11-2-linear-regression-1

April 27, 2024

1 Seatwork 11.1 Exploratory Data Analysis for Machine Learning

[1]:
```
pip install ucimlrepo
```

```
Collecting ucimlrepo
  Downloading ucimlrepo-0.0.6-py3-none-any.whl (8.0 kB)
Installing collected packages: ucimlrepo
Successfully installed ucimlrepo-0.0.6
```

[2]:
```
from ucimlrepo import fetch_ucirepo
# fetch dataset
automobile = fetch_ucirepo(id=10)
# data (as pandas dataframes)
X = automobile.data.features
y = automobile.data.targets
```

[3]:
```
X
```

[3]:

	price	highway-mpg	city-mpg	peak-rpm	horsepower	compression-ratio	\
0	13495.0	27	21	5000.0	111.0	9.0	
1	16500.0	27	21	5000.0	111.0	9.0	
2	16500.0	26	19	5000.0	154.0	9.0	
3	13950.0	30	24	5500.0	102.0	10.0	
4	17450.0	22	18	5500.0	115.0	8.0	
..	…	…	…	…	…	…	
200	16845.0	28	23	5400.0	114.0	9.5	
201	19045.0	25	19	5300.0	160.0	8.7	
202	21485.0	23	18	5500.0	134.0	8.8	
203	22470.0	27	26	4800.0	106.0	23.0	
204	22625.0	25	19	5400.0	114.0	9.5	

	stroke	bore	fuel-system	engine-size	…	length	wheel-base	\
0	2.68	3.47	mpfi	130	…	168.8	88.6	
1	2.68	3.47	mpfi	130	…	168.8	88.6	
2	3.47	2.68	mpfi	152	…	171.2	94.5	
3	3.40	3.19	mpfi	109	…	176.6	99.8	
4	3.40	3.19	mpfi	136	…	176.6	99.4	

```
..      …   …         …           …  …     …        …
200    3.15  3.78        mpfi          141  …   188.8      109.1
201    3.15  3.78        mpfi          141  …   188.8      109.1
202    2.87  3.58        mpfi          173  …   188.8      109.1
203    3.40  3.01         idi          145  …   188.8      109.1
204    3.15  3.78        mpfi          141  …   188.8      109.1

     engine-location  drive-wheels   body-style  num-of-doors  aspiration  \
0              front           rwd  convertible           2.0         std
1              front           rwd  convertible           2.0         std
2              front           rwd    hatchback           2.0         std
3              front           fwd        sedan           4.0         std
4              front           4wd        sedan           4.0         std
..               …           …          …           …         …
200            front           rwd        sedan           4.0         std
201            front           rwd        sedan           4.0       turbo
202            front           rwd        sedan           4.0         std
203            front           rwd        sedan           4.0       turbo
204            front           rwd        sedan           4.0       turbo

    fuel-type         make normalized-losses
0         gas  alfa-romero               NaN
1         gas  alfa-romero               NaN
2         gas  alfa-romero               NaN
3         gas         audi             164.0
4         gas         audi             164.0
..        …          …               …
200       gas        volvo              95.0
201       gas        volvo              95.0
202       gas        volvo              95.0
203    diesel        volvo              95.0
204       gas        volvo              95.0

[205 rows x 25 columns]
```

[4]:
```
y
```

[4]:
```
     symboling
0            3
1            3
2            1
3            2
4            2
..         …
200         -1
201         -1
202         -1
```

```
203         -1
204         -1

[205 rows x 1 columns]
```

[6]:
```
!pip install hvplot
```

```
Collecting hvplot
  Downloading hvplot-0.9.2-py2.py3-none-any.whl (1.8 MB)
                           1.8/1.8 MB
8.2 MB/s eta 0:00:00
Requirement already satisfied: bokeh>=1.0.0 in
/usr/local/lib/python3.10/dist-packages (from hvplot) (3.3.4)
Requirement already satisfied: colorcet>=2 in /usr/local/lib/python3.10/dist-
packages (from hvplot) (3.1.0)
Requirement already satisfied: holoviews>=1.11.0 in
/usr/local/lib/python3.10/dist-packages (from hvplot) (1.17.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from hvplot) (2.0.3)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.10/dist-
packages (from hvplot) (1.25.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from hvplot) (24.0)
Requirement already satisfied: panel>=0.11.0 in /usr/local/lib/python3.10/dist-
packages (from hvplot) (1.3.8)
Requirement already satisfied: param<3.0,>=1.12.0 in
/usr/local/lib/python3.10/dist-packages (from hvplot) (2.1.0)
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.10/dist-
packages (from bokeh>=1.0.0->hvplot) (3.1.3)
Requirement already satisfied: contourpy>=1 in /usr/local/lib/python3.10/dist-
packages (from bokeh>=1.0.0->hvplot) (1.2.1)
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.10/dist-
packages (from bokeh>=1.0.0->hvplot) (9.4.0)
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.10/dist-
packages (from bokeh>=1.0.0->hvplot) (6.0.1)
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.10/dist-
packages (from bokeh>=1.0.0->hvplot) (6.3.3)
Requirement already satisfied: xyzservices>=2021.09.1 in
/usr/local/lib/python3.10/dist-packages (from bokeh>=1.0.0->hvplot) (2024.4.0)
Requirement already satisfied: pyviz-comms>=0.7.4 in
/usr/local/lib/python3.10/dist-packages (from holoviews>=1.11.0->hvplot) (3.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas->hvplot) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->hvplot) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->hvplot) (2024.1)
```

```
Requirement already satisfied: markdown in /usr/local/lib/python3.10/dist-
packages (from panel>=0.11.0->hvplot) (3.6)
Requirement already satisfied: markdown-it-py in /usr/local/lib/python3.10/dist-
packages (from panel>=0.11.0->hvplot) (3.0.0)
Requirement already satisfied: linkify-it-py in /usr/local/lib/python3.10/dist-
packages (from panel>=0.11.0->hvplot) (2.0.3)
Requirement already satisfied: mdit-py-plugins in
/usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (0.4.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from panel>=0.11.0->hvplot) (2.31.0)
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.10/dist-
packages (from panel>=0.11.0->hvplot) (4.66.2)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages
(from panel>=0.11.0->hvplot) (6.1.0)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.10/dist-packages (from panel>=0.11.0->hvplot) (4.11.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from Jinja2>=2.9->bokeh>=1.0.0->hvplot)
(2.1.5)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas->hvplot) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-
packages (from bleach->panel>=0.11.0->hvplot) (0.5.1)
Requirement already satisfied: uc-micro-py in /usr/local/lib/python3.10/dist-
packages (from linkify-it-py->panel>=0.11.0->hvplot) (1.0.3)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-
packages (from markdown-it-py->panel>=0.11.0->hvplot) (0.1.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->panel>=0.11.0->hvplot) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot)
(2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->panel>=0.11.0->hvplot)
(2024.2.2)
Installing collected packages: hvplot
Successfully installed hvplot-0.9.2
```

[7]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import hvplot.pandas
from sklearn.model_selection import train_test_split
```

```
from sklearn import metrics
from sklearn.linear_model import LinearRegression
%matplotlib inline
```

[8]:
```
df = pd.concat([X, y], axis = 1)
df
```

[8]:

	price	highway-mpg	city-mpg	peak-rpm	horsepower	compression-ratio	\
0	13495.0	27	21	5000.0	111.0	9.0	
1	16500.0	27	21	5000.0	111.0	9.0	
2	16500.0	26	19	5000.0	154.0	9.0	
3	13950.0	30	24	5500.0	102.0	10.0	
4	17450.0	22	18	5500.0	115.0	8.0	
..	…	…	…	…	…	…	
200	16845.0	28	23	5400.0	114.0	9.5	
201	19045.0	25	19	5300.0	160.0	8.7	
202	21485.0	23	18	5500.0	134.0	8.8	
203	22470.0	27	26	4800.0	106.0	23.0	
204	22625.0	25	19	5400.0	114.0	9.5	

	stroke	bore	fuel-system	engine-size	…	wheel-base	engine-location	\
0	2.68	3.47	mpfi	130	…	88.6	front	
1	2.68	3.47	mpfi	130	…	88.6	front	
2	3.47	2.68	mpfi	152	…	94.5	front	
3	3.40	3.19	mpfi	109	…	99.8	front	
4	3.40	3.19	mpfi	136	…	99.4	front	
..	…	…	…	…	…	…	…	
200	3.15	3.78	mpfi	141	…	109.1	front	
201	3.15	3.78	mpfi	141	…	109.1	front	
202	2.87	3.58	mpfi	173	…	109.1	front	
203	3.40	3.01	idi	145	…	109.1	front	
204	3.15	3.78	mpfi	141	…	109.1	front	

	drive-wheels	body-style	num-of-doors	aspiration	fuel-type	\
0	rwd	convertible	2.0	std	gas	
1	rwd	convertible	2.0	std	gas	
2	rwd	hatchback	2.0	std	gas	
3	fwd	sedan	4.0	std	gas	
4	4wd	sedan	4.0	std	gas	
..	…	…	…	…	…	
200	rwd	sedan	4.0	std	gas	
201	rwd	sedan	4.0	turbo	gas	
202	rwd	sedan	4.0	std	gas	
203	rwd	sedan	4.0	turbo	diesel	
204	rwd	sedan	4.0	turbo	gas	

	make	normalized-losses	symboling

```
0    alfa-romero                NaN          3
1    alfa-romero                NaN          3
2    alfa-romero                NaN          1
3           audi              164.0          2
4           audi              164.0          2
..           …                 …       …
200       volvo                95.0         -1
201       volvo                95.0         -1
202       volvo                95.0         -1
203       volvo                95.0         -1
204       volvo                95.0         -1

[205 rows x 26 columns]
```

2 Cleaning

[9]:
```
missing_values = df.isnull().sum()
print(missing_values)
```

```
price                4
highway-mpg          0
city-mpg             0
peak-rpm             2
horsepower           2
compression-ratio    0
stroke               4
bore                 4
fuel-system          0
engine-size          0
num-of-cylinders     0
engine-type          0
curb-weight          0
height               0
width                0
length               0
wheel-base           0
engine-location      0
drive-wheels         0
body-style           0
num-of-doors         2
aspiration           0
fuel-type            0
make                 0
normalized-losses   41
symboling            0
dtype: int64
```

[14]:
```
null_columns = []
for x in df.columns:
    if df[x].isnull().any():
        null_columns.append(x)
```

[15]:
```
null_columns
```

[15]:
```
['price',
 'peak-rpm',
 'horsepower',
 'stroke',
 'bore',
 'num-of-doors',
 'normalized-losses']
```

[19]:
```
for x in null_columns:
  df[x] = df[x].fillna(df[x].mean())
```

[20]:
```
df
```

[20]:

	price	highway-mpg	city-mpg	peak-rpm	horsepower	compression-ratio	\
0	13495.0	27	21	5000.0	111.0	9.0	
1	16500.0	27	21	5000.0	111.0	9.0	
2	16500.0	26	19	5000.0	154.0	9.0	
3	13950.0	30	24	5500.0	102.0	10.0	
4	17450.0	22	18	5500.0	115.0	8.0	
..	…	…	…	…	…	…	
200	16845.0	28	23	5400.0	114.0	9.5	
201	19045.0	25	19	5300.0	160.0	8.7	
202	21485.0	23	18	5500.0	134.0	8.8	
203	22470.0	27	26	4800.0	106.0	23.0	
204	22625.0	25	19	5400.0	114.0	9.5	

	stroke	bore	fuel-system	engine-size	…	wheel-base	engine-location	\
0	2.68	3.47	mpfi	130	…	88.6	front	
1	2.68	3.47	mpfi	130	…	88.6	front	
2	3.47	2.68	mpfi	152	…	94.5	front	
3	3.40	3.19	mpfi	109	…	99.8	front	
4	3.40	3.19	mpfi	136	…	99.4	front	
..	…	…	…	…	…	…	…	
200	3.15	3.78	mpfi	141	…	109.1	front	
201	3.15	3.78	mpfi	141	…	109.1	front	
202	2.87	3.58	mpfi	173	…	109.1	front	
203	3.40	3.01	idi	145	…	109.1	front	
204	3.15	3.78	mpfi	141	…	109.1	front	

	drive-wheels	body-style	num-of-doors	aspiration	fuel-type	\

```
0             rwd  convertible          2.0         std        gas
1             rwd  convertible          2.0         std        gas
2             rwd    hatchback          2.0         std        gas
3             fwd        sedan          4.0         std        gas
4             4wd        sedan          4.0         std        gas
..            …          …            …         …         …
200           rwd        sedan          4.0         std        gas
201           rwd        sedan          4.0       turbo        gas
202           rwd        sedan          4.0         std        gas
203           rwd        sedan          4.0       turbo     diesel
204           rwd        sedan          4.0       turbo        gas

            make normalized-losses symboling
0    alfa-romero             122.0         3
1    alfa-romero             122.0         3
2    alfa-romero             122.0         1
3           audi             164.0         2
4           audi             164.0         2
..           …                …       …
200        volvo              95.0        -1
201        volvo              95.0        -1
202        volvo              95.0        -1
203        volvo              95.0        -1
204        volvo              95.0        -1

[205 rows x 26 columns]
```

[22]:
```
df.isnull().sum()
```

[22]:
```
price               0
highway-mpg         0
city-mpg            0
peak-rpm            0
horsepower          0
compression-ratio   0
stroke              0
bore                0
fuel-system         0
engine-size         0
num-of-cylinders    0
engine-type         0
curb-weight         0
height              0
width               0
length              0
wheel-base          0
engine-location     0
```

```
drive-wheels         0
body-style           0
num-of-doors         0
aspiration           0
fuel-type            0
make                 0
normalized-losses    0
symboling            0
dtype: int64
```

#Linear Regression annalyis

[23]:
```
df.columns
```

[23]:
```
Index(['price', 'highway-mpg', 'city-mpg', 'peak-rpm', 'horsepower',
       'compression-ratio', 'stroke', 'bore', 'fuel-system', 'engine-size',
       'num-of-cylinders', 'engine-type', 'curb-weight', 'height', 'width',
       'length', 'wheel-base', 'engine-location', 'drive-wheels', 'body-style',
       'num-of-doors', 'aspiration', 'fuel-type', 'make', 'normalized-losses',
       'symboling'],
      dtype='object')
```

[24]:
```
df.head(20)
```

[24]:

	price	highway-mpg	city-mpg	peak-rpm	horsepower	\
0	13495.000000	27	21	5000.0	111.0	
1	16500.000000	27	21	5000.0	111.0	
2	16500.000000	26	19	5000.0	154.0	
3	13950.000000	30	24	5500.0	102.0	
4	17450.000000	22	18	5500.0	115.0	
5	15250.000000	25	19	5500.0	110.0	
6	17710.000000	25	19	5500.0	110.0	
7	18920.000000	25	19	5500.0	110.0	
8	23875.000000	20	17	5500.0	140.0	
9	13207.129353	22	16	5500.0	160.0	
10	16430.000000	29	23	5800.0	101.0	
11	16925.000000	29	23	5800.0	101.0	
12	20970.000000	28	21	4250.0	121.0	
13	21105.000000	28	21	4250.0	121.0	
14	24565.000000	25	20	4250.0	121.0	
15	30760.000000	22	16	5400.0	182.0	
16	41315.000000	22	16	5400.0	182.0	
17	36880.000000	20	15	5400.0	182.0	
18	5151.000000	53	47	5100.0	48.0	
19	6295.000000	43	38	5400.0	70.0	

	compression-ratio	stroke	bore	fuel-system	engine-size	…	wheel-base	\

0	9.0	2.68	3.47	mpfi	130	…	88.6
1	9.0	2.68	3.47	mpfi	130	…	88.6
2	9.0	3.47	2.68	mpfi	152	…	94.5
3	10.0	3.40	3.19	mpfi	109	…	99.8
4	8.0	3.40	3.19	mpfi	136	…	99.4
5	8.5	3.40	3.19	mpfi	136	…	99.8
6	8.5	3.40	3.19	mpfi	136	…	105.8
7	8.5	3.40	3.19	mpfi	136	…	105.8
8	8.3	3.40	3.13	mpfi	131	…	105.8
9	7.0	3.40	3.13	mpfi	131	…	99.5
10	8.8	2.80	3.50	mpfi	108	…	101.2
11	8.8	2.80	3.50	mpfi	108	…	101.2
12	9.0	3.19	3.31	mpfi	164	…	101.2
13	9.0	3.19	3.31	mpfi	164	…	101.2
14	9.0	3.19	3.31	mpfi	164	…	103.5
15	8.0	3.39	3.62	mpfi	209	…	103.5
16	8.0	3.39	3.62	mpfi	209	…	103.5
17	8.0	3.39	3.62	mpfi	209	…	110.0
18	9.5	3.03	2.91	2bbl	61	…	88.4
19	9.6	3.11	3.03	2bbl	90	…	94.5

	engine-location	drive-wheels	body-style	num-of-doors	aspiration	\
0	front	rwd	convertible	2.0	std	
1	front	rwd	convertible	2.0	std	
2	front	rwd	hatchback	2.0	std	
3	front	fwd	sedan	4.0	std	
4	front	4wd	sedan	4.0	std	
5	front	fwd	sedan	2.0	std	
6	front	fwd	sedan	4.0	std	
7	front	fwd	wagon	4.0	std	
8	front	fwd	sedan	4.0	turbo	
9	front	4wd	hatchback	2.0	turbo	
10	front	rwd	sedan	2.0	std	
11	front	rwd	sedan	4.0	std	
12	front	rwd	sedan	2.0	std	
13	front	rwd	sedan	4.0	std	
14	front	rwd	sedan	4.0	std	
15	front	rwd	sedan	4.0	std	
16	front	rwd	sedan	2.0	std	
17	front	rwd	sedan	4.0	std	
18	front	fwd	hatchback	2.0	std	
19	front	fwd	hatchback	2.0	std	

	fuel-type	make	normalized-losses	symboling
0	gas	alfa-romero	122.0	3
1	gas	alfa-romero	122.0	3
2	gas	alfa-romero	122.0	1

```
3         gas         audi             164.0         2
4         gas         audi             164.0         2
5         gas         audi             122.0         2
6         gas         audi             158.0         1
7         gas         audi             122.0         1
8         gas         audi             158.0         1
9         gas         audi             122.0         0
10        gas          bmw             192.0         2
11        gas          bmw             192.0         0
12        gas          bmw             188.0         0
13        gas          bmw             188.0         0
14        gas          bmw             122.0         1
15        gas          bmw             122.0         0
16        gas          bmw             122.0         0
17        gas          bmw             122.0         0
18        gas    chevrolet             121.0         2
19        gas    chevrolet              98.0         1

[20 rows x 26 columns]
```

[25]:
```
df.describe()
```

[25]:
```
              price  highway-mpg    city-mpg     peak-rpm  horsepower  \
count    205.000000   205.000000  205.000000   205.000000  205.000000
mean   13207.129353    30.751220   25.219512  5125.369458  104.256158
std     7868.768212     6.886443    6.542142   476.979093   39.519211
min     5118.000000    16.000000   13.000000  4150.000000   48.000000
25%     7788.000000    25.000000   19.000000  4800.000000   70.000000
50%    10595.000000    30.000000   24.000000  5200.000000   95.000000
75%    16500.000000    34.000000   30.000000  5500.000000  116.000000
max    45400.000000    54.000000   49.000000  6600.000000  288.000000

       compression-ratio      stroke        bore  engine-size  \
count         205.000000  205.000000  205.000000   205.000000
mean           10.142537    3.255423    3.329751   126.907317
std             3.972040    0.313597    0.270844    41.642693
min             7.000000    2.070000    2.540000    61.000000
25%             8.600000    3.110000    3.150000    97.000000
50%             9.000000    3.290000    3.310000   120.000000
75%             9.400000    3.410000    3.580000   141.000000
max            23.000000    4.170000    3.940000   326.000000

       num-of-cylinders  curb-weight      height       width      length  \
count        205.000000   205.000000  205.000000  205.000000  205.000000
mean           4.380488  2555.565854   53.724878   65.907805  174.049268
std            1.080854   520.680204    2.443522    2.145204   12.337289
min            2.000000  1488.000000   47.800000   60.300000  141.100000
```

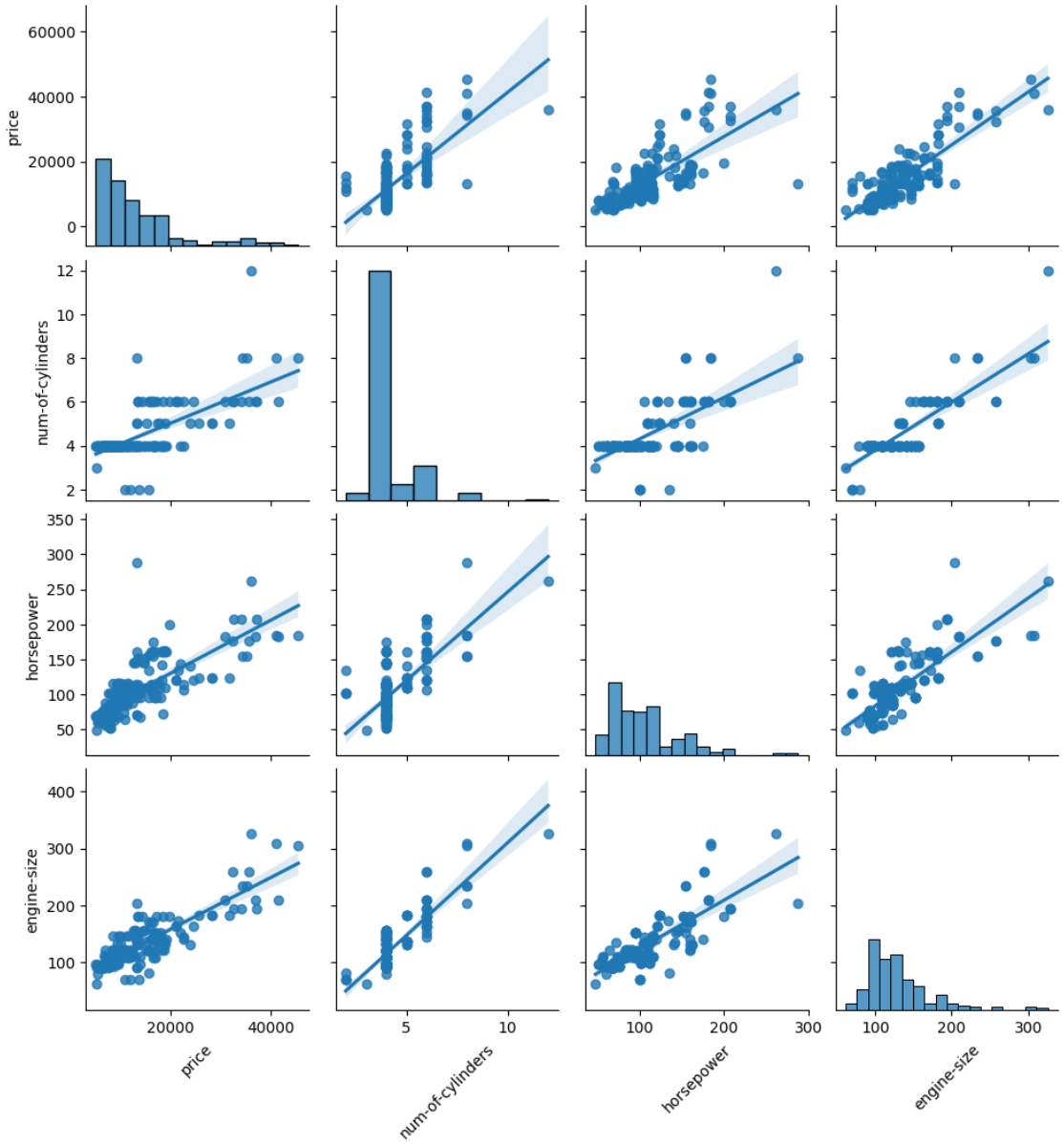
```
25%            4.000000  2145.000000   52.000000   64.100000  166.300000
50%            4.000000  2414.000000   54.100000   65.500000  173.200000
75%            4.000000  2935.000000   55.500000   66.900000  183.100000
max           12.000000  4066.000000   59.800000   72.300000  208.100000

       wheel-base  num-of-doors  normalized-losses   symboling
count  205.000000    205.000000         205.000000  205.000000
mean    98.756585      3.123153         122.000000    0.834146
std      6.021776      0.989952          31.681008    1.245307
min     86.600000      2.000000          65.000000   -2.000000
25%     94.500000      2.000000         101.000000    0.000000
50%     97.000000      4.000000         122.000000    1.000000
75%    102.400000      4.000000         137.000000    2.000000
max    120.900000      4.000000         256.000000    3.000000
```

[26]:
```
df.info()
```

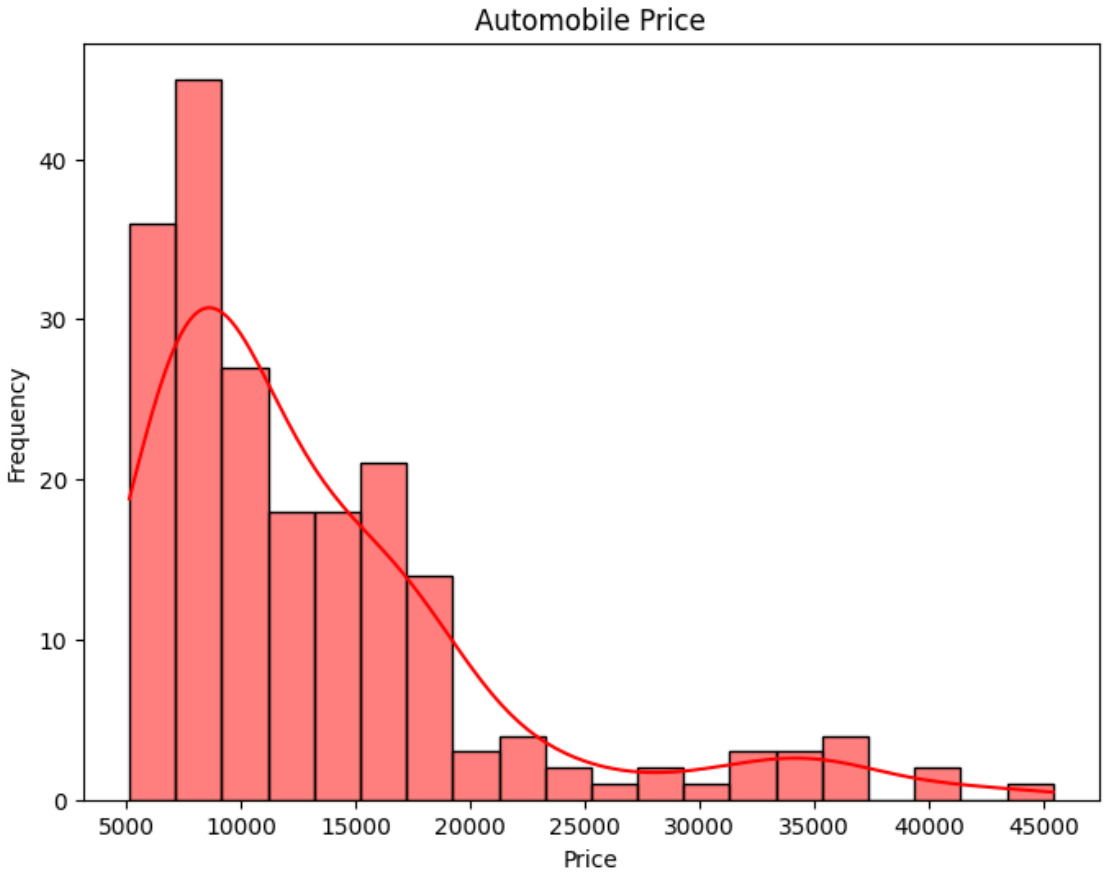
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   price              205 non-null    float64
 1   highway-mpg        205 non-null    int64
 2   city-mpg           205 non-null    int64
 3   peak-rpm           205 non-null    float64
 4   horsepower         205 non-null    float64
 5   compression-ratio  205 non-null    float64
 6   stroke             205 non-null    float64
 7   bore               205 non-null    float64
 8   fuel-system        205 non-null    object
 9   engine-size        205 non-null    int64
 10  num-of-cylinders   205 non-null    int64
 11  engine-type        205 non-null    object
 12  curb-weight        205 non-null    int64
 13  height             205 non-null    float64
 14  width              205 non-null    float64
 15  length             205 non-null    float64
 16  wheel-base         205 non-null    float64
 17  engine-location    205 non-null    object
 18  drive-wheels       205 non-null    object
 19  body-style         205 non-null    object
 20  num-of-doors       205 non-null    float64
 21  aspiration         205 non-null    object
 22  fuel-type          205 non-null    object
 23  make               205 non-null    object
 24  normalized-losses  205 non-null    float64
```

```
 25  symboling          205 non-null    int64
dtypes: float64(12), int64(6), object(8)
memory usage: 41.8+ KB
```

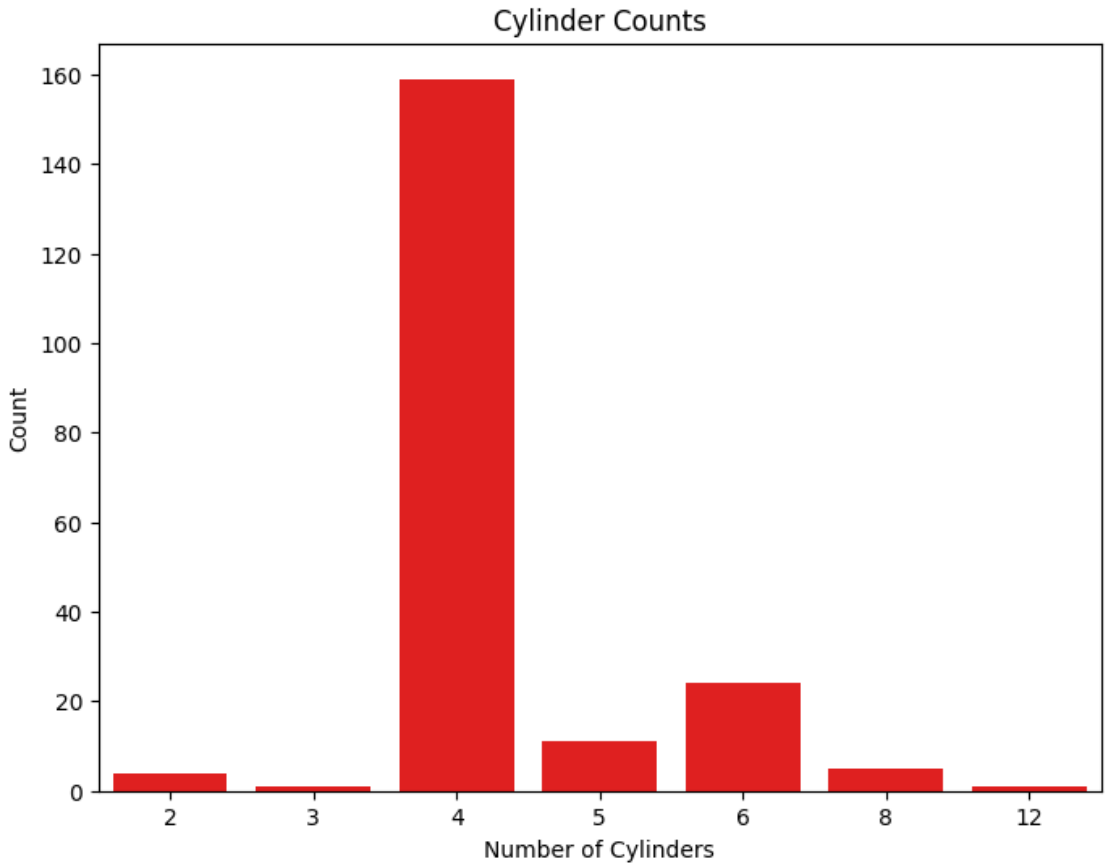
[29]:
```
pairplot = sns.pairplot(df[['price', 'num-of-cylinders', 'horsepower',␣
↪'engine-size']], kind='reg')

for ax in pairplot.axes.flatten():
    ax.set_xlabel(ax.get_xlabel(), rotation = 45)

plt.show()
```



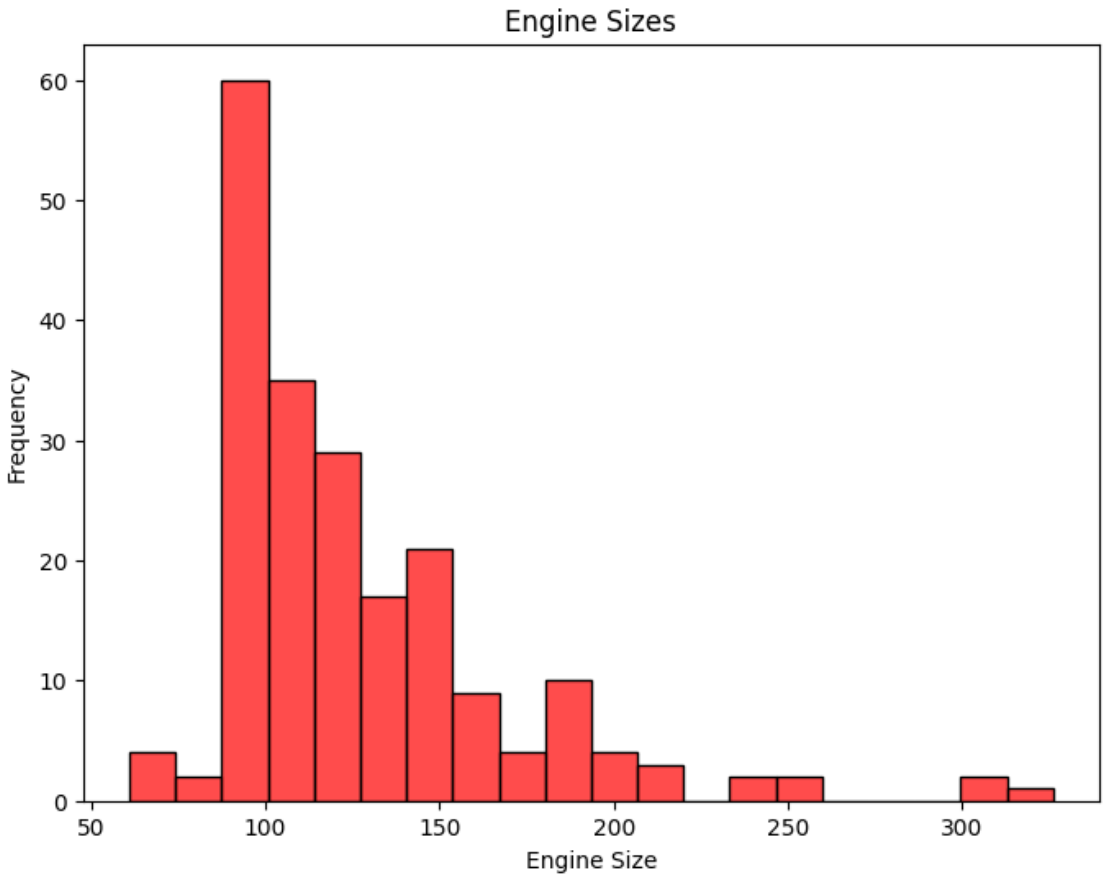
[30]:
```
import matplotlib.pyplot as plt

# Plotting the histogram for automobile prices
plt.figure(figsize=(8, 6))
sns.histplot(df['price'], bins=20, kde=True, color='red')
plt.title('Automobile Price')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```



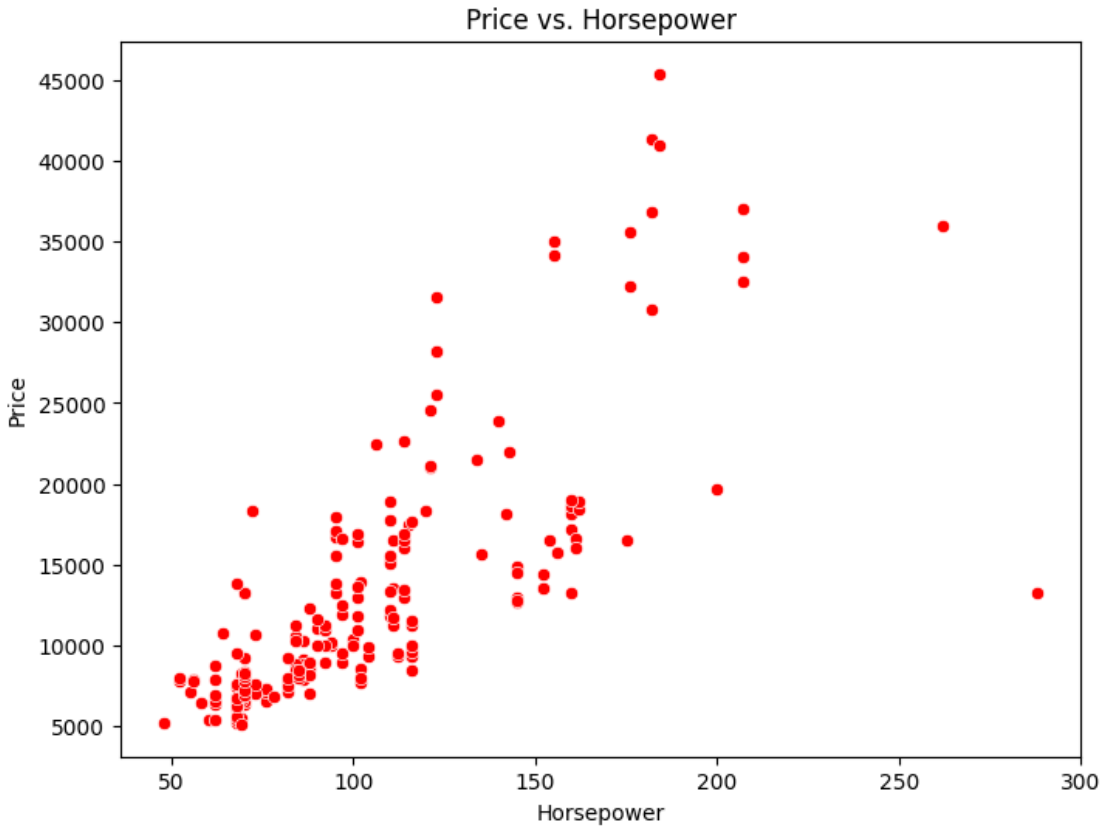
[34]:
```
plt.figure(figsize=(8, 6))
sns.countplot(x='num-of-cylinders', color='red',data=df)
plt.title('Cylinder Counts')
plt.xlabel('Number of Cylinders')
plt.ylabel('Count')
plt.show()
```



[35]:
```
plt.figure(figsize=(8, 6))
sns.histplot(df['engine-size'], bins=20, color='red', alpha=0.7)

plt.title('Engine Sizes')
plt.xlabel('Engine Size')
plt.ylabel('Frequency')

plt.show()
```

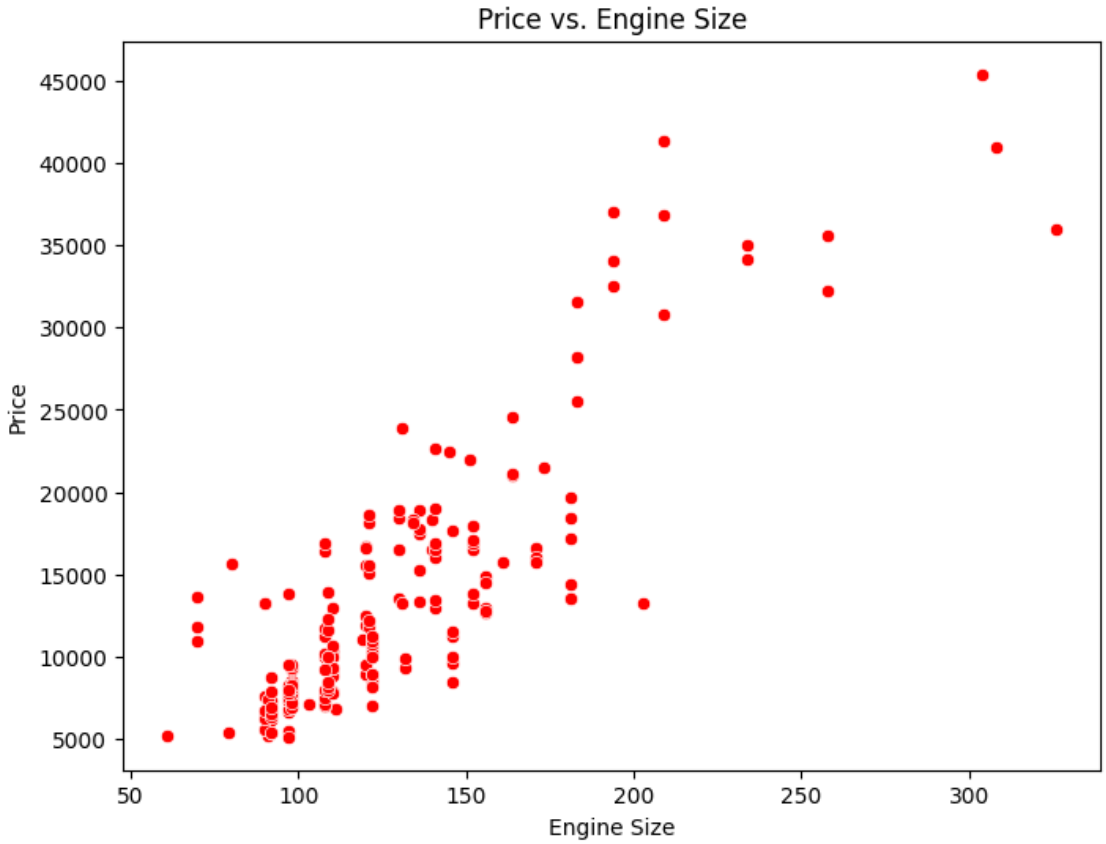


[36]:
```
# Create scatterplot for Price vs. Horsepower
plt.figure(figsize=(8, 6))
sns.scatterplot(x='horsepower', y='price', data=df, color='red')
plt.title('Price vs. Horsepower')
plt.xlabel('Horsepower')
plt.ylabel('Price')
plt.show()
```



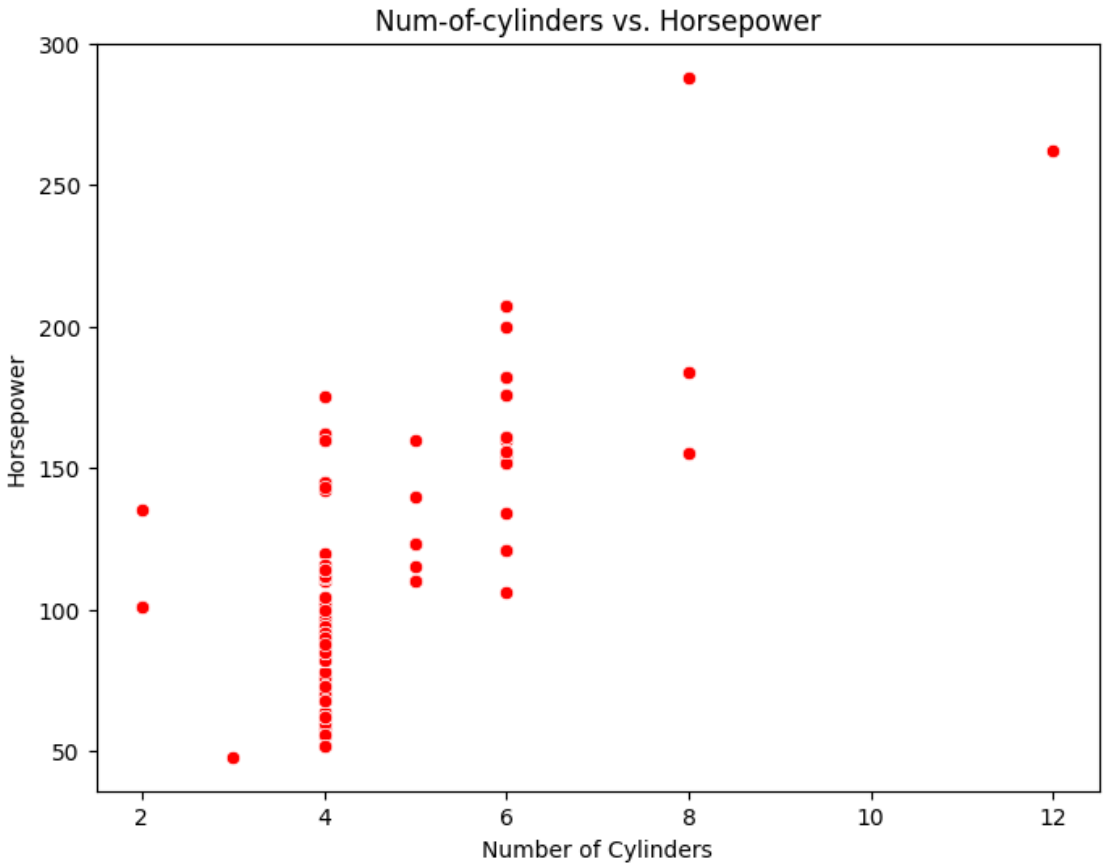
indicate that cars with higher horsepower tend to be more expensive

[37]:
```
# Create scatterplot for Price vs. Engine Size
plt.figure(figsize=(8, 6))
sns.scatterplot(x='engine-size', y='price', data=df, color='red')
plt.title('Price vs. Engine Size')
plt.xlabel('Engine Size')
plt.ylabel('Price')
plt.show()
```



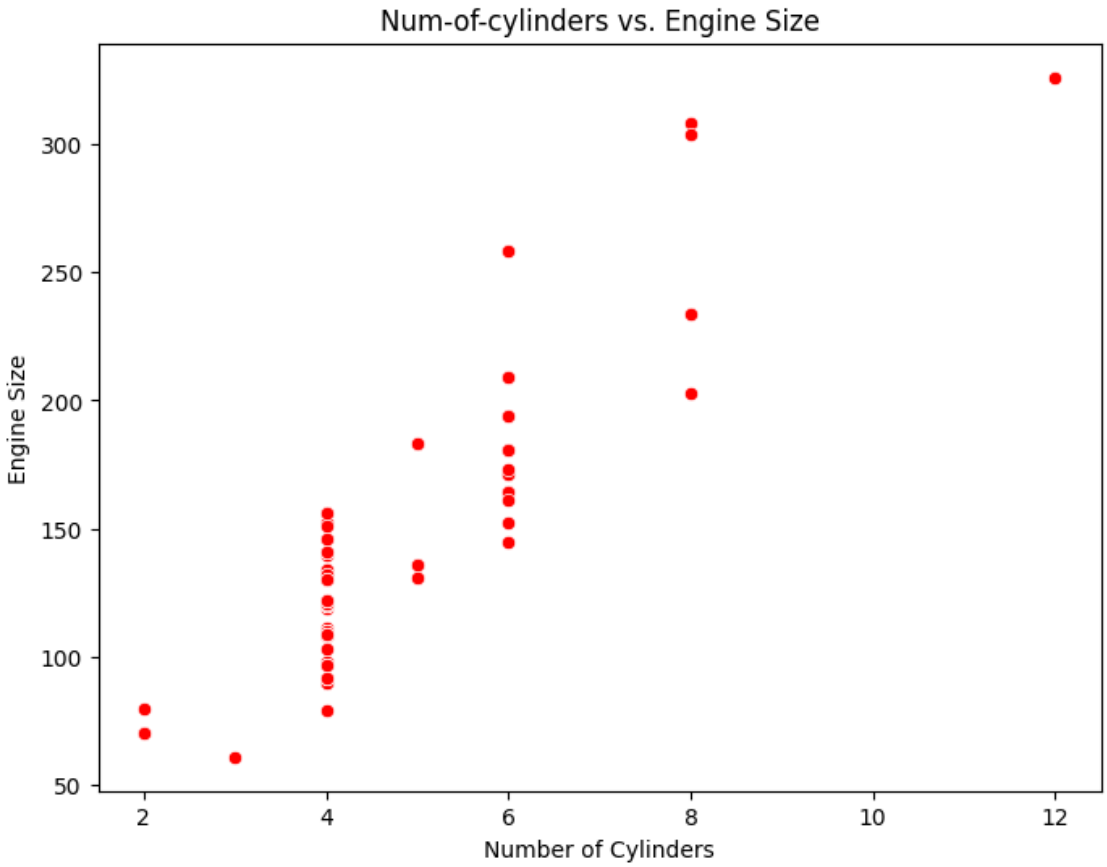
larger engines can be more expensive.

[38]:
```
# Create scatterplot for Num-of-cylinders vs. Horsepower
plt.figure(figsize=(8, 6))
sns.scatterplot(x='num-of-cylinders', y='horsepower', data=df, color='red')
plt.title('Num-of-cylinders vs. Horsepower')
plt.xlabel('Number of Cylinders')
plt.ylabel('Horsepower')
plt.show()
```



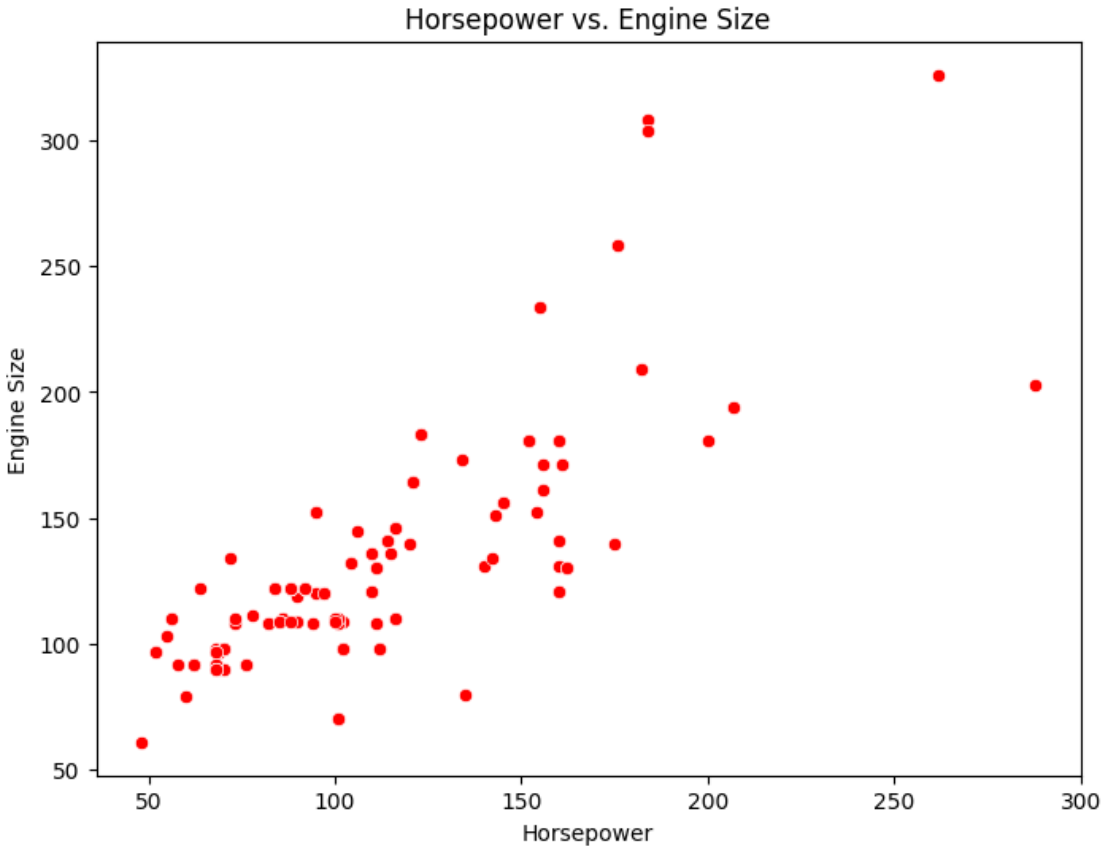
automobile with more cylinders have higher horsepower.

[39]:
```
# Create scatterplot for Num-of-cylinders vs. Engine Size
plt.figure(figsize=(8, 6))
sns.scatterplot(x='num-of-cylinders', y='engine-size', data=df, color='red')
plt.title('Num-of-cylinders vs. Engine Size')
plt.xlabel('Number of Cylinders')
plt.ylabel('Engine Size')
plt.show()
```



manufacturers tend to increase engine size by adding more cylinders

[40]:
```
# Create scatterplot for Horsepower vs. Engine Size
plt.figure(figsize=(8, 6))
sns.scatterplot(x='horsepower', y='engine-size', data=df, color='red')
plt.title('Horsepower vs. Engine Size')
plt.xlabel('Horsepower')
plt.ylabel('Engine Size')
plt.show()
```



helps determine that larger engine sizes tend to produce higher horsepower outputs.

[44]:
```
print("X =", X.shape,"\nY =", y.shape)
```

```
X = (205, 25)
Y = (205, 1)
```

[76]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[77]:
```
X_train.shape
```

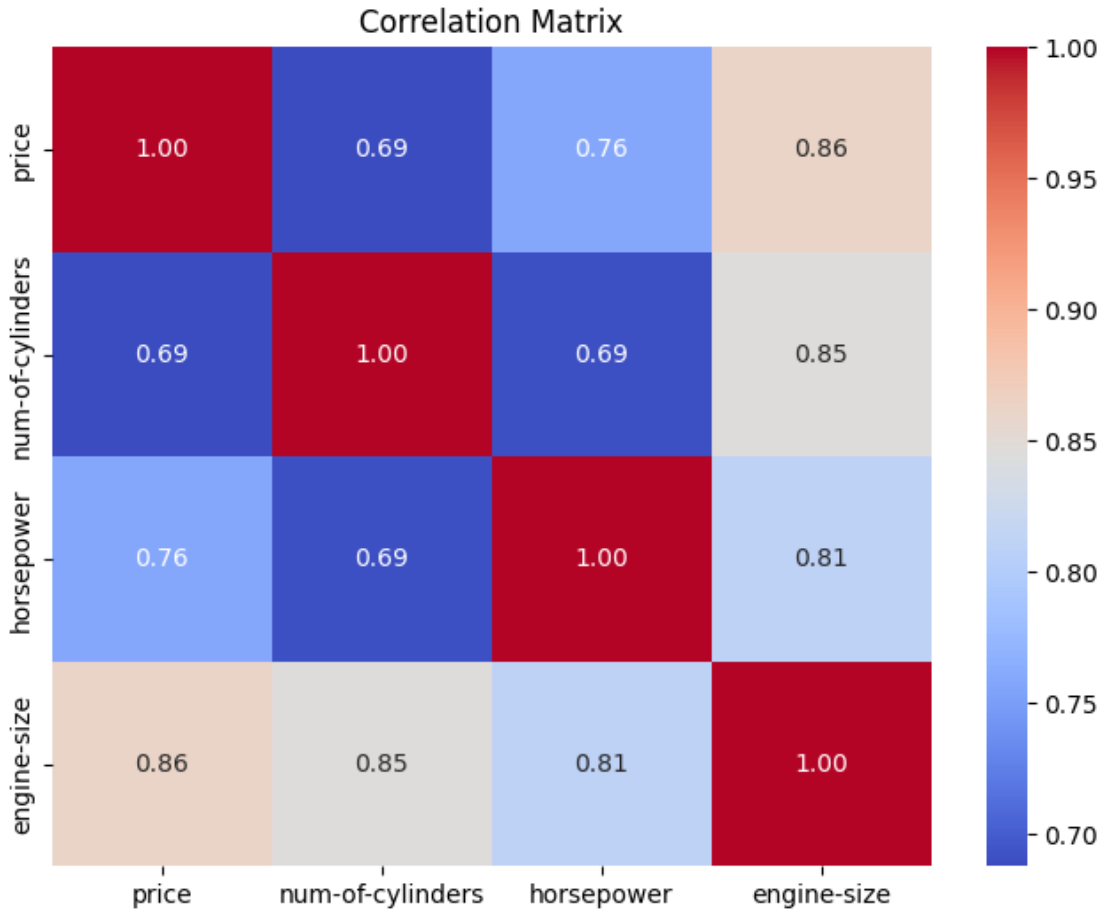
[77]: (164, 3)

[78]:
```
X_test.shape
```

[78]: (41, 3)

[80]:
```
model = LinearRegression()
```

[67]:
```
correlation_matrix = df[['price', 'num-of-cylinders', 'horsepower',␣
 ↪'engine-size']].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```



##Price vs. num-of-cylinders (0.687770): There is a moderately positive correlation between the price and the number of cylinders in the engine.

More cylinders often mean larger and more powerful engine, which can result in a higher price for the vehicle.

##Price vs. horsepower (0.757917): There is a strong positive correlation between the price and the horsepower of the car.

Higher horsepower is often will be better performance, which can justify a higher price tag for the car.

##Price vs. engine-size (0.861752): There is a very strong positive correlation between

the price and the engine size.

Larger engines typically have more power and can offer better performance, leading to higher prices.

##Num-of-cylinders vs. horsepower (0.691208): There is a moderately positive correlation between the number of cylinders and the horsepower of the car.

More cylinders generally mean more power generated by the engine, resulting in higher horsepower.

##Num-of-cylinders vs. engine-size (0.846031): There is a strong positive correlation between the number of cylinders and the engine size.

More cylinders mean a larger total displacement of the engine, which leads to a larger engine size.

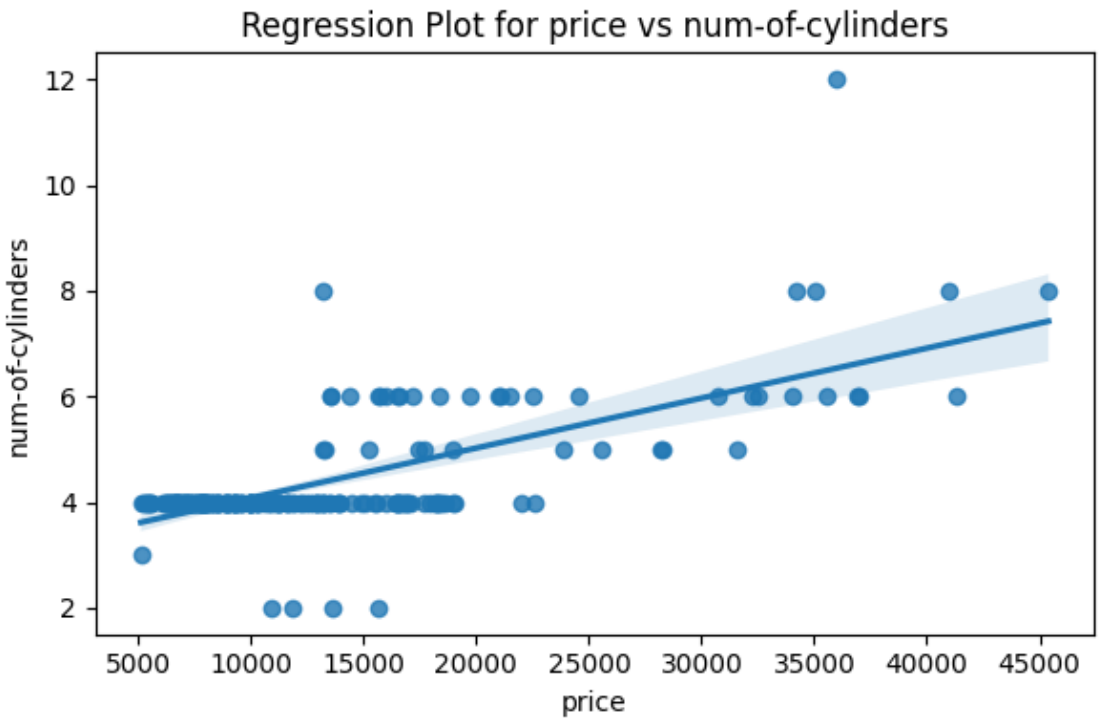
##Horsepower vs. engine-size (0.810713): There is a strong positive correlation between the horsepower and the engine size.

A larger engine size allows for more fuel, resulting in higher power output.

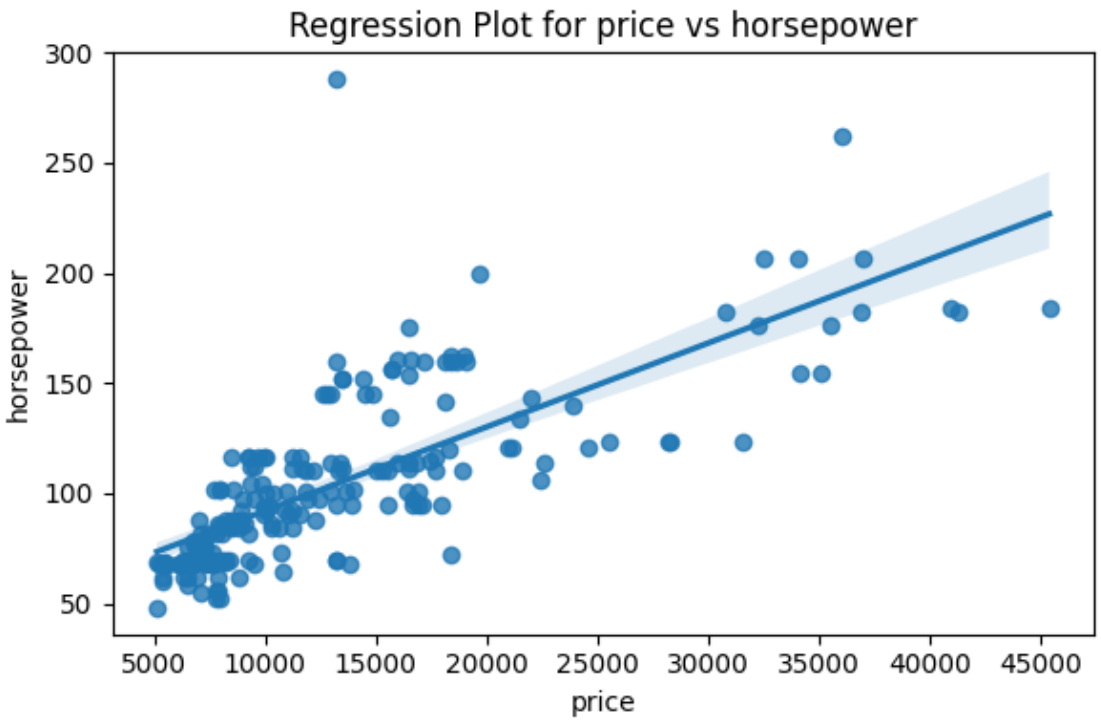
[84]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

X = df[['num-of-cylinders', 'horsepower', 'engine-size']]
y = df['price']

y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

```
Mean Squared Error: 16171180.212878957
Coefficients: [-1201.27854017    37.88655387   157.55524143]
Intercept: -5478.095778018269
```

[92]:
```
# Scatter plot with regression line for price vs num-of-cylinders
plt.figure(figsize=(6, 4))
sns.regplot(x='price', y='num-of-cylinders', data=df)
plt.title('Regression Plot for price vs num-of-cylinders')
plt.xlabel('price')
plt.ylabel('num-of-cylinders')
plt.tight_layout()
plt.show()
```



[93]:
```
# Scatter plot with regression line for price vs horsepower
plt.figure(figsize=(6, 4))
sns.regplot(x='price', y='horsepower', data=df)
plt.title('Regression Plot for price vs horsepower')
plt.xlabel('price')
plt.ylabel('horsepower')
plt.tight_layout()
plt.show()
```



[94]:
```
# Scatter plot with regression line for price vs engine-size
plt.figure(figsize=(6, 4))
sns.regplot(x='price', y='engine-size', data=df)
plt.title('Regression Plot for price vs engine-size')
plt.xlabel('price')
plt.ylabel('engine-size')
plt.tight_layout()
plt.show()
```

According to the graphs that the larger and more powerful the automobile tend to be have higher prices.