giot5xc72

July 28, 2023

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

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

[2]:

	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]
```

[3]:
```
df.head()
```

[3]:
```
       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

[4]:
```
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
```

[5]:
```
df.describe()
```

[5]:
```
       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
```

[6]:
```
df.columns
```

[6]:
```
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')
```

[7]:
```
df1=df.dropna(axis=1)
df1
```

[7]:
```
          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]
```

[8]:
```
df1.columns
```

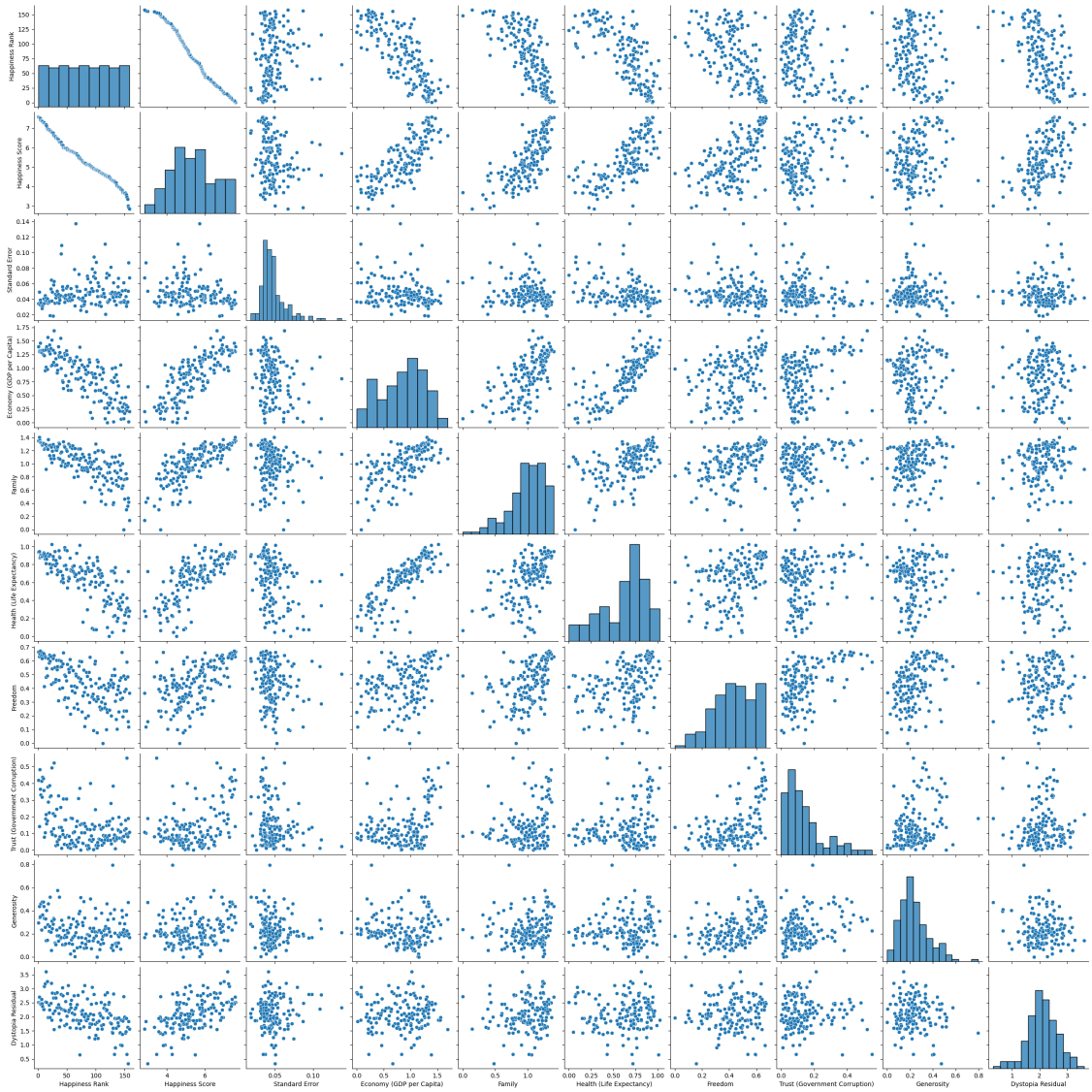
[8]:
```
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Standard Error', 'Economy (GDP per Capita)', 'Family',
       'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
       'Generosity', 'Dystopia Residual'],
      dtype='object')
```

[9]:
```
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

[10]:
```
sns.pairplot(df1)
```

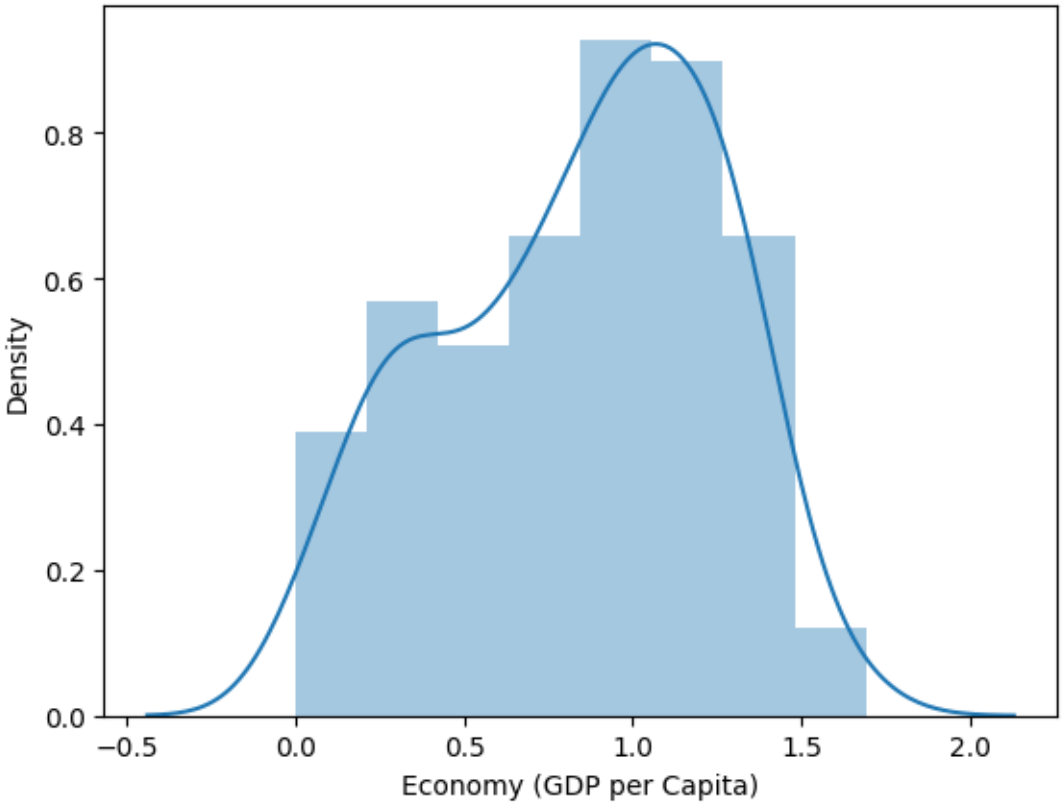
[10]:
```
<seaborn.axisgrid.PairGrid at 0x781e4b961e70>
```



[11]:
```
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)'])
```

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



[12]:
```
sns.heatmap(df1.corr())
```

[12]: <Axes: >

3 TO TRAIN THE MODEL AND MODEL BULDING

[13]:
```
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']
```

[14]:
```
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)
```

[15]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

[15]:
```
LinearRegression()
```

[16]:
```
lr.intercept_
```

[16]:
```
-0.0013960613633940966
```

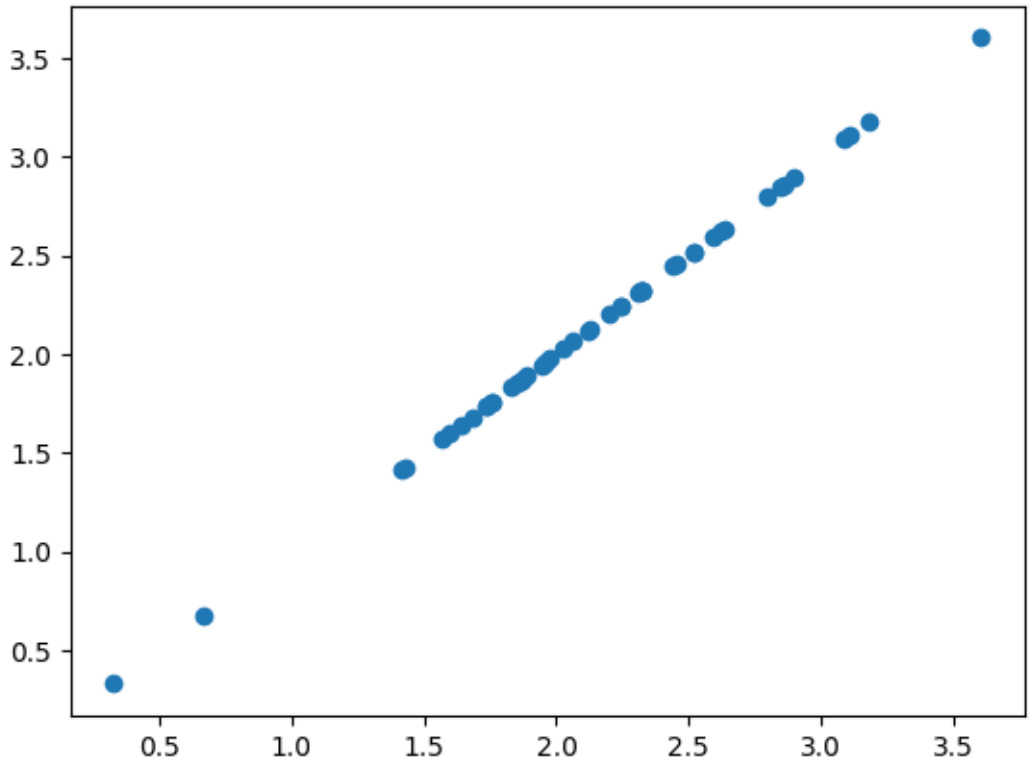
[17]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

[17]:

	Co-efficient
Happiness Rank	0.000004
Happiness Score	1.000166
Standard Error	0.001067
Economy (GDP per Capita)	-1.000135
Family	-0.999855
Health (Life Expectancy)	-0.999928
Freedom	-0.999978
Trust (Government Corruption)	-0.999785
Generosity	-0.999894

[18]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

[18]:
```
<matplotlib.collections.PathCollection at 0x781e3f505420>
```



4 ACCURACY

[19]:
```
lr.score(x_test,y_test)
```

[19]: 0.9999997950060738

[20]:
```
lr.score(x_train,y_train)
```

[20]: 0.9999997136489213

[21]:
```
from sklearn.linear_model import Ridge,Lasso
```

[22]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

[22]: Ridge(alpha=10)

[23]:
```
rr.score(x_test,y_test)
```

[23]: 0.6434824901189916

[24]:
```
rr.score(x_train,y_train)
```

[24]: 0.6547058476290912

[25]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

[25]: Lasso(alpha=10)

[26]:
```
la.score(x_test,y_test)
```

[26]: 0.052089493639094764

[27]:
```
la.score(x_train,y_train)
```

[27]: 0.05835228899089184