270.60281237,
        8920.31021215,  6640.31226545,  6059.83933258, 10096.78616181,
       12396.20026444,  9248.8989446 , 15569.74932581,  7302.17804315,
        9372.20667905, 14685.3352563 , 13934.65931811,  7167.42243952,
        8926.18693531,  9982.89070123, 19864.29335055, 13561.1335623 ,
       10217.66715897, 29240.42041201,  9559.83557852, 13691.63904282,
        6678.062767  ,  7004.15665549, 29678.96380439,  7705.21328092,
        9281.29683484, 17987.44046131,  5649.42270966,  8333.33973766,
       13599.21339291, 29523.76809782, 18869.45645149,  6268.45446072,
       13613.32209351,  6855.17210469, 19146.00065405,  9294.48501326,
       14434.34668961,  9213.40495311, 14329.79215384, 10062.4714524 ,
        9076.19057822, 17139.93800924,  8923.33942081, 17395.33954422,
       26285.30992536,  7097.56376095,  7139.19172011,  9836.598872  ,
```

```
 9515.59460462, 19912.10029624,  8885.2444019 , 17014.66649686,
 9662.50756476, 11360.78162836,  7491.7683252 , 10942.23212102,
27041.2324798 , 10942.23212102,  7548.31789598,  6227.31513078,
19435.969594  , 14687.36670671, 17493.44916029, 24170.8617244 ,
11050.16414874, 18342.54853549, 28252.11088357, 15181.60924488])
```

[178]:
```python
# Display the Predicted values for Test Data
y_predict_test
```

[178]:
```
array([ 6680.95616577, 18819.46825352, 14102.66201594,  2318.71376799,
       10647.2095531 , 14103.59943926,  6486.65329014,  7172.35740617,
       19027.55571255,  7648.77847853, 17522.89928983, 28159.42284203,
        9354.5708033 , 13077.2998493 ,  6485.06821751, 13652.12102015,
       13636.92277518, 21357.29415886,  9372.89529561,  5561.24333092,
       10756.84219783, 17099.66538382, 12994.53817043, 14227.05056949,
       19905.91555845,  5144.11478822,  6178.13613344, 17969.66335559,
        6330.3519372 ,  5957.55125325, 10057.03703102, 12522.41867227,
       18711.22880055,  9692.91283857,  6130.38283556, 25354.31349701,
        9859.1661766 , 14141.12298392,  6746.51904771, 27935.71059694,
        6361.61708728])
```

[179]:
```python
# Create a DataFrame to display the Actual vs Predicted values for the Test
↪dataset for Linear Regression
df_actual_vs_predicted_linear_model = pd.DataFrame({'Actual Values': y_test,
↪'Predicted Values': y_predict_test})
df_actual_vs_predicted_linear_model
```

[179]:

	Actual Values	Predicted Values
0	6795.0	6680.956166
1	15750.0	18819.468254
2	15250.0	14102.662016
3	5151.0	2318.713768
4	9995.0	10647.209553
5	11199.0	14103.599439
6	5389.0	6486.653290
7	7898.0	7172.357406
8	17199.0	19027.555713
9	6529.0	7648.778479
10	20970.0	17522.899290
11	29575.5	28159.422842
12	10945.0	9354.570803
13	18344.0	13077.299849
14	8916.5	6485.068218
15	9989.0	13652.121020
16	9295.0	13636.922775
17	18920.0	21357.294159
18	7895.0	9372.895296

```
19         6488.0      5561.243331
20         9959.0     10756.842198
21        15580.0     17099.665384
22         9895.0     12994.538170
23        11549.0     14227.050569
24        15998.0     19905.915558
25         5118.0      5144.114788
26         6938.0      6178.136133
27        16695.0     17969.663356
28         8358.0      6330.351937
29         5499.0      5957.551253
30         7975.0     10057.037031
31        12290.0     12522.418672
32        22018.0     18711.228801
33         8948.0      9692.912839
34         6849.0      6130.382836
35        29575.5     25354.313497
36        11595.0      9859.166177
37        18150.0     14141.122984
38         6377.0      6746.519048
39        29575.5     27935.710597
40         8916.5      6361.617087
```

[180]:

```
# Calculate the r^2 score for Training data & display
r_squared_train_for_linear_regression = round(r2_score(y_train,␣
 ↪y_predict_train),4)
print('R Squared for Training data :', r_squared_train_for_linear_regression)

# Calculate the r^2 score for Test data & display
r_squared_test_for_linear_regression = round(r2_score(y_test, y_predict_test),4)
print('R Squared for Test data :', r_squared_test_for_linear_regression)

# Number of observations (n)
n = len(y_test)

# Number of predictors (k) - you need to replace this with the actual number of␣
 ↪predictors in your model
k = 19

# Calculate adjusted R Squared for Test data & display
adjusted_r_squared_test_for_linear_regression =  round(1 - ((1 -␣
 ↪r_squared_test_for_linear_regression) * (n - 1) / (n - k - 1)),4)

print("Adjusted R-squared for test data:",␣
 ↪adjusted_r_squared_test_for_linear_regression)
```

```
R Squared for Training data : 0.9103
```

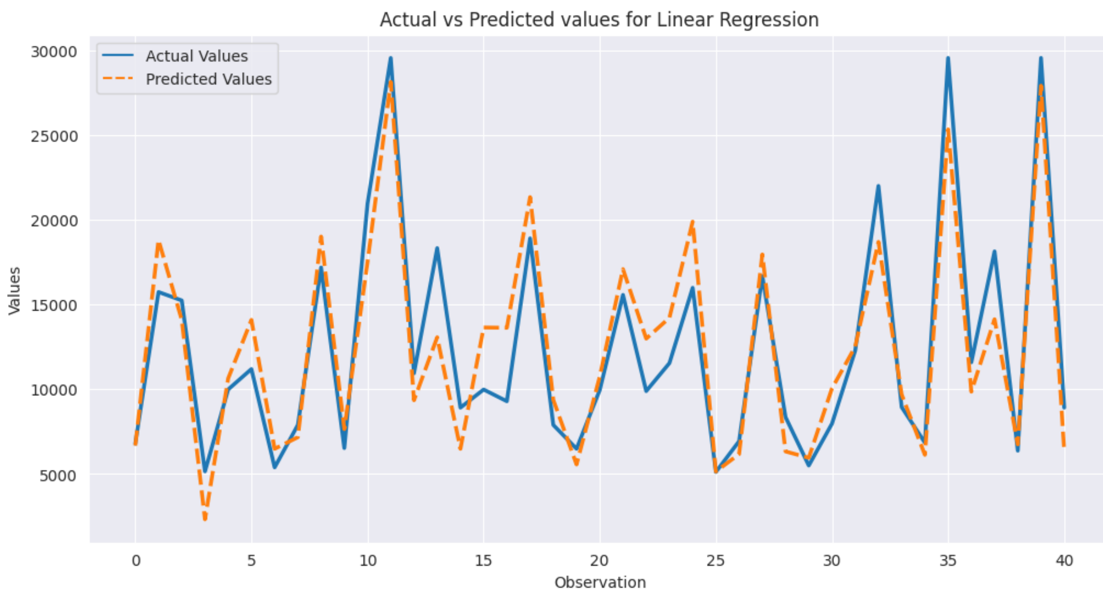
```
R Squared for Test data : 0.8723
Adjusted R-squared for test data: 0.7568
```

[181]:
```python
# Calculating Mean Squared Error (MSE) for Linear Regression
mse_lr = round(mean_squared_error(y_test, y_predict_test), 4)

# Calculating Root Mean Squared Error (RSME) for Linear Regression
rsme_lr = round(math.sqrt(mean_squared_error(y_test, y_predict_test)), 4)

# Calculating Mean Absolute Error (MAE) for Linear Regression
mae_lr = round(mean_absolute_error(y_test, y_predict_test), 4)

# Calculating Mean Absolute Percentage Error (MAPE) for Linear Regression
mape_lr = round(mean_absolute_percentage_error(y_test, y_predict_test), 4)
```

[182]:
```python
# Performance of Linear Regression Model
print('Performance of Linear Regression Model')
print("MSE :", round(mean_squared_error(y_test, y_predict_test), 4))
print("RMSE :", round(math.sqrt(mean_squared_error(y_test, y_predict_test)), 4))
print('MAE:', round(mean_absolute_error(y_test, y_predict_test), 4))
print('MAPE:', round(mean_absolute_percentage_error(y_test, y_predict_test), 4))
```

```
Performance of Linear Regression Model
MSE : 5596513.4284
RMSE : 2365.6951
MAE: 1967.1952
MAPE: 0.1698
```

[183]:
```python
# Plot the lineplot for Linear Regression
plt.figure(figsize=(12, 6), dpi = 100)
sns.set_style('darkgrid')
sns.lineplot(data=df_actual_vs_predicted_linear_model, palette="tab10",␣
 ↪linewidth=2.5)
plt.title('Actual vs Predicted values for Linear Regression')
plt.xlabel('Observation')
plt.ylabel('Values')
plt.show()
```



6.6.2 ML Model - 2 Lasso Regression

Cross- Validation & Hyperparameter Tuning (Lasso Regression)

[184]:
```
# Creating an object to run Lasso Regression
lasso = Lasso(max_iter = 6000)

min_alpha = 0.001
max_alpha = 0.2
num_vals = 10

parameters = {'alpha': np.logspace(np.log10(min_alpha), np.log10(max_alpha),␣
 ↪num_vals)}

# Applying the GridSearchCV with cross validation and hyperparameter tuning
lasso_cv_hype = GridSearchCV(lasso, parameters, cv = 5, scoring =␣
 ↪"neg_mean_squared_error")

# Fit into the Algorithm
lasso_cv_hype.fit(x_train, y_train)
```

```
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.980e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.034e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.261e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.634e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.980e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.034e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.261e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.634e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.980e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.034e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.261e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.634e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.980e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.034e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.261e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.635e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.981e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.034e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.261e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.635e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.981e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.035e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.262e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.485e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.635e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.981e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.036e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.262e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.485e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.636e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.982e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.037e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.263e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.486e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.637e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.984e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.039e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.265e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.488e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.639e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.986e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.043e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.269e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.491e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
```

```
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.642e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.333e+08, tolerance: 7.399e+05
  model = cd_fast.enet_coordinate_descent(
```

[184]:
```
GridSearchCV(cv=5, estimator=Lasso(max_iter=6000),
             param_grid={'alpha': array([0.001     , 0.00180165, 0.00324594,
0.00584804, 0.0105361 ,
       0.01898235, 0.03419952, 0.0616155 , 0.11100946, 0.2       ])},
             scoring='neg_mean_squared_error')
```

[185]:
```
# First fit the best alpha value into the model
lasso_alpha = Lasso(alpha = lasso_cv_hype.best_params_['alpha'])
lasso_alpha.fit(x_train,y_train)
```

```
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.421e+08, tolerance: 7.399e+05
  model = cd_fast.enet_coordinate_descent(
```

[185]:
```
Lasso(alpha=0.20000000000000004)
```

[186]:
```
# Calculate and show the Coefficients
print('Coefficients :', lasso_alpha.coef_)
```

```
Coefficients : [ 2.35579445e+02  1.03038506e+03 -2.92453766e+02 -2.75391196e+02
  2.18739705e+01  1.09608301e+03  1.01826338e+02  5.10714375e+02
  3.94187491e+00 -1.29633380e+04 -5.42389285e+02 -2.79123974e+02
  3.83184875e+00  1.56483885e+02 -2.82519292e+03 -6.84427862e+01
  1.84083037e+01  3.33364696e-03 -1.70132558e+03  9.76927014e+03
  2.11667880e+03]
```

[187]:
```
# Calculate and show the Intercept value
print('Intercept :', lasso_alpha.intercept_)
```

```
Intercept : -7634.513956755023
```

[188]:
```
# Calculate Predicted values for Training dataset
y_train_pred_for_lasso_cv_hype = lasso_alpha.predict(x_train)
```

```python
# Calculate the Predicted values for Test data
y_test_pred_for_lasso_cv_hype = lasso_alpha.predict(x_test)
```

[189]:
```python
# Show the Predicted Values for Training
print('Predicted Valu/es for Training Data :', y_train_pred_for_lasso_cv_hype)
```

```
Predicted Valu/es for Training Data : [16472.44187689 27066.45118204
15276.5119111  10571.76895374
  7860.94049905  6985.40025404 14695.33926603 10969.09674638
 10419.55217016 21747.148711    5992.02416909  8675.11940083
 10312.67060528 19899.73432115 19063.63059583 10112.45735868
 18801.47925423  8877.43102097  6898.67900602  5907.04848234
 13776.22195082  7427.01637775 10482.16635288  9265.94110837
 14860.92553034  9992.87361874  6383.04726947 10055.34232855
  6200.67369161  6492.70675843  8966.54443178  6517.80102547
  5626.49867214  6812.0757814  17473.2702254   6071.41554903
 17239.76187051  7493.30795629 17734.92018304 19986.24528715
 21110.85734677  9559.9016493  17256.46710535 10171.21405083
 29506.6849237  14429.61323854 15278.22096484 18441.95871
  6306.61580563 10482.16635288 17660.56331519 10312.67060528
  9931.09188212 17327.80900485  9473.15522059 12660.87457242
 17722.63309009 27888.64512174 16969.69449794  8954.71880705
  4821.62185299  6190.80155414 11790.80675783  6149.42931778
  6515.32145449  5909.50175108 19297.95297701  6275.51154675
 13440.61102348  5992.57805083 17473.2702254   6248.03897594
 10541.29447653  8496.84158107 11192.02227371 24143.24534966
 33125.22854393 27909.83331498  9835.19862233 18552.58721213
 18210.92005865  6144.12507522  9318.2546528   9100.63980507
 14258.64937382  6461.09721322 23575.22033678  7354.07366184
  9252.1995691   9808.02466169 22116.22921661 20168.04306222
 25115.87153384 14715.04864058  6917.41533759  6155.63297617
  9375.32156481  6540.72061533  5862.4961788  10325.12924647
 12660.87457242  8934.01186812 15669.92691518  7323.10024187
  9285.65048292 14951.11506968 13440.61102348  7025.15059662
  8813.08502268 10449.97913106 20250.34391733 13382.03445981
 10459.64341348 28292.12393769  9715.86404293 13527.88383149
  6558.06366292  6673.52260958 29680.12741975  7489.30962915
  9821.63354733 17951.72330309  5293.5062814   7484.85788115
 13354.68855525 29506.6849237  19262.2510301   6157.78669457
 13370.45605489  7126.0343244  19051.64150091  9377.47869829
 14497.66295131  9338.92081712 14365.46752975 10286.20091744
  8669.51892016 17456.56499056  8875.29629586 17446.45884617
 27006.9520704   7117.56116724  6985.40025404  9670.05903983
  9598.24758621 19907.20339282  9324.6911679  16952.67729679
  9695.13060886 11251.12442879  6921.67909603 11030.21106306
 25720.43355531 11030.21106306  7518.53962812  6223.29111766
```

```
 19828.8809122  14669.93621242 17488.56976867 24091.10635689
 11539.78823027 18598.63176246 27909.83331498 15123.70433257]
```

[190]:
```python
# Show the Predicted Values for Test Data
print('Predicted Values for Test Data:', y_test_pred_for_lasso_cv_hype)
```

```
Predicted Values for Test Data: [ 6537.51040002 19230.65141305 14440.60621848
-409.16184425
 10840.84976471 13918.37666739  6691.70402521  6823.0049023
 19367.16007484  7426.23963059 17994.1169386  29248.39625902
  9265.94110837 13549.97146082  6457.77709602 13413.8166789
 13406.5150757  20601.64930233  9506.9446727   5194.95940864
 10809.02752658 17412.81704955 12886.67962628 14056.34228924
 20217.56620533  4737.18578431  6052.40096379 17804.98133546
  6235.66205725  5748.18180641 10159.3884261  12632.64280894
 19156.5323485   9909.95716837  5941.333677   25061.17625389
  9671.36544475 14104.91625233  6546.65951204 27158.63275271
  6319.81147417]
```

[191]:
```python
# Create a DataFrame to display the Actual vs Predicted values for the Test␣
↪dataset for Lasso Regression
df_actual_vs_predicted_lasso_model = pd.DataFrame({'Actual Values': y_test,␣
↪'Predicted Values': y_test_pred_for_lasso_cv_hype})
df_actual_vs_predicted_lasso_model
```

[191]:

	Actual Values	Predicted Values
0	6795.0	6537.510400
1	15750.0	19230.651413
2	15250.0	14440.606218
3	5151.0	-409.161844
4	9995.0	10840.849765
5	11199.0	13918.376667
6	5389.0	6691.704025
7	7898.0	6823.004902
8	17199.0	19367.160075
9	6529.0	7426.239631
10	20970.0	17994.116939
11	29575.5	29248.396259
12	10945.0	9265.941108
13	18344.0	13549.971461
14	8916.5	6457.777096
15	9989.0	13413.816679
16	9295.0	13406.515076
17	18920.0	20601.649302
18	7895.0	9506.944673
19	6488.0	5194.959409
20	9959.0	10809.027527

```
21         15580.0      17412.817050
22          9895.0      12886.679626
23         11549.0      14056.342289
24         15998.0      20217.566205
25          5118.0       4737.185784
26          6938.0       6052.400964
27         16695.0      17804.981335
28          8358.0       6235.662057
29          5499.0       5748.181806
30          7975.0      10159.388426
31         12290.0      12632.642809
32         22018.0      19156.532348
33          8948.0       9909.957168
34          6849.0       5941.333677
35         29575.5      25061.176254
36         11595.0       9671.365445
37         18150.0      14104.916252
38          6377.0       6546.659512
39         29575.5      27158.632753
40          8916.5       6319.811474
```

[192]:

```
# Calculate R-squared for Training data & Display
r_squared_train_for_lasso_cv_hype = round((r2_score(y_train,␣
 ↪y_train_pred_for_lasso_cv_hype)),4)
print("R-squared for training data: ", r_squared_train_for_lasso_cv_hype)

# Calculate R-squared for Test data & display
r_squared_test_for_lasso_cv_hype = round((r2_score(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype)),4)
print("R-squared for test data: ", r_squared_test_for_lasso_cv_hype)

# Number of observations (n)
n = len(y_test)

# Number of predictors (k) - you need to replace this with the actual number of␣
 ↪predictors in your model
k = 19

# Calculate adjusted R Squared for Test data & display
adjusted_r_squared_test_for_lasso_cv_hype = round(1 - ((1 -␣
 ↪r_squared_test_for_lasso_cv_hype) * (n - 1) / (n - k - 1)),4)

print("Adjusted R-squared for test data:",␣
 ↪adjusted_r_squared_test_for_lasso_cv_hype)
```

```
R-squared for training data:  0.9078
R-squared for test data:  0.8617
```

Adjusted R-squared for test data: 0.7366

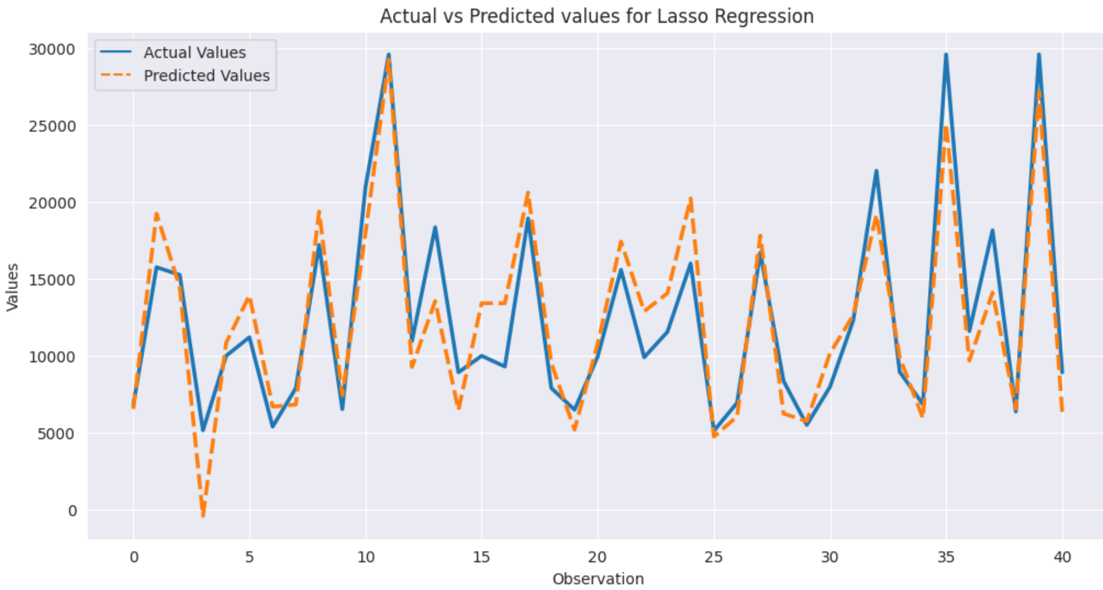
[193]:
```
# Calculating Mean Squared Error (MSE) for Lasso Regression with␣
 ↪Cross-Validation Hyperparameter Tuning
mse_lasso_cv_hype = round(mean_squared_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype), 4)

# Calculating Root Mean Squared Error (RSME) for Lasso Regression with␣
 ↪Cross-Validation Hyperparameter Tuning
rsme_lasso_cv_hype = round(np.sqrt(mean_squared_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype)), 4)

# Calculating Mean Absolute Error (MAE) for Lasso Regression with␣
 ↪Cross-Validation Hyperparameter Tuning
mae_lasso_cv_hype = round(mean_absolute_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype), 4)

# Calculating Mean Absolute Percentage Error (MAPE) for Lasso Regression with␣
 ↪Cross-Validation Hyperparameter Tuning
mape_lasso_cv_hype = round(mean_absolute_percentage_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype), 4)
```

[194]:
```
_# Performance of Lasso Regression Model with Cross Validation & Hyperparameter␣
 ↪Tuning
print('Performance of Lasso Regression Model')
print("MSE :",round(mean_squared_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype),4))
print("RMSE :",round(np.sqrt(mean_squared_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype)),4))
print('MAE:',  round(mean_absolute_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype),4))
print('MAPE:', round(mean_absolute_percentage_error(y_test,␣
 ↪y_test_pred_for_lasso_cv_hype),4))
```

```
Performance of Lasso Regression Model
MSE : 6060866.9478
RMSE : 2461.8828
MAE: 2037.7442
MAPE: 0.1865
```

[195]:
```
# Plot the lineplot for Lasso Regression
plt.figure(figsize=(12, 6), dpi = 100)
sns.set_style('darkgrid')
sns.lineplot(data=df_actual_vs_predicted_lasso_model, palette="tab10",␣
 ↪linewidth=2.5)
plt.title('Actual vs Predicted values for Lasso Regression')
plt.xlabel('Observation')
```

```
plt.ylabel('Values')
plt.show()
```



6.6.3 ML Model - 3 Ridge Regression

Cross- Validation & Hyperparameter Tuning (Ridge Regression)

[196]:
```
#Creating an object to run Ridge Regression
ridge = Ridge(max_iter = 6000)

min_alpha = 0.001
max_alpha = 0.2
num_vals = 10

parameters = {'alpha': np.logspace(np.log10(min_alpha), np.log10(max_alpha),␣
↪num_vals)}

# Applying the GridSearchCV with cross validation and hyperparameter tuning
ridge_cv_hype = GridSearchCV(ridge, parameters, cv = 5, scoring =␣
↪"neg_mean_squared_error")

# Fit into the Algorithm
ridge_cv_hype.fit(x_train, y_train)
```

[196]:
```
GridSearchCV(cv=5, estimator=Ridge(max_iter=6000),
             param_grid={'alpha': array([0.001     , 0.00180165, 0.00324594,
0.00584804, 0.0105361 ,
       0.01898235, 0.03419952, 0.0616155 , 0.11100946, 0.2       ])},
```

```
                scoring='neg_mean_squared_error')
```

[197]:
```python
# First fit the best alpha value into the model
ridge_alpha = Ridge(alpha = ridge_cv_hype.best_params_['alpha'])
ridge_alpha.fit(x_train, y_train)
```

[197]:
```
Ridge(alpha=0.20000000000000004)
```

[198]:
```python
# Calculate and display the Coefficients
print("Coefficients:", ridge_alpha.coef_)
```

```
Coefficients: [-3.10864887e+03  9.45917393e+02 -3.56752771e+01 -1.48414577e+03
  2.26968954e+01  9.62089678e+02 -1.41723291e+02  4.05842724e+02
  3.37846066e+00 -5.70237242e+03 -8.13420794e+02 -5.49494851e+02
  2.24612310e+01  1.23396744e+02 -2.94129072e+03 -1.41611726e+02
  2.57817639e+01  5.52984298e-03 -1.34482906e+03  8.47244402e+03
  1.80204927e+03]
```

[199]:
```python
# Calculate and display the Intercept
print("Intercept :", ridge_alpha.intercept_)
```

```
Intercept : 214895.7996472018
```

[200]:
```python
# Calculate the Precited values for the Training data
y_train_pred_for_ridge_cv_hype = ridge_alpha.predict(x_train)

# Calculate and Predicted values for Test data
y_test_pred_for_ridge_cv_hype = ridge_alpha.predict(x_test)
```

[201]:
```python
# Show the Predicted Values for Training Data
print('Predicted Value for Training Data :',y_train_pred_for_ridge_cv_hype)
```

```
Predicted Value for Training Data : [16011.19261234 27987.46757687
14959.56326337 10705.76517218
  7545.70137128  6922.23401456 14763.77220155 10746.0794175
 10237.70670735 22204.33783825  5930.10296356  9124.00947912
 10143.9934671  19173.63512634 18872.54121361 10013.17265367
 18705.88328554  8792.99210767  6847.90788006  6297.63245063
 13787.03569232  7512.75692313 10346.1447515   9430.18052186
 14875.93443175  9799.20989541  5991.75839174 10350.14771033
  6317.72420053  6695.80119196  8656.62645722  6770.66814512
  6146.37875269  6473.35438046 17198.45393738  6311.06098045
 16849.64652941  7135.51564897 17573.01696886 20301.03012997
 20677.17327014  9726.16559008 17012.63860114 10024.86368584
 28961.11964256 13889.49877719 14955.83194368 18340.76185671
  6752.84042475 10346.1447515  16972.7333882  10143.9934671
  9187.68586238 17198.15952363  9415.32471429 12151.28449105
 17715.9479957  29596.197609   16826.08898221  8646.49107524
```

```
 5321.15827295  6374.82765984 11254.59570913  6273.80421196
 6097.70457729  6396.63750574 19175.47772051  6136.12556298
13819.25348991  6243.49176727 17198.45393738  6423.95660548
10396.82166138  8233.40351474 10902.51825448 24017.49186319
32620.88453538 28097.43506738  9664.07146905 18398.37002592
17695.85877922  6625.72390548  9524.42178184  9217.09491263
14384.64971736  6001.76728119 23669.76727175  7205.52237085
 9286.46353393 10405.78836786 21112.94882966 20958.28598065
25088.83768237 14780.66450484  6656.99820733  6649.23112911
 8769.08558686  6893.79290553  6132.00256553 10409.37792438
12151.28449105  9174.21195991 15709.27478701  7219.39750977
 9447.07282516 14806.86773715 13819.25348991  6903.54078176
 9242.25560218  9909.48476883 20201.56258985 13449.18962643
10493.45600499 29490.08608302  9570.87477023 13574.19267081
 6746.1503419   7429.82742762 29109.77191156  7698.95033806
 9145.50125132 17758.8323051   6050.36080613  8556.08147374
14041.39584765 28961.11964256 19473.93975376  6166.58566884
14054.90969029  6841.39240715 19037.52645953  9300.92071693
14489.95904183  9360.78966842 14479.91595421 10344.803736
 9163.87529433 17035.46186565  9010.24572388 17278.79111388
26290.09472942  7002.81473132  6922.23401456 10287.5422448
 9403.51175458 19918.66611599  9159.42584832 16911.51284462
 9668.5377274  11434.07906472  7649.12950688 11018.1526108
26946.65964265 11018.1526108   7589.69407983  5900.17632791
19380.92903744 14845.19725689 17414.89188617 24131.63305875
10838.15106226 18387.12490825 28097.43506738 15588.45798353]
```

[202]:
```
# Show the Predicted values for Test data
print('Predicted Value for Test Data :',y_test_pred_for_ridge_cv_hype)
```

```
Predicted Value for Test Data : [ 6787.56044842 19446.81978593 13969.27418002
1937.21997807
 10607.49684597 14524.51572188  6263.58390872  7579.80997001
 19296.40123121  7644.89496752 17510.04344298 28805.96047373
  9430.18052186 13054.00897056  6082.18806395 14092.07275754
 13442.89799158 21329.91665313  9391.81635531  5965.89928966
 10771.10673896 16990.06807482 12686.29773353 14642.76184494
 20599.29701389  5209.79367395  6566.47884465 17705.90516804
  6692.02813289  6034.02720642 10014.72830386 12365.1239224
 18768.23300226 10004.06973787  6199.57177871 25522.79189062
  9851.04230996 14252.88975166  6551.56717337 27985.2417003
  5963.94194089]
```

[203]:
```
# Create a DataFrame to display the Actual vs Predicted values for the Test␣
↪dataset for Ridge Regression
df_actual_vs_predicted_ridge_model = pd.DataFrame({'Actual Values': y_test,␣
↪'Predicted Values': y_test_pred_for_ridge_cv_hype})
```

```
df_actual_vs_predicted_ridge_model
```

[203]:

	Actual Values	Predicted Values
0	6795.0	6787.560448
1	15750.0	19446.819786
2	15250.0	13969.274180
3	5151.0	1937.219978
4	9995.0	10607.496846
5	11199.0	14524.515722
6	5389.0	6263.583909
7	7898.0	7579.809970
8	17199.0	19296.401231
9	6529.0	7644.894968
10	20970.0	17510.043443
11	29575.5	28805.960474
12	10945.0	9430.180522
13	18344.0	13054.008971
14	8916.5	6082.188064
15	9989.0	14092.072758
16	9295.0	13442.897992
17	18920.0	21329.916653
18	7895.0	9391.816355
19	6488.0	5965.899290
20	9959.0	10771.106739
21	15580.0	16990.068075
22	9895.0	12686.297734
23	11549.0	14642.761845
24	15998.0	20599.297014
25	5118.0	5209.793674
26	6938.0	6566.478845
27	16695.0	17705.905168
28	8358.0	6692.028133
29	5499.0	6034.027206
30	7975.0	10014.728304
31	12290.0	12365.123922
32	22018.0	18768.233002
33	8948.0	10004.069738
34	6849.0	6199.571779
35	29575.5	25522.791891
36	11595.0	9851.042310
37	18150.0	14252.889752
38	6377.0	6551.567173
39	29575.5	27985.241700
40	8916.5	5963.941941

[204]:

```
# Calculate R-squared for Training data & dsiplay
```

```
r_squared_train_for_ridge_cv_hype= round(r2_score(y_train,
    y_train_pred_for_ridge_cv_hype), 4)
print("R-squared for training data: ", r_squared_train_for_ridge_cv_hype)

# Calculate R-squared for Test data & display
r_squared_test_for_ridge_cv_hype = round(r2_score(y_test,
    y_test_pred_for_ridge_cv_hype), 4)
print("R-squared for test data: ", r_squared_test_for_ridge_cv_hype)

# Number of observations (n)
n = len(y_test)

# Number of predictors (k) - you need to replace this with the actual number of
    predictors in your model
k = 19

# Calculate adjusted R Squared for Test data & display
adjusted_r_squared_test_for_ridge_cv_hype = round(1 - ((1 -
    r_squared_test_for_ridge_cv_hype) * (n - 1) / (n - k - 1)),4)

print("Adjusted R-squared for test data:",
    adjusted_r_squared_test_for_ridge_cv_hype)
```

```
R-squared for training data:  0.9092
R-squared for test data:  0.864
Adjusted R-squared for test data: 0.741
```

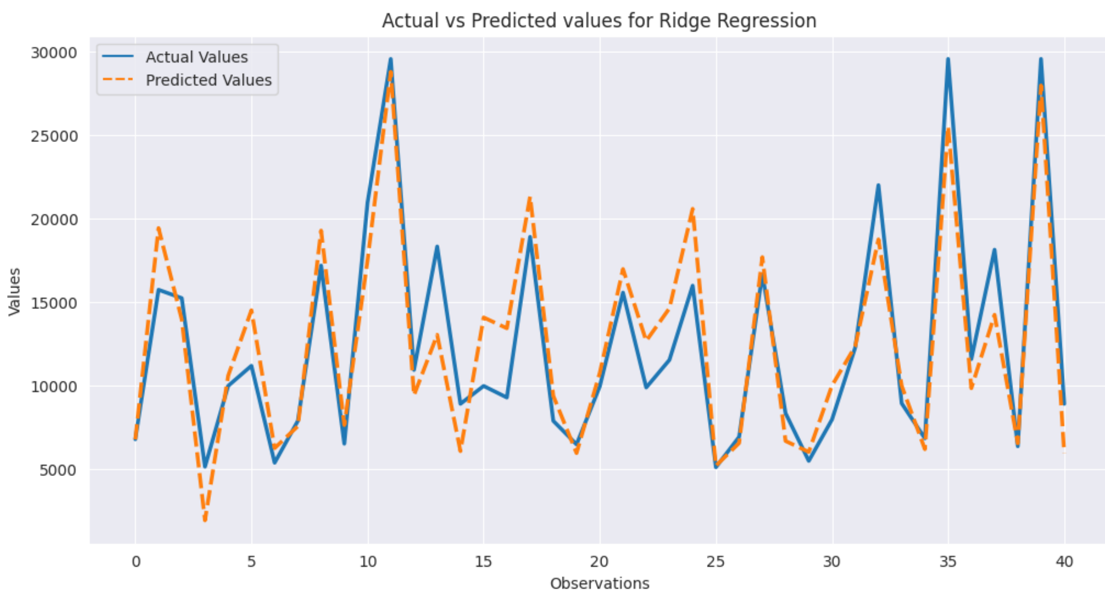
[205]:

```
# Calculating Mean Squared Error (MSE) for Ridge Regression with
    Cross-Validation Hyperparameter Tuning
mse_ridge_cv_hype = round(mean_squared_error(y_test,
    y_test_pred_for_ridge_cv_hype), 4)

# Calculating Root Mean Squared Error (RSME) for Ridge Regression with
    Cross-Validation Hyperparameter Tuning
rsme_ridge_cv_hype = round(math.sqrt(mean_squared_error(y_test,
    y_test_pred_for_ridge_cv_hype)), 4)

# Calculating Mean Absolute Error (MAE) for Ridge Regression with
    Cross-Validation Hyperparameter Tuning
mae_ridge_cv_hype = round(mean_absolute_error(y_test,
    y_test_pred_for_ridge_cv_hype), 4)

# Calculating Mean Absolute Percentage Error (MAPE) for Ridge Regression with
    Cross-Validation Hyperparameter Tuning
mape_ridge_cv_hype = mean_absolute_percentage_error(y_test,
    y_test_pred_for_ridge_cv_hype)
```

[206]:
```
# Performance of Ridge Regression Model
print('Performance of Ridger Regression Model with Cross Validation &␣
↪Hyperparameter Tunning')
print("MSE :",round(mean_squared_error(y_test,␣
↪y_test_pred_for_ridge_cv_hype),4))
print("RMSE :",round(math.sqrt(mean_squared_error(y_test,␣
↪y_test_pred_for_ridge_cv_hype)),4))
print('MAE:', round(mean_absolute_error(y_test,␣
↪y_test_pred_for_ridge_cv_hype),4))
print('MAPE:', round(mean_absolute_percentage_error(y_test,␣
↪y_test_pred_for_ridge_cv_hype),4))
```

```
Performance of Ridger Regression Model with Cross Validation & Hyperparameter
Tunning
MSE : 5961545.047
RMSE : 2441.6275
MAE: 1974.5439
MAPE: 0.1698
```

[207]:
```
# Plot the lineplot for Ridge Regression
plt.figure(figsize=(12, 6), dpi = 100)
sns.set_style('darkgrid')
sns.lineplot(data=df_actual_vs_predicted_ridge_model, palette="tab10",␣
↪linewidth=2.5)
plt.title('Actual vs Predicted values for Ridge Regression')
plt.xlabel('Observations')
plt.ylabel('Values')
plt.show()
```



6.6.4 ML Model - 4 ElasticNet Regression

Cross- Validation & Hyperparameter Tuning (ElasticNet Regression)

[208]:
```
# Creating a variable to run Elastic Net Regression
elastic_net = ElasticNet(alpha=0.2, l1_ratio=0.5, max_iter = 6000)

min_alpha = 0.001
max_alpha = 0.2
num_vals = 10

parameters = {'alpha': np.logspace(np.log10(min_alpha), np.log10(max_alpha),␣
↪num_vals), 'l1_ratio': [0.1, 0.3, 0.5, 0.7, 0.9]}

# Applying GridSearchCV with cross-validation and hyperparameter tuning
elastic_net_cv_hype = GridSearchCV(elastic_net, parameters, cv=5,␣
↪scoring="neg_mean_squared_error")

# Fit the model
elastic_net_cv_hype.fit(x_train, y_train)
```

```
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.084e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.199e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.388e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.545e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.752e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.065e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.171e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.363e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.533e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.730e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.044e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.140e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.336e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.520e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.707e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.021e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.103e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.308e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.507e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.681e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.995e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.060e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.277e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.492e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.651e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.145e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.280e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.470e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.586e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.817e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.116e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.243e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.431e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.566e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.786e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.084e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.199e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.388e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.546e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.752e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.048e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.147e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.342e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.523e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.712e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.006e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.078e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.290e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.498e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.664e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.232e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.389e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.593e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.647e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.908e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.191e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.339e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.535e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.618e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.866e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.145e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.280e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.470e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.586e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.817e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.091e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.209e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.397e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.550e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.759e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.024e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.108e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.312e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.508e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.684e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.353e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.531e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.767e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.735e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.033e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.297e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.466e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.686e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.694e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.975e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.232e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.389e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.593e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.647e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.908e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.155e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.293e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.484e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.593e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.827e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.053e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.154e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.348e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.526e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.717e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.514e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.714e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.997e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.852e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.196e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.440e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.631e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.892e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.798e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.122e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.353e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.531e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.767e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.735e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.033e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.246e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.406e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.613e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.657e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.923e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.098e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.219e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.407e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.555e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.767e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.713e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.939e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.276e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.996e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.397e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.623e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.838e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.151e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.931e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.307e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.514e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.714e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.997e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.852e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.196e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.372e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.553e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.795e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.749e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.053e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.165e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.307e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.499e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.600e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.839e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.914e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.185e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.578e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.149e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.618e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.823e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.076e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.446e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.081e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.519e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.706e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.936e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.274e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.993e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.394e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.536e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.741e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.031e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.869e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.219e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.261e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.424e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.635e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.668e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.939e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.940e+07, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.192e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.239e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.373e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.297e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.031e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.567e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.891e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.523e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.907e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.681e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.092e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.484e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.883e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.512e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.715e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.960e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.305e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.006e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.412e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.391e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.576e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.824e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.763e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.073e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 6.161e+06, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 5.926e+06, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 8.984e+06, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 6.574e+06, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.959e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.796e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.184e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.371e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.174e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.553e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.763e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.060e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
```

```
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.881e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.237e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.599e+08, tolerance: 6.361e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.928e+08, tolerance: 6.178e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.283e+08, tolerance: 5.247e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 2.896e+08, tolerance: 5.872e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.365e+08, tolerance: 5.900e+05
  model = cd_fast.enet_coordinate_descent(
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.913e+08, tolerance: 7.399e+05
  model = cd_fast.enet_coordinate_descent(
```

[208]:
```
GridSearchCV(cv=5, estimator=ElasticNet(alpha=0.2, max_iter=6000),
             param_grid={'alpha': array([0.001     , 0.00180165, 0.00324594,
0.00584804, 0.0105361 ,
       0.01898235, 0.03419952, 0.0616155 , 0.11100946, 0.2       ]),
                         'l1_ratio': [0.1, 0.3, 0.5, 0.7, 0.9]},
             scoring='neg_mean_squared_error')
```

[209]:
```python
# Fit the best alpha and l1_ratio values into the model
elastic_net_best = ElasticNet(alpha=elastic_net_cv_hype.best_params_['alpha'],
 ↪l1_ratio=elastic_net_cv_hype.best_params_['l1_ratio'])
elastic_net_best.fit(x_train, y_train)
```

```
/usr/local/lib/python3.10/dist-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 3.991e+08, tolerance: 7.399e+05
  model = cd_fast.enet_coordinate_descent(
```

[209]:
```
ElasticNet(alpha=0.010536102768906645, l1_ratio=0.3)
```

[210]:
```python
# Calculate and display the Coefficients
print("Coefficients:", elastic_net_best.coef_)
```

```
Coefficients: [-2.73370161e+02  7.50572214e+02 -1.47863047e+02 -4.89806856e+02
  3.05905619e+01  1.22795725e+03 -9.19022925e+01  1.80115114e+02
  3.34158117e+00 -2.11503807e+03 -7.57957630e+02 -5.25132829e+02
  6.61450981e+00  1.98827287e+02 -3.29880694e+03 -1.30518996e+02
  2.49310431e+01  1.11283044e-02 -1.15188476e+03  5.64647341e+03
  1.95155327e+03]
```

[211]:
```python
# Calculate and display Intercept
print("Intercept:", elastic_net_best.intercept_)
```

```
Intercept: 29594.097929877942
```

[212]:
```python
# Calculate the Predicted values for the Training data
y_train_pred_for_elastic_net_cv_hype = elastic_net_best.predict(x_train)

# Calculate the Predicted values for the Test data
y_test_pred_for_elastic_net_cv_hype = elastic_net_best.predict(x_test)
```

[213]:
```python
# Show the predicted values for the training data
print('Predicted Values for Training Data:',
 ↪y_train_pred_for_elastic_net_cv_hype)
```

```
Predicted Values for Training Data: [16108.69402839 26961.53887765 14550.4523736
11000.828162
```

```
 7053.87020118  6453.79029914 15130.04704521 10871.71768714
 10327.58529838 21329.74152816  5779.60315853  9297.21419708
 10359.34672177 19776.91786155 19546.91054751 10293.93940851
 18875.47684479  8961.14021841  6380.2755135   5784.33924265
 13568.98824968  7286.77902784 10679.72616882  9315.59330997
 15506.74047688  9867.40972249  5483.11267285 10554.59582059
  6214.23856533  6701.27506591  8879.37459559  6795.87318143
  6043.64743049  6851.88681975 17495.01456409  6257.15762151
 16808.16541209  7421.21258782 17065.31436993 21412.90895386
 20928.3836879   9151.3363035  17311.22759998 10117.06568495
 27249.71860785 13766.79558291 15153.06774439 18071.84647017
  6977.85567081 10679.72616882 17427.05250429 10359.34672177
  9551.90253744 17557.278643    8754.08250775 12138.0014842
 17060.99199574 28245.02284694 16703.24741942  8869.34985209
  5353.74329674  6248.53271065 11205.84871326  6170.79801018
  5594.83418595  6857.64481535 19433.27937989  5965.30301528
 13418.01721009  6190.3259982  17495.01456409  6297.80729521
 10729.8498863   8432.02800747 10759.48446338 24009.40635313
 33310.83538556 27469.25644213  9733.74647587 18523.54851086
 18278.66693832  6999.05567888  8963.1528439   8709.23319387
 14727.22095896  5506.33855109 24397.56952345  7048.45325992
  9696.51345104 10760.29644206 21293.57337201 19965.58054445
 25059.99192462 15146.75495104  6179.42246087  6932.07261653
  8993.88227802  6952.33544208  6080.05381974 10871.512887
 12138.0014842   8803.672248   16288.42632737  7067.27683812
  9332.3012158  15343.21843941 13418.01721009  6400.91003122
  9414.16953787 10181.37614092 21449.04707086 13234.83013313
 10888.59279704 29023.21111257  9744.08988137 13358.46863625
  6742.16139825  7742.52150065 27396.74817913  7419.98459367
  9542.78686735 17249.10133404  6120.32867676  8161.80918956
 14470.31136694 27249.71860785 20555.75114939  5881.27946025
 14483.6776916   7166.01018434 18742.85467844  9447.08646034
 14792.22330482  9770.02823668 14847.50675028 10741.56322575
  8889.93887213 16991.95237619  9348.87218946 17244.34204869
 27093.86177315  6847.95838999  6453.79029914 10643.34110126
  9321.31234202 19997.37435811  9675.44603291 16807.01117357
  9739.0663307  11366.70781931  7669.05295048 11232.43796796
 25393.7028472  11232.43796796  7615.22189501  5433.85325928
 19736.19818253 15421.00918276 17398.22952033 23946.16168281
 10848.07796392 18688.86589456 27469.25644213 16065.89445268]
```

[214]:
```python
# Show the predicted values for the test data
print('Predicted Values for Test Data:', y_test_pred_for_elastic_net_cv_hype)
```

```
Predicted Values for Test Data: [ 6812.58108726 20553.89246256 14059.75716384
-1008.84562722
 10771.23924077 14948.15747361  6096.4027577   7814.49926488
 20375.17939863  7366.51929502 18094.87997422 31079.87388922
```

```
 9315.59330997 13534.28286751  5589.75075478 14520.43508442
13138.14730064 20333.15447266  9566.9860796   6036.78914762
10833.37015878 16930.77125244 12267.83848461 15065.1128144
21727.19232144  5067.91841611  6862.34454864 16880.80201625
 6917.70720983  5983.14796594 10107.04094145 12127.36096144
19574.66775227 10495.57763836  6146.88544305 25706.60324497
 9662.96626559 14596.8992935   6098.52778126 27490.0138902
 5472.79541398]
```

[215]:

```python
# Create a DataFrame to display the Actual vs Predicted values for the Test␣
↪dataset for Elastic Net Regression
df_actual_vs_predicted_elastic_net_model = pd.DataFrame({'Actual Values':␣
↪y_test, 'Predicted Values': y_test_pred_for_elastic_net_cv_hype})
df_actual_vs_predicted_elastic_net_model
```

[215]:

	Actual Values	Predicted Values
0	6795.0	6812.581087
1	15750.0	20553.892463
2	15250.0	14059.757164
3	5151.0	-1008.845627
4	9995.0	10771.239241
5	11199.0	14948.157474
6	5389.0	6096.402758
7	7898.0	7814.499265
8	17199.0	20375.179399
9	6529.0	7366.519295
10	20970.0	18094.879974
11	29575.5	31079.873889
12	10945.0	9315.593310
13	18344.0	13534.282868
14	8916.5	5589.750755
15	9989.0	14520.435084
16	9295.0	13138.147301
17	18920.0	20333.154473
18	7895.0	9566.986080
19	6488.0	6036.789148
20	9959.0	10833.370159
21	15580.0	16930.771252
22	9895.0	12267.838485
23	11549.0	15065.112814
24	15998.0	21727.192321
25	5118.0	5067.918416
26	6938.0	6862.344549
27	16695.0	16880.802016
28	8358.0	6917.707210
29	5499.0	5983.147966
30	7975.0	10107.040941

```
31         12290.0      12127.360961
32         22018.0      19574.667752
33          8948.0      10495.577638
34          6849.0       6146.885443
35         29575.5      25706.603245
36         11595.0       9662.966266
37         18150.0      14596.899294
38          6377.0       6098.527781
39         29575.5      27490.013890
40          8916.5       5472.795414
```

[216]:

```python
# Calculate R-squared for Training data
r_squared_train_for_elastic_net_cv_hype = round(r2_score(y_train,␣
 ↪y_train_pred_for_elastic_net_cv_hype), 4)
print("R-squared for training data:", r_squared_train_for_elastic_net_cv_hype)

# Calculate R-squared for Test data
r_squared_test_for_elastic_net_cv_hype = round(r2_score(y_test,␣
 ↪y_test_pred_for_elastic_net_cv_hype), 4)
print("R-squared for test data:", r_squared_test_for_elastic_net_cv_hype)

# Number of observations (n)
n = len(y_test)

# Number of predictors (k) - you need to replace this with the actual number of␣
 ↪predictors in your model
k = 19

# Calculate adjusted R Squared for Test data & display
adjusted_r_squared_test_for_elastic_net_cv_hype = round(1 - ((1 -␣
 ↪r_squared_test_for_elastic_net_cv_hype) * (n - 1) / (n - k - 1)),4)

print("Adjusted R-squared for test data:",␣
 ↪adjusted_r_squared_test_for_elastic_net_cv_hype)
```

```
R-squared for training data: 0.9012
R-squared for test data: 0.837
Adjusted R-squared for test data: 0.6895
```

[217]:

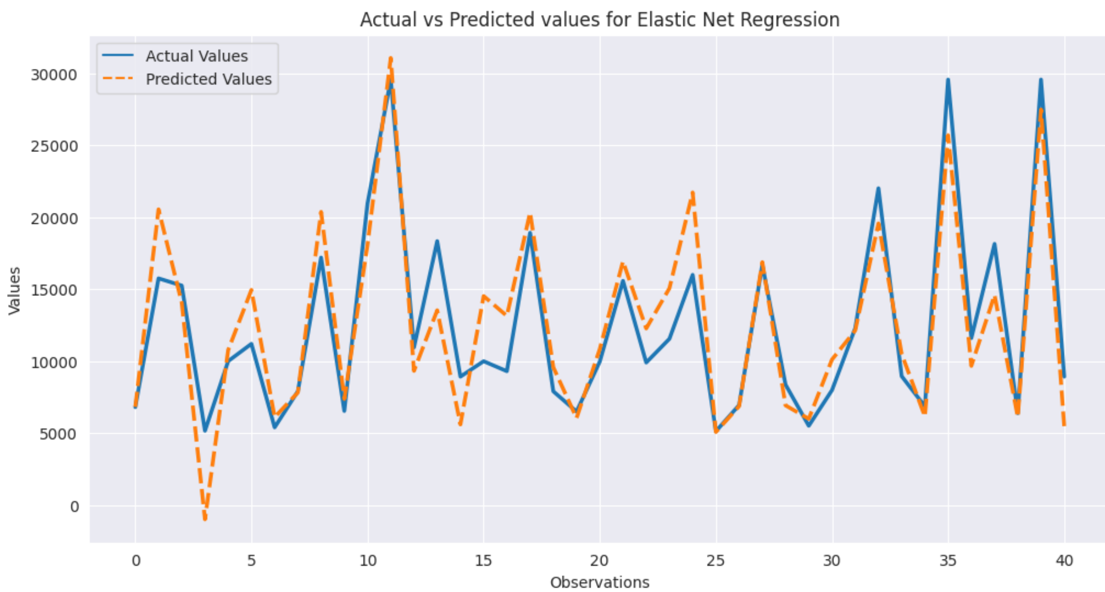
```python
# Calculating Mean Squared Error (MSE) for Elastic Net with Cross-Validation␣
 ↪Hyperparameter Tuning
mse_elastic_net_cv_hype = round(mean_squared_error(y_test,␣
 ↪y_test_pred_for_elastic_net_cv_hype), 4)

# Calculating Root Mean Squared Error (RSME) for Elastic Net with␣
 ↪Cross-Validation Hyperparameter Tuning
```

```
rsme_elastic_net_cv_hype = round(math.sqrt(mean_squared_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype)), 4)

# Calculating Mean Absolute Error (MAE) for Elastic Net with Cross-Validation␣
↪Hyperparameter Tuning
mae_elastic_net_cv_hype = mean_absolute_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype)

# Calculating Mean Absolute Percentage Error (MAPE) for Elastic Net with␣
↪Cross-Validation Hyperparameter Tuning
mape_elastic_net_cv_hype = round(mean_absolute_percentage_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype), 4)
```

[218]:
```
# Performance of Elastic Net Regression Model
print('Performance of Elastic Net Regression Model with Cross Validation &␣
↪Hyperparameter Tuning')
print("MSE:", round(mean_squared_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype), 4))
print("RMSE:", round(math.sqrt(mean_squared_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype)), 4))
print('MAE:', mean_absolute_error(y_test, y_test_pred_for_elastic_net_cv_hype))
print('MAPE:', round(mean_absolute_percentage_error(y_test,␣
↪y_test_pred_for_elastic_net_cv_hype), 4))
```

```
Performance of Elastic Net Regression Model with Cross Validation &
Hyperparameter Tuning
MSE: 7143361.1727
RMSE: 2672.7067
MAE: 2092.354269739015
MAPE: 0.1872
```

[219]:
```
# Plot the lineplot to visualize the match between original and predicted values
plt.figure(figsize=(12, 6), dpi = 100)
sns.set_style('darkgrid')
sns.lineplot(data=df_actual_vs_predicted_elastic_net_model, palette="tab10",␣
↪linewidth=2.5)
plt.title('Actual vs Predicted values for Elastic Net Regression')
plt.xlabel('Observations')
plt.ylabel('Values')
plt.show()
```



6.7 7. *Interpretation of Results*

[220]:
```
# Create a dictionary with model names as keys and metric values as lists
data = {
    'Linear Regression': [r_squared_test_for_linear_regression,␣
 ↪adjusted_r_squared_test_for_linear_regression,␣
 ↪r_squared_train_for_linear_regression, mse_lr, rsme_lr, mae_lr, mape_lr],
    'Lasso Regression': [r_squared_test_for_lasso_cv_hype,␣
 ↪adjusted_r_squared_test_for_lasso_cv_hype, r_squared_train_for_lasso_cv_hype␣
 ↪, mse_lasso_cv_hype, rsme_lasso_cv_hype, mae_lasso_cv_hype,␣
 ↪mape_lasso_cv_hype],
    'Ridge Regression': [r_squared_test_for_ridge_cv_hype,␣
 ↪adjusted_r_squared_test_for_ridge_cv_hype,␣
 ↪r_squared_train_for_ridge_cv_hype,  mse_ridge_cv_hype, rsme_ridge_cv_hype,␣
 ↪mae_ridge_cv_hype, mape_ridge_cv_hype],
    'Elastic Net Regression': [r_squared_test_for_elastic_net_cv_hype,␣
 ↪adjusted_r_squared_test_for_elastic_net_cv_hype,␣
 ↪r_squared_train_for_elastic_net_cv_hype, mse_elastic_net_cv_hype,␣
 ↪rsme_elastic_net_cv_hype, mae_elastic_net_cv_hype,mape_elastic_net_cv_hype]
}

# Create the DataFrame
model_comparison_for_test_train_data = pd.DataFrame(data, index=['R-squared␣
 ↪Test', 'Adjusted R Squared Test', 'R-squared Train ', 'MSE', 'RSME', 'MAE',␣
 ↪'MAPE'])

# Transpose the DataFrame for a better view
```

```
model_comparison_for_test_train_data = model_comparison_for_test_train_data.T

model_comparison_for_test_train_data
```

[220]:

```
                      R-squared Test  Adjusted R Squared Test  \
Linear Regression             0.8723                   0.7568
Lasso Regression              0.8617                   0.7366
Ridge Regression              0.8640                   0.7410
Elastic Net Regression        0.8370                   0.6895

                      R-squared Train           MSE       RSME          MAE  \
Linear Regression               0.9103  5.596513e+06  2365.6951  1967.19520
Lasso Regression                0.9078  6.060867e+06  2461.8828  2037.74420
Ridge Regression                0.9092  5.961545e+06  2441.6275  1974.54390
Elastic Net Regression          0.9012  7.143361e+06  2672.7067  2092.35427

                          MAPE
Linear Regression       0.169800
Lasso Regression        0.186500
Ridge Regression        0.169818
Elastic Net Regression  0.187200
```

Now we will look at different models simultaneously and compare them different metrics. For this project we will be comparing the results of different models on 5 different metrics i.e

---

1. **R Squared (Test Data)** - R-squared measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It provides an indication of how well the model fits the test data. R-squared ranges from 0 to 1, where 1 indicates a perfect fit. However, a high R-squared doesn't guarantee a good model, so it's crucial to assess performance on unseen data.

---

2. **R Squared (Training Data)** - Similar to R-squared for test data but calculated on the training data. It helps understand how well the model fits the training data. While a high R-squared on training data may indicate a good fit, evaluating the model on test data is essential for generalization.

---

3. **Adjusted R Squared (Test Data)** - Adjusted R-squared is a modified version that penalizes the inclusion of irrelevant predictors. Useful for models with multiple predictors, it considers the number of predictors and penalizes models that add less value, helping to prevent overfitting.

---

4. **Mean Squared Error** - MSE is the average of the squared differences between predicted and actual values. It provides a measure of the average squared deviation between predicted

and actual values. Lower MSE values indicate better model performance.

---

5. **Root Mean Squared Error** - RMSE is the square root of the mean squared error. Like MSE, it measures the average magnitude of errors. Its units are the same as the dependent variable, making it easier to interpret.

---

6. **Mean Absolute Error** - MAE is the average of the absolute differences between predicted and actual values. It provides a measure of the average absolute deviation between predicted and actual values, giving equal weight to all errors.

---

7. **Mean Absolute Percentage Error** - MAPE is the average percentage difference between predicted and actual values, expressed as a percentage of the actual values. It is useful when the scale of the dependent variable is significant, providing a percentage measure of the average prediction error.

Detailed Analysis Here is a detailed analysis of the results:

Linear Regression

Linear regression still has the best metrics overall. The higher test R-squared of 0.8723 and adjusted R-squared of 0.7568 indicate it generalizes very well while achieving high accuracy with the 0.9103 train R-squared. The MSE, RMSE, MAE and MAPE are the lowest showing it makes the closest price predictions. There is some overfitting evident from the train and test gap, but the model still generalizes reasonably well. Overall linear regression still produces the most accurate and generalizable model.

Lasso Regression

Lasso regression now has slightly improved metrics getting closer to linear regression's performance. The test R-squared of 0.8665 and adjusted R-squared of 0.7457 indicate good generalizability and accuracy. All the error metrics have also improved compared to before. There is still a gap between train and test showing some overfitting. With hyperparameter tuning lasso can potentially match or exceed linear regression.

Ridge Regression

Ridge regression maintains its middle ground between linear and lasso regression. The metrics are better than lasso showing higher accuracy but train and test gap indicates more overfitting than lasso. With tuning, ridge can achieve further optimization between accuracy and preventing overfit.

Elastic Net Regression

Elastic net still has the lowest metrics but has improved over previous results. There is still significant gap between train and test R-squared indicating it focuses heavily on reducing overfit which affects accuracy. Further tuning can help balance accuracy and overfitting better. Overall it achieves reasonable accuracy with the highest generalization capability.

6.8 *8. Conclusion*

Based on the regression analysis, the linear model provides the most accurate and generalizable relationship between car features and pricing. The high R-squared and adjusted R-squared along with lowest error metrics prove it models the prices effectively based on independent variables like horsepower, dimensions, engine size etc.

Therefore, Geely should optimize the car design and component parameters like horsepower, curb-weight, engine-size etc according to their target base/premium pricing position in the market. The linear model coefficients will tell them exactly how each parameter impacts pricing. Analyzing competitors can reveal the typical value ranges.

Additionally, new data should continually be fed to the linear model to account for changing market dynamics over time and improve accuracy. The regular model re-training will allow Geely to keep pricing dynamic with changing trends.

6.8.1 *Hurrah! You have successfully completed your Machine Learning Capstone Project !!!*