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

In [2]:
```
df=pd.read_csv(r"C:\Users\Admin\Downloads\2015 - 2015.csv")
df
```

Out[2]:

	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 [3]:

```
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
```

In [4]:

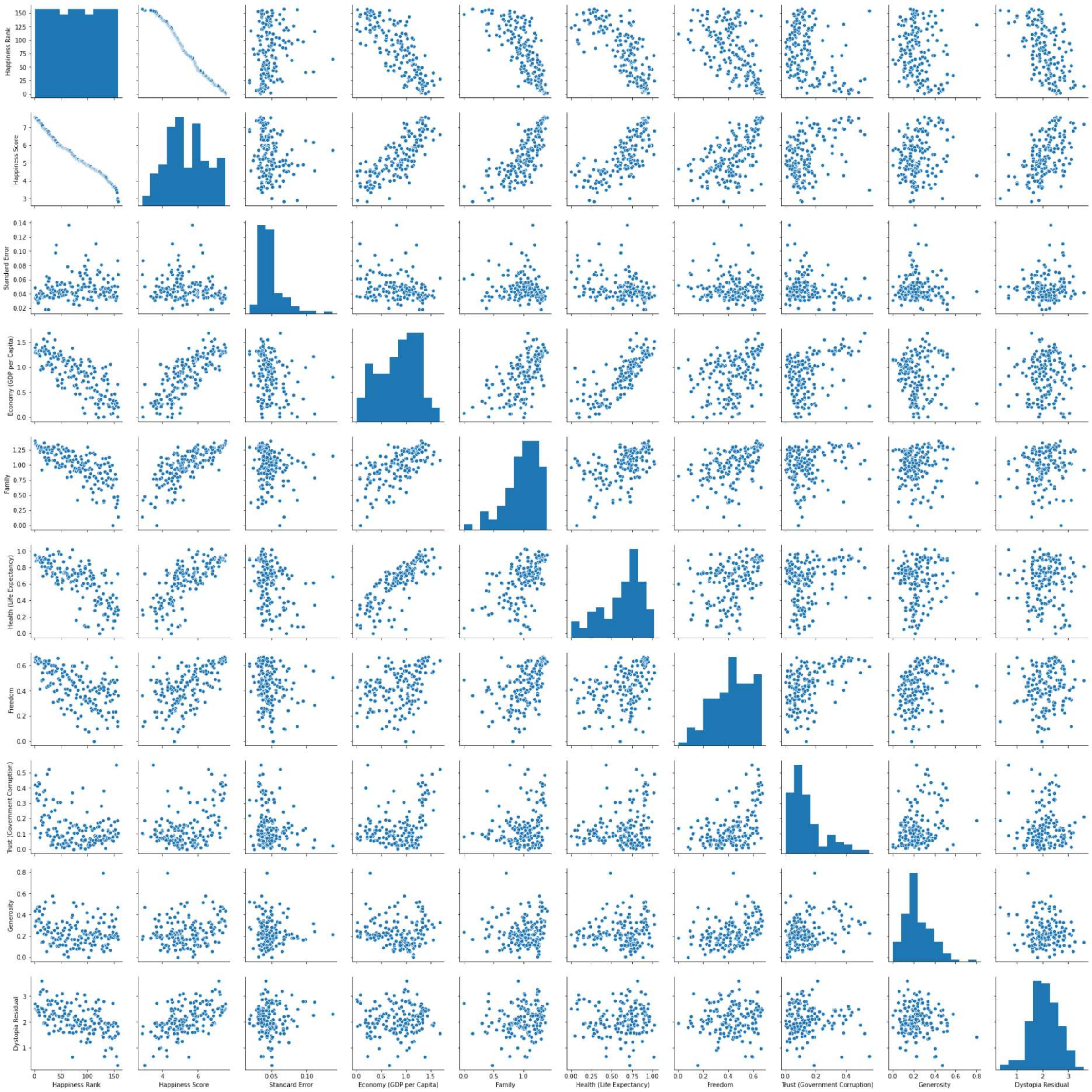
```
df.columns
```

Out[4]:

```
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')
```

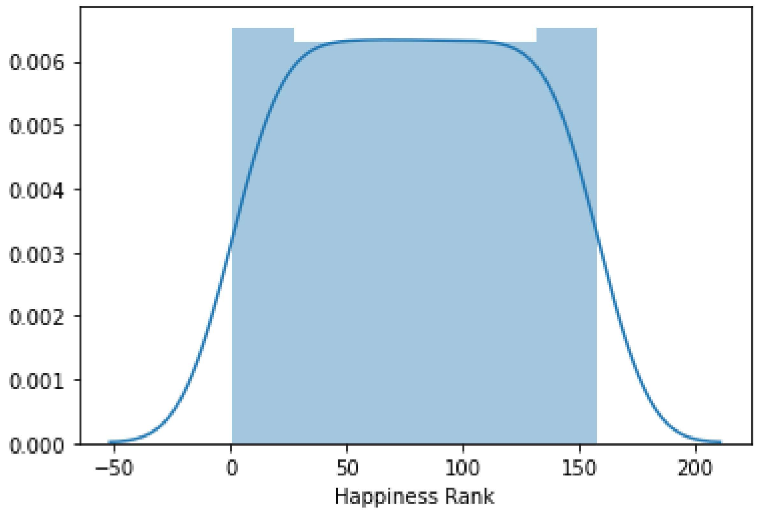
In [5]:
```
sns.pairplot(df)
```

Out[5]:
```
<seaborn.axisgrid.PairGrid at 0x246c34580d0>
```



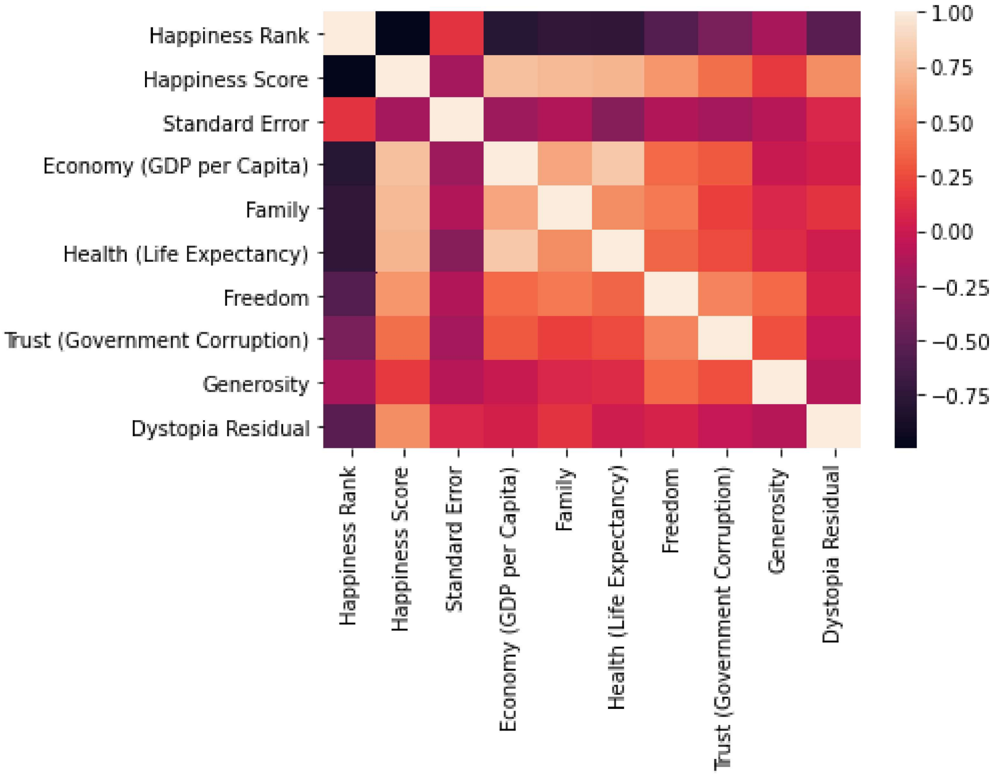
In [6]:
```
sns.distplot(df['Happiness Rank'])
```

Out[6]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x246be692af0>
```



In [7]:
```
sns.heatmap(df.corr())
```

Out[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x246c7c97340>
```



In [8]:
```
x=df[['Happiness Score', 'Family']]
y=df[['Happiness Rank']]
```

In [9]:
```
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)
```

In [10]:

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

Out[10]:

```
LinearRegression()
```

In [11]:

```
print(lr.intercept_)
```

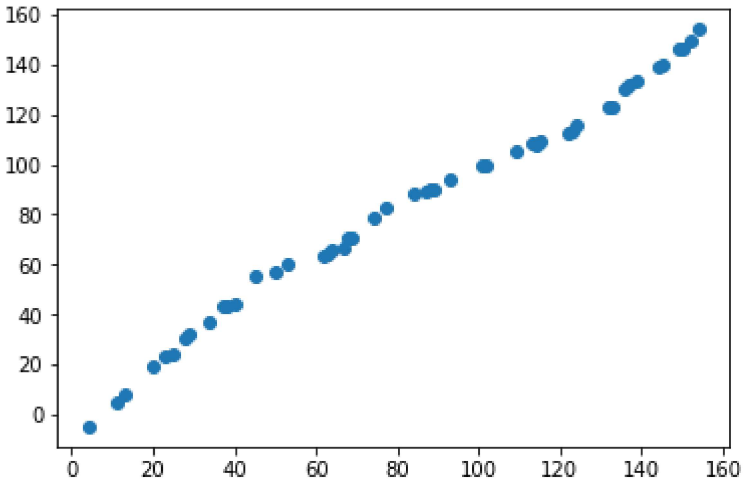
```
[289.30053438]
```

In [12]:

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

Out[12]:

```
<matplotlib.collections.PathCollection at 0x246c8771f10>
```



In [13]:

```
print(lr.score(x_test,y_test))
```

```
0.9867308381291591
```

In [14]:

```
print(lr.score(x_train,y_train))
```

```
0.9828007348418877
```

In [15]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [16]:

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

Out[16]:

```
Ridge(alpha=10)
```

In [17]:

```
rr.score(x_test,y_test)
```

Out[17]:

```
0.9771858524318255
```

In [18]:

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

Out[18]:

```
Lasso(alpha=10)
```

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

Out[19]: 0.9362269408351096
```

```
In [ ]:
```