Practical-2

1. Ensure that the numpy, scikit learn, and matplotlib libraries are available in your system. Create the requirements.txt file and make a note of the versions of these libraries.

```
[ ] import numpy as np
    np.__version__
```

```
'1.23.5'
```

```
[ ] import sklearn
    sklearn.__version__
```

```
'1.2.2'
```

```
[ ] import matplotlib as plt
    plt.__version__
```

```
'3.7.1'
```

```
[ ] import pandas as pd
    pd.__version__
```

```
'1.5.3'
```

2. Write a python code to import the Sample.txt data. Further, apply the following processes on the imported data

```
data = pd.read_csv('/content/Housing.csv')
print(data)
```

```
        price  area  bedrooms  bathrooms  stories mainroad guestroom basement  \
0    13300000  7420         4          2        3      yes        no       no
1    12250000  8960         4          4        4      yes        no       no
2    12250000  9960         3          2        2      yes        no      yes
3    12215000  7500         4          2        2      yes        no      yes
4    11410000  7420         4          1        2      yes       yes      yes
..        ...   ...       ...        ...      ...      ...       ...      ...
540   1820000  3000         2          1        1      yes        no      yes
541   1767150  2400         3          1        1       no        no       no
542   1750000  3620         2          1        1      yes        no       no
543   1750000  2910         3          1        1       no        no       no
544   1750000  3850         3          1        2      yes        no       no
```

```
[ ] x.shape,y.shape
```

```
((545,), (545,))
```

```
x_train_normalized = scaler.transform(x_train_reshape)
x_train_normalized
```

```
array([[ 3.84168192e-01],
       [ 9.29180700e-01],
       [-6.07754573e-01],
       [-1.15549214e+00],
       [-6.37730261e-01],
       [-6.54671275e-02],
       [-5.92312552e-01],
       [-3.54914395e-02],
       [-8.96611202e-01],
       [ 3.93251733e-01],
       [-9.23861828e-01],
       [ 1.15626924e+00],
       [ 8.38345282e-01],
       [-7.19482137e-01],
       [-9.78363078e-01],
       [ 5.88547882e-01],
       [-1.25086933e+00],
       [ 6.62442286e-02],
       [ 1.03091637e+00],
       [ 3.15010667e+00],
       [ 9.56431325e-01],
       [-9.78363078e-01],
       [ 2.08265196e-02],
       [ 1.47419321e+00],
       [ 6.22611164e-01],
       [-6.83147970e-01],
       [-6.89960626e-01],
       [-5.24185988e-01],
```

```
x_test_normalized = scaler.transform(x_test_reshape)
x_test_normalized
```

```
array([[ 0.33875048],
       [ 0.61125674],
       [-0.5060189 ],
       [-0.0700089 ],
       [-0.54235307],
       [ 0.7111757 ],
       [ 1.52869446],
       [-0.07455067],
       [-0.86936058],
       [-1.11461621],
       [ 1.55594508],
       [-0.55416168],
       [-0.65135557],
       [-0.93294537],
       [-0.5378113 ],
       [-0.69223151],
       [-1.43254017],
       [ 0.38416819],
       [ 0.31604163],
       [ 0.38416819],
       [-0.06092536],
       [ 0.75886429],
       [-0.68360215],
       [-0.61502141],
       [ 1.40606664],
       [ 2.18270947],
       [-0.93748714],
       [-0.97836308],
       [ 3.65424324],
```

```
[ ]  np.mean(x_test_normalized)
```

```
-0.008182479453515227
```

```
np.mean(x_train_normalized)
```

```
1.2222638803212733e-16
```

```
[ ]  np.std(x_test_normalized)
```

```
0.9210591725069316
```

```
[ ]  np.std(x_train_normalized)
```

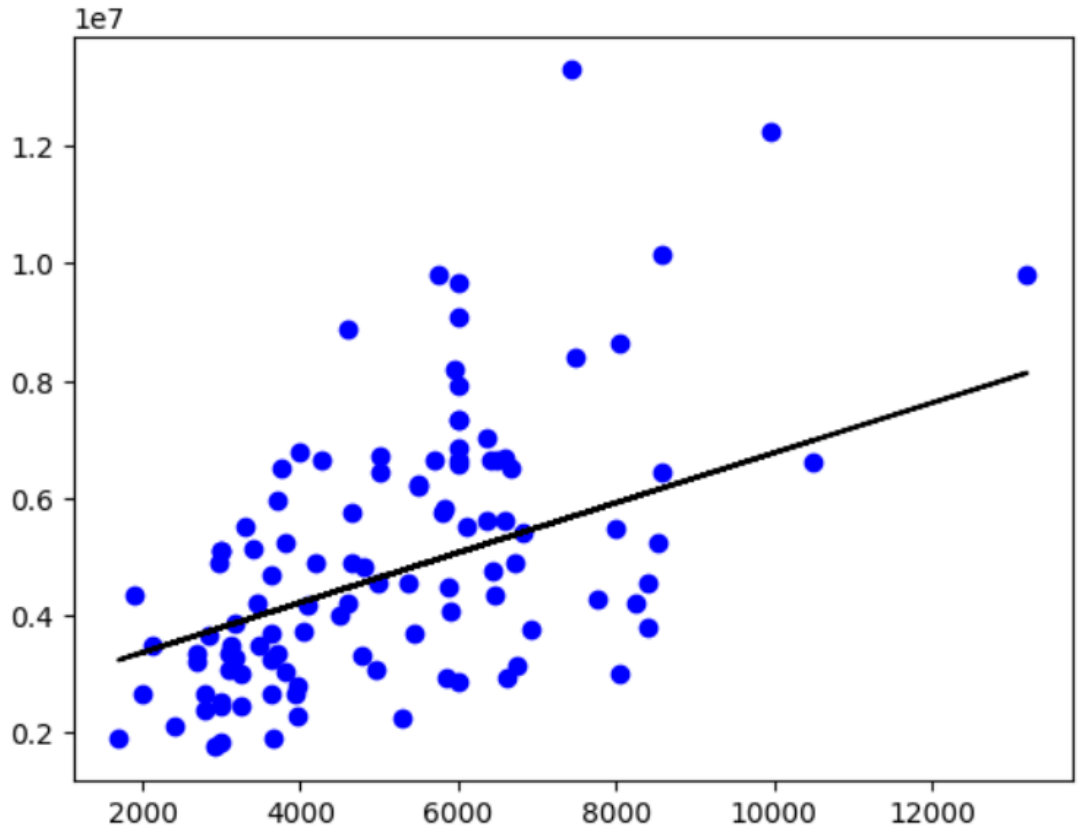
```
1.0000000000000002
```

```
[ ]  mae = mean_absolute_error(y_test, Predictions)

     #squared True returns MSE value, False returns RMSE value.
     mse = mean_squared_error(y_test, Predictions) #default=True
     rmse = mean_squared_error(y_test, Predictions,squared=False)
     print("MAE:",mae)
     print("MSE:",mse)
     print("RMSE:",rmse)
```

```
MAE: 1474748.1337969352
MSE: 3675286604768.1855
RMSE: 1917103.7021424235
```

```
import matplotlib.pyplot as plt
plt.scatter(x_test, y_test, color ='b')
plt.plot(x_test, Predictions, color ='k')
plt.show()
```



test_normalization.ipynb file screenshots:

```
x_test_normalized
```

```
array([[ 0.33875048],
       [ 0.61125674],
       [-0.5060189 ],
       [-0.0700089 ],
       [-0.54235307],
       [ 0.7111757 ],
       [ 1.52869446],
       [-0.07455067],
       [-0.86936058],
       [-1.11461621],
       [ 1.55594508],
       [-0.55416168],
       [-0.65135557],
       [-0.93294537],
       [-0.5378113 ],
       [-0.69223151],
       [-1.43254017],
       [ 0.38416819],
       [ 0.31604163],
       [ 0.38416819],
       [-0.06092536],
       [ 0.75886429],
       [-0.68360215],
       [-0.61502141],
       [ 1.40606664],
       [ 2.18270947],
       [-0.93748714],
       [-0.97836308],
       [ 3.65424324],
```

MI-Ops Practical Screenshots

```
np.mean(x_test_normalized)
```

```
-0.008182479453515227
```

```
[ ] np.std(x_test_normalized)
```

```
0.9210591725069316
```

```
[ ] x_test
```

```
array([[ 5900],
       [ 6500],
       [ 4040],
       [ 5000],
       [ 3960],
       [ 6720],
       [ 8520],
       [ 4990],
       [ 3240],
       [ 2700],
       [ 8580],
       [ 3934],
       [ 3720],
       [ 3100],
       [ 3970],
       [ 3630],
       [ 2000],
       [ 6000],
       [ 5850],
       [ 6000],
       [ 5020],
```

```
x_test_normalized_Main == x_test_normalized
```

```
array([[ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
```

```
[ ] y_test.shape
```

```
(109,)
```

```
mae = mean_absolute_error(y_test, predictions)

#squared True returns MSE value, False returns RMSE value.
mse = mean_squared_error(y_test, predictions) #default=True
rmse = mean_squared_error(y_test, predictions,squared=False)
print("MAE:",mae)
print("MSE:",mse)
print("RMSE:",rmse)
```

```
MAE: 1474748.1337969352
MSE: 3675286604768.1855
RMSE: 1917103.7021424235
```