Multiple Linear Regression with sklearn

data located in real_estate_price_size_year.csv

To do:

- Create a multiple linear regression
- Display the intercept and coefficient(s)
- Find the R-squared and Adjusted R-squared
- Compare the R-squared and the Adjusted R-squared
- Compare the R-squared of this regression and the simple linear regression where only 'size' was used
- Using the model make a prediction about an apartment with size 750 sq.ft. from 2009
- Find the univariate / multivariate p-values of the two variables. Describe.
- Create a summary table

In this exercise, the dependent variable is 'price', while the independent variables are 'size' and 'year'.

Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()

from sklearn.linear_model import LinearRegression
```

Load the data

In [2]:
```
data = pd.read_csv('real_estate_price_size_year.csv')
data.head()
```

Out[2]:

	price	size	year
0	234314.144	643.09	2015
1	228581.528	656.22	2009
2	281626.336	487.29	2018
3	401255.608	1504.75	2015
4	458674.256	1275.46	2009

In [3]:
```
data.describe()
```

Out[3]:

	price	size	year
count	100.000000	100.000000	100.000000
mean	292289.470160	853.024200	2012.600000
std	77051.727525	297.941951	4.729021
min	154282.128000	479.750000	2006.000000
25%	234280.148000	643.330000	2009.000000
50%	280590.716000	696.405000	2015.000000
75%	335723.696000	1029.322500	2018.000000
max	500681.128000	1842.510000	2018.000000

Regression

Declare the dependent and the independent variables

In [4]:
```
x = data[['size','year']]
y = data['price']
```

Regression

In [5]:
```
reg = LinearRegression()
reg.fit(x,y)
```

Out[5]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

Find the intercept

In [6]:
```
reg.intercept_
```

Out[6]:
```
-5772267.01746328
```

Find the coefficients

In [7]:
```
reg.coef_
```

Out[7]:
```
array([ 227.70085401, 2916.78532684])
```

Calculate the R-squared

In [8]:
```
reg.score(x,y)
```

Out[8]:
```
0.7764803683276792
```

Calculate the Adjusted R-squared

In [9]:
```
# Let's use the handy function we created
def adj_r2(x,y):
    r2 = reg.score(x,y)
    n = x.shape[0]
    p = x.shape[1]
    adjusted_r2 = 1-(1-r2)*(n-1)/(n-p-1)
    return adjusted_r2
```

In [10]:
```
adj_r2(x,y)
```

Out[10]:
```
0.7718717161282499
```

Compare the R-squared and the Adjusted R-squared

It seems the the R-squared is only slightly larger than the Adjusted R-squared, implying that we were not penalized a lot for the inclusion of 2 independent variables.

Compare the Adjusted R-squared with the R-squared of the simple linear regression

Comparing the Adjusted R-squared with the R-squared of the simple linear regression (when only 'size' was used - a couple of lectures ago), we realize that 'Year' is not bringing too much value to the result.

Making predictions

Find the predicted price of an apartment that has a size of 750 sq.ft. from 2009.

In [11]:
```
reg.predict([[750,2009]])
```

Out[11]:
```
array([258330.34465995])
```

Calculate the univariate p-values of the variables

In [12]:
```
from sklearn.feature_selection import f_regression
```

In [13]:
```
f_regression(x,y)
```

Out[13]:
```
(array([285.92105192,   0.85525799]), array([8.12763222e-31, 3.57340758e-01]))
```

In [14]:
```
p_values = f_regression(x,y)[1]
p_values
```

Out[14]:
```
array([8.12763222e-31, 3.57340758e-01])
```

In [15]:
```
p_values.round(3)
```

Out[15]:
```
array([0.   , 0.357])
```

Create a summary table with your findings

In [16]:
```
reg_summary = pd.DataFrame(data = x.columns.values, columns=['Features'])
reg_summary ['Coefficients'] = reg.coef_
reg_summary ['p-values'] = p_values.round(3)
reg_summary
```

Out[16]:

	Features	Coefficients	p-values
0	size	227.700854	0.000
1	year	2916.785327	0.357

It seems that 'Year' is not event significant, therefore we should remove it from the model.