Import libraries

```
In [2]:  import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sns
```

```
In [3]:  data = pd.read_csv(r"C:\Users\user\Downloads\2015.csv")
         data
```

Out[3]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Fre
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.6
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.6
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.6
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.6
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.6
...	...	...	...	...	...	...	...	...	
153	Rwanda	Sub-Saharan Africa	154	3.465	0.03464	0.22208	0.77370	0.42864	0.5
154	Benin	Sub-Saharan Africa	155	3.340	0.03656	0.28665	0.35386	0.31910	0.4
155	Syria	Middle East and Northern Africa	156	3.006	0.05015	0.66320	0.47489	0.72193	0.1
156	Burundi	Sub-Saharan Africa	157	2.905	0.08658	0.01530	0.41587	0.22396	0.1
157	Togo	Sub-Saharan Africa	158	2.839	0.06727	0.20868	0.13995	0.28443	0.3

158 rows × 12 columns

In [4]:
```
data.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
```
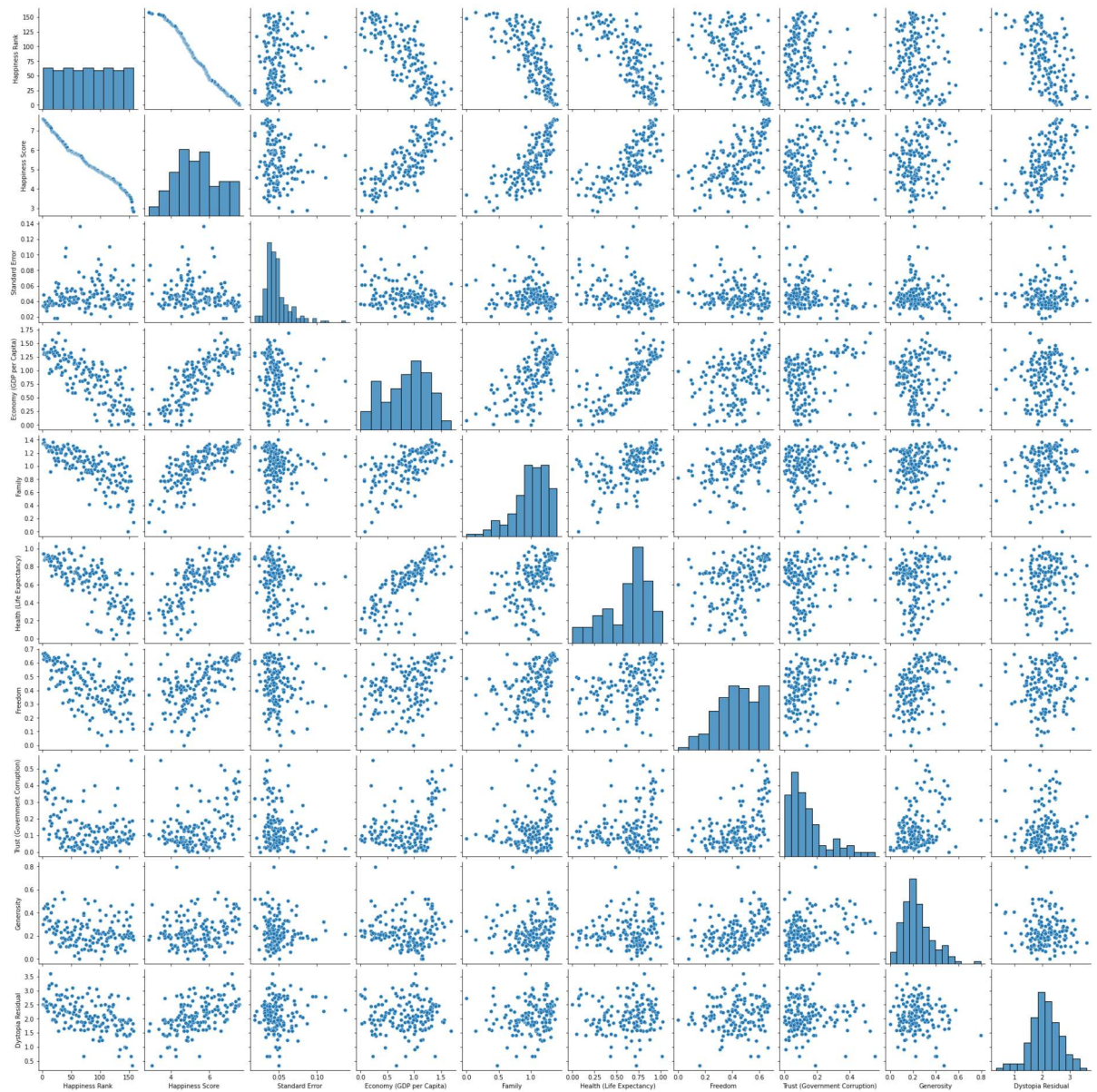
In [6]:
```
data.columns
```

Out[6]:
```
Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score',
       'Standard Error', 'Economy (GDP per Capita)', 'Family',
       'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruptio
n)',
       'Generosity', 'Dystopia Residual'],
      dtype='object')
```
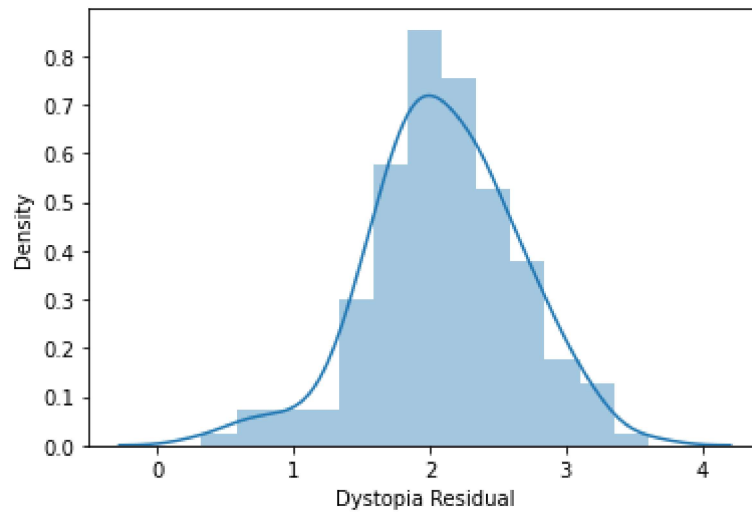
EDA and visualization

In [7]:   `sns.pairplot(data)`

Out[7]:   `<seaborn.axisgrid.PairGrid at 0x1884d737790>`

In [11]: `sns.distplot(data['Dystopia Residual'])`

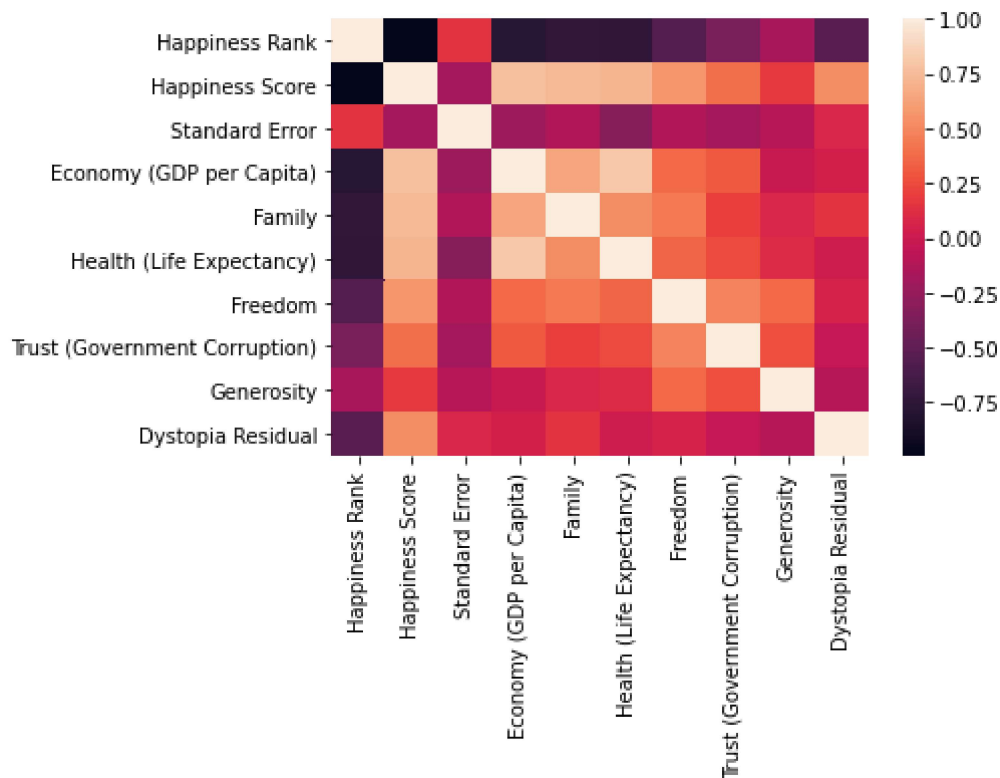Out[11]: `<AxesSubplot:xlabel='Dystopia Residual', ylabel='Density'>`

In [13]:
```
data1 = data[['Happiness Rank', 'Happiness Score',
        'Standard Error', 'Economy (GDP per Capita)', 'Family',
        'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
        'Generosity','Dystopia Residual']]
```

In [15]: `sns.heatmap(data1.corr())`

Out[15]: `<AxesSubplot:>`

model building

```
In [16]: x = data1[['Happiness Rank', 'Happiness Score',
                'Standard Error', 'Economy (GDP per Capita)', 'Family',
                'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)',
                'Generosity']]
         y = data1['Dystopia Residual']
```

```
In [17]: 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)
```

```
In [18]: from sklearn.linear_model import LinearRegression

         lr = LinearRegression()
         lr.fit(x_train,y_train)
```

```
Out[18]: LinearRegression()
```

```
In [19]: print(lr.intercept_)
```

```
         -0.0008287282561960652
```

```
In [20]: coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
         coeff
```

Out[20]:

	Co-efficient
Happiness Rank	0.000002
Happiness Score	1.000081
Standard Error	-0.000111
Economy (GDP per Capita)	-1.000286
Family	-0.999908
Health (Life Expectancy)	-0.999682
Freedom	-0.999731
Trust (Government Corruption)	-0.999853
Generosity	-0.999870

In [21]:
```python
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[21]: <matplotlib.collections.PathCollection at 0x188583049a0>
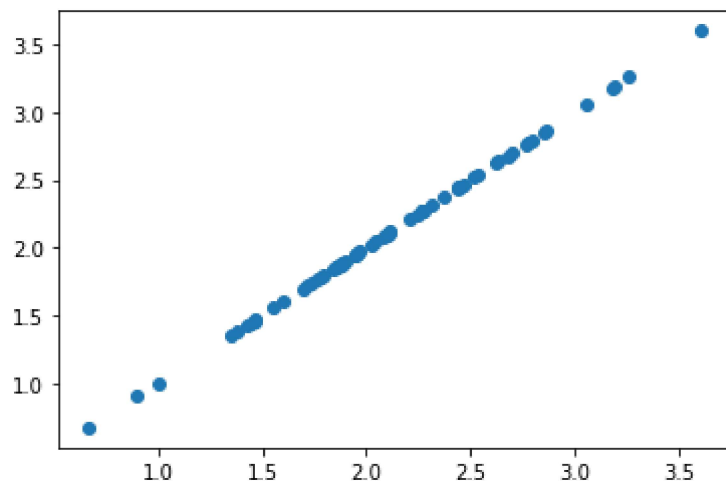
In [22]:
```python
print(lr.score(x_test,y_test))
```

0.999999732573885

In [ ]: