Open in Colab

In [69]:
```
import pandas as pd
import numpy as np

path = "https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data"
dataf = pd.read_csv(path) #na_values are used to use NaN inplace of ?? in dataset. Nan is standard name for missing
print("The first 5 rows of the dataframe")
headers = ["symboling","normalized-losses","make","fuel-type","aspiration", "num-of-doors","body-style",
         "drive-wheels","engine-location","wheel-base", "length","width","height","curb-weight","engine-type",
         "num-of-cylinders", "engine-size","fuel-system","bore","stroke","compression-ratio","horsepower",
         "peak-rpm","city-mpg","highway-mpg","price"]
#dataf.columns shows labels of columns in dataframe
dataf.columns = headers
dataf.head(10)
```

```
The first 5 rows of the dataframe
```

Out[69]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | ? | alfa-romero | gas | std | two | convertible | rwd | front | 88.6 | ... | 130 | mpfi | 3.47 | 2.68 | |
| **1** | 1 | ? | alfa-romero | gas | std | two | hatchback | rwd | front | 94.5 | ... | 152 | mpfi | 2.68 | 3.47 | |
| **2** | 2 | 164 | audi | gas | std | four | sedan | fwd | front | 99.8 | ... | 109 | mpfi | 3.19 | 3.40 | |
| **3** | 2 | 164 | audi | gas | std | four | sedan | 4wd | front | 99.4 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **4** | 2 | ? | audi | gas | std | two | sedan | fwd | front | 99.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **5** | 1 | 158 | audi | gas | std | four | sedan | fwd | front | 105.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **6** | 1 | ? | audi | gas | std | four | wagon | fwd | front | 105.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **7** | 1 | 158 | audi | gas | turbo | four | sedan | fwd | front | 105.8 | ... | 131 | mpfi | 3.13 | 3.40 | |
| **8** | 0 | ? | audi | gas | turbo | two | hatchback | 4wd | front | 99.5 | ... | 131 | mpfi | 3.13 | 3.40 | |
| **9** | 2 | 192 | bmw | gas | std | two | sedan | rwd | front | 101.2 | ... | 108 | mpfi | 3.50 | 2.80 | |

10 rows × 26 columns

In [70]:
```
dataf = dataf.replace('?',np.nan)
dataf
```

Out[70]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | NaN | alfa-romero | gas | std | two | convertible | rwd | front | 88.6 | ... | 130 | mpfi | 3.47 | 2.68 |
| **1** | 1 | NaN | alfa-romero | gas | std | two | hatchback | rwd | front | 94.5 | ... | 152 | mpfi | 2.68 | 3.47 |
| **2** | 2 | 164 | audi | gas | std | four | sedan | fwd | front | 99.8 | ... | 109 | mpfi | 3.19 | 3.40 |
| **3** | 2 | 164 | audi | gas | std | four | sedan | 4wd | front | 99.4 | ... | 136 | mpfi | 3.19 | 3.40 |
| **4** | 2 | NaN | audi | gas | std | two | sedan | fwd | front | 99.8 | ... | 136 | mpfi | 3.19 | 3.40 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **199** | -1 | 95 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **200** | -1 | 95 | volvo | gas | turbo | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **201** | -1 | 95 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 173 | mpfi | 3.58 | 2.87 |
| **202** | -1 | 95 | volvo | diesel | turbo | four | sedan | rwd | front | 109.1 | ... | 145 | idi | 3.01 | 3.40 |
| **203** | -1 | 95 | volvo | gas | turbo | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |

204 rows × 26 columns

In [71]:
```
dataf.isna().sum()
```

Out[71]:
```
symboling            0
normalized-losses    40
make                 0
fuel-type            0
aspiration           0
num-of-doors         2
```

```
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
bore                 4
stroke               4
compression-ratio    0
horsepower           2
peak-rpm             2
city-mpg             0
highway-mpg          0
price                4
```

In [72]:

```
dataf['normalized-losses'] =  dataf['normalized-losses'].astype('float')
mean = dataf['normalized-losses'].mean()
dataf['normalized-losses'].replace(np.nan,mean,inplace=True)
dataf
```

Out[72]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | 122.0 | alfa-romero | gas | std | two | convertible | rwd | front | 88.6 | ... | 130 | mpfi | 3.47 | 2.68 |
| **1** | 1 | 122.0 | alfa-romero | gas | std | two | hatchback | rwd | front | 94.5 | ... | 152 | mpfi | 2.68 | 3.47 |
| **2** | 2 | 164.0 | audi | gas | std | four | sedan | fwd | front | 99.8 | ... | 109 | mpfi | 3.19 | 3.40 |
| **3** | 2 | 164.0 | audi | gas | std | four | sedan | 4wd | front | 99.4 | ... | 136 | mpfi | 3.19 | 3.40 |
| **4** | 2 | 122.0 | audi | gas | std | two | sedan | fwd | front | 99.8 | ... | 136 | mpfi | 3.19 | 3.40 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **199** | -1 | 95.0 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **200** | -1 | 95.0 | volvo | gas | turbo | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **201** | -1 | 95.0 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 173 | mpfi | 3.58 | 2.87 |
| **202** | -1 | 95.0 | volvo | diesel | turbo | four | sedan | rwd | front | 109.1 | ... | 145 | idi | 3.01 | 3.40 |

In [73]:
```
miss2_data = dataf.isnull()
print(miss2_data["normalized-losses"].value_counts())
```

```
False    204
Name: normalized-losses, dtype: int64
```

In [74]:
```
dataf['normalized-losses'].isna().sum
```

Out[74]:
```
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
199    False
200    False
201    False
202    False
203    False
Name: normalized-losses, Length: 204, dtype: bool>
```

In [75]:
```
dataf['bore'] =  dataf['bore'].astype('float')
mean = dataf['bore'].mean()
dataf['bore'].replace(np.nan,mean,inplace=True)
dataf.head(10)
```

Out[75]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | 122.0 | alfa-romero | gas | std | two | convertible | rwd | front | 88.6 | ... | 130 | mpfi | 3.47 | 2.68 | |
| **1** | 1 | 122.0 | alfa-romero | gas | std | two | hatchback | rwd | front | 94.5 | ... | 152 | mpfi | 2.68 | 3.47 | |
| **2** | 2 | 164.0 | audi | gas | std | four | sedan | fwd | front | 99.8 | ... | 109 | mpfi | 3.19 | 3.40 | |
| **3** | 2 | 164.0 | audi | gas | std | four | sedan | 4wd | front | 99.4 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **4** | 2 | 122.0 | audi | gas | std | two | sedan | fwd | front | 99.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **5** | 1 | 158.0 | audi | gas | std | four | sedan | fwd | front | 105.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **6** | 1 | 122.0 | audi | gas | std | four | wagon | fwd | front | 105.8 | ... | 136 | mpfi | 3.19 | 3.40 | |
| **7** | 1 | 158.0 | audi | gas | turbo | four | sedan | fwd | front | 105.8 | ... | 131 | mpfi | 3.13 | 3.40 | |
| **8** | 0 | 122.0 | audi | gas | turbo | two | hatchback | 4wd | front | 99.5 | | 131 | mpfi | 3.13 | 3.40 | |

In [76]:
```
dataMiss = dataf.isnull()
print(dataMiss["bore"].value_counts())
```

```
False    204
Name: bore, dtype: int64
```

In [77]:
```
dataf['bore'].isnull().sum
```

Out[77]:
```
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
199    False
200    False
201    False
202    False
203    False
Name: bore, Length: 204, dtype: bool>
```

In [78]:
```
dataf['stroke'] =  dataf['stroke'].astype('float')
mean = dataf['stroke'].mean()
dataf['stroke'].replace(np.nan,mean,inplace=True)
dataf
dataMiss = dataf.isnull()
print(dataMiss["stroke"].value_counts())
dataf['stroke'].isnull().sum
```

Out[78]:
```
False    204
Name: stroke, dtype: int64
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
199    False
200    False
201    False
202    False
203    False
Name: stroke, Length: 204, dtype: bool>
```

In [79]:
```
dataf['horsepower'] =  dataf['horsepower'].astype('float')
mean = dataf['horsepower'].mean()
dataf['horsepower'].replace(np.nan,mean,inplace=True)
dataf
dataMiss = dataf.isnull()
print(dataMiss["horsepower"].value_counts())
dataf['horsepower'].isnull().sum
```

Out[79]:
```
False    204
Name: horsepower, dtype: int64
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
199    False
200    False
201    False
```

```
202    False
203    False
Name: horsepower, Length: 204, dtype: bool>
```

In [80]:
```
dataf['peak-rpm'] =  dataf['peak-rpm'].astype('float')
mean = dataf['peak-rpm'].mean()
dataf['peak-rpm'].replace(np.nan,mean,inplace=True)
dataf
dataMiss = dataf.isnull()
print(dataMiss["peak-rpm"].value_counts())
dataf['peak-rpm'].isnull().sum
```

Out[80]:
```
False    204
Name: peak-rpm, dtype: int64
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
199    False
200    False
201    False
202    False
203    False
Name: peak-rpm, Length: 204, dtype: bool>
```

In [81]:
```
dataf['price'] =  dataf['price'].astype('float')
mean = dataf['price'].mean()
dataf['price'].replace(np.nan,mean,inplace=True)
dataf
dataMiss = dataf.isnull()
print(dataMiss["price"].value_counts())
dataf['price'].isnull().sum
```

Out[81]:
```
False    204
Name: price, dtype: int64
<bound method NDFrame._add_numeric_operations.<locals>.sum of 0      False
1      False
2      False
3      False
4      False
       ...
```

```
199    False
200    False
201    False
202    False
203    False
Name: price, Length: 204, dtype: bool
```

In [82]:
```
x = dataf['num-of-doors'].value_counts().idxmax()
```

In [83]:
```
dataf['num-of-doors'].replace(np.nan,x,inplace=True)
dataf
```

Out[83]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | 122.0 | alfa-romero | gas | std | two | convertible | rwd | front | 88.6 | ... | 130 | mpfi | 3.47 | 2.68 |
| **1** | 1 | 122.0 | alfa-romero | gas | std | two | hatchback | rwd | front | 94.5 | ... | 152 | mpfi | 2.68 | 3.47 |
| **2** | 2 | 164.0 | audi | gas | std | four | sedan | fwd | front | 99.8 | ... | 109 | mpfi | 3.19 | 3.40 |
| **3** | 2 | 164.0 | audi | gas | std | four | sedan | 4wd | front | 99.4 | ... | 136 | mpfi | 3.19 | 3.40 |
| **4** | 2 | 122.0 | audi | gas | std | two | sedan | fwd | front | 99.8 | ... | 136 | mpfi | 3.19 | 3.40 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **199** | -1 | 95.0 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **200** | -1 | 95.0 | volvo | gas | turbo | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |
| **201** | -1 | 95.0 | volvo | gas | std | four | sedan | rwd | front | 109.1 | ... | 173 | mpfi | 3.58 | 2.87 |
| **202** | -1 | 95.0 | volvo | diesel | turbo | four | sedan | rwd | front | 109.1 | ... | 145 | idi | 3.01 | 3.40 |
| **203** | -1 | 95.0 | volvo | gas | turbo | four | sedan | rwd | front | 109.1 | ... | 141 | mpfi | 3.78 | 3.15 |

204 rows × 26 columns

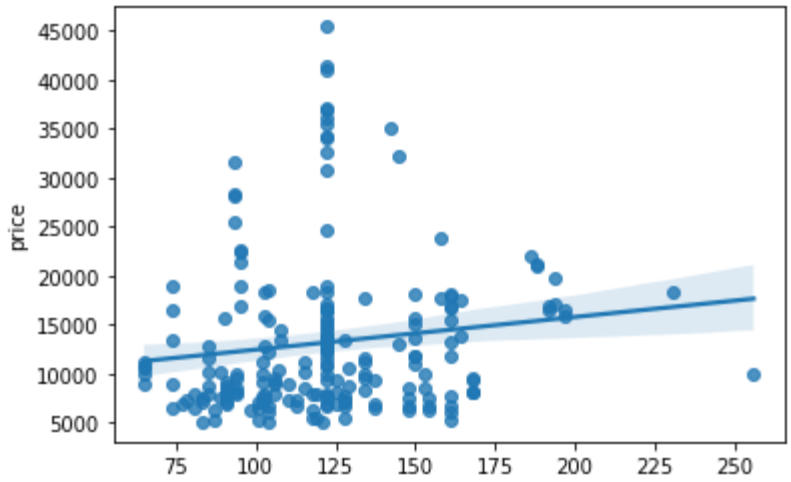
In [84]:
```
dataf.isna().sum()
```

Out[84]:
```
symboling            0
normalized-losses    0
make                 0
fuel-type            0
aspiration           0
num-of-doors         0
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
bore                 0
stroke               0
compression-ratio    0
horsepower           0
peak-rpm             0
city-mpg             0
highway-mpg          0
price                0
dtype: int64
```

In [85]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.regplot(dataf["normalized-losses"],dataf["price"])
#From this we conclude that normalized-losses very importance
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as ke
yword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

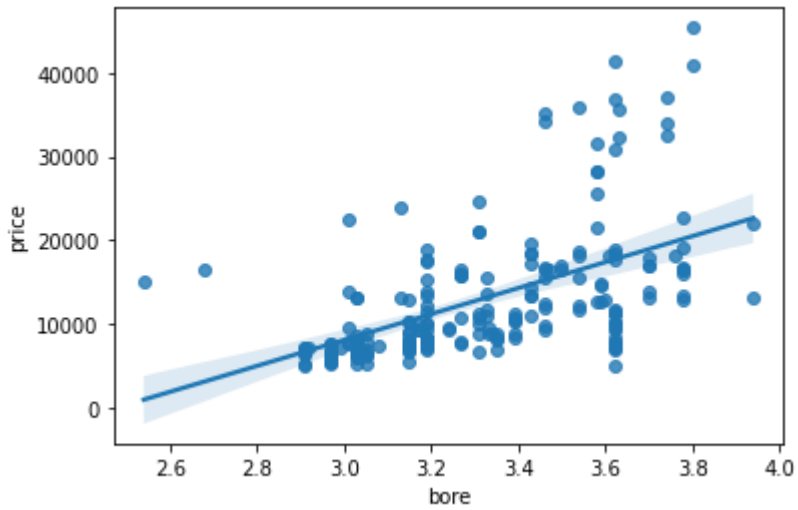
Out[85]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40cc6e390>
```




In [86]:
```
sns.regplot(dataf["bore"],dataf["price"])
#From this we conclude that bore very importance
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as ke
yword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

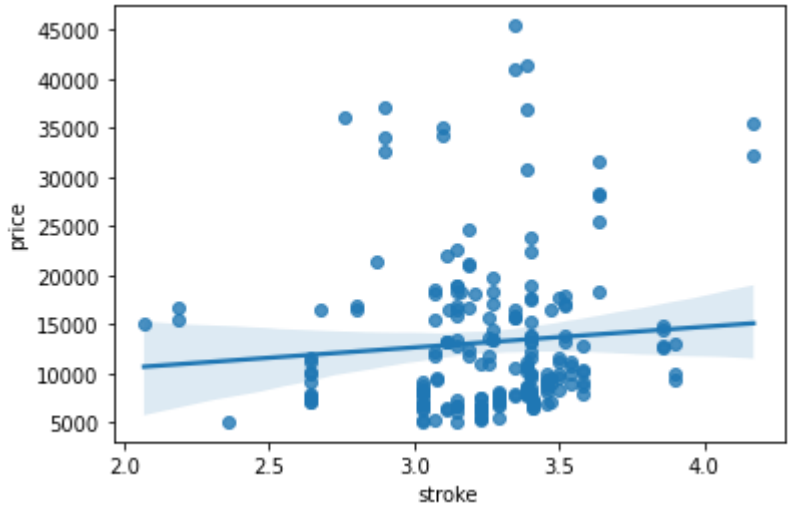
Out[86]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40cc57c50>
```



In [87]:
```
sns.regplot(dataf["stroke"],dataf["price"])
#From this we conclude that stroke not very importance
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as ke
yword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

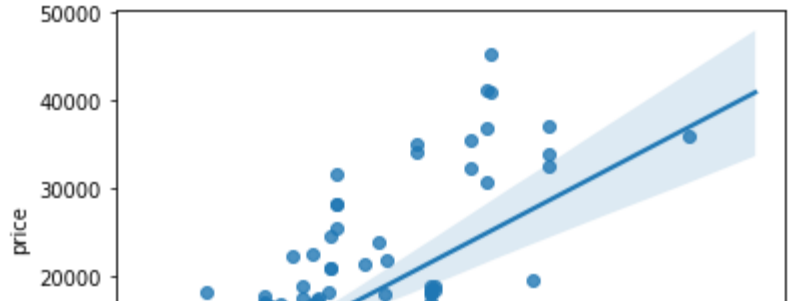
Out[87]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40cbc9fd0>
```




In [88]:
```
sns.regplot(dataf["horsepower"],dataf["price"])
#From this we conclude that horsepower very importance
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as ke
yword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

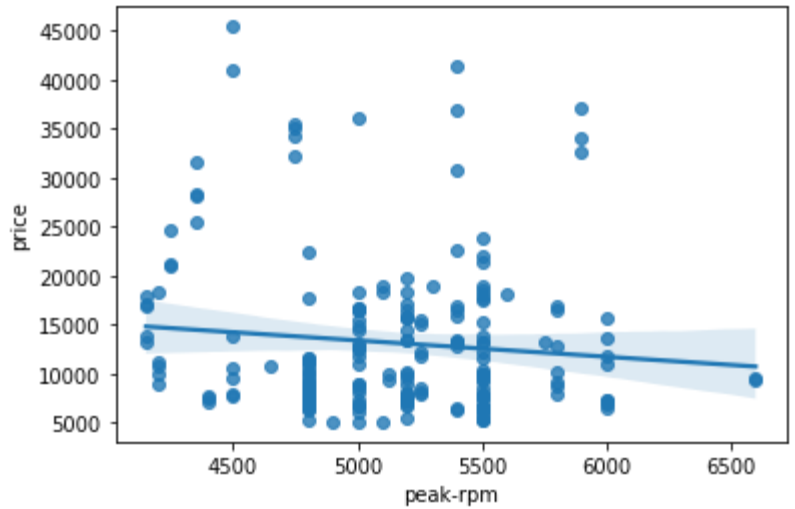
Out[88]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40cb3f150>
```




In [89]:
```
sns.regplot(dataf["peak-rpm"],dataf["price"])
#From this we conclude that peak-rpm not very importance
```

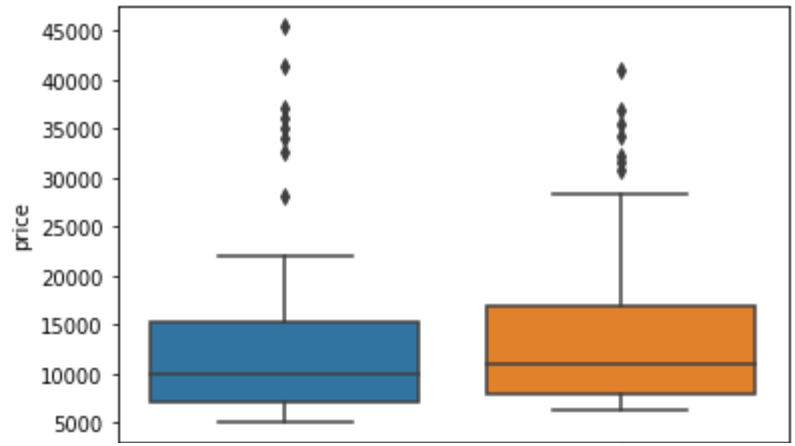
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as ke
yword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[89]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40caae490>
```



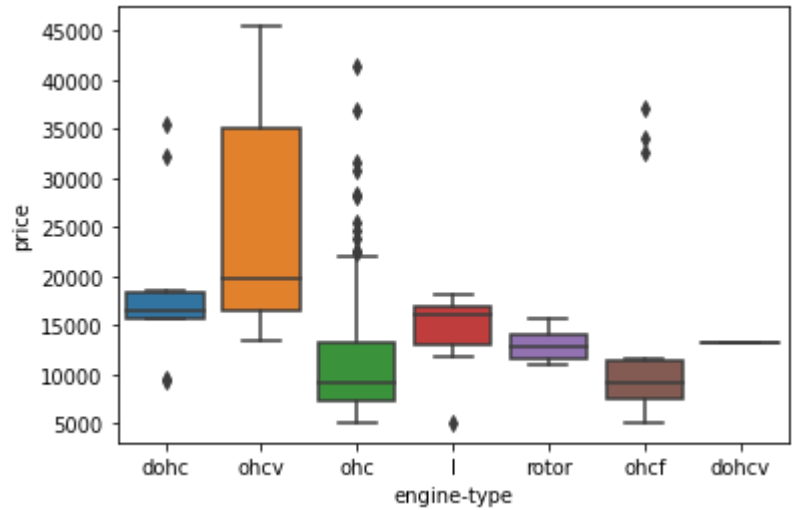

In [90]:
```
sns.boxplot(x="num-of-doors",y="price",data=dataf)
```

Out[90]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40ca8aad0>
```



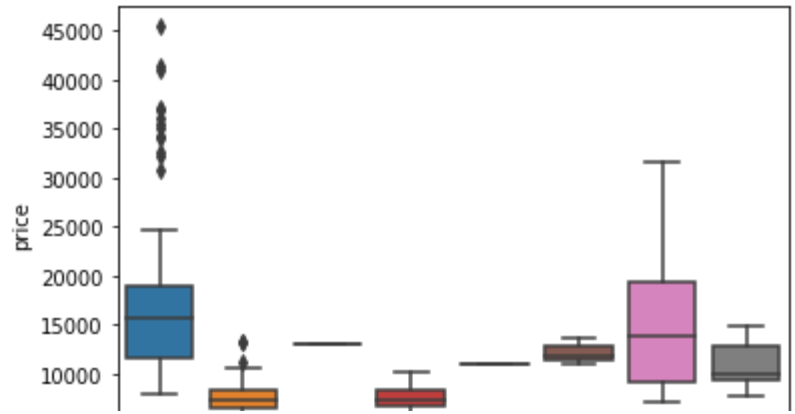

In [91]:
```
sns.boxplot(x="engine-type",y="price",data=dataf)
```

Out[91]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40ca1e4d0>
```



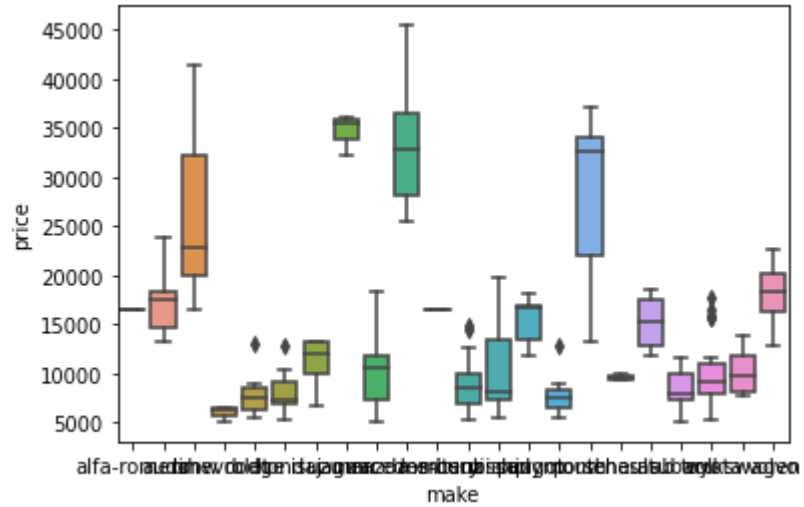

In [92]:
```
sns.boxplot(x="fuel-system",y="price",data=dataf)
```

Out[92]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c8b0910>
```



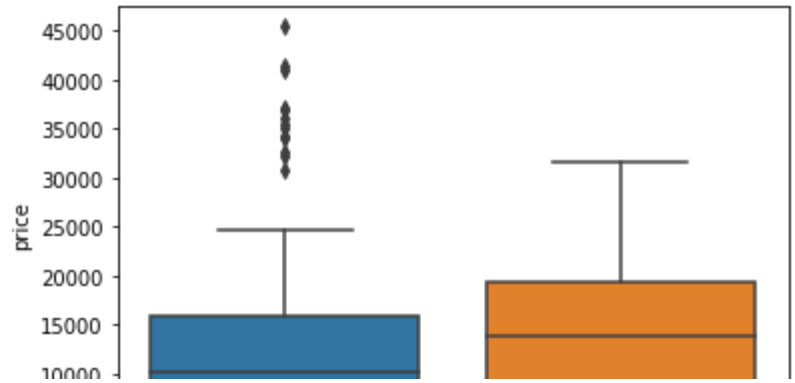

In [93]:
```
sns.boxplot(x="make",y="price",data=dataf)
```

Out[93]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c819e90>
```



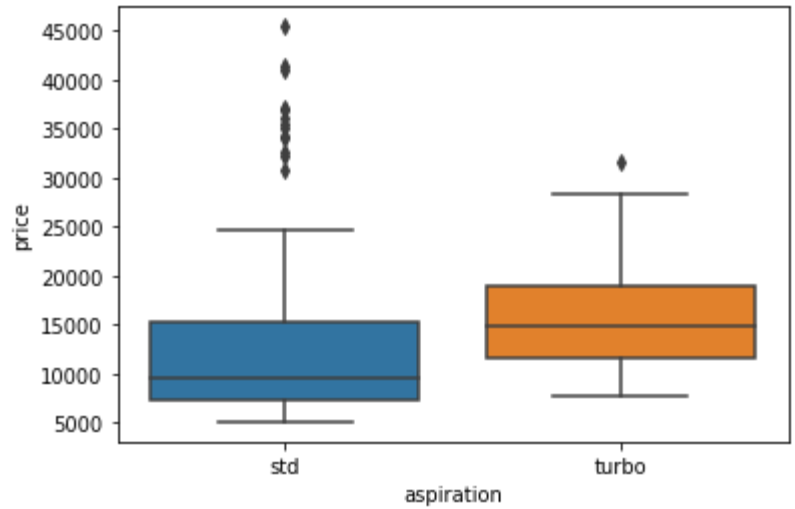

In [94]:
```
sns.boxplot(x="fuel-type",y="price",data=dataf)
```

Out[94]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c52e090>
```



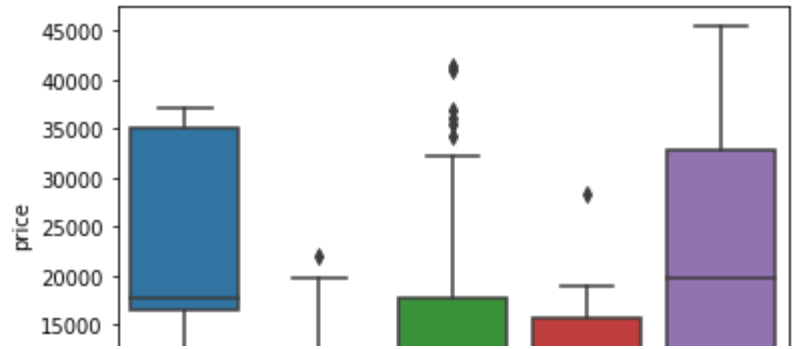

In [95]:
```
sns.boxplot(x="aspiration",y="price",data=dataf)
```

Out[95]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c547e90>
```



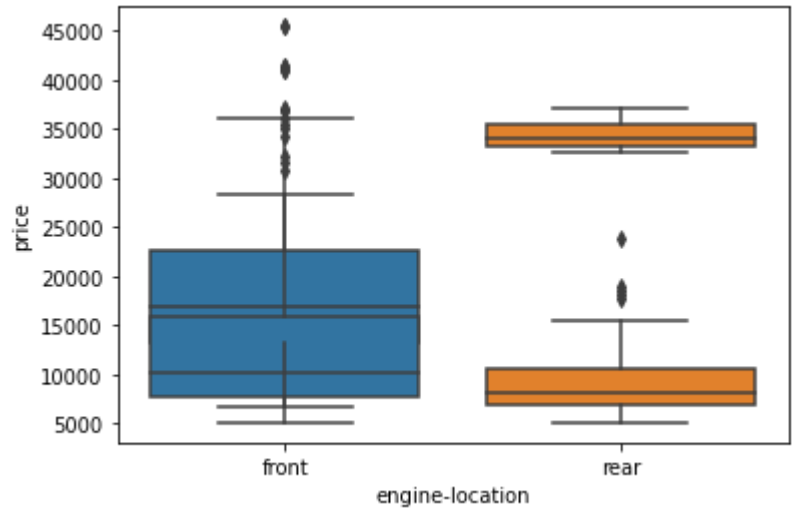

In [96]:
```
sns.boxplot(x="body-style",y="price",data=dataf)
```

Out[96]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c431390>
```



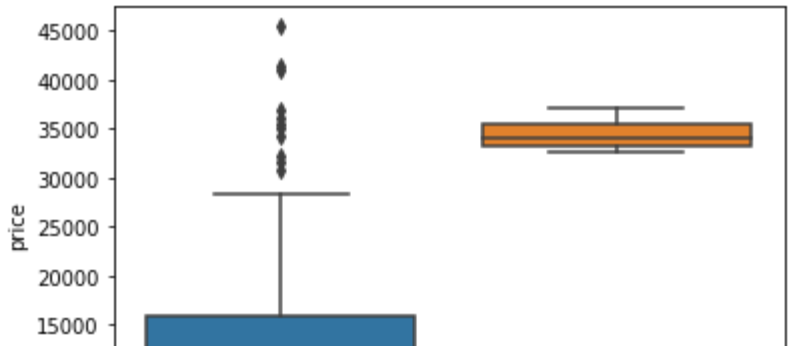

In [97]:
```
sns.boxplot(x="drive-wheels",y="price",data=dataf)
sns.boxplot(x="engine-location",y="price",data=dataf)
```

Out[97]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c4add50>
```




In [98]:
```
sns.boxplot(x="engine-location",y="price",data=dataf)
```

Out[98]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe40c2d3990>
```




In [99]:
```
dataf.corr()["price"]
```

Out[99]:
```
symboling           -0.083136
normalized-losses    0.133999
wheel-base           0.587607
length               0.683372
width                0.730130
height               0.136123
curb-weight          0.820831
engine-size          0.861753
bore                 0.532562
stroke               0.083115
compression-ratio    0.071058
horsepower           0.757943
peak-rpm            -0.100833
city-mpg            -0.668021
highway-mpg         -0.690937
price                1.000000
Name: price, dtype: float64
```

In [100…
```
#here we are doing ANOVA test for drive-wheels attribute
grouped_test2=dataf[['drive-wheels', 'price']].groupby(['drive-wheels'])
#we can use the function 'f_oneway' in the module 'stats' to obtain the F-test score and P-value
from scipy import stats
#because drive-wheels has unique values ('fwd','rwd','4wd') we pass it as argument of ANOVA function
f_val, p_val = stats.f_oneway(grouped_test2.get_group('fwd')['price'], grouped_test2.get_group('rwd')['price'], gro

print( "ANOVA results: F=", f_val, ", P =", p_val)
```

```
ANOVA results: F= 68.02912294265208 , P = 2.734359810651548e-23
```

In [101…

```
#dfn=number of columns-1
from scipy import stats
dfn1=3-1
dfd1=grouped_test2.get_group('fwd')['price'].shape[0]+grouped_test2.get_group('4wd')['price'].shape[0]+grouped_test
stats.f.ppf(q=1-.05, dfn=dfn1, dfd=dfd1) #for all categorical data
```

Out[101…

```
3.040828049372274
```

In [103…

```
#here we are doing ANOVA test for drive-wheels attribute
grouped_test2=dataf[['body-style', 'price']].groupby(['body-style'])
#we can use the function 'f_oneway' in the module 'stats' to obtain the F-test score and P-value
from scipy import stats
#because drive-wheels has unique values ('fwd','rwd','4wd') we pass it as argument of ANOVA function
f_val, p_val = stats.f_oneway(grouped_test2.get_group('convertible')['price'], grouped_test2.get_group('hatchback')

print( "ANOVA results: F=", f_val, ", P =", p_val)
```

```
ANOVA results: F= 13.515413171342297 , P = 3.6083258742924143e-06
```

In [104…

```
from scipy import stats
dfn1=3-1
dfd1=grouped_test2.get_group('convertible')['price'].shape[0]+grouped_test2.get_group('hatchback')['price'].shape[0
stats.f.ppf(q=1-.05, dfn=dfn1, dfd=dfd1) #for all categorical data
```

Out[104…

```
3.0497921314802525
```

In [106…

```
#here we are doing ANOVA test for drive-wheels attribute
grouped_test2=dataf[['fuel-type', 'price']].groupby(['fuel-type'])
#we can use the function 'f_oneway' in the module 'stats' to obtain the F-test score and P-value
from scipy import stats
#because drive-wheels has unique values ('fwd','rwd','4wd') we pass it as argument of ANOVA function
f_val, p_val = stats.f_oneway(grouped_test2.get_group('gas')['price'], grouped_test2.get_group('diesel')['price'])

print("ANOVA results: F=", f_val, ", P =", p_val)
```

```
ANOVA results: F= 2.4876668121727095 , P = 0.11630645825051396
```

In [107…

```
from scipy import stats
dfn1=3-1
dfd1=grouped_test2.get_group('gas')['price'].shape[0]+grouped_test2.get_group('diesel')['price'].shape[0]-3
stats.f.ppf(q=1-.05, dfn=dfn1, dfd=dfd1) #for all categorical data
```

Out[107…

```
3.040828049372274
```