sdkver7fm

July 31, 2023

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:
```
df=pd.read_csv("/content/2_2015.csv")
df
```

[ ]:
```
         Country                           Region  Happiness Rank  \
0    Switzerland                   Western Europe               1
1        Iceland                   Western Europe               2
2        Denmark                   Western Europe               3
3         Norway                   Western Europe               4
4         Canada                    North America               5
..             …                                …             …
153       Rwanda               Sub-Saharan Africa             154
154        Benin               Sub-Saharan Africa             155
155        Syria  Middle East and Northern Africa             156
156      Burundi               Sub-Saharan Africa             157
157         Togo               Sub-Saharan Africa             158

     Happiness Score  Standard Error  Economy (GDP per Capita)   Family  \
0              7.587         0.03411                   1.39651  1.34951
1              7.561         0.04884                   1.30232  1.40223
2              7.527         0.03328                   1.32548  1.36058
3              7.522         0.03880                   1.45900  1.33095
4              7.427         0.03553                   1.32629  1.32261
..                 …               …                        …       …
153            3.465         0.03464                   0.22208  0.77370
154            3.340         0.03656                   0.28665  0.35386
155            3.006         0.05015                   0.66320  0.47489
156            2.905         0.08658                   0.01530  0.41587
157            2.839         0.06727                   0.20868  0.13995

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)  \
0                     0.94143  0.66557                        0.41978
1                     0.94784  0.62877                        0.14145
```

```
2                     0.87464  0.64938                        0.48357
3                     0.88521  0.66973                        0.36503
4                     0.90563  0.63297                        0.32957
..                        …      …                            …
153                   0.42864  0.59201                        0.55191
154                   0.31910  0.48450                        0.08010
155                   0.72193  0.15684                        0.18906
156                   0.22396  0.11850                        0.10062
157                   0.28443  0.36453                        0.10731

     Generosity  Dystopia Residual
0       0.29678            2.51738
1       0.43630            2.70201
2       0.34139            2.49204
3       0.34699            2.46531
4       0.45811            2.45176
..          …                …
153     0.22628            0.67042
154     0.18260            1.63328
155     0.47179            0.32858
156     0.19727            1.83302
157     0.16681            1.56726

[158 rows x 12 columns]
```

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

```
[ ]:        Country          Region  Happiness Rank  Happiness Score  \
     0  Switzerland  Western Europe               1            7.587
     1      Iceland  Western Europe               2            7.561
     2      Denmark  Western Europe               3            7.527
     3       Norway  Western Europe               4            7.522
     4       Canada   North America               5            7.427

        Standard Error  Economy (GDP per Capita)   Family  \
     0         0.03411                   1.39651  1.34951
     1         0.04884                   1.30232  1.40223
     2         0.03328                   1.32548  1.36058
     3         0.03880                   1.45900  1.33095
     4         0.03553                   1.32629  1.32261

        Health (Life Expectancy)  Freedom  Trust (Government Corruption)  \
     0                   0.94143  0.66557                        0.41978
     1                   0.94784  0.62877                        0.14145
     2                   0.87464  0.64938                        0.48357
     3                   0.88521  0.66973                        0.36503
     4                   0.90563  0.63297                        0.32957
```

```
   Generosity  Dystopia Residual
0     0.29678            2.51738
1     0.43630            2.70201
2     0.34139            2.49204
3     0.34699            2.46531
4     0.45811            2.45176
```

1 DATA CLEANING AND DATA PREPROCESSING

[ ]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 12 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   Country                        158 non-null    object
 1   Region                         158 non-null    object
 2   Happiness Rank                 158 non-null    int64
 3   Happiness Score                158 non-null    float64
 4   Standard Error                 158 non-null    float64
 5   Economy (GDP per Capita)       158 non-null    float64
 6   Family                         158 non-null    float64
 7   Health (Life Expectancy)       158 non-null    float64
 8   Freedom                        158 non-null    float64
 9   Trust (Government Corruption)  158 non-null    float64
 10  Generosity                     158 non-null    float64
 11  Dystopia Residual              158 non-null    float64
dtypes: float64(9), int64(1), object(2)
memory usage: 14.9+ KB
```

[ ]:

```
df.describe()
```

[ ]:

```
       Happiness Rank  Happiness Score  Standard Error  \
count      158.000000       158.000000      158.000000
mean        79.493671         5.375734        0.047885
std         45.754363         1.145010        0.017146
min          1.000000         2.839000        0.018480
25%         40.250000         4.526000        0.037268
50%         79.500000         5.232500        0.043940
75%        118.750000         6.243750        0.052300
max        158.000000         7.587000        0.136930

       Economy (GDP per Capita)      Family  Health (Life Expectancy)  \
count                158.000000  158.000000                158.000000
```

```
mean                   0.846137    0.991046                   0.630259
std                    0.403121    0.272369                   0.247078
min                    0.000000    0.000000                   0.000000
25%                    0.545808    0.856823                   0.439185
50%                    0.910245    1.029510                   0.696705
75%                    1.158448    1.214405                   0.811013
max                    1.690420    1.402230                   1.025250

          Freedom  Trust (Government Corruption)  Generosity  \
count  158.000000                     158.000000  158.000000
mean     0.428615                       0.143422    0.237296
std      0.150693                       0.120034    0.126685
min      0.000000                       0.000000    0.000000
25%      0.328330                       0.061675    0.150553
50%      0.435515                       0.107220    0.216130
75%      0.549092                       0.180255    0.309883
max      0.669730                       0.551910    0.795880

       Dystopia Residual
count         158.000000
mean            2.098977
std             0.553550
min             0.328580
25%             1.759410
50%             2.095415
75%             2.462415
max             3.602140
```

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

```
[ ]: Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
            'Standard Error', 'Economy (GDP per Capita)', 'Family',
            'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
            'Generosity', 'Dystopia Residual'],
           dtype='object')
```

```
[ ]: df1=df.dropna(axis=1)
     df1
```

```
[ ]:         Country                           Region  Happiness Rank  \
     0    Switzerland                   Western Europe               1
     1        Iceland                   Western Europe               2
     2        Denmark                   Western Europe               3
     3         Norway                   Western Europe               4
     4         Canada                    North America               5
     ..           …                               …              …
     153       Rwanda               Sub-Saharan Africa             154
```

```
154        Benin               Sub-Saharan Africa             155
155        Syria  Middle East and Northern Africa             156
156      Burundi               Sub-Saharan Africa             157
157         Togo               Sub-Saharan Africa             158

     Happiness Score  Standard Error  Economy (GDP per Capita)   Family  \
0              7.587         0.03411                   1.39651  1.34951
1              7.561         0.04884                   1.30232  1.40223
2              7.527         0.03328                   1.32548  1.36058
3              7.522         0.03880                   1.45900  1.33095
4              7.427         0.03553                   1.32629  1.32261
..               …             …                       …      …
153            3.465         0.03464                   0.22208  0.77370
154            3.340         0.03656                   0.28665  0.35386
155            3.006         0.05015                   0.66320  0.47489
156            2.905         0.08658                   0.01530  0.41587
157            2.839         0.06727                   0.20868  0.13995

     Health (Life Expectancy)  Freedom  Trust (Government Corruption)  \
0                     0.94143  0.66557                        0.41978
1                     0.94784  0.62877                        0.14145
2                     0.87464  0.64938                        0.48357
3                     0.88521  0.66973                        0.36503
4                     0.90563  0.63297                        0.32957
..                        …      …                            …
153                   0.42864  0.59201                        0.55191
154                   0.31910  0.48450                        0.08010
155                   0.72193  0.15684                        0.18906
156                   0.22396  0.11850                        0.10062
157                   0.28443  0.36453                        0.10731

     Generosity  Dystopia Residual
0       0.29678            2.51738
1       0.43630            2.70201
2       0.34139            2.49204
3       0.34699            2.46531
4       0.45811            2.45176
..          …                …
153     0.22628            0.67042
154     0.18260            1.63328
155     0.47179            0.32858
156     0.19727            1.83302
157     0.16681            1.56726

[158 rows x 12 columns]
```

[ ]:

```
df1.columns
```

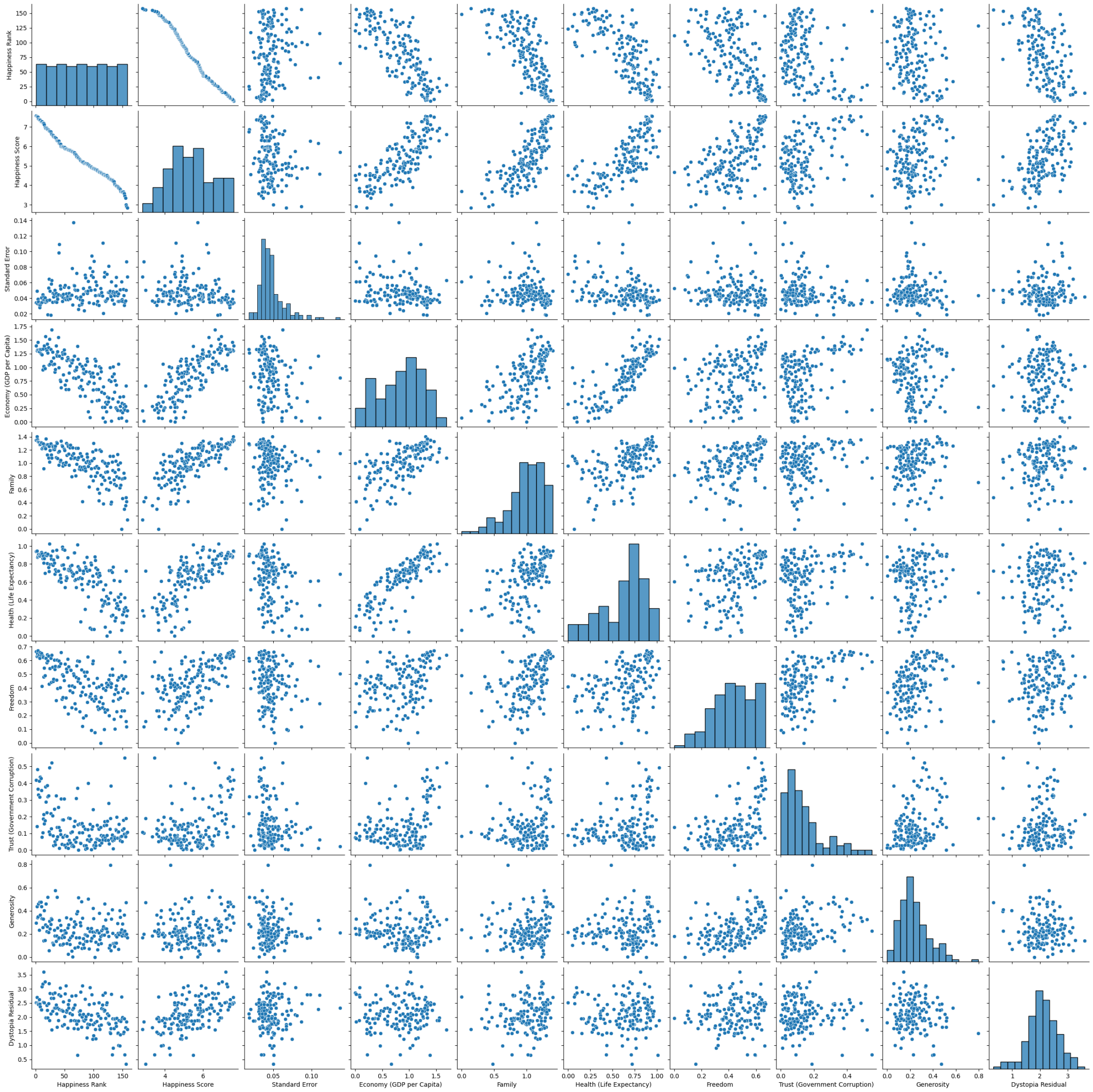
```
[ ]: Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
            'Standard Error', 'Economy (GDP per Capita)', 'Family',
            'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
            'Generosity', 'Dystopia Residual'],
           dtype='object')
```

```
[ ]: df1=df1[[ 'Happiness Rank', 'Happiness Score',
            'Standard Error', 'Economy (GDP per Capita)', 'Family',
            'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
            'Generosity', 'Dystopia Residual']]
```

2 EDA AND VISUALIZATION

```
[ ]: sns.pairplot(df1)
```

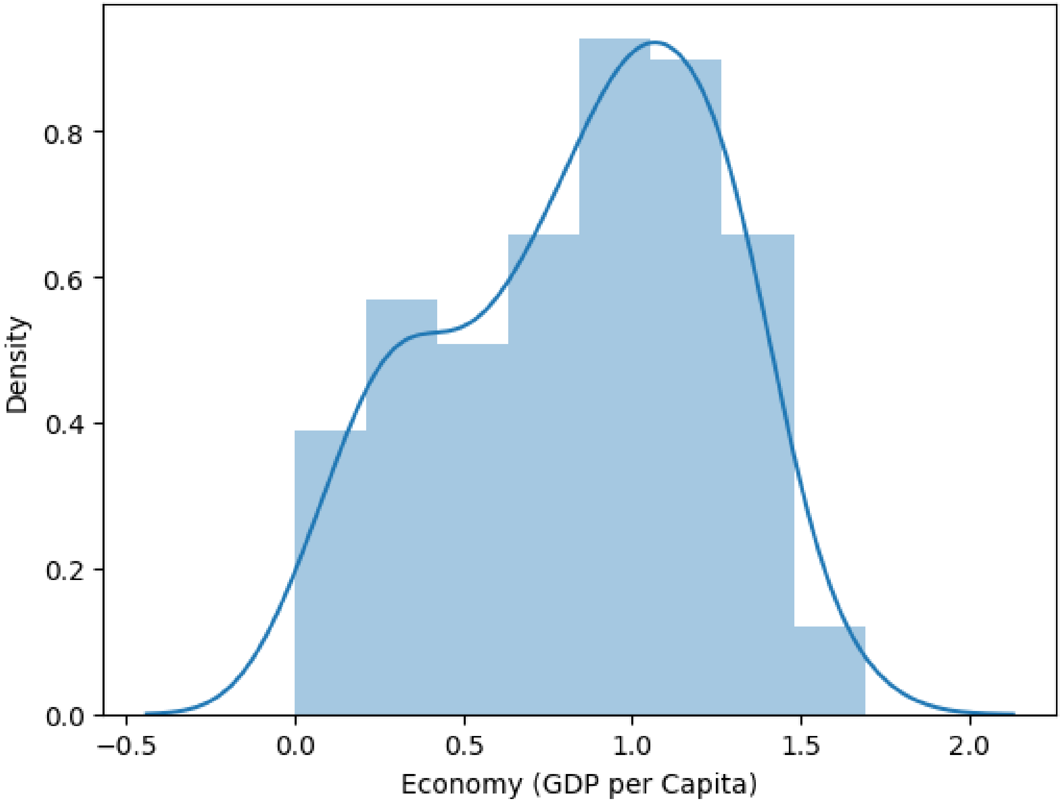
```
[ ]: <seaborn.axisgrid.PairGrid at 0x78aa12495810>
```



```
[ ]: sns.distplot(df1['Economy (GDP per Capita)'])
```

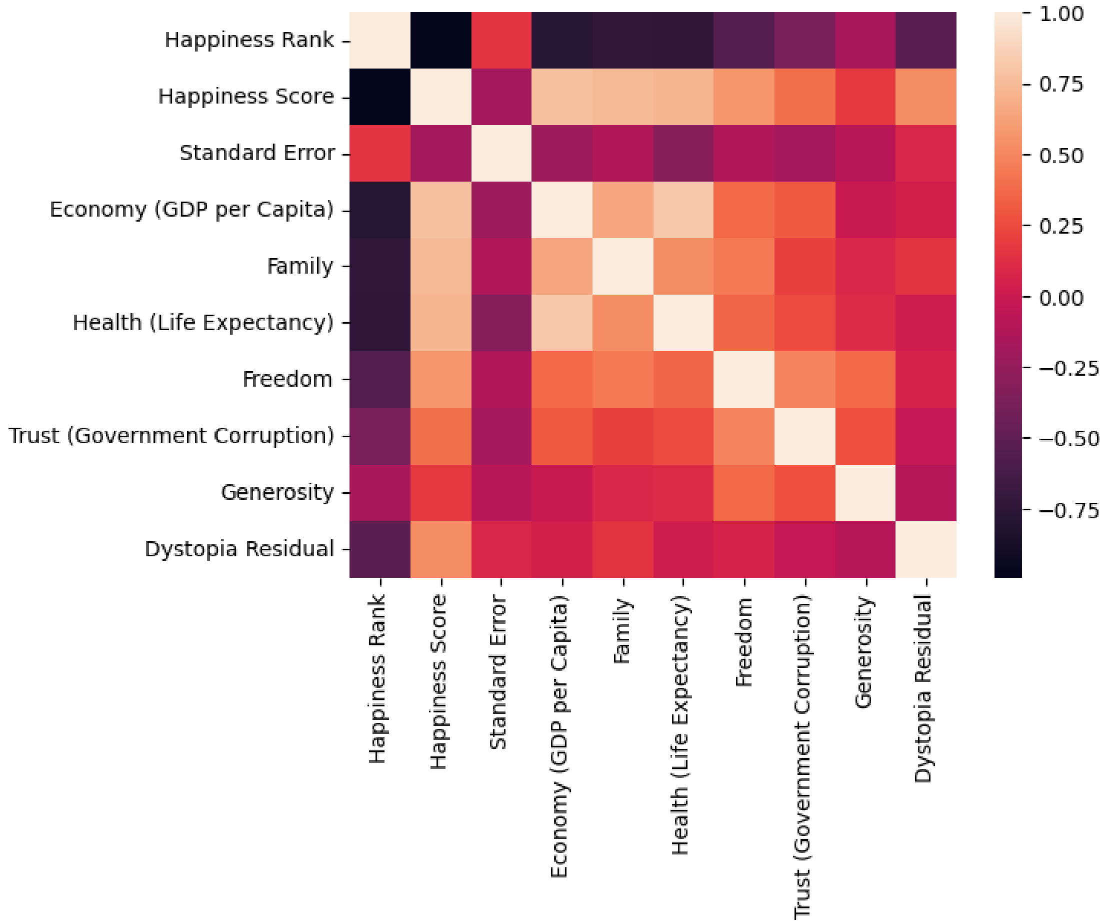
```
<ipython-input-11-94a07f5b2384>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df1['Economy (GDP per Capita)'])
```

```
[ ]: <Axes: xlabel='Economy (GDP per Capita)', ylabel='Density'>
```



```
[ ]: sns.heatmap(df1.corr())
```

```
[ ]: <Axes: >
```



3 TO TRAIN THE MODEL AND MODEL BULDING

```
[ ]: x=df[['Happiness Rank', 'Happiness Score',
            'Standard Error', 'Economy (GDP per Capita)', 'Family',
            'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
            'Generosity',]]
     y=df['Dystopia Residual']
```

```
[ ]: from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

```
[ ]: from sklearn.linear_model import LinearRegression
     lr=LinearRegression()
     lr.fit(x_train,y_train)
```

```
[ ]: LinearRegression()
```

[ ]:
```
lr.intercept_
```

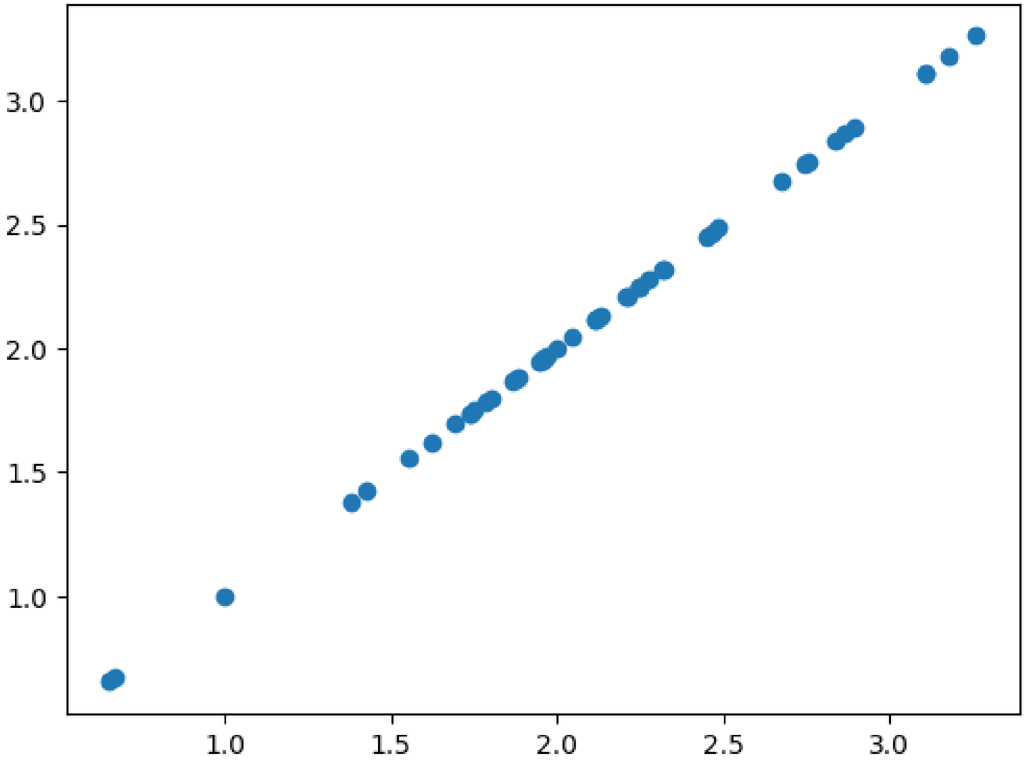
[ ]:
```
-0.00045276788722903305
```

[ ]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

[ ]:
```
                              Co-efficient
Happiness Rank                    0.000001
Happiness Score                   1.000068
Standard Error                   -0.000824
Economy (GDP per Capita)         -1.000162
Family                           -1.000085
Health (Life Expectancy)         -0.999768
Freedom                          -0.999633
Trust (Government Corruption)    -1.000152
Generosity                       -0.999952
```

[ ]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

[ ]:
```
<matplotlib.collections.PathCollection at 0x78aa06113d60>
```



4 ACCURACY

```
[ ]: lr.score(x_test,y_test)
```

```
[ ]: 0.9999997029885788
```

```
[ ]: lr.score(x_train,y_train)
```

```
[ ]: 0.9999997644485602
```

```
[ ]: from sklearn.linear_model import Ridge,Lasso
```

```
[ ]: rr=Ridge(alpha=10)
     rr.fit(x_train,y_train)
```

```
[ ]: Ridge(alpha=10)
```

```
[ ]: rr.score(x_test,y_test)
```

```
[ ]: 0.6083965596410505
```

```
[ ]: rr.score(x_train,y_train)
```

```
[ ]: 0.6654787639452585
```

```
[ ]: la=Lasso(alpha=10)
     la.fit(x_train,y_train)
```

```
[ ]: Lasso(alpha=10)
```

```
[ ]: la.score(x_test,y_test)
```

```
[ ]: 0.10331856539439044
```

```
[ ]: la.score(x_train,y_train)
```

```
[ ]: 0.1292605237518104
```

```
[ ]: from sklearn.linear_model import ElasticNet
     en=ElasticNet()
     en.fit(x_train,y_train)
```

```
[ ]: ElasticNet()
```

```
[ ]: print(en.coef_)
     print(en.intercept_)
```

```
[-0.00600835  0.          0.         -0.         -0.         -0.
 -0.         -0.         -0.        ]
2.5794025545873085
```

```
[ ]: prediction = en.predict(x_test)
     prediction
```

```
[ ]: array([2.22490994, 1.82835887, 2.40516042, 1.70218353, 1.65411674,
            2.37511867, 2.45923557, 1.96655091, 1.91848411, 2.13478469,
            2.48326896, 2.26696838, 1.97255926, 2.1107513 , 1.73823363,
            1.90646741, 1.93650916, 2.50730236, 1.85840062, 2.14680139,
            2.0386511 , 2.32104353, 1.87041732, 2.21289324, 2.25495168,
            2.41116877, 2.0987346 , 2.34507693, 2.04465945, 1.87642567,
            2.53734411, 1.73222528, 2.38713537, 2.47726061, 1.94251751,
            1.75625868, 2.42318547, 1.72020858, 2.20688489, 2.02062605,
            1.85239227, 2.54936081, 1.93050081, 2.39915207, 2.41717712,
            2.09272625, 2.30902683, 2.28499343])
```

```
[ ]: en.score(x_test,y_test)
```

```
[ ]: 0.23949751888228477
```

```
[ ]: from sklearn import metrics
     print("Mean Absolute Error: ", metrics.mean_absolute_error(y_test,prediction))
     print("Mean Squared Error: ", metrics.mean_squared_error(y_test,prediction))
     print("Root Mean Squared Error: ", np.sqrt(metrics.
      ↪mean_squared_error(y_test,prediction)))
```

```
Mean Absolute Error:  0.3860739575661493
Mean Squared Error:  0.25217282492057863
Root Mean Squared Error:  0.5021681241582132
```