Linear Regression with Python

The data contains the following columns:

- 'mean_of_area_income': Avg. Income of residents of the city house is located in.
- 'mean_of_area_house_age': Avg Age of Houses in same city
- 'mean_area_number_of_rooms': Avg Number of Rooms for Houses in same city
- 'mean_area_number_of_bedrooms': Avg Number of Bedrooms for Houses in same city
- 'population': Population of city house is located in
- 'selling_price': Price that the house sold at
- 'Adress': Address for the house

```
import pandas as pd
import numpy as np
```

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
%matplotlib inline
```

[## 3 Ways to Load CSV files into Colab:]

1) From Github (Files < 25MB)

```
#url = 'copied_raw_GH_link'
#df1 = pd.read_csv(url)
# Dataset is now stored in a Pandas Dataframe

url = 'https://raw.githubusercontent.com/GePajarinen/Machine-Learning-with-Python/mair
df = pd.read_csv(url)
```

```
df.head()
```

	mean_of_area_income	mean_of_area_house_age	mean_area_number_of_rooms	mean_a
0	79545.458574	5.682861	7.009188	
1	79248.642455	6.002900	6.730821	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   mean_of_area_income           5000 non-null   float64
 1   mean_of_area_house_age        5000 non-null   float64
 2   mean_area_number_of_rooms     5000 non-null   float64
 3   mean_area_number_of_bedrooms  5000 non-null   float64
 4   population                    5000 non-null   float64
 5   selling_price                 5000 non-null   float64
 6   Adress                        5000 non-null   object
dtypes: float64(6), object(1)
memory usage: 312.5+ KB
```

```
df.describe()
```

	mean_of_area_income	mean_of_area_house_age	mean_area_number_of_rooms	me
count	5000.000000	5000.000000	5000.000000	
mean	68583.108984	5.977222	6.987792	
std	10657.991214	0.991456	1.005833	
min	17796.631190	2.644304	3.236194	
25%	61480.562388	5.322283	6.299250	
50%	68804.286404	5.970429	7.002902	
75%	75783.338666	6.650808	7.665871	
max	107701.748378	9.519088	10.759588	

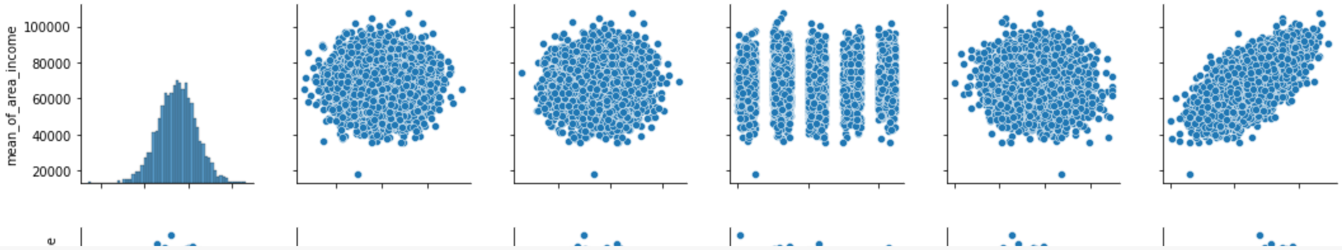
```
df.columns
```

```
Index(['mean_of_area_income', 'mean_of_area_house_age',
       'mean_area_number_of_rooms', 'mean_area_number_of_bedrooms',
       'population', 'selling_price', 'Adress'],
      dtype='object')
```

```
sns.pairplot(df)
```

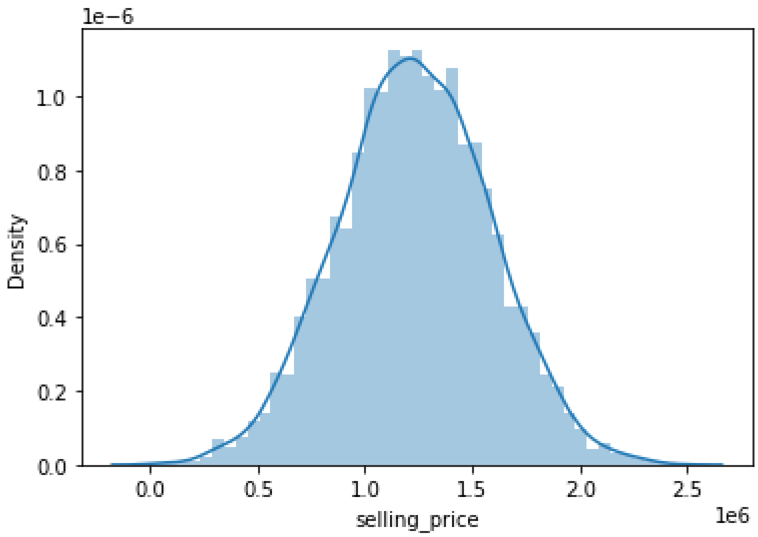
```
sns.pairplot(df)
```

<seaborn.axisgrid.PairGrid at 0x7f80baff4a10>



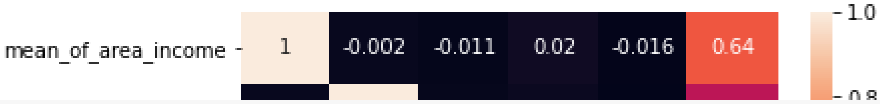

```
sns.distplot(df["selling_price"])
```

/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarn
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f80b25e7bd0>




```
sns.heatmap(df.corr(), annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f80ace89090>
```




```
X = df[['mean_of_area_income', 'mean_of_area_house_age',
       'mean_area_number_of_rooms', 'mean_area_number_of_bedrooms',
       'population']]
```


```
X = df.drop(["Adress","selling_price"], axis = 1)
```

```
y = df ["selling_price"]
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=
```

Podia olhar mais sobre o porquê dessas variáveis de tamanho e random.

```
from sklearn.linear_model import LinearRegression
```

```
lm = LinearRegression()
```

```
lm.fit(X_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
print(lm.intercept_)
```

```
-2640159.796852695
```

```
lm.coef_
```

```
array([2.15282755e+01, 1.64883282e+05, 1.22368678e+05, 2.23380186e+03,
       1.51504200e+01])
```

```
X_train.columns
```

```
Index(['mean_of_area_income', 'mean_of_area_house_age',
       'mean_area_number_of_rooms', 'mean_area_number_of_bedrooms',
       'population'],
      dtype='object')
```

```
cdf = pd.DataFrame(lm.coef_, X.columns, columns = ["Coeff"])
```

```
cdf
```

	Coeff
mean_of_area_income	21.528276
mean_of_area_house_age	164883.282027
mean_area_number_of_rooms	122368.678027
mean_area_number_of_bedrooms	2233.801864
population	15.150420

Interpreting the coefficients:

- Holding all other features fixed, a 1 unit increase in **mean_of_area_income** is associated with an **increase of $21.52 **.
- Holding all other features fixed, a 1 unit increase in **mean_of_area_house_age** is associated with an **increase of $164883.28 **.
- Holding all other features fixed, a 1 unit increase in **mean_area_number_of_rooms** is associated with an **increase of $122368.67 **.
- Holding all other features fixed, a 1 unit increase in **mean_area_number_of_bedrooms** is associated with an **increase of $2233.80 **.
- Holding all other features fixed, a 1 unit increase in **population** is associated with an **increase of $15.15 **.

```
from sklearn.datasets import load_boston
```

```
boston = load_boston()
```

```
boston.keys()
```

```
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
```

```
print(boston["data"])
```

```
[[6.3200e-03 1.8000e+01 2.3100e+00 ... 1.5300e+01 3.9690e+02 4.9800e+00]
 [2.7310e-02 0.0000e+00 7.0700e+00 ... 1.7800e+01 3.9690e+02 9.1400e+00]
 [2.7290e-02 0.0000e+00 7.0700e+00 ... 1.7800e+01 3.9283e+02 4.0300e+00]
 ...
 [6.0760e-02 0.0000e+00 1.1930e+01 ... 2.1000e+01 3.9690e+02 5.6400e+00]
 [1.0959e-01 0.0000e+00 1.1930e+01 ... 2.1000e+01 3.9345e+02 6.4800e+00]
 [4.7410e-02 0.0000e+00 1.1930e+01 ... 2.1000e+01 3.9690e+02 7.8800e+00]]
```

```
print(boston["target"])
```

```
[24.  21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 15.  18.9 21.7 20.4
 18.2 19.9 23.1 17.5 20.2 18.2 13.6 19.6 15.2 14.5 15.6 13.9 16.6 14.8
 18.4 21.  12.7 14.5 13.2 13.1 13.5 18.9 20.  21.  24.7 30.8 34.9 26.6
 25.3 24.7 21.2 19.3 20.  16.6 14.4 19.4 19.7 20.5 25.  23.4 18.9 35.4
 24.7 31.6 23.3 19.6 18.7 16.  22.2 25.  33.  23.5 19.4 22.  17.4 20.9
 24.2 21.7 22.8 23.4 24.1 21.4 20.  20.8 21.2 20.3 28.  23.9 24.8 22.9
 23.9 26.6 22.5 22.2 23.6 28.7 22.6 22.  22.9 25.  20.6 28.4 21.4 38.7
 43.8 33.2 27.5 26.5 18.6 19.3 20.1 19.5 19.5 20.4 19.8 19.4 21.7 22.8
 18.8 18.7 18.5 18.3 21.2 19.2 20.4 19.3 22.  20.3 20.5 17.3 18.8 21.4
 15.7 16.2 18.  14.3 19.2 19.6 23.  18.4 15.6 18.1 17.4 17.1 13.3 17.8
 14.  14.4 13.4 15.6 11.8 13.8 15.6 14.6 17.8 15.4 21.5 19.6 15.3 19.4
 17.  15.6 13.1 41.3 24.3 23.3 27.  50.  50.  50.  22.7 25.  50.  23.8
 23.8 22.3 17.4 19.1 23.1 23.6 22.6 29.4 23.2 24.6 29.9 37.2 39.8 36.2
 37.9 32.5 26.4 29.6 50.  32.  29.8 34.9 37.  30.5 36.4 31.1 29.1 50.
 33.3 30.3 34.6 34.9 32.9 24.1 42.3 48.5 50.  22.6 24.4 22.5 24.4 20.
 21.7 19.3 22.4 28.1 23.7 25.  23.3 28.7 21.5 23.  26.7 21.7 27.5 30.1
 44.8 50.  37.6 31.6 46.7 31.5 24.3 31.7 41.7 48.3 29.  24.  25.1 31.5
 23.7 23.3 22.  20.1 22.2 23.7 17.6 18.5 24.3 20.5 24.5 26.2 24.4 24.8
 29.6 42.8 21.9 20.9 44.  50.  36.  30.1 33.8 43.1 48.8 31.  36.5 22.8
 30.7 50.  43.5 20.7 21.1 25.2 24.4 35.2 32.4 32.  33.2 33.1 29.1 35.1
 45.4 35.4 46.  50.  32.2 22.  20.1 23.2 22.3 24.8 28.5 37.3 27.9 23.9
 21.7 28.6 27.1 20.3 22.5 29.  24.8 22.  26.4 33.1 36.1 28.4 33.4 28.2
 22.8 20.3 16.1 22.1 19.4 21.6 23.8 16.2 17.8 19.8 23.1 21.  23.8 23.1
 20.4 18.5 25.  24.6 23.  22.2 19.3 22.6 19.8 17.1 19.4 22.2 20.7 21.1
 19.5 18.5 20.6 19.  18.7 32.7 16.5 23.9 31.2 17.5 17.2 23.1 24.5 26.6
 22.9 24.1 18.6 30.1 18.2 20.6 17.8 21.7 22.7 22.6 25.  19.9 20.8 16.8
 21.9 27.5 21.9 23.1 50.  50.  50.  50.  50.  13.8 13.8 15.  13.9 13.3
 13.1 10.2 10.4 10.9 11.3 12.3  8.8  7.2 10.5  7.4 10.2 11.5 15.1 23.2
  9.7 13.8 12.7 13.1 12.5  8.5  5.   6.3  5.6  7.2 12.1  8.3  8.5  5.
 11.9 27.9 17.2 27.5 15.  17.2 17.9 16.3  7.   7.2  7.5 10.4  8.8  8.4
 16.7 14.2 20.8 13.4 11.7  8.3 10.2 10.9 11.   9.5 14.5 14.1 16.1 14.3
 11.7 13.4  9.6  8.7  8.4 12.8 10.5 17.1 18.4 15.4 10.8 11.8 14.9 12.6
 14.1 13.  13.4 15.2 16.1 17.8 14.9 14.1 12.7 13.5 14.9 20.  16.4 17.7
 19.5 20.2 21.4 19.9 19.  19.1 19.1 20.1 19.9 19.6 23.2 29.8 13.8 13.3
 16.7 12.  14.6 21.4 23.  23.7 25.  21.8 20.6 21.2 19.1 20.6 15.2  7.
  8.1 13.6 20.1 21.8 24.5 23.1 19.7 18.3 21.2 17.5 16.8 22.4 20.6 23.9
 22.  11.9]
```

Does this make sense? Probably not because I made up this data. If you want real data to repeat this sort of analysis, check out the boston dataset: