Oasis Infobyte ¶

Task 3 : Car Price Prediction

The price of a car depends on a lot of factors like the goodwill of the **brand of the car** , **features of the car** , **horsepower** and **the mileage** it gives and many more. Car priceprediction is one of the major research areas in machine learning. So if you want to learn how to train a car price prediction model then this project is for you.

1. Import all necessary

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

2. Import dataframe

In [2]:
```
df = pd.read_csv("car.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	engir
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	

5 rows × 26 columns

3. Check for Null Values

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
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
```

we don't have null values

4. Check for Dublicate row

In [5]:
```
df.duplicated().sum()
```

Out[5]:
```
0
```

we don't have dublicate row

5. Summery of data

In [6]:
```
df.describe()
```

Out[6]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	eng
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205
mean	103.000000	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854	126
std	59.322565	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41
min	1.000000	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61
25%	52.000000	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97
50%	103.000000	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120
75%	154.000000	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141
max	205.000000	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326

In [7]:
```
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
```

6. Check column name

In [8]:
```
df.columns
```

Out[8]:
```
Index(['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'],
      dtype='object')
```

7. Check the datatype

In [9]:
```
df.dtypes
```

Out[9]:
```
car_ID              int64
symboling           int64
CarName            object
fueltype           object
aspiration         object
doornumber         object
carbody            object
drivewheel         object
enginelocation     object
wheelbase         float64
carlength         float64
carwidth          float64
carheight         float64
curbweight          int64
enginetype         object
cylindernumber     object
enginesize          int64
fuelsystem         object
boreratio         float64
stroke            float64
compressionratio  float64
horsepower          int64
peakrpm             int64
citympg             int64
highwaympg          int64
price             float64
dtype: object
```

8.Shape of dataset

In [10]:
```
df.shape
```

Out[10]:
```
(205, 26)
```

9. Find Corelation of data

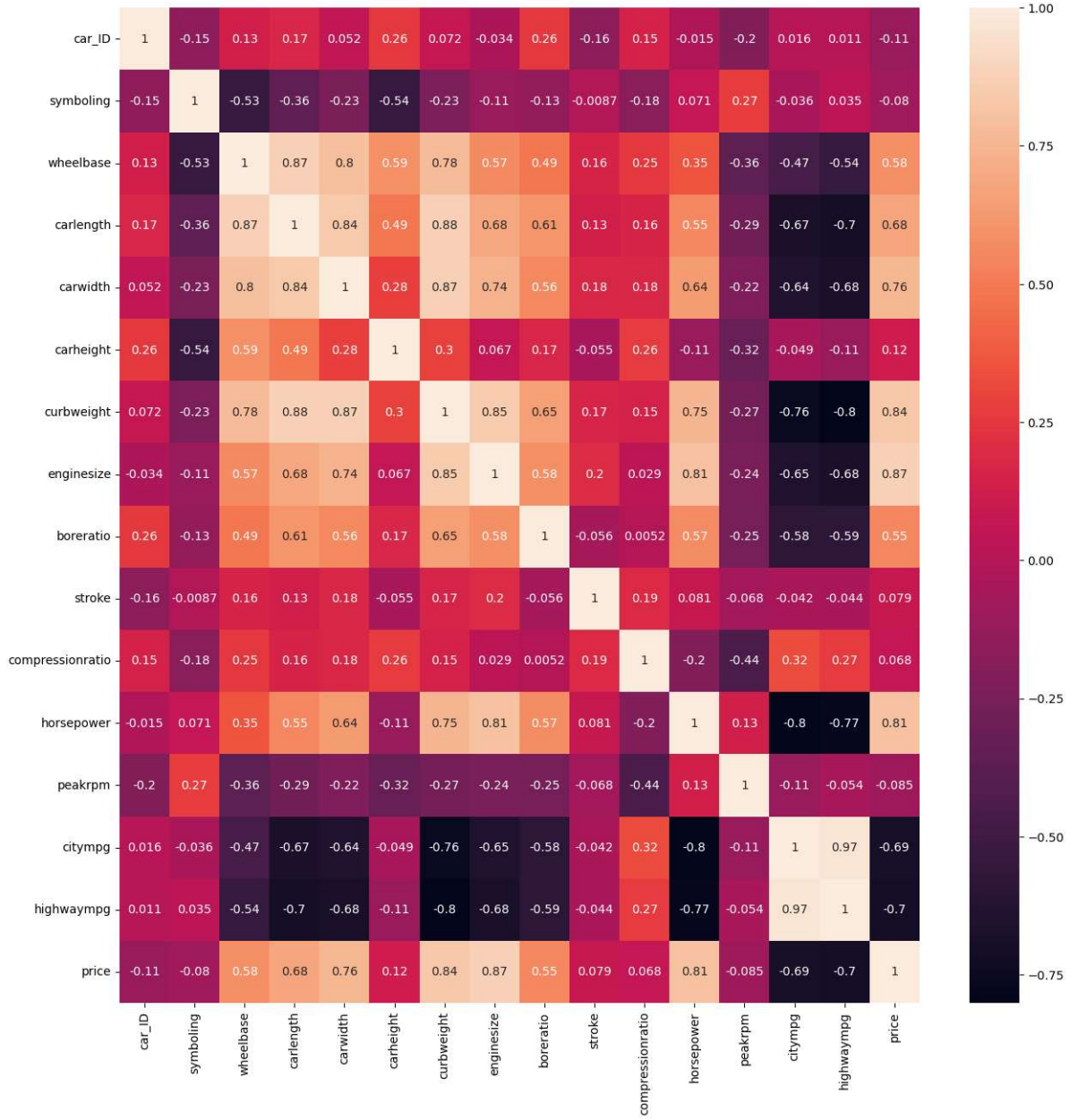
In [11]:
```
df.corr()
```

Out[11]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight
car_ID	1.000000	-0.151621	0.129729	0.170636	0.052387	0.255960	0.071962
symboling	-0.151621	1.000000	-0.531954	-0.357612	-0.232919	-0.541038	-0.227691
wheelbase	0.129729	-0.531954	1.000000	0.874587	0.795144	0.589435	0.776386
carlength	0.170636	-0.357612	0.874587	1.000000	0.841118	0.491029	0.877728
carwidth	0.052387	-0.232919	0.795144	0.841118	1.000000	0.279210	0.867032
carheight	0.255960	-0.541038	0.589435	0.491029	0.279210	1.000000	0.295572
curbweight	0.071962	-0.227691	0.776386	0.877728	0.867032	0.295572	1.000000
enginesize	-0.033930	-0.105790	0.569329	0.683360	0.735433	0.067149	0.850594
boreratio	0.260064	-0.130051	0.488750	0.606454	0.559150	0.171071	0.648480
stroke	-0.160824	-0.008735	0.160959	0.129533	0.182942	-0.055307	0.168790
compressionratio	0.150276	-0.178515	0.249786	0.158414	0.181129	0.261214	0.151362
horsepower	-0.015006	0.070873	0.353294	0.552623	0.640732	-0.108802	0.750739
peakrpm	-0.203789	0.273606	-0.360469	-0.287242	-0.220012	-0.320411	-0.266243
citympg	0.015940	-0.035823	-0.470414	-0.670909	-0.642704	-0.048640	-0.757414
highwaympg	0.011255	0.034606	-0.544082	-0.704662	-0.677218	-0.107358	-0.797465
price	-0.109093	-0.079978	0.577816	0.682920	0.759325	0.119336	0.835305

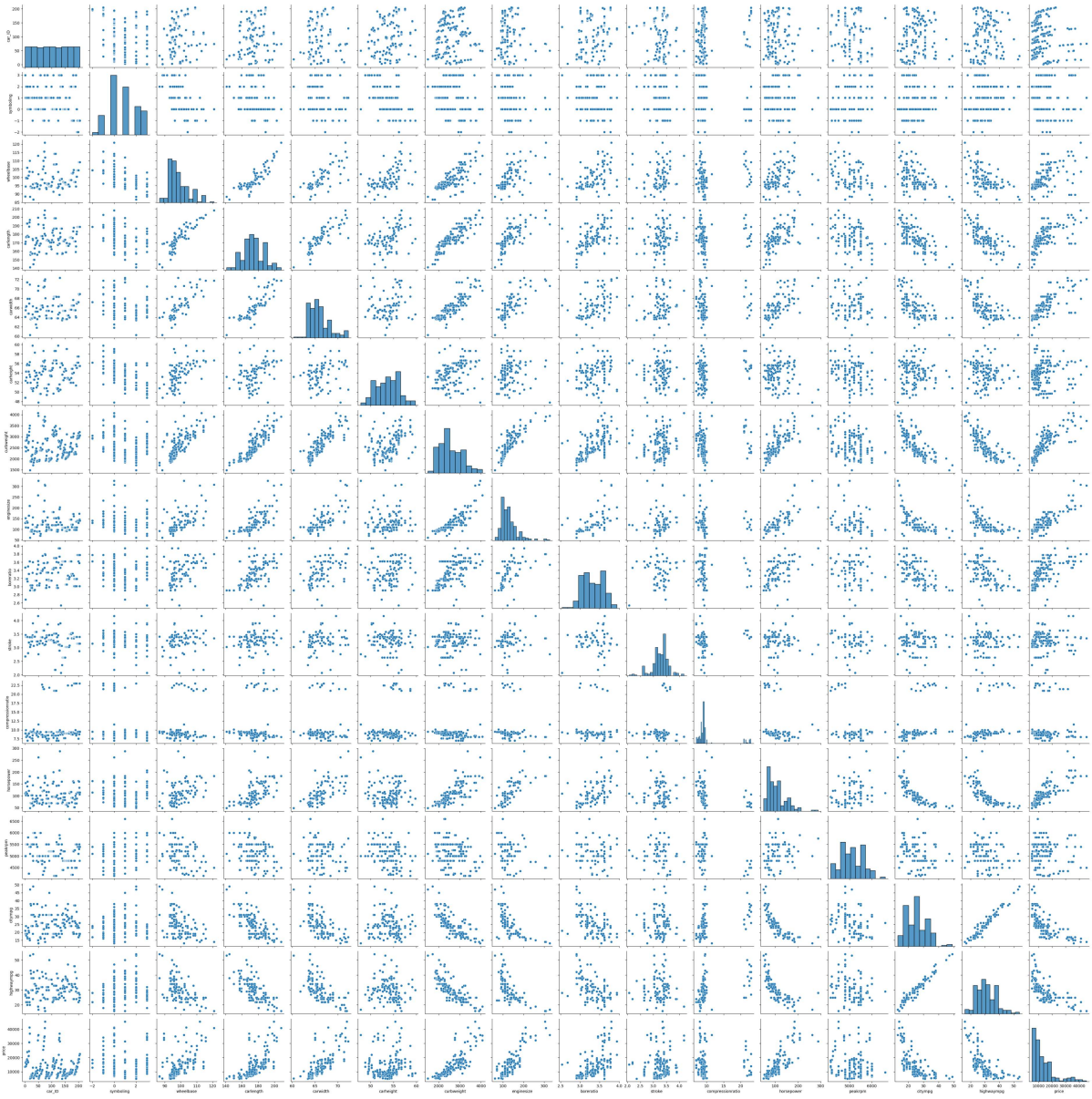
In [12]:
```
plt.figure(figsize=(15,15))
sns.heatmap(df.corr() , annot=True)
```

Out[12]: <AxesSubplot:>



In [13]:
```
sns.pairplot(df)
```

Out[13]:
```
<seaborn.axisgrid.PairGrid at 0x1ea918d6130>
```



10. Analysing the 'price' column

In [14]:
```
df["price"]
```

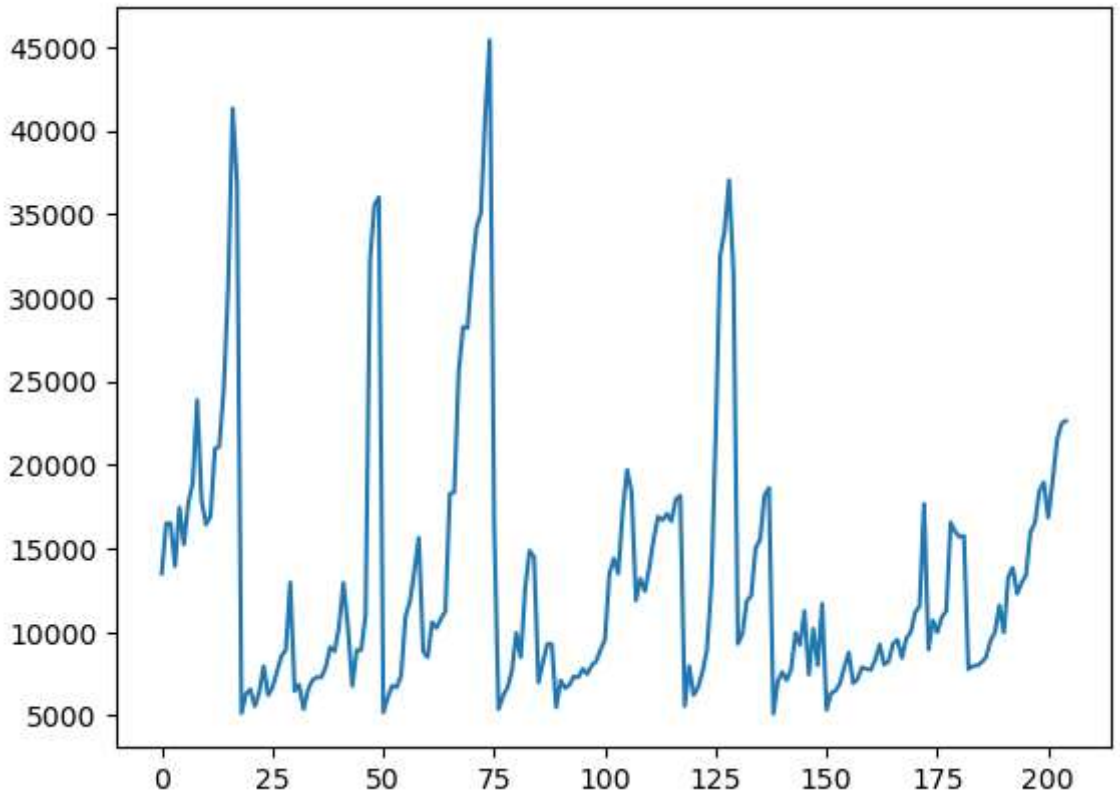
Out[14]:
```
0      13495.0
1      16500.0
2      16500.0
3      13950.0
4      17450.0
        ...
200    16845.0
201    19045.0
202    21485.0
203    22470.0
204    22625.0
Name: price, Length: 205, dtype: float64
```

In [15]:
```
df["price"].describe()
```

Out[15]:
```
count      205.000000
mean     13276.710571
std       7988.852332
min       5118.000000
25%       7788.000000
50%      10295.000000
75%      16503.000000
max      45400.000000
Name: price, dtype: float64
```

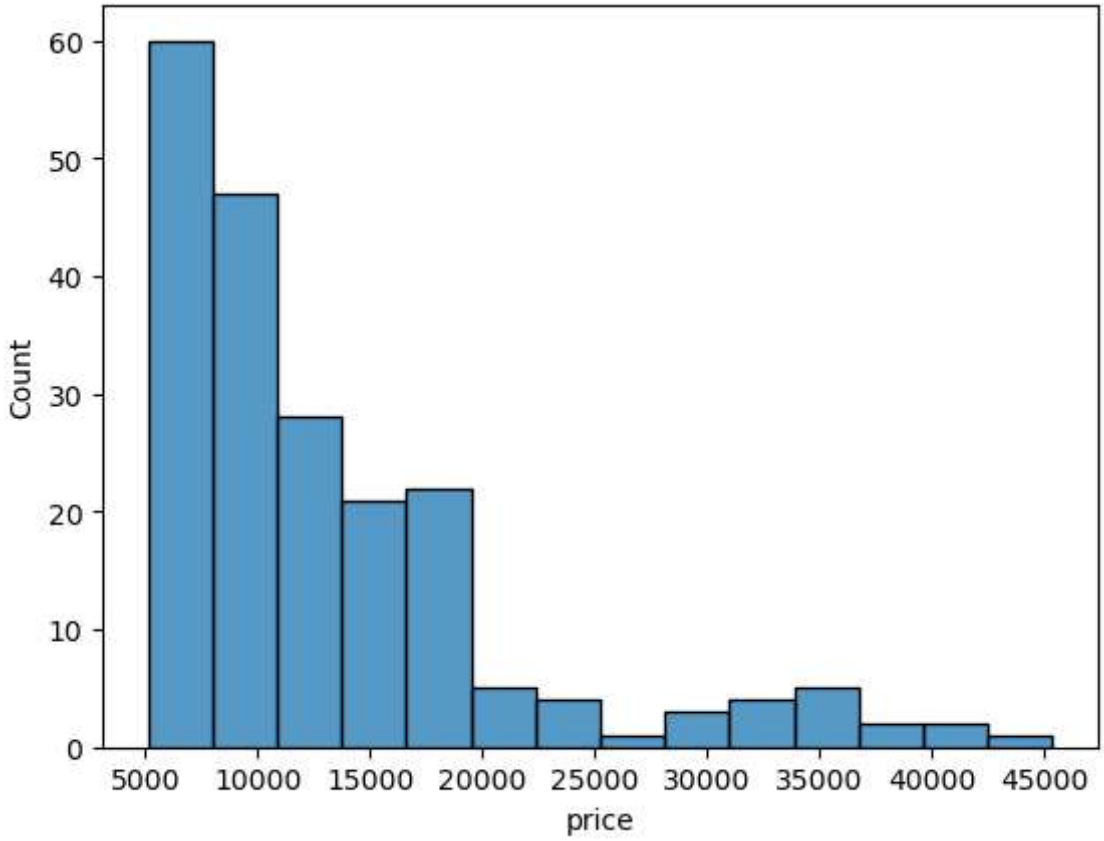
In [16]:
```
plt.plot(df["price"])
```

Out[16]:
```
[<matplotlib.lines.Line2D at 0x1ea9f1652b0>]
```



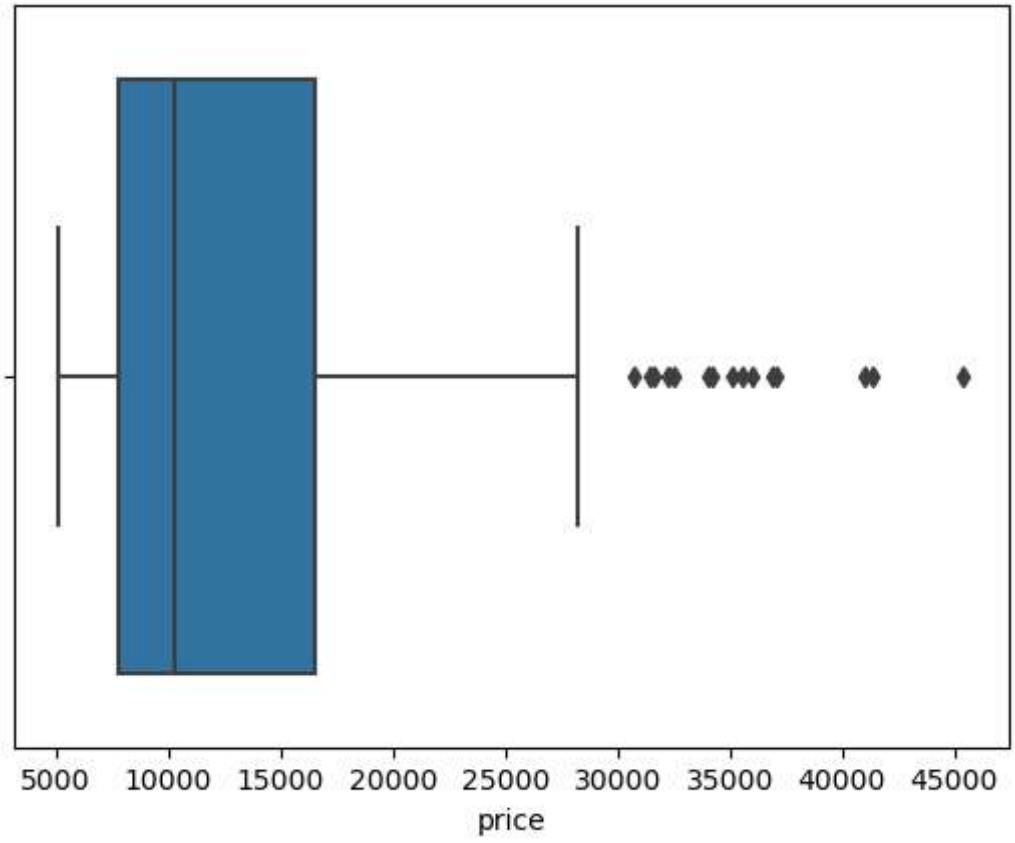
In [17]:
```
sns.histplot(df["price"])
```

Out[17]:
```
<AxesSubplot:xlabel='price', ylabel='Count'>
```



In [18]:
```
sns.boxplot(df["price"])
```

Out[18]:
```
<AxesSubplot:xlabel='price'>
```



11. Analysing the Independent column

In [19]:
```
df.columns
```

Out[19]:
```
Index(['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'],
      dtype='object')
```

In [20]:
```
df["symboling"].unique()
```

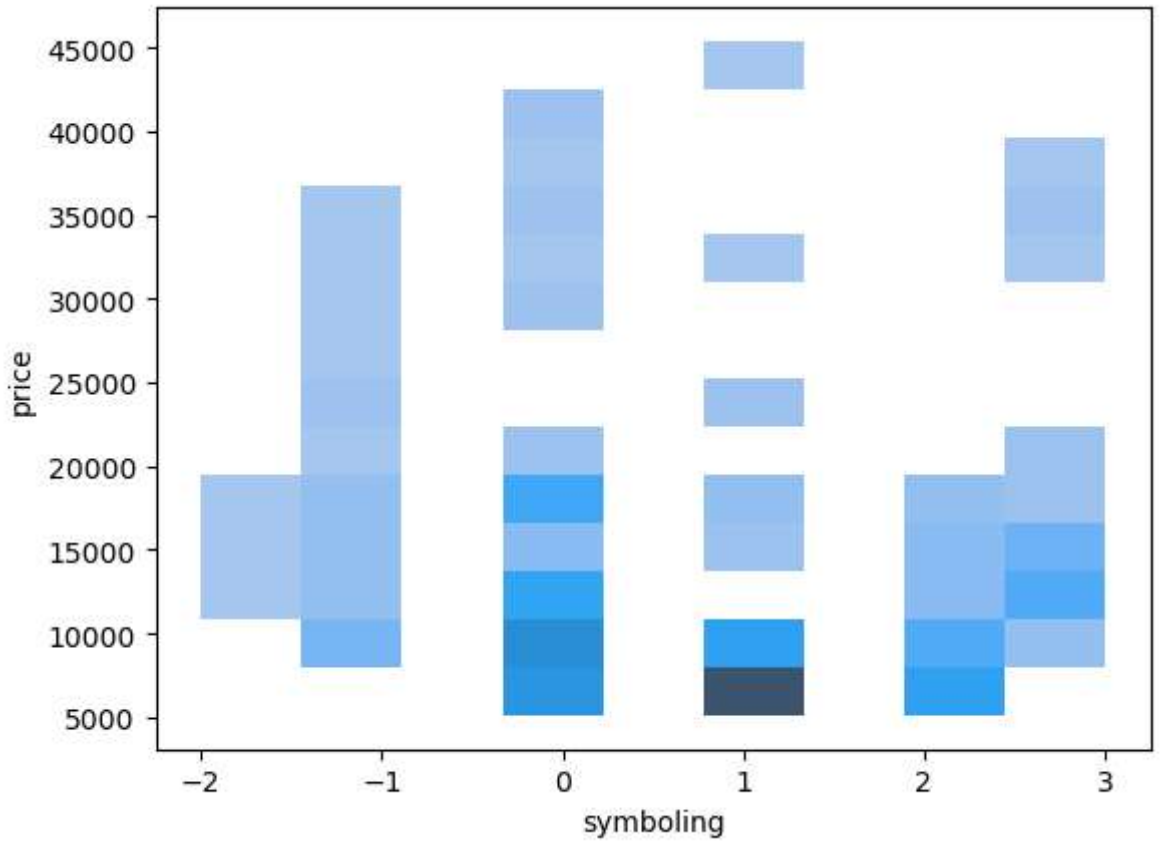
Out[20]:
```
array([ 3,  1,  2,  0, -1, -2], dtype=int64)
```

In [21]:
```
df["symboling"].value_counts()
```

Out[21]:
```
 0    67
 1    54
 2    32
 3    27
-1    22
-2     3
Name: symboling, dtype: int64
```

In [22]:
```
sns.histplot(x = "symboling" , y = "price" , data = df)
```

Out[22]:
```
<AxesSubplot:xlabel='symboling', ylabel='price'>
```



In [23]:
```
df["fueltype"].unique()
```

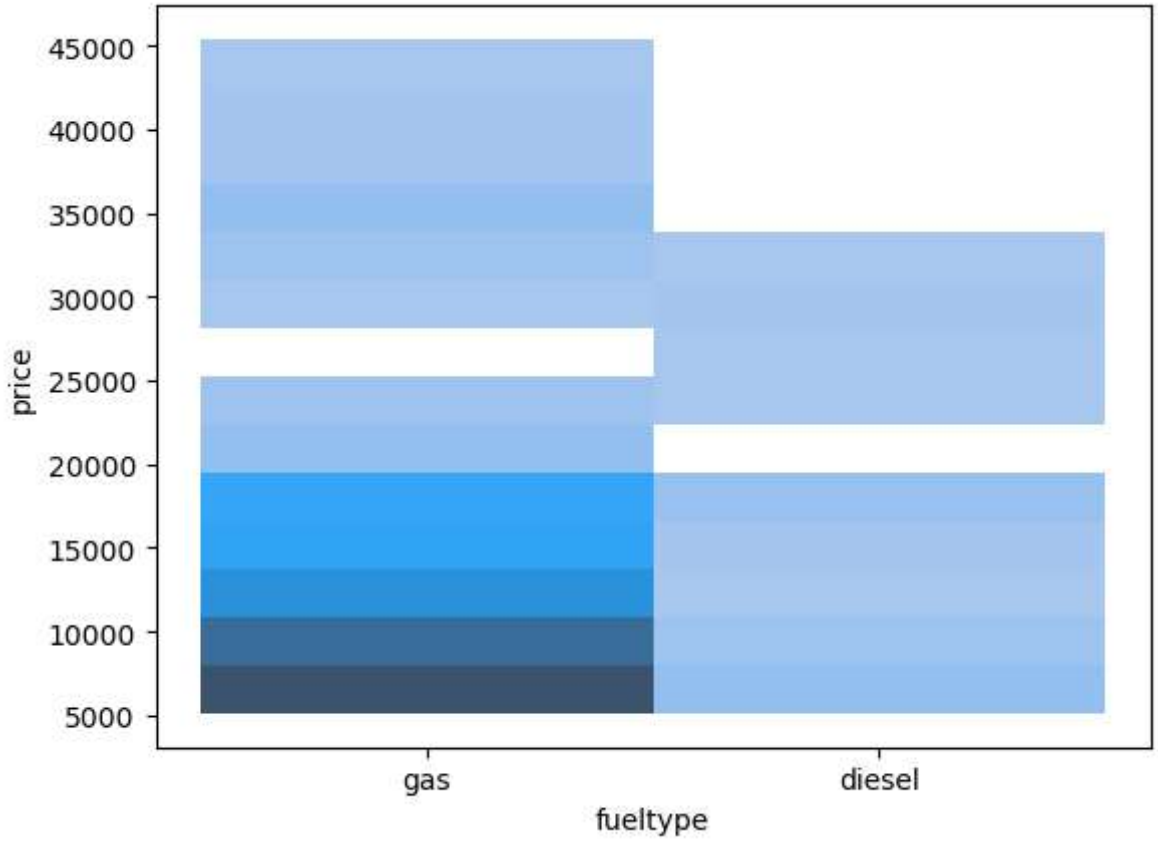
Out[23]:
```
array(['gas', 'diesel'], dtype=object)
```

In [24]:
```
df["fueltype"].value_counts()
```

Out[24]:
```
gas       185
diesel     20
Name: fueltype, dtype: int64
```

In [25]:
```
sns.histplot(x = "fueltype" , y = "price" , data = df)
```

Out[25]:
```
<AxesSubplot:xlabel='fueltype', ylabel='price'>
```



In [26]:
```
df["aspiration"].unique()
```

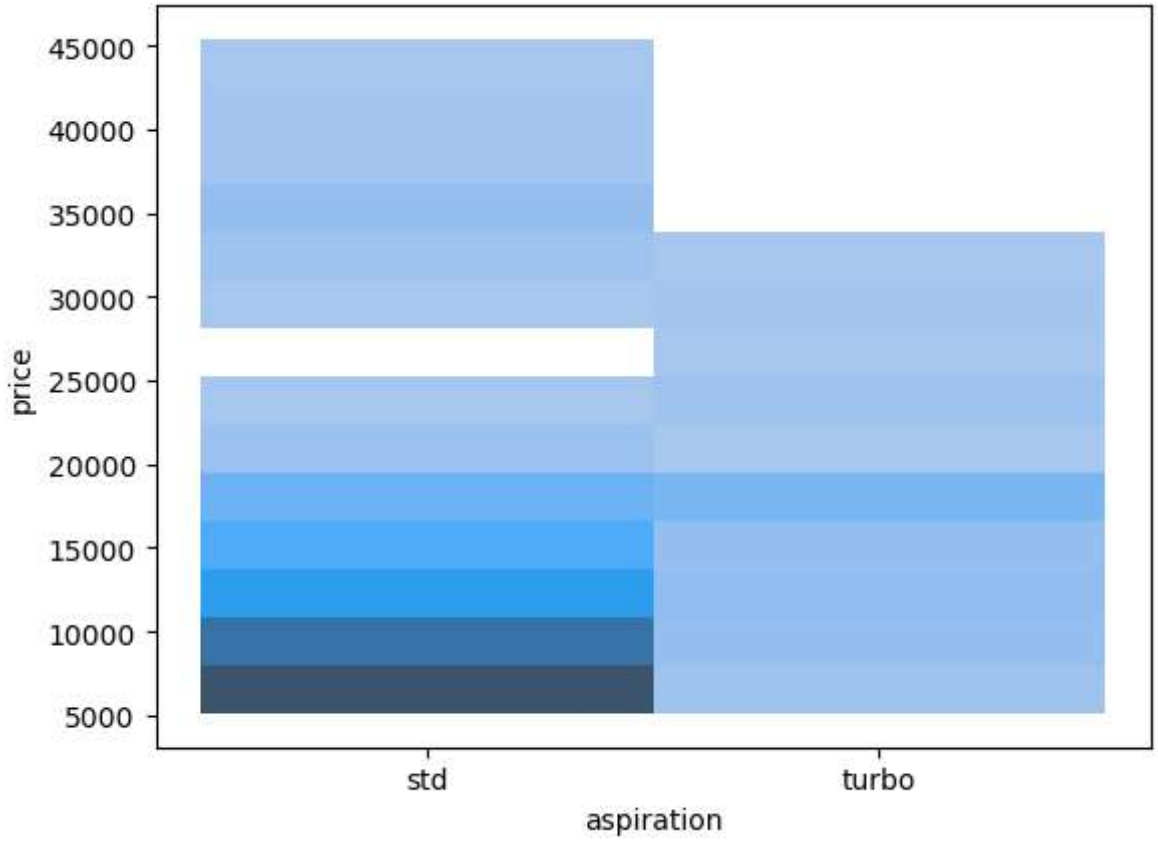
Out[26]:
```
array(['std', 'turbo'], dtype=object)
```

In [27]:
```
df["aspiration"].value_counts()
```

Out[27]:
```
std      168
turbo     37
Name: aspiration, dtype: int64
```

In [28]:
```
sns.histplot(x="aspiration" , y="price" , data = df )
```

Out[28]:
```
<AxesSubplot:xlabel='aspiration', ylabel='price'>
```



In [29]:
```
df["doornumber"].unique()
```

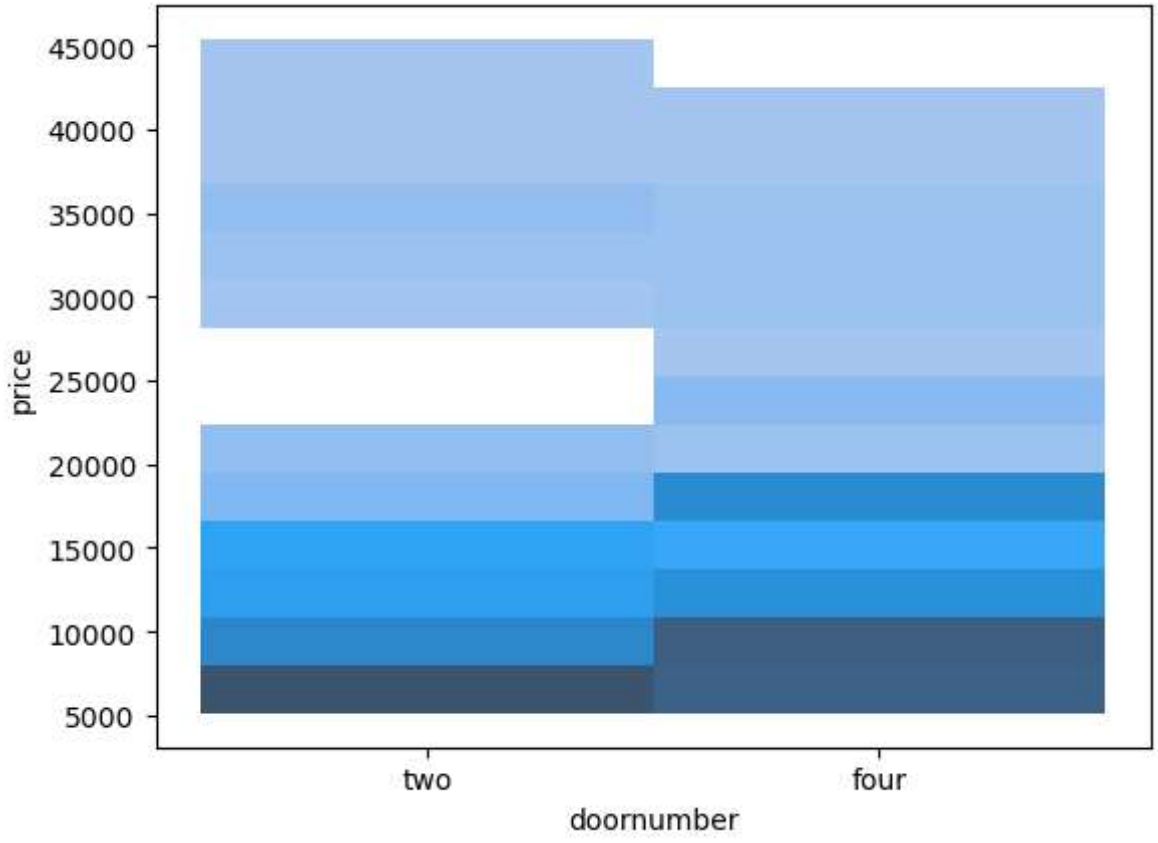
Out[29]:
```
array(['two', 'four'], dtype=object)
```

In [30]:
```
df["doornumber"].value_counts()
```

Out[30]:
```
four    115
two      90
Name: doornumber, dtype: int64
```

In [31]:
```
sns.histplot(x="doornumber" , y = "price" , data = df)
```

Out[31]:
```
<AxesSubplot:xlabel='doornumber', ylabel='price'>
```



In [32]:
```
df["carbody"].unique()
```

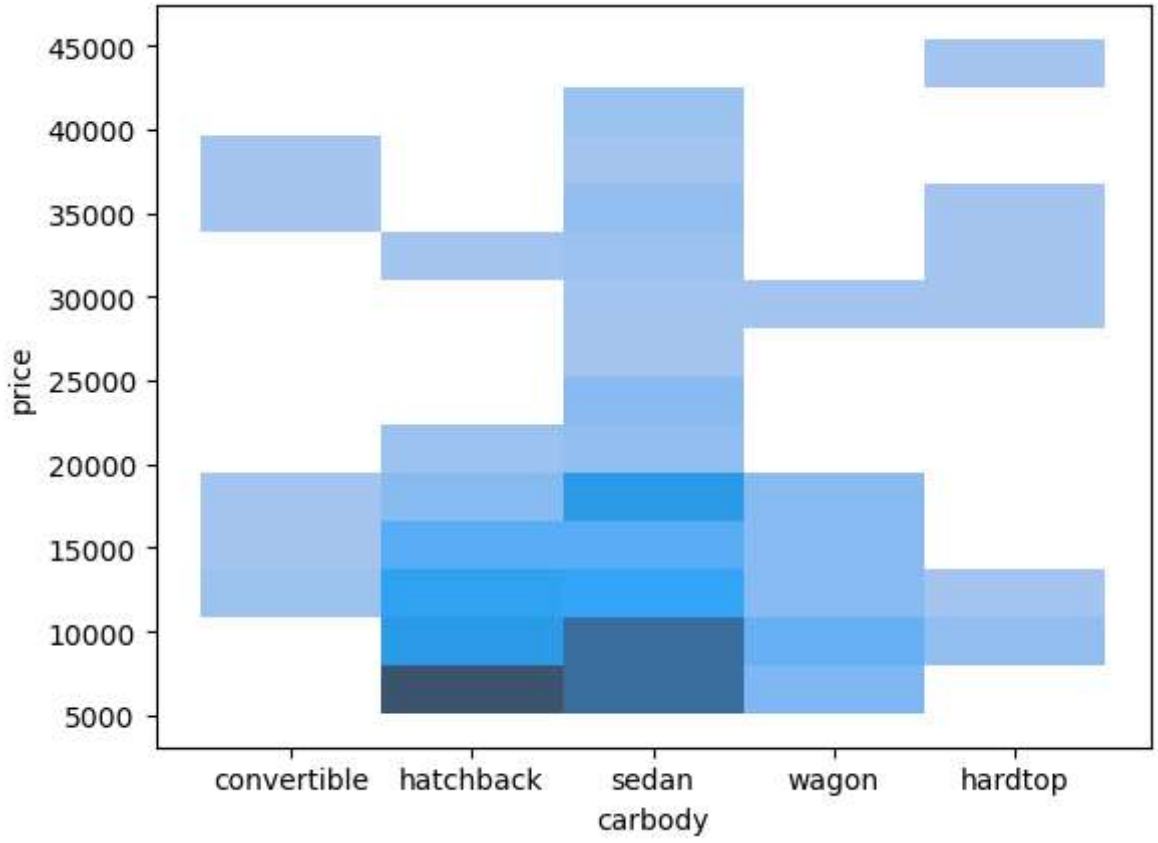
Out[32]:
```
array(['convertible', 'hatchback', 'sedan', 'wagon', 'hardtop'],
      dtype=object)
```

In [33]:
```
df["carbody"].value_counts()
```

Out[33]:
```
sedan          96
hatchback      70
wagon          25
hardtop         8
convertible     6
Name: carbody, dtype: int64
```

In [34]:
```
sns.histplot(x = "carbody" , y = "price" , data = df)
```

Out[34]:
```
<AxesSubplot:xlabel='carbody', ylabel='price'>
```



In [35]:
```
df["drivewheel"].unique()
```

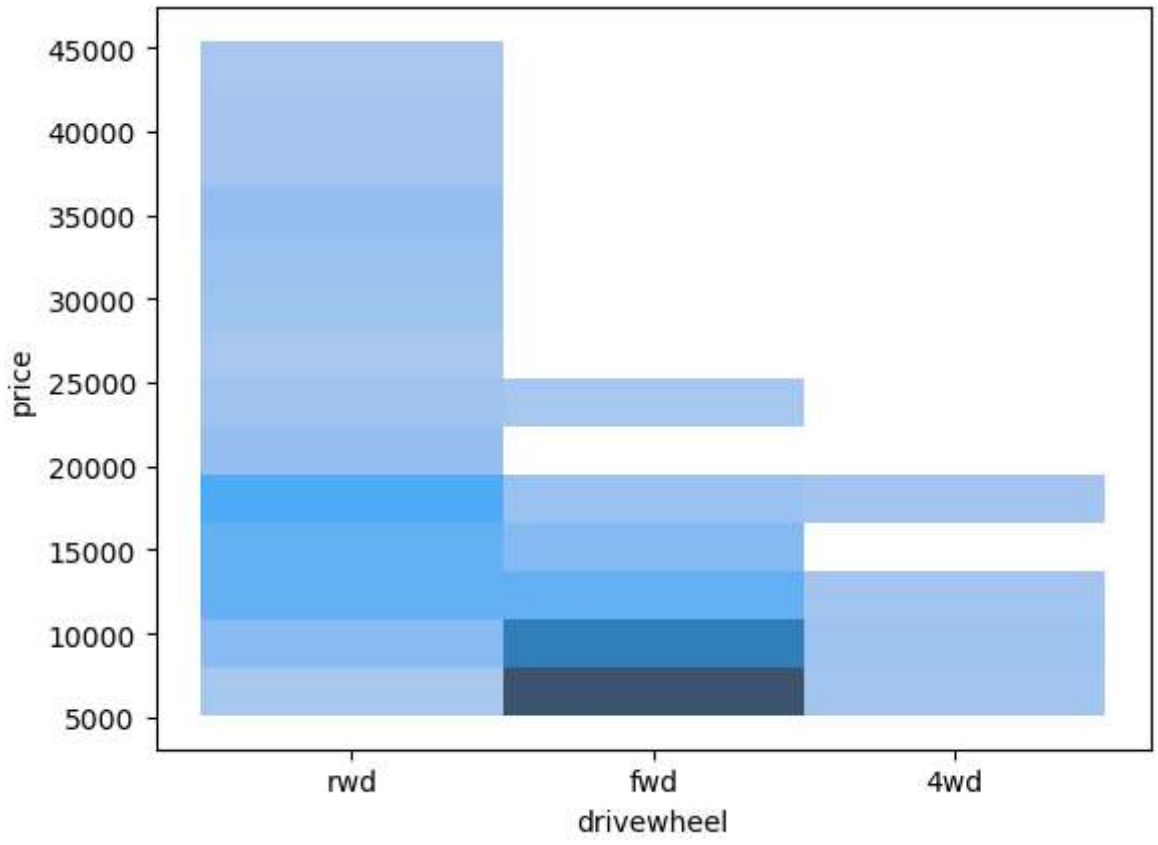
Out[35]:
```
array(['rwd', 'fwd', '4wd'], dtype=object)
```

In [36]:
```
df["drivewheel"].value_counts()
```

Out[36]:
```
fwd    120
rwd     76
4wd      9
Name: drivewheel, dtype: int64
```

In [37]:
```
sns.histplot(x = "drivewheel" , y = "price" , data = df )
```

Out[37]:
```
<AxesSubplot:xlabel='drivewheel', ylabel='price'>
```



In [38]:
```
df["enginetype"].unique()
```

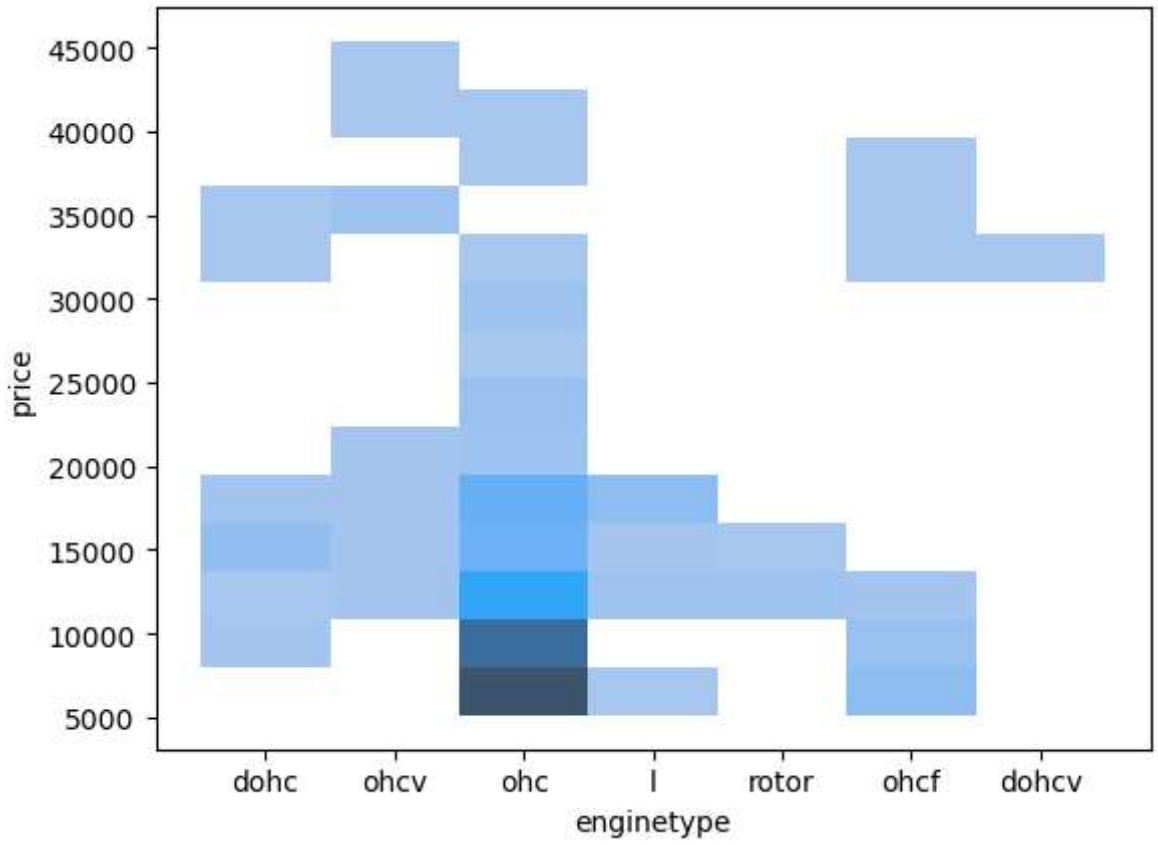
Out[38]:
```
array(['dohc', 'ohcv', 'ohc', 'l', 'rotor', 'ohcf', 'dohcv'], dtype=object)
```

In [39]:
```
df["enginetype"].value_counts()
```

Out[39]:
```
ohc      148
ohcf      15
ohcv      13
dohc      12
l         12
rotor      4
dohcv      1
Name: enginetype, dtype: int64
```

In [40]:
```
sns.histplot(x="enginetype" , y = "price" , data = df)
```

Out[40]:
```
<AxesSubplot:xlabel='enginetype', ylabel='price'>
```



In [41]:
```
df["cylindernumber"].unique()
```

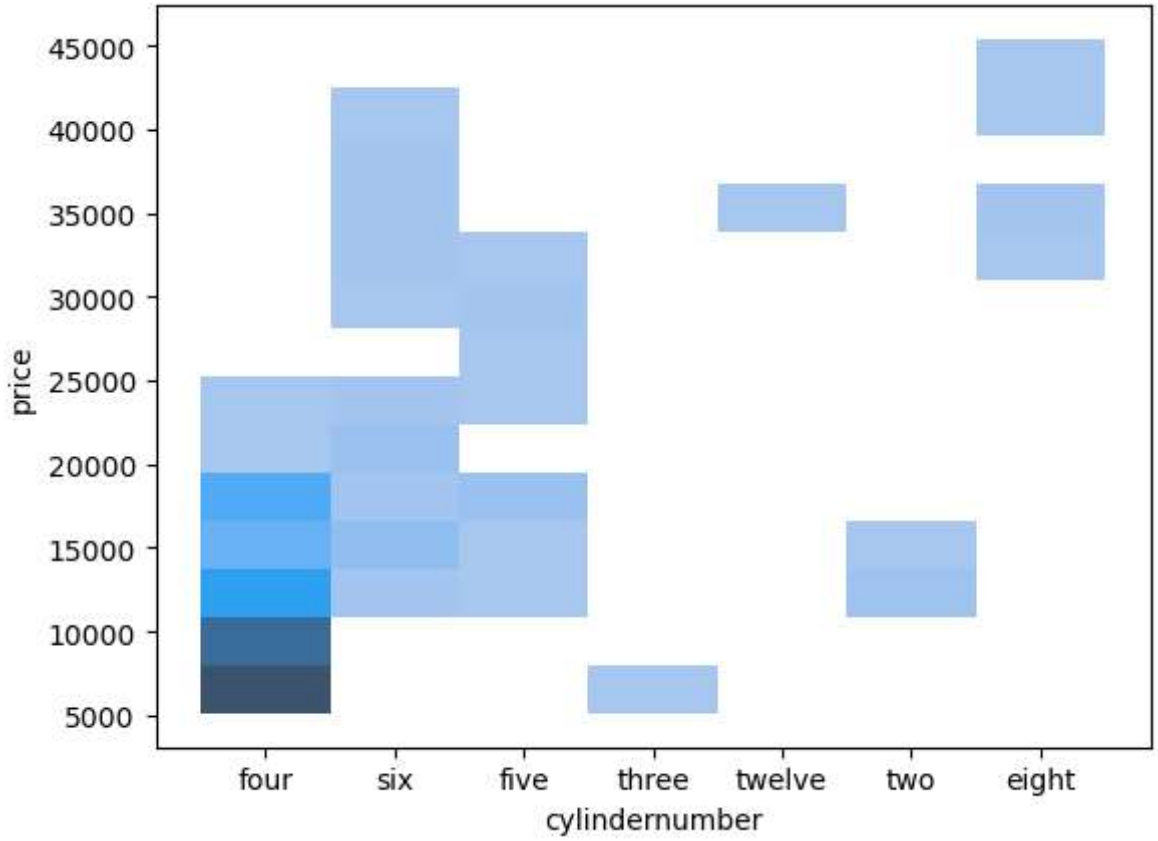
Out[41]:
```
array(['four', 'six', 'five', 'three', 'twelve', 'two', 'eight'],
      dtype=object)
```

In [42]:
```
df["cylindernumber"].value_counts()
```

Out[42]:
```
four      159
six        24
five       11
eight       5
two         4
three       1
twelve      1
Name: cylindernumber, dtype: int64
```

In [43]:
```
sns.histplot(x="cylindernumber" , y = "price" , data = df)
```

Out[43]:
```
<AxesSubplot:xlabel='cylindernumber', ylabel='price'>
```



In [44]:
```
df["enginesize"].unique()
```

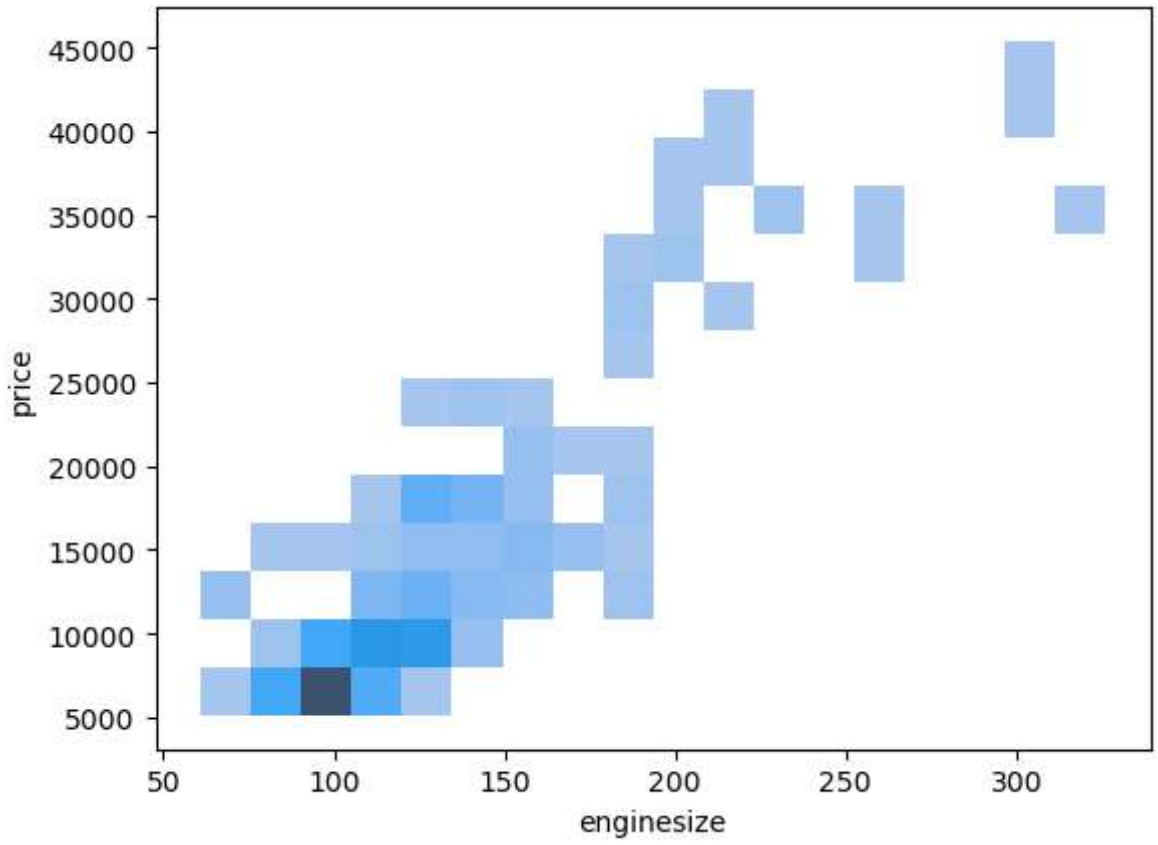
Out[44]:
```
array([130, 152, 109, 136, 131, 108, 164, 209,  61,  90,  98, 122, 156,
        92,  79, 110, 111, 119, 258, 326,  91,  70,  80, 140, 134, 183,
       234, 308, 304,  97, 103, 120, 181, 151, 194, 203, 132, 121, 146,
       171, 161, 141, 173, 145], dtype=int64)
```

In [45]:
```
df["enginesize"].value_counts()
```

Out[45]:
```
122    15
92     15
97     14
98     14
108    13
90     12
110    12
109     8
120     7
141     7
152     6
181     6
146     6
121     6
156     5
136     5
91      5
183     4
130     4
171     3
70      3
194     3
209     3
164     3
258     2
140     2
134     2
234     2
132     2
131     2
173     1
203     1
161     1
80      1
151     1
103     1
304     1
308     1
326     1
119     1
111     1
79      1
61      1
145     1
Name: enginesize, dtype: int64
```

In [46]:
```
sns.histplot(x="enginesize" , y = "price" , data = df)
```

Out[46]:
```
<AxesSubplot:xlabel='enginesize', ylabel='price'>
```



In [47]:
```
df["boreratio"].unique()
```

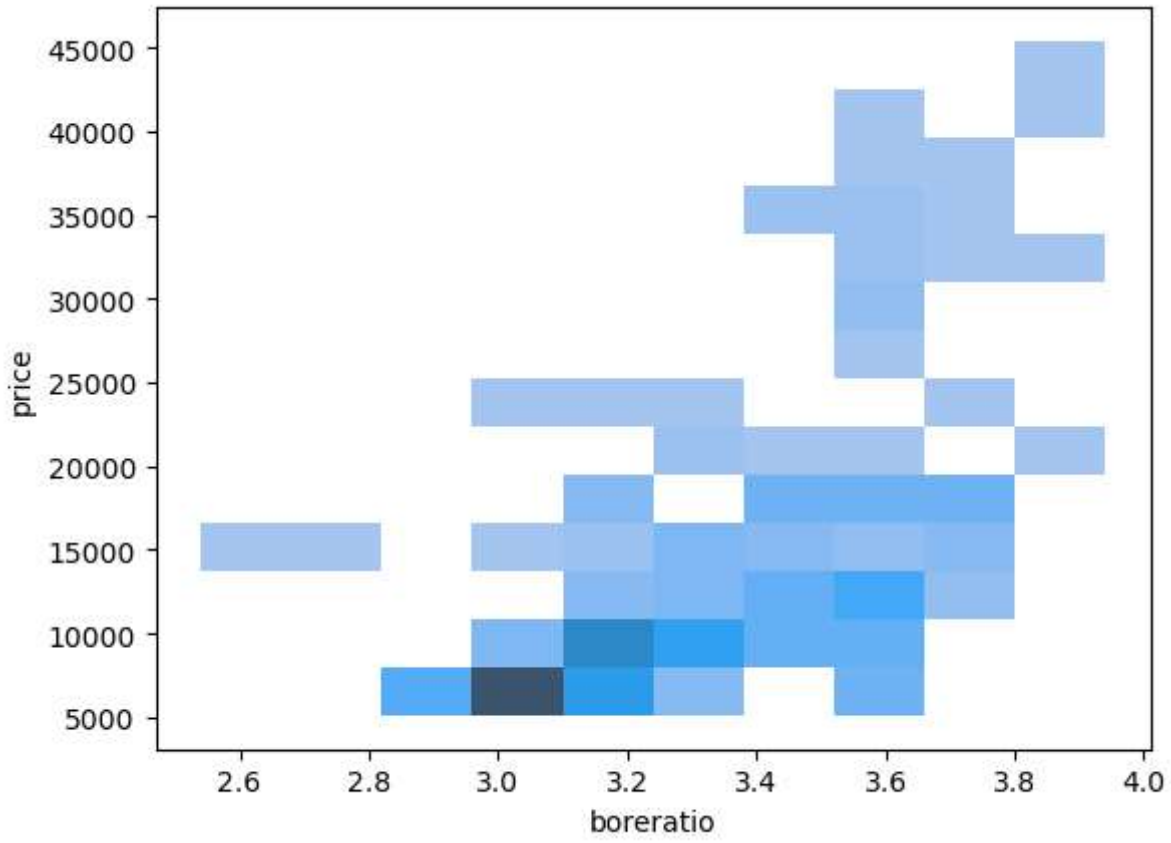
Out[47]:
```
array([3.47, 2.68, 3.19, 3.13, 3.5 , 3.31, 3.62, 2.91, 3.03, 2.97, 3.34,
       3.6 , 2.92, 3.15, 3.43, 3.63, 3.54, 3.08, 3.33, 3.39, 3.76, 3.58,
       3.46, 3.8 , 3.78, 3.17, 3.35, 3.59, 2.99, 3.7 , 3.61, 3.94, 3.74,
       2.54, 3.05, 3.27, 3.24, 3.01])
```

In [48]:
```
df["boreratio"].value_counts()
```

Out[48]:
```
3.62    23
3.19    20
3.15    15
3.03    12
2.97    12
3.46     9
3.31     8
3.43     8
3.78     8
3.27     7
2.91     7
3.58     6
3.39     6
3.33     6
3.05     6
3.54     6
3.70     5
3.01     5
3.35     4
3.17     3
3.59     3
3.74     3
3.47     2
3.94     2
3.24     2
3.63     2
3.13     2
3.80     2
3.50     2
2.54     1
3.08     1
3.61     1
3.34     1
2.68     1
3.60     1
2.92     1
3.76     1
2.99     1
Name: boreratio, dtype: int64
```

In [49]:
```
sns.histplot(x="boreratio" , y = "price" , data = df)
```

Out[49]:
```
<AxesSubplot:xlabel='boreratio', ylabel='price'>
```



In [50]:
```
df["stroke"].unique()
```

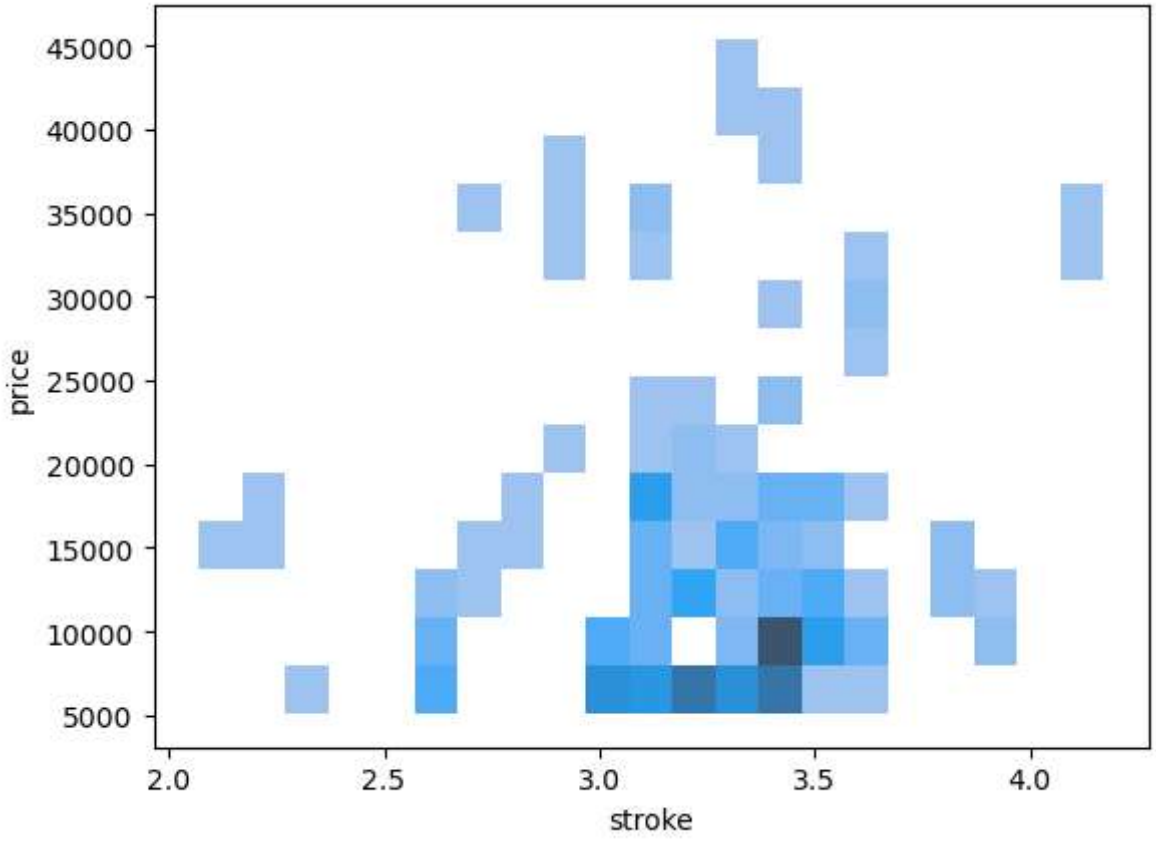
Out[50]:
```
array([2.68 , 3.47 , 3.4  , 2.8  , 3.19 , 3.39 , 3.03 , 3.11 , 3.23 ,
       3.46 , 3.9  , 3.41 , 3.07 , 3.58 , 4.17 , 2.76 , 3.15 , 3.255,
       3.16 , 3.64 , 3.1  , 3.35 , 3.12 , 3.86 , 3.29 , 3.27 , 3.52 ,
       2.19 , 3.21 , 2.9  , 2.07 , 2.36 , 2.64 , 3.08 , 3.5  , 3.54 ,
       2.87 ])
```

In [51]:
```
df["stroke"].value_counts()
```

Out[51]:
```
3.400    20
3.230    14
3.150    14
3.030    14
3.390    13
2.640    11
3.290     9
3.350     9
3.460     8
3.110     6
3.270     6
3.410     6
3.070     6
3.580     6
3.190     6
3.500     6
3.640     5
3.520     5
3.860     4
3.540     4
3.470     4
3.255     4
3.900     3
2.900     3
3.100     2
4.170     2
2.800     2
2.190     2
3.080     2
2.680     2
2.360     1
3.160     1
2.070     1
3.210     1
3.120     1
2.760     1
2.870     1
Name: stroke, dtype: int64
```

In [52]:
```
sns.histplot(x="stroke" , y = "price" , data = df)
```

Out[52]:
```
<AxesSubplot:xlabel='stroke', ylabel='price'>
```



In [53]:
```
df["horsepower"].unique()
```

Out[53]:
```
array([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], dtype=int64)
```

In [54]:
```
df["horsepower"].value_counts()
```

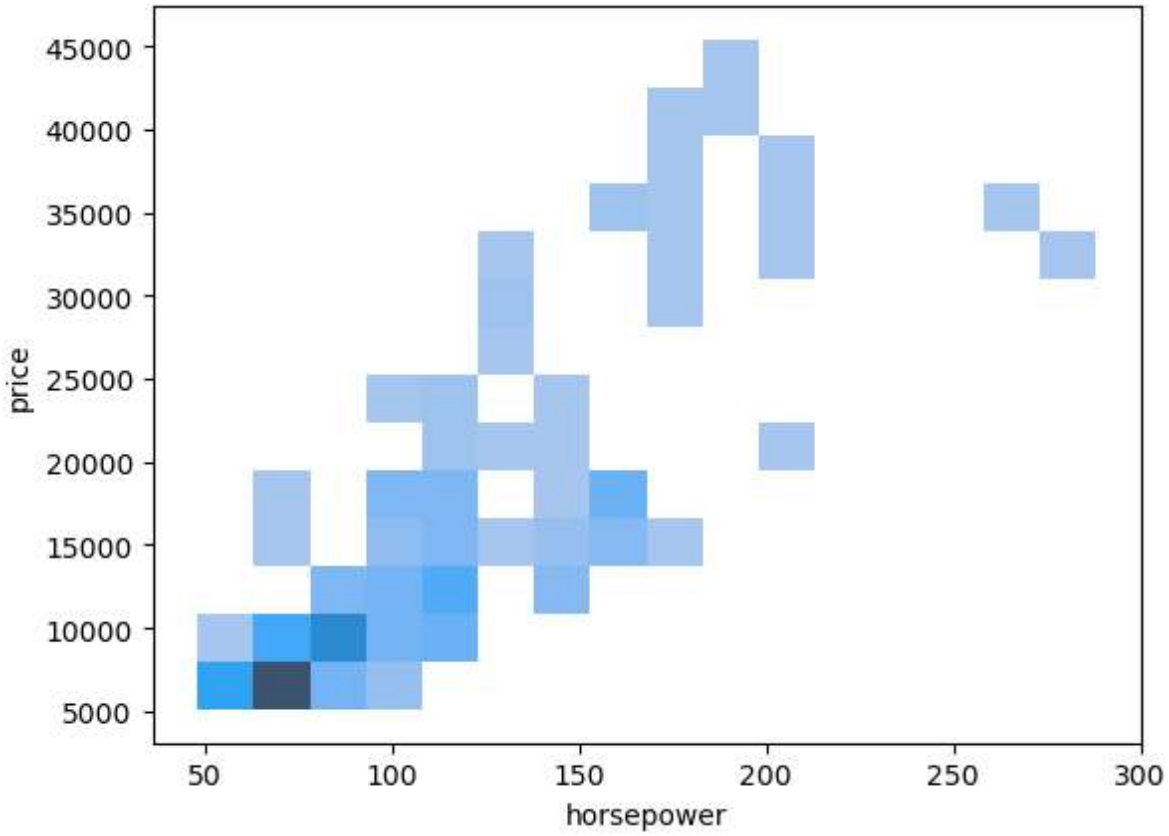
Out[54]:

68	19
70	11
69	10
116	9
110	8
95	7
114	6
160	6
101	6
62	6
88	6
145	5
76	5
97	5
84	5
90	5
82	5
102	5
92	4
111	4
123	4
86	4
207	3
73	3
182	3
121	3
85	3
152	3
176	2
94	2
56	2
112	2
161	2
184	2
155	2
156	2
52	2
100	2
162	2
140	1
115	1
134	1
78	1
142	1
288	1
143	1
48	1
200	1
58	1
55	1
60	1
175	1
154	1
72	1
120	1
64	1
135	1

```
262     1
106     1
Name: horsepower, dtype: int64
```

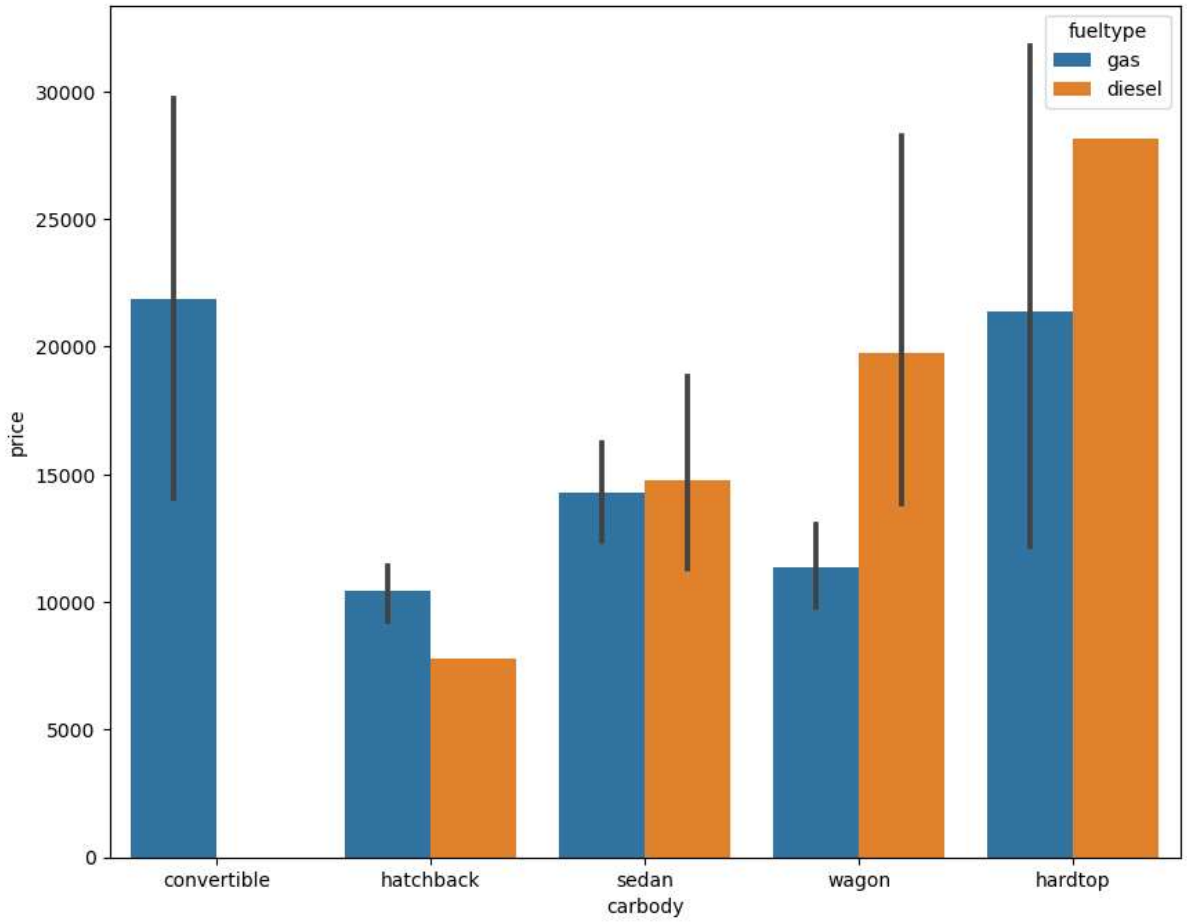
In [55]:
```
sns.histplot(x="horsepower" , y = "price" , data = df)
```

Out[55]:
```
<AxesSubplot:xlabel='horsepower', ylabel='price'>
```



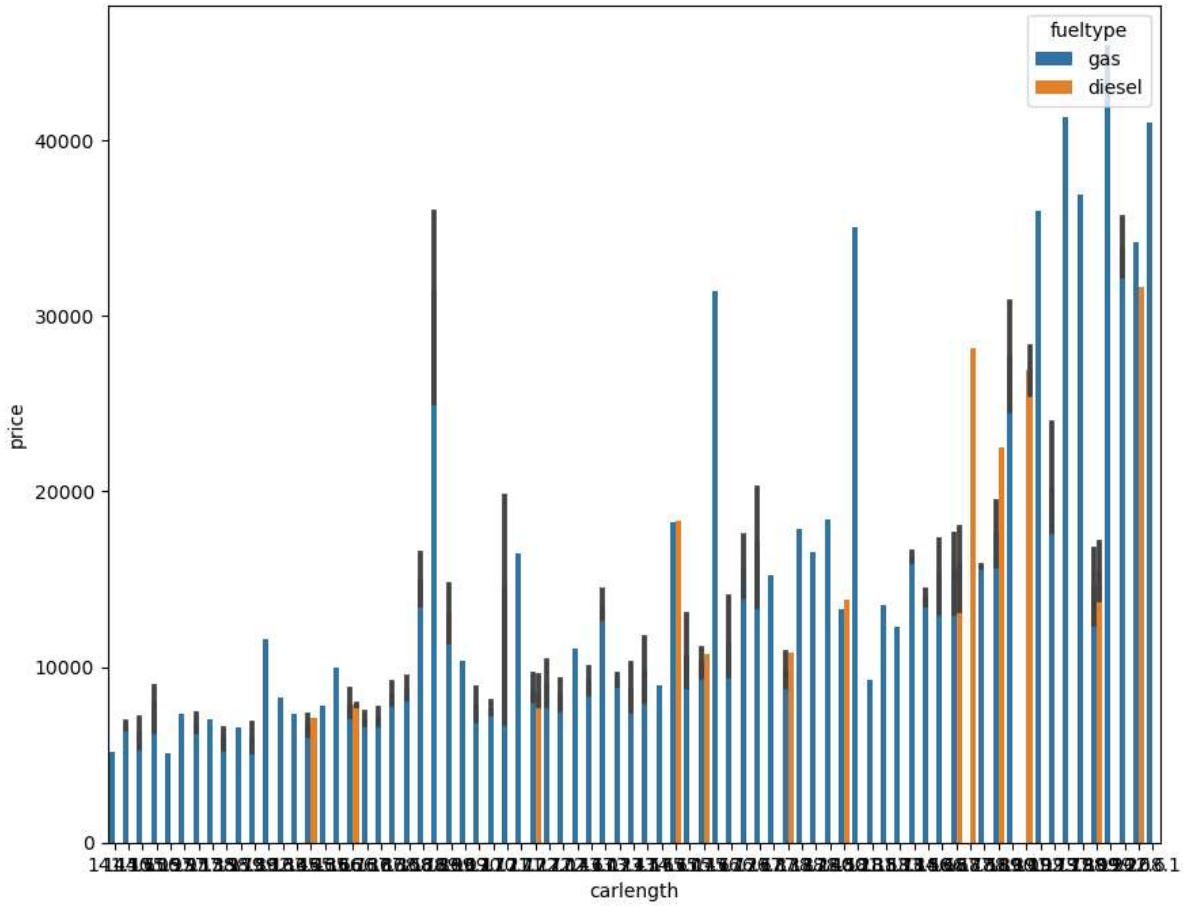
In [56]:
```
plt.figure(figsize=(10,8))
sns.barplot(x= "carbody" , y = "price" , data = df , hue = "fueltype"  )
```

Out[56]: <AxesSubplot:xlabel='carbody', ylabel='price'>



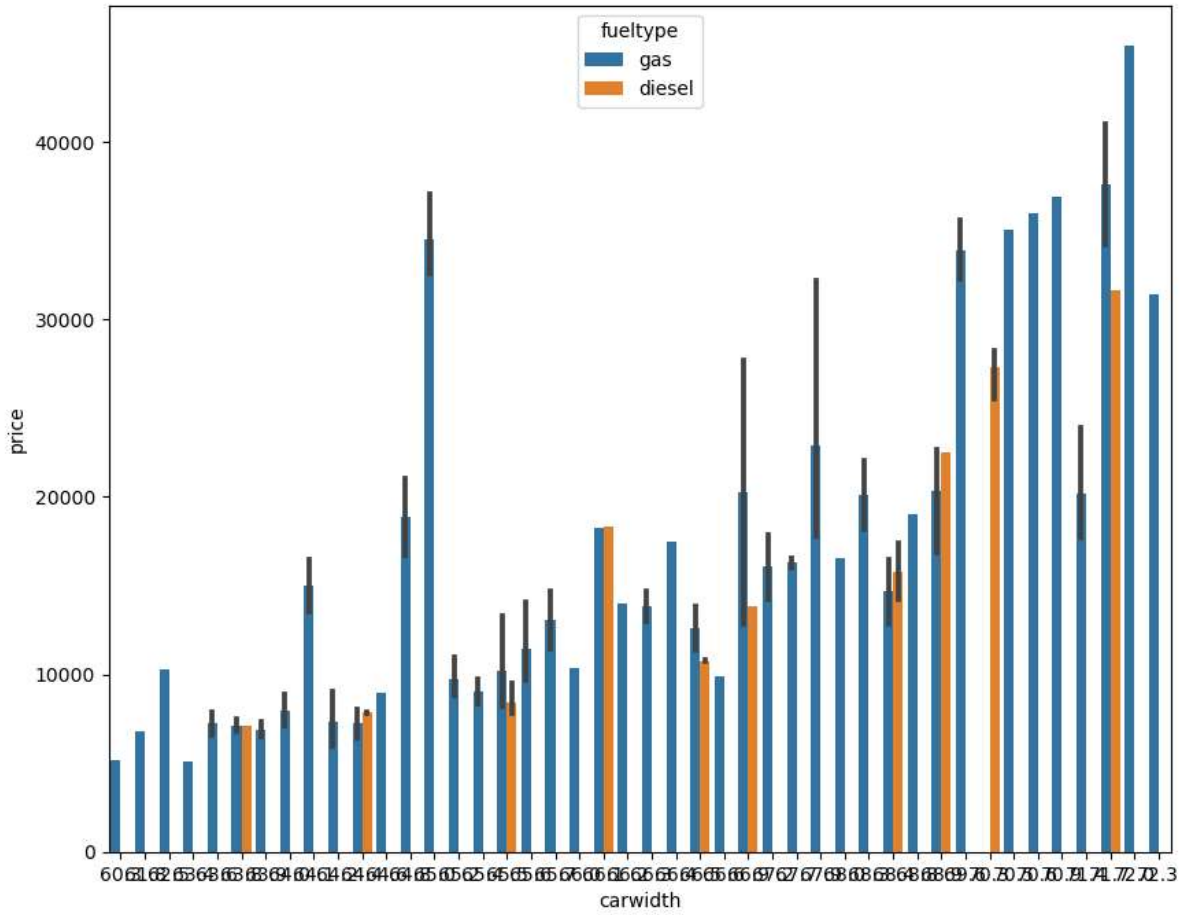
In [57]:
```
plt.figure(figsize=(10,8))
sns.barplot(x= "carlength" , y = "price" , data = df , hue = "fueltype"  )
```

Out[57]: <AxesSubplot:xlabel='carlength', ylabel='price'>



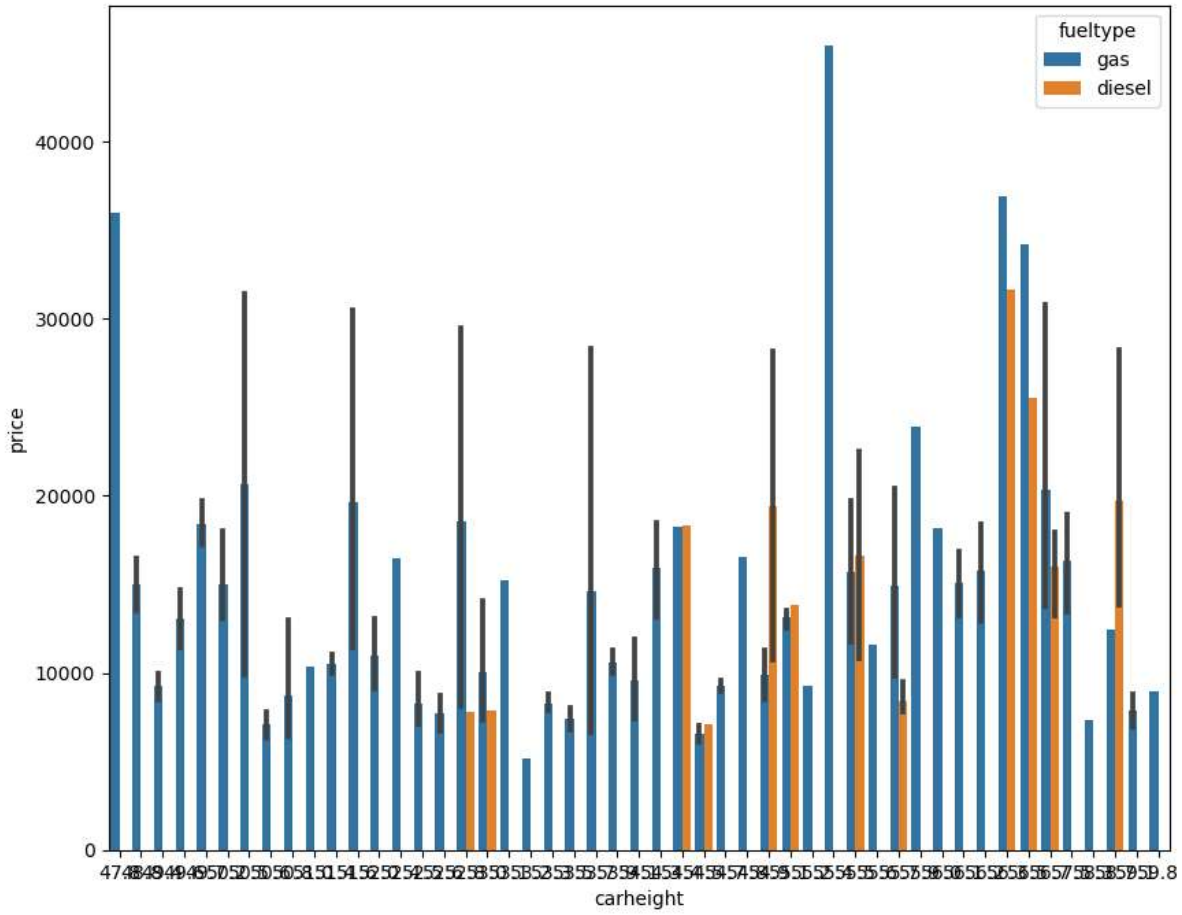
In [58]:
```
plt.figure(figsize=(10,8))
sns.barplot(x= "carwidth" , y = "price" , data = df , hue = "fueltype"  )
```

Out[58]: <AxesSubplot:xlabel='carwidth', ylabel='price'>



In [59]:
```
plt.figure(figsize=(10,8))
sns.barplot(x= "carheight" , y = "price" , data = df , hue = "fueltype"  )
```

Out[59]: <AxesSubplot:xlabel='carheight', ylabel='price'>



12. Split the data

In [60]:
```
x = df.drop( "price"      , axis = 1)
```

In [61]:
```
y = df["price"]
```

In [62]:
```
x
```

Out[62]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	en
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	
...	...	...	...	...	...	...	...	...	...
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	

205 rows × 25 columns

In [63]:
```
x.dtypes
```

Out[63]:
```
car_ID              int64
symboling           int64
CarName            object
fueltype           object
aspiration         object
doornumber         object
carbody            object
drivewheel         object
enginelocation     object
wheelbase         float64
carlength         float64
carwidth          float64
carheight         float64
curbweight          int64
enginetype         object
cylindernumber     object
enginesize          int64
fuelsystem         object
boreratio         float64
stroke            float64
compressionratio  float64
horsepower          int64
peakrpm             int64
citympg             int64
highwaympg          int64
dtype: object
```

In [64]:
```
y
```

Out[64]:
```
0      13495.0
1      16500.0
2      16500.0
3      13950.0
4      17450.0
        ...
200    16845.0
201    19045.0
202    21485.0
203    22470.0
204    22625.0
Name: price, Length: 205, dtype: float64
```

13. Let's Standardize iris data

In [65]:
```
x
```

Out[65]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	en
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	
...	...	...	...	...	...	...	...	...	...
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	

205 rows × 25 columns

In [66]:
```
y
```

Out[66]:
```
0      13495.0
1      16500.0
2      16500.0
3      13950.0
4      17450.0
        ...
200    16845.0
201    19045.0
202    21485.0
203    22470.0
204    22625.0
Name: price, Length: 205, dtype: float64
```

14. encoding the Categorical data

In [67]:
```
x.dtypes
```

Out[67]:
```
car_ID              int64
symboling           int64
CarName            object
fueltype           object
aspiration         object
doornumber         object
carbody            object
drivewheel         object
enginelocation     object
wheelbase         float64
carlength         float64
carwidth          float64
carheight         float64
curbweight          int64
enginetype         object
cylindernumber     object
enginesize          int64
fuelsystem         object
boreratio         float64
stroke            float64
compressionratio  float64
horsepower          int64
peakrpm             int64
citympg             int64
highwaympg          int64
dtype: object
```

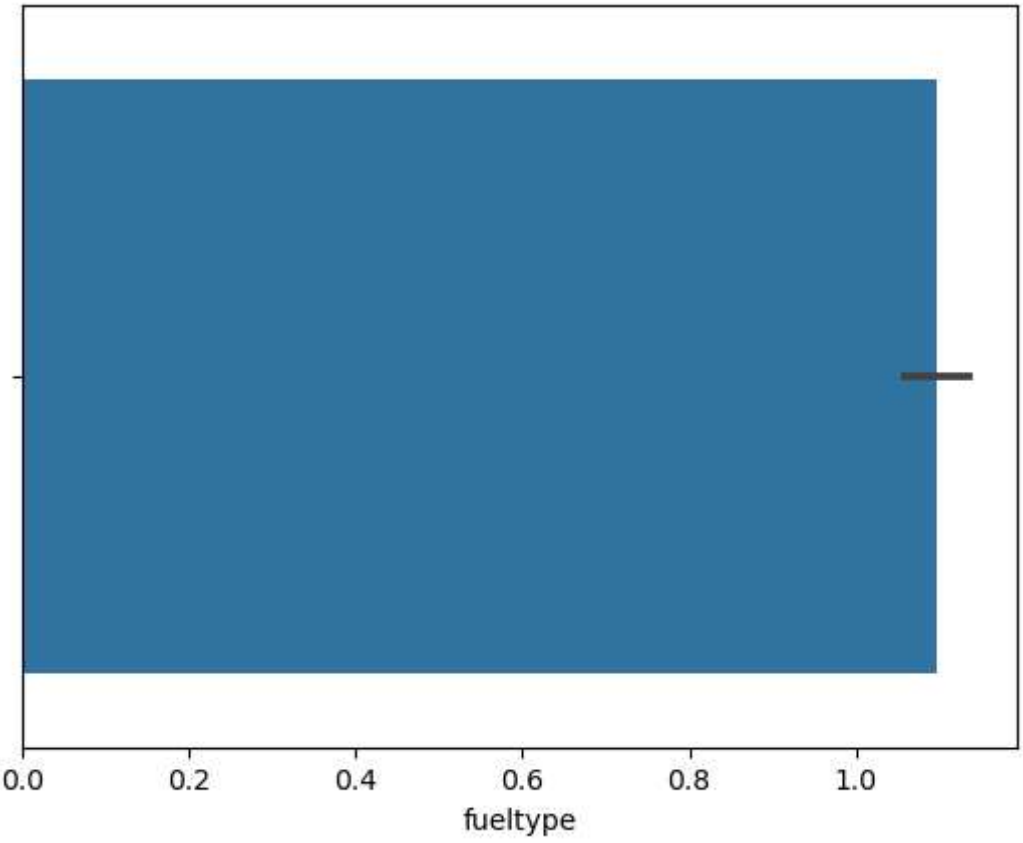
In [68]:
```
x["fueltype"].unique()
```

Out[68]:
```
array(['gas', 'diesel'], dtype=object)
```

In [69]:
```
x.replace( { "gas" : 1 ,"diesel" : 2  } , inplace=True )
```

In [70]:
```
sns.barplot(x["fueltype"])
```

Out[70]:
```
<AxesSubplot:xlabel='fueltype'>
```



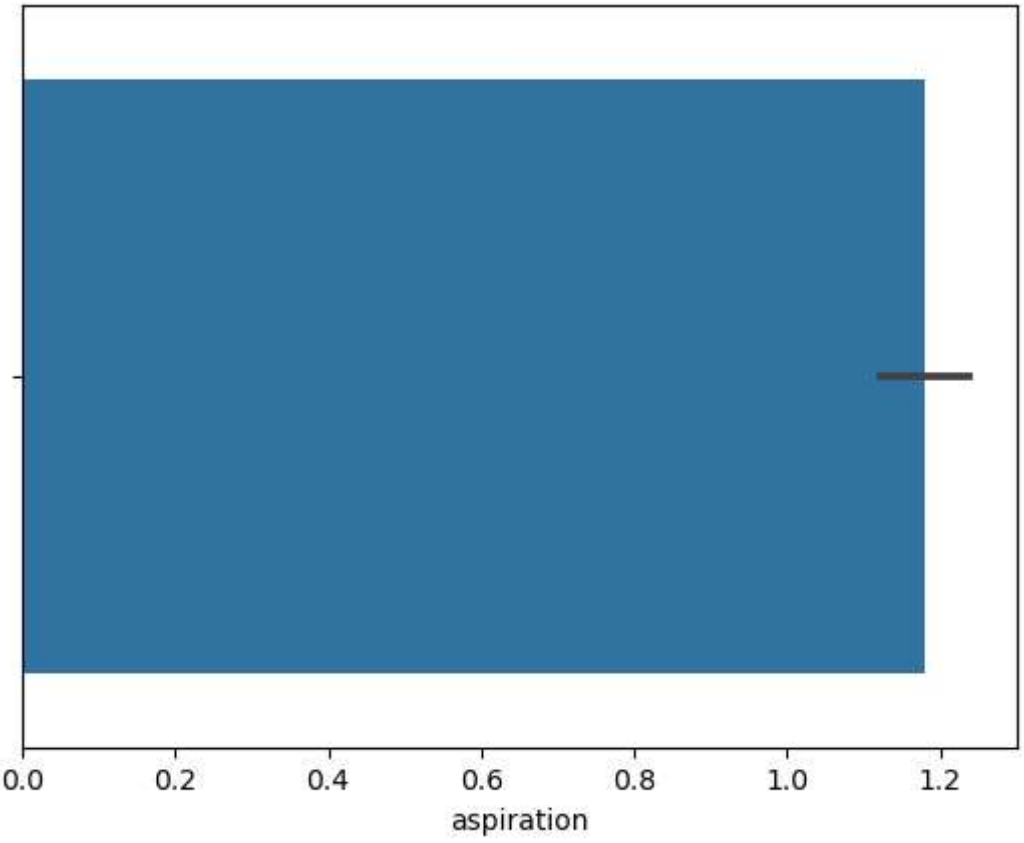
In [71]:
```
x["aspiration"].unique()
```

Out[71]:
```
array(['std', 'turbo'], dtype=object)
```

In [72]:
```
x.replace( { "std" : 1 ,"turbo" : 2  } , inplace=True )
```

In [73]:
```
sns.barplot(x["aspiration"])
```

Out[73]:
```
<AxesSubplot:xlabel='aspiration'>
```



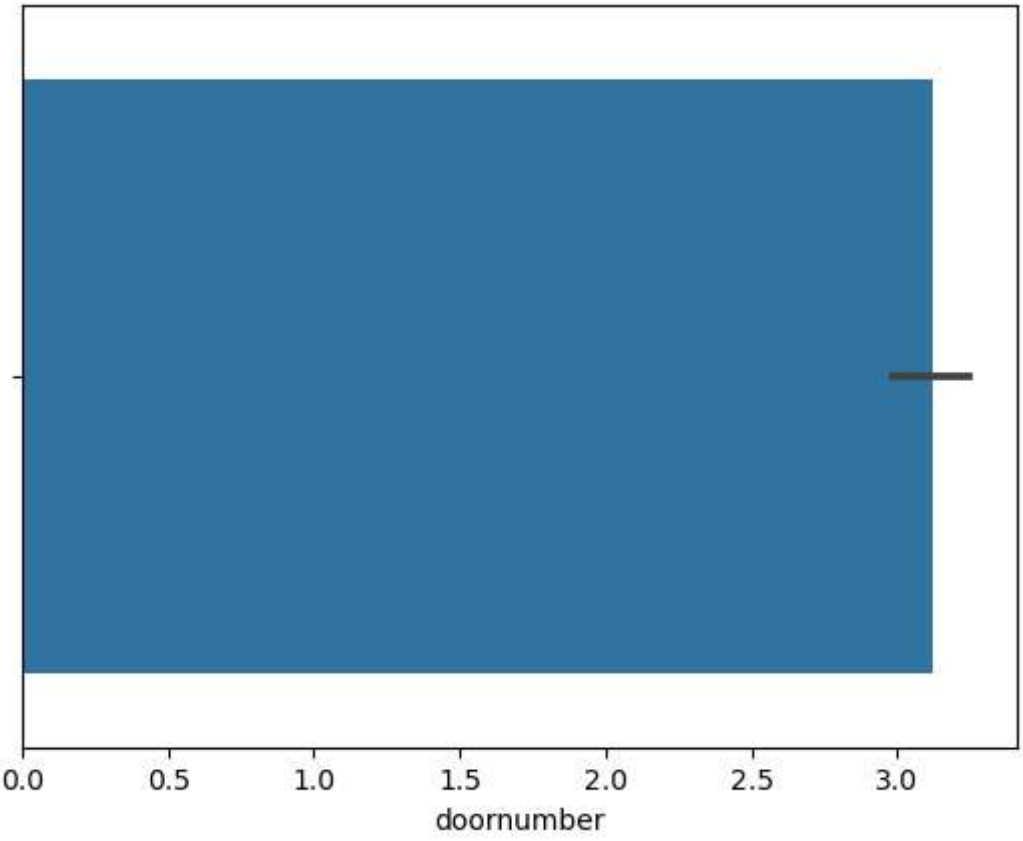
In [74]:
```
x["doornumber"].unique()
```

Out[74]:
```
array(['two', 'four'], dtype=object)
```

In [75]:
```
x.replace( { "two" : 2 ,"four" : 4  } , inplace=True )
```

In [76]:
```
sns.barplot(x["doornumber"])
```

Out[76]:
```
<AxesSubplot:xlabel='doornumber'>
```



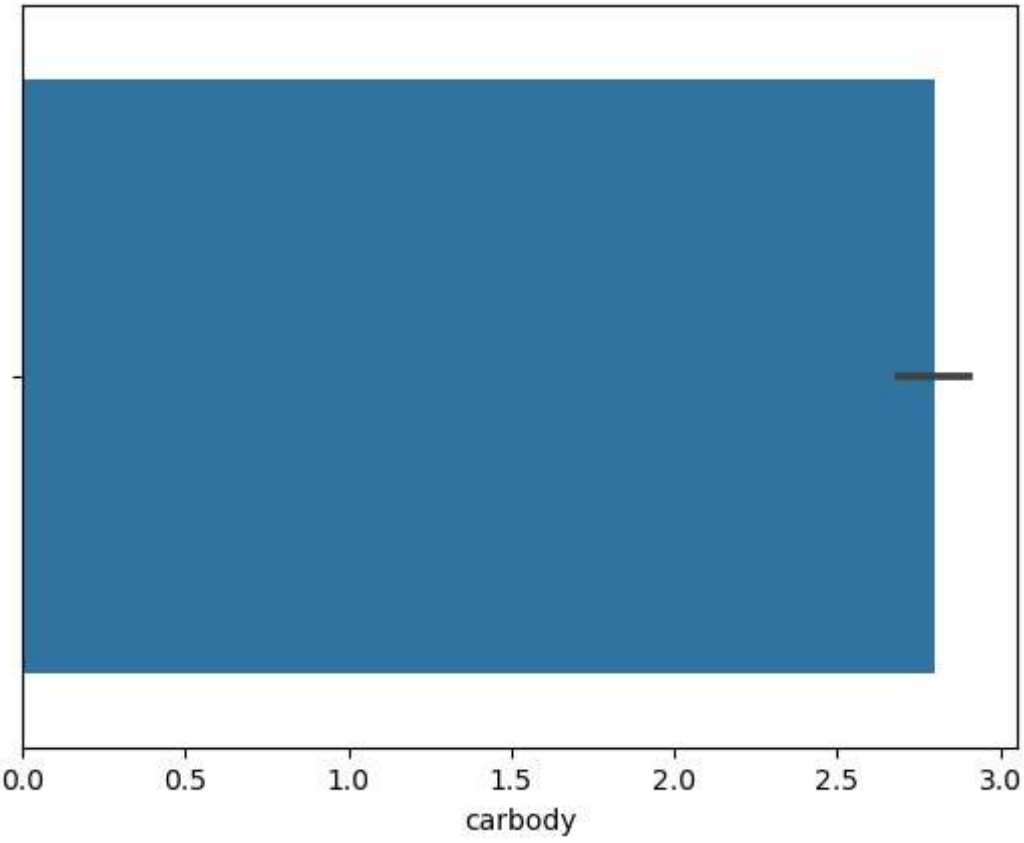
In [77]:
```
x["carbody"].unique()
```

Out[77]:
```
array(['convertible', 'hatchback', 'sedan', 'wagon', 'hardtop'],
      dtype=object)
```

In [78]:
```
x.replace( { "convertible" : 1 ,"hatchback" : 2 , "sedan" : 3 , "wagon" : 4  ,
```

In [79]:
```
sns.barplot(x["carbody"])
```

Out[79]: <AxesSubplot:xlabel='carbody'>



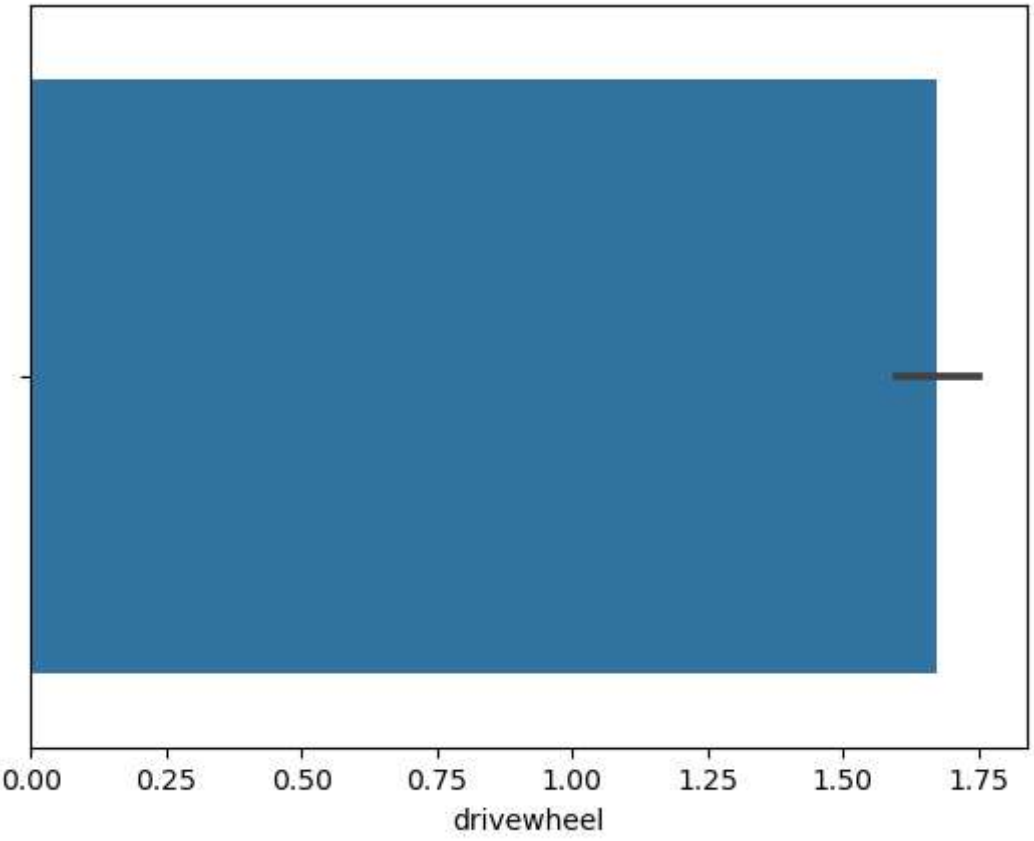
In [80]:
```
x["drivewheel"].unique()
```

Out[80]: array(['rwd', 'fwd', '4wd'], dtype=object)

In [81]:
```
x.replace( { "rwd" : 1 ,"fwd" : 2 , "4wd" : 3 } , inplace=True )
```

In [82]:
```
sns.barplot(x["drivewheel"])
```

Out[82]: <AxesSubplot:xlabel='drivewheel'>



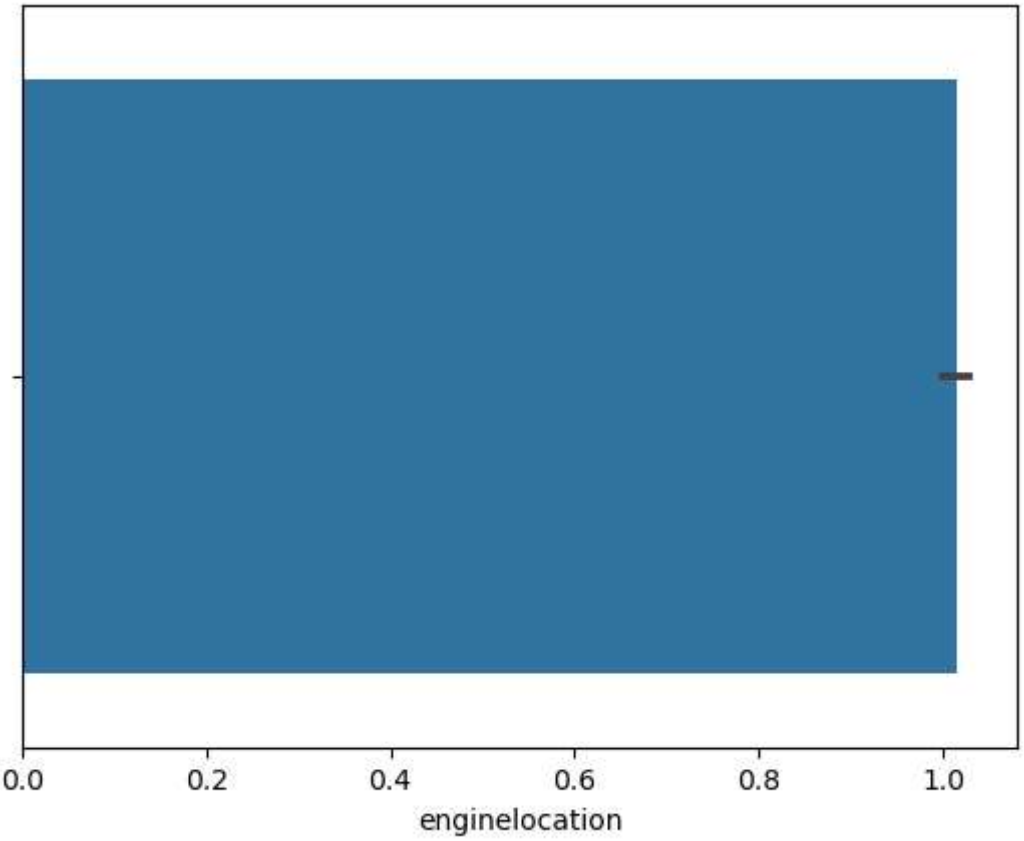
In [83]:
```
x["enginelocation"].unique()
```

Out[83]: array(['front', 'rear'], dtype=object)

In [84]:
```
x.replace( { "front" : 1 ,"rear" : 2  } , inplace=True )
```

In [85]:
```
sns.barplot(x["enginelocation"])
```

Out[85]: <AxesSubplot:xlabel='enginelocation'>



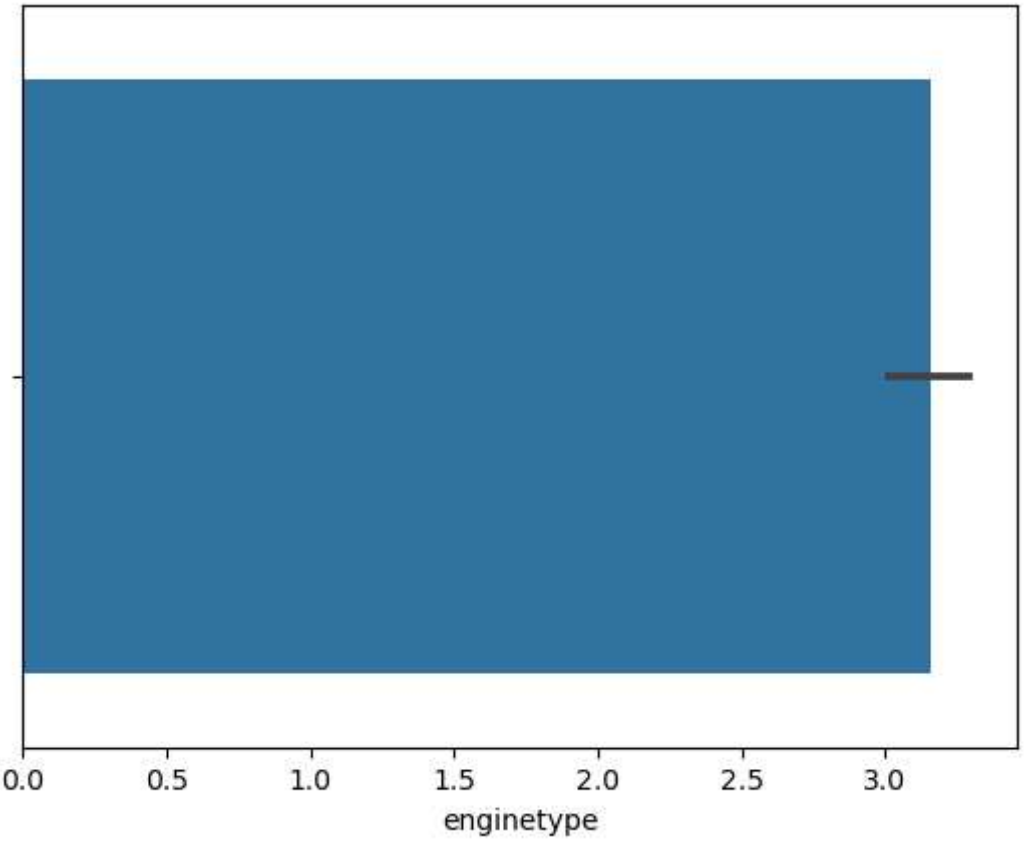
In [86]:
```
x["enginetype"].unique()
```

Out[86]: array(['dohc', 'ohcv', 'ohc', 'l', 'rotor', 'ohcf', 'dohcv'], dtype=object)

In [87]:
```
x.replace( { "dohc" : 1 ,"ohcv" : 2 , "ohc" : 3 , "l" : 4  , "rotor" : 5 , "ohc
```

In [88]:
```
sns.barplot(x["enginetype"])
```

Out[88]: <AxesSubplot:xlabel='enginetype'>



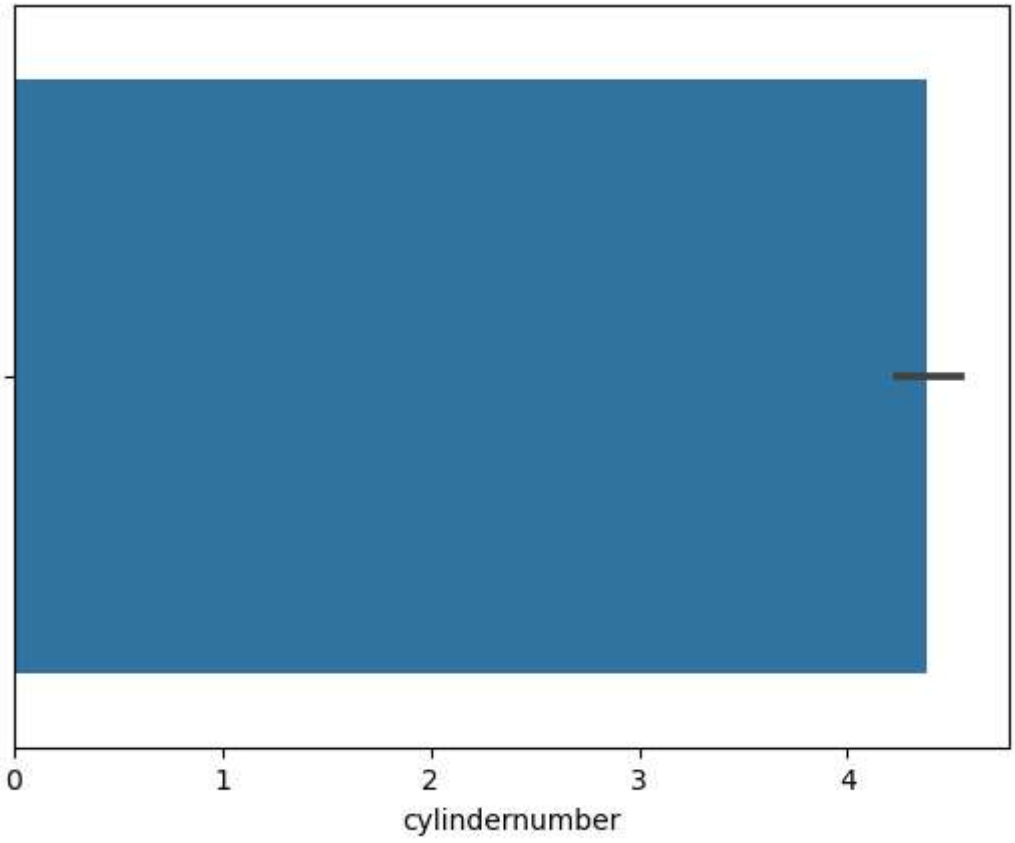
In [89]:
```
x["cylindernumber"].unique()
```

Out[89]:
```
array([4, 'six', 'five', 'three', 'twelve', 2, 'eight'], dtype=object)
```

In [90]:
```
x.replace( { "six" : 6 ,"five" : 5 , "three" : 3 , "twelve" : 12  , "eight" : 8
```

In [91]:
```
sns.barplot(x["cylindernumber"])
```

Out[91]:
```
<AxesSubplot:xlabel='cylindernumber'>
```



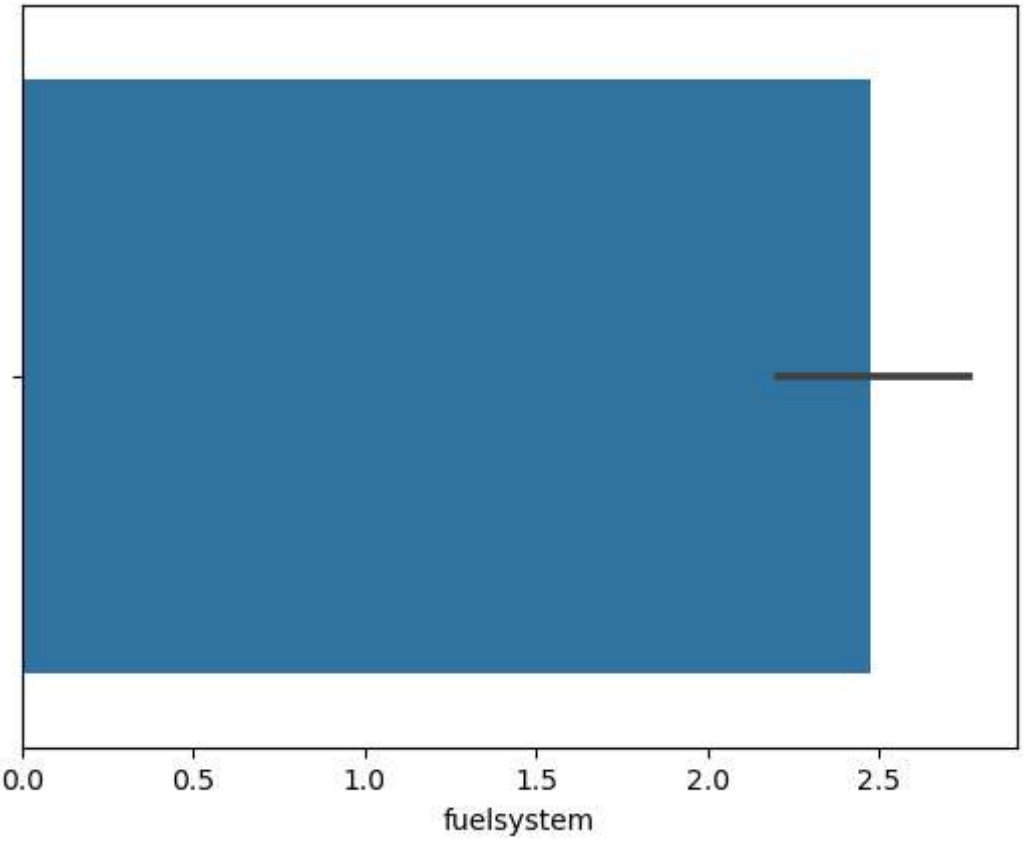
In [92]:
```
x["fuelsystem"].unique()
```

Out[92]:
```
array(['mpfi', '2bbl', 'mfi', '1bbl', 'spfi', '4bbl', 'idi', 'spdi'],
      dtype=object)
```

In [93]:
```
x.replace( { "mpfi" : 1 ,"2bbl" : 2 , "mfi" : 3 , "1bbl" : 4  , "spfi" : 5 , "4
```

In [94]:
```
sns.barplot(x["fuelsystem"])
```

Out[94]:
```
<AxesSubplot:xlabel='fuelsystem'>
```



In [95]:
```
x["aspiration"].unique()
```

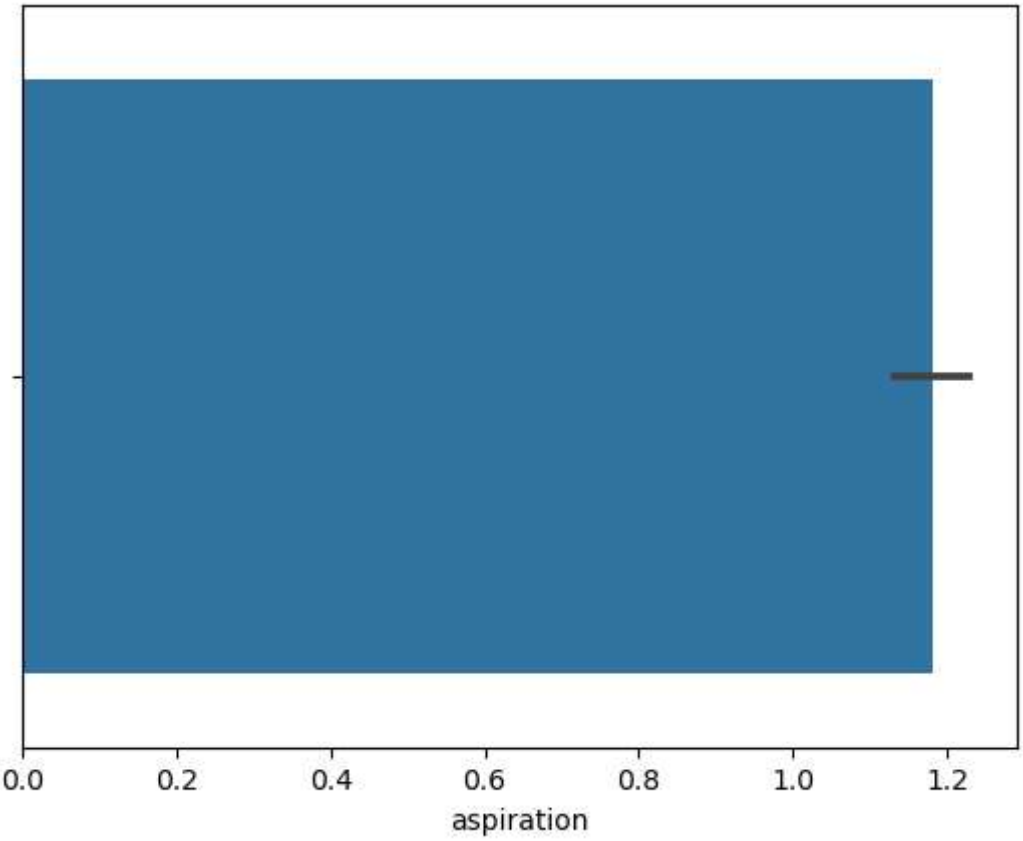
Out[95]:
```
array([1, 2], dtype=int64)
```

In [96]:
```
x["aspiration"].value_counts()
```

Out[96]:
```
1    168
2     37
Name: aspiration, dtype: int64
```

In [97]:
```
sns.barplot(x["aspiration"])
```

Out[97]: <AxesSubplot:xlabel='aspiration'>



In [98]:
```
x.replace( { "turbo" : 2} , inplace=True )
```

In [99]:
```
x["CarName"].unique()
```

Out[99]:
```
array(['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'], dtype=object)
```

In [100]:
```
x["CarName"].value_counts()
```

Out[100]:
```
toyota corona           6
toyota corolla          6
peugeot 504             6
subaru dl               4
mitsubishi mirage g4    3
                       ..
mazda glc 4             1
mazda rx2 coupe         1
maxda glc deluxe        1
maxda rx3               1
volvo 246               1
Name: CarName, Length: 147, dtype: int64
```

In [101]:
```
Car_name = df.groupby("CarName")["price"].mean().to_dict()
```

In [102]:
```
Car_name
```

Out[102]:
```
{'Nissan versa': 5499.0,
 'alfa-romero Quadrifoglio': 16500.0,
 'alfa-romero giulia': 13495.0,
 'alfa-romero stelvio': 16500.0,
 'audi 100 ls': 13950.0,
 'audi 100ls': 17580.0,
 'audi 4000': 23875.0,
 'audi 5000': 18920.0,
 'audi 5000s (diesel)': 17859.167,
 'audi fox': 15250.0,
 'bmw 320i': 16677.5,
 'bmw x1': 20970.0,
 'bmw x3': 28992.5,
 'bmw x4': 30760.0,
 'bmw x5': 41315.0,
 'bmw z4': 24565.0,
 'buick century': 28176.0,
 'buick century luxus (sw)': 28248.0,
 'buick century special': 40960.0,
```

In [103]:
```
x.replace( Car_name , inplace=True )
```

In [104]:
```
x
```

Out[104]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	engine
0	1	3	13495.0	1	1	2	1	1	
1	2	3	16500.0	1	1	2	1	1	
2	3	1	16500.0	1	1	2	2	1	
3	4	2	13950.0	1	1	4	3	2	
4	5	2	17580.0	1	1	4	3	3	
...	...	...	...	...	...	...	...	...	...
200	201	-1	14892.5	1	1	4	3	1	
201	202	-1	16230.0	1	2	4	3	1	
202	203	-1	18735.0	1	1	4	3	1	
203	204	-1	22470.0	2	2	4	3	1	
204	205	-1	20522.5	1	2	4	3	1	

205 rows × 25 columns

In [105]:
```
x.dtypes
```

Out[105]:
```
car_ID              int64
symboling           int64
CarName           float64
fueltype            int64
aspiration          int64
doornumber          int64
carbody             int64
drivewheel          int64
enginelocation      int64
wheelbase         float64
carlength         float64
carwidth          float64
carheight         float64
curbweight          int64
enginetype          int64
cylindernumber      int64
enginesize          int64
fuelsystem          int64
boreratio         float64
stroke            float64
compressionratio  float64
horsepower          int64
peakrpm             int64
citympg             int64
highwaympg          int64
dtype: object
```

15. Train-Test-split

In [106]:
```
from sklearn.model_selection import  train_test_split

x_train , x_test , y_train , y_test =  train_test_split(x,y,test_size=0.20)
```

In [107]:
```
x_train
```

Out[107]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	en
204	205	-1	20522.500000	1	2	4	3	1	
163	164	1	12308.000000	1	1	2	3	1	
134	135	3	13605.000000	1	1	2	2	2	
167	168	2	8449.000000	1	1	2	5	1	
80	81	3	11242.333333	1	2	2	2	2	
...	...	...	...	...	...	...	...	...	...
50	51	1	5195.000000	1	1	2	2	2	
85	86	1	6989.000000	1	1	4	3	2	
88	89	-1	11242.333333	1	1	4	3	2	
65	66	0	15062.500000	1	1	4	3	1	
9	10	0	17859.167000	1	2	2	2	3	

164 rows × 25 columns

In [108]:
```
x_test
```

Out[108]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	en
28	29	-1	8921.000000	1	1	4	4	2	
146	147	0	7463.000000	1	1	4	4	2	
58	59	3	15645.000000	1	1	2	2	1	
114	115	0	17075.000000	2	2	4	4	1	
10	11	2	16677.500000	1	1	2	3	1	
166	167	1	9538.000000	1	1	2	2	1	
22	23	1	6377.000000	1	1	2	2	2	
178	179	3	12308.000000	1	1	2	2	1	
73	74	0	40960.000000	1	1	4	3	1	
200	201	-1	14892.500000	1	1	4	3	1	
43	44	0	6785.000000	1	1	4	3	1	
18	19	2	5151.000000	1	1	2	2	2	
186	187	2	8495.000000	1	1	4	3	2	
20	21	0	6575.000000	1	1	4	3	2	
156	157	0	9148.000000	1	1	4	3	2	
195	196	-1	16230.000000	1	1	4	4	1	
128	129	3	37028.000000	1	1	2	1	1	
81	82	3	10352.333333	1	1	2	2	2	
8	9	1	23875.000000	1	2	4	3	2	
124	125	3	12764.000000	1	2	2	2	1	
104	105	3	17199.000000	1	1	2	2	1	
45	46	0	8916.500000	1	1	4	3	2	
177	178	-1	9148.000000	1	1	4	2	2	
102	103	0	14399.000000	1	1	4	4	2	
106	107	1	13324.000000	1	1	2	2	1	
59	60	1	10345.000000	1	1	2	2	2	
61	62	1	10595.000000	1	1	2	2	2	
56	57	3	15062.500000	1	1	2	2	1	
165	166	1	9298.000000	1	1	2	3	1	
144	145	0	9233.000000	1	1	4	3	3	
180	181	-1	12839.500000	1	1	4	3	1	
107	108	0	15435.833333	1	1	4	3	1	
95	96	1	7799.000000	1	1	2	2	2	
108	109	0	13200.000000	2	2	4	3	1	
171	172	2	9530.000000	1	1	2	2	1	

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	en
97	98	1	7999.000000	1	1	4	4	2	
137	138	2	15235.000000	1	2	4	3	2	
127	128	3	32714.250000	1	1	2	5	1	
3	4	2	13950.000000	1	1	4	3	2	
110	111	0	15435.833333	2	2	4	4	1	
151	152	1	9366.333333	1	1	2	2	2	

41 rows × 25 columns

In [109]:
```
y_train
```

Out[109]:
```
204    22625.000
163     8058.000
134    15040.000
167     8449.000
80      9959.000
          ...
50      5195.000
85      6989.000
88      9279.000
65     18280.000
9      17859.167
Name: price, Length: 164, dtype: float64
```

In [110]:
```
y_test
```

Out[110]:
```
28      8921.0
146     7463.0
58     15645.0
114    17075.0
10     16430.0
166     9538.0
22      6377.0
178    16558.0
73     40960.0
200    16845.0
43      6785.0
18      5151.0
186     8495.0
20      6575.0
156     6938.0
195    13415.0
128    37028.0
81      8499.0
8      23875.0
124    12764.0
104    17199.0
45      8916.5
177    11248.0
102    14399.0
106    18399.0
59      8845.0
61     10595.0
56     11845.0
165     9298.0
144     9233.0
180    15690.0
107    11900.0
95      7799.0
108    13200.0
171    11549.0
97      7999.0
137    18620.0
127    34028.0
3      13950.0
110    13860.0
151     6338.0
Name: price, dtype: float64
```

16 Apply Models

Linear Regression

In [111]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

In [112]:
```
model.fit(x_train, y_train)
```

Out[112]:
```
LinearRegression()
```

In [113]:
```
y_Pred = model.predict(x_test)
```

check model

In [114]:
```
from sklearn.metrics import r2_score
```

In [115]:
```
r2 = r2_score(y_test,y_Pred)
```

In [116]:
```
r2
```

Out[116]:
```
0.9545419693248489
```

In [117]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

In [118]:
```
MAE = mean_absolute_error(y_test,y_Pred)
MAE
```

Out[118]:
```
1289.3063096894616
```

In [119]:
```
np.sqrt(MAE) # root mean squared error
```

Out[119]:
```
35.90691172587057
```

In [120]:
```
MSE = mean_squared_error(y_test,y_Pred)
MSE
```

Out[120]:
```
2824452.350283293
```

In [121]:
```
model.score()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_2828\3947582719.py in <module>
----> 1 model.score()

TypeError: score() missing 2 required positional arguments: 'X' and 'y'
```

17. result summary

In [122]:
```
import statsmodels.api as sm
x_train_Sm =sm.add_constant(x_train)
x_train_Sm =sm.add_constant(x_train)

ls=sm.OLS(y_train,x_train).fit()
print(ls.summary())
```

```
                            OLS Regression Results
==============================================================================
==========
Dep. Variable:                  price   R-squared (uncentered):
0.993
Model:                            OLS   Adj. R-squared (uncentered):
0.991
Method:                 Least Squares   F-statistic:
764.7
Date:                Thu, 06 Apr 2023   Prob (F-statistic):
3.97e-136
Time:                        16:02:33   Log-Likelihood:
-1409.8
No. Observations:                 164   AIC:
2870.
Df Residuals:                     139   BIC:
2947.
Df Model:                          25
Covariance Type:            nonrobust
==============================================================================
=======
                     coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-------
car_ID              -3.2379      2.399     -1.349      0.179      -7.982
1.506
symboling          133.1248    151.471      0.879      0.381    -166.360
432.610
CarName              0.7417      0.037     19.844      0.000       0.668
0.816
fueltype          3816.8925   3821.527      0.999      0.320   -3738.945
1.14e+04
aspiration         -75.5352    534.722     -0.141      0.888   -1132.776
981.706
doornumber         345.3689    178.347      1.936      0.055      -7.255
697.993
carbody           -163.1828    175.734     -0.929      0.355    -510.641
184.275
drivewheel        -135.0243    349.387     -0.386      0.700    -825.825
555.776
enginelocation    1049.5758   1672.216      0.628      0.531   -2256.692    4
355.844
wheelbase           70.0847     62.183      1.127      0.262     -52.862
193.032
carlength          -18.9120     30.896     -0.612      0.541     -79.999
42.175
carwidth            -7.9637    121.686     -0.065      0.948    -248.559
232.631
carheight          -12.3235     75.383     -0.163      0.870    -161.368
136.721
curbweight           0.5710      0.892      0.640      0.523      -1.192
2.334
enginetype         128.4921    189.170      0.679      0.498    -245.532
502.516
cylindernumber    -931.1352    347.860     -2.677      0.008   -1618.916    -
243.355
```

```
enginesize          54.2766     15.479      3.506      0.001      23.671
84.882
fuelsystem        -104.3066     89.223     -1.169      0.244    -280.716
72.102
boreratio        -2111.4250    906.739     -2.329      0.021   -3904.210    -
318.640
stroke           -1047.3797    516.829     -2.027      0.045   -2069.243
-25.517
compressionratio  -147.1131    264.330     -0.557      0.579    -669.740
375.514
horsepower          17.6581     10.248      1.723      0.087      -2.604
37.920
peakrpm              0.5614      0.394      1.424      0.157      -0.218
1.341
citympg            102.5401    101.897      1.006      0.316     -98.929
304.009
highwaympg        -132.5900     91.179     -1.454      0.148    -312.868
47.688
==============================================================================
=
Omnibus:                       26.431   Durbin-Watson:                   2.03
7
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              103.48
3
Skew:                           0.466   Prob(JB):                     3.38e-2
3
Kurtosis:                       6.778   Cond. No.                     5.59e+0
5
==============================================================================
=

Notes:
[1] R² is computed without centering (uncentered) since the model does not co
ntain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
[3] The condition number is large, 5.59e+05. This might indicate that there a
re
strong multicollinearity or other numerical problems.
```

In [ ]: