Importing All Libraries

In [34]:
```
import numpy as nr
import pandas as pq
import matplotlib.pyplot as plr
import seaborn as sjh
```

Loading The Dataset

In [2]:
```
df1 = pq.read_csv("D:\Dataset\World_Happiness 1.csv")
df1
```

Out[2]:

	Country name	Regional indicator	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dys res
0	Finland	Western Europe	7.741	7.815	7.667	1.844	1.572	0.695	0.859	0.142	0.546	
1	Denmark	Western Europe	7.583	7.665	7.500	1.908	1.520	0.699	0.823	0.204	0.548	
2	Iceland	Western Europe	7.525	7.618	7.433	1.881	1.617	0.718	0.819	0.258	0.182	
3	Sweden	Western Europe	7.344	7.422	7.267	1.878	1.501	0.724	0.838	0.221	0.524	
4	Israel	Middle East and North Africa	7.341	7.405	7.277	1.803	1.513	0.740	0.641	0.153	0.193	
...	...	...	...	...	...	...	...	...	...	...	...	
138	Congo (Kinshasa)	Sub-Saharan Africa	3.295	3.462	3.128	0.534	0.665	0.262	0.473	0.189	0.072	
139	Sierra Leone	Sub-Saharan Africa	3.245	3.366	3.124	0.654	0.566	0.253	0.469	0.181	0.053	
140	Lesotho	Sub-Saharan Africa	3.186	3.469	2.904	0.771	0.851	0.000	0.523	0.082	0.085	
141	Lebanon	Middle East and North Africa	2.707	2.797	2.616	1.377	0.577	0.556	0.173	0.068	0.029	-
142	Afghanistan	South Asia	1.721	1.775	1.667	0.628	0.000	0.242	0.000	0.091	0.088	

143 rows × 12 columns

Data Cleaning Using Pandas

In [3]:
```
df1.isnull().sum()
```

Out[3]:
```
Country name                      0
Regional indicator                0
Ladder score                      0
upperwhisker                      0
lowerwhisker                      0
Log GDP per capita                0
Social support                    0
Healthy life expectancy           0
Freedom to make life choices      0
Generosity                        0
Perceptions of corruption         0
Dystopia + residual               0
dtype: int64
```

In [4]:
```
df1.head(19)
```

Out[4]:

	Country name	Regional indicator	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dys res
0	Finland	Western Europe	7.741	7.815	7.667	1.844	1.572	0.695	0.859	0.142	0.546	
1	Denmark	Western Europe	7.583	7.665	7.500	1.908	1.520	0.699	0.823	0.204	0.548	
2	Iceland	Western Europe	7.525	7.618	7.433	1.881	1.617	0.718	0.819	0.258	0.182	
3	Sweden	Western Europe	7.344	7.422	7.267	1.878	1.501	0.724	0.838	0.221	0.524	
4	Israel	Middle East and North Africa	7.341	7.405	7.277	1.803	1.513	0.740	0.641	0.153	0.193	
5	Netherlands	Western Europe	7.319	7.383	7.256	1.901	1.462	0.706	0.725	0.247	0.372	
6	Norway	Western Europe	7.302	7.389	7.215	1.952	1.517	0.704	0.835	0.224	0.484	
7	Luxembourg	Western Europe	7.122	7.213	7.031	2.141	1.355	0.708	0.801	0.146	0.432	
8	Switzerland	Western Europe	7.060	7.147	6.973	1.970	1.425	0.747	0.759	0.173	0.498	
9	Australia	North America and ANZ	7.057	7.141	6.973	1.854	1.461	0.692	0.756	0.225	0.323	
10	New Zealand	North America and ANZ	7.029	7.105	6.954	1.810	1.527	0.673	0.746	0.226	0.480	
11	Costa Rica	Latin America and Caribbean	6.955	7.051	6.860	1.561	1.373	0.661	0.797	0.109	0.123	
12	Kuwait	Middle East and North Africa	6.951	7.060	6.843	1.845	1.364	0.661	0.827	0.200	0.172	
13	Austria	Western Europe	6.905	6.986	6.824	1.885	1.336	0.696	0.703	0.214	0.305	
14	Canada	North America and ANZ	6.900	6.984	6.815	1.840	1.459	0.701	0.730	0.230	0.368	
15	Belgium	Western Europe	6.894	6.961	6.827	1.868	1.440	0.690	0.729	0.170	0.311	
16	Ireland	Western Europe	6.838	6.927	6.749	2.129	1.390	0.700	0.758	0.205	0.418	
17	Czechia	Central and Eastern Europe	6.822	6.903	6.741	1.783	1.511	0.638	0.787	0.177	0.068	
18	Lithuania	Central and Eastern Europe	6.818	6.896	6.739	1.766	1.454	0.598	0.533	0.044	0.116	

In [5]:
```
df1.tail(16)
```

Out[5]:

	Country name	Regional indicator	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dys res
127	Sri Lanka	South Asia	3.898	4.018	3.778	1.361	1.179	0.586	0.583	0.144	0.031	
128	Bangladesh	South Asia	3.886	3.994	3.778	1.122	0.249	0.513	0.775	0.140	0.167	
129	Ethiopia	Sub-Saharan Africa	3.861	3.997	3.725	0.792	0.915	0.420	0.441	0.270	0.101	
130	Tanzania	Sub-Saharan Africa	3.781	3.917	3.644	0.820	0.706	0.380	0.709	0.191	0.257	
131	Comoros	Sub-Saharan Africa	3.566	3.754	3.378	0.896	0.328	0.370	0.172	0.128	0.160	
132	Yemen	Middle East and North Africa	3.561	3.714	3.408	0.671	1.281	0.293	0.362	0.080	0.113	
133	Zambia	Sub-Saharan Africa	3.502	3.636	3.368	0.899	0.809	0.264	0.727	0.168	0.109	
134	Eswatini	Sub-Saharan Africa	3.502	3.673	3.331	1.255	0.925	0.176	0.284	0.059	0.116	
135	Malawi	Sub-Saharan Africa	3.421	3.561	3.281	0.617	0.410	0.349	0.571	0.135	0.136	
136	Botswana	Sub-Saharan Africa	3.383	3.558	3.209	1.445	0.969	0.241	0.567	0.014	0.082	
137	Zimbabwe	Sub-Saharan Africa	3.341	3.457	3.226	0.748	0.850	0.232	0.487	0.096	0.131	
138	Congo (Kinshasa)	Sub-Saharan Africa	3.295	3.462	3.128	0.534	0.665	0.262	0.473	0.189	0.072	
139	Sierra Leone	Sub-Saharan Africa	3.245	3.366	3.124	0.654	0.566	0.253	0.469	0.181	0.053	
140	Lesotho	Sub-Saharan Africa	3.186	3.469	2.904	0.771	0.851	0.000	0.523	0.082	0.085	
141	Lebanon	Middle East and North Africa	2.707	2.797	2.616	1.377	0.577	0.556	0.173	0.068	0.029	-
142	Afghanistan	South Asia	1.721	1.775	1.667	0.628	0.000	0.242	0.000	0.091	0.088	

In [6]:
```
df1.describe()
```

Out[6]:

	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dystopia + residual
count	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000
mean	5.527580	5.641175	5.413972	1.376306	1.130192	0.516643	0.680657	0.156476	0.154455	1.564902
std	1.170717	1.155008	1.187133	0.430993	0.337642	0.166079	0.756869	0.109612	0.125393	0.545784
min	1.721000	1.775000	1.667000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.073000
25%	4.726000	4.845500	4.606000	1.077500	0.919500	0.386000	0.526000	0.092500	0.068500	1.300500
50%	5.785000	5.895000	5.674000	1.430000	1.236000	0.549000	0.641000	0.138000	0.121000	1.640000
75%	6.416000	6.507500	6.319000	1.745000	1.385500	0.644000	0.741000	0.200000	0.196000	1.879500
max	7.741000	7.815000	7.667000	2.141000	1.617000	0.857000	9.445000	0.899000	0.575000	2.998000

In [7]:
```
df1.nunique()
```

Out[7]:
```
Country name                    143
Regional indicator               10
Ladder score                    140
upperwhisker                    140
lowerwhisker                    136
Log GDP per capita              137
Social support                  127
Healthy life expectancy         122
Freedom to make life choices    125
Generosity                      112
Perceptions of corruption       114
Dystopia + residual             136
dtype: int64
```

In [9]:
```
df1.shape
```

Out[9]:
```
(143, 12)
```

In [10]:
```
df1.columns
```

Out[10]:
```
Index(['Country name', 'Regional indicator', 'Ladder score', 'upperwhisker',
       'lowerwhisker', 'Log GDP per capita', 'Social support',
       'Healthy life expectancy', 'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption', 'Dystopia + residual'],
      dtype='object')
```

In [11]:
```
df1.loc[139,"Generosity"] = 0.9421
df1
```

Out[11]:

	Country name	Regional indicator	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dys res
0	Finland	Western Europe	7.741	7.815	7.667	1.844	1.572	0.695	0.859	0.1420	0.546	
1	Denmark	Western Europe	7.583	7.665	7.500	1.908	1.520	0.699	0.823	0.2040	0.548	
2	Iceland	Western Europe	7.525	7.618	7.433	1.881	1.617	0.718	0.819	0.2580	0.182	
3	Sweden	Western Europe	7.344	7.422	7.267	1.878	1.501	0.724	0.838	0.2210	0.524	
4	Israel	Middle East and North Africa	7.341	7.405	7.277	1.803	1.513	0.740	0.641	0.1530	0.193	
...	...	...	...	...	...	...	...	...	...	...	...	
138	Congo (Kinshasa)	Sub-Saharan Africa	3.295	3.462	3.128	0.534	0.665	0.262	0.473	0.1890	0.072	
139	Sierra Leone	Sub-Saharan Africa	3.245	3.366	3.124	0.654	0.566	0.253	0.469	0.9421	0.053	
140	Lesotho	Sub-Saharan Africa	3.186	3.469	2.904	0.771	0.851	0.000	0.523	0.0820	0.085	
141	Lebanon	Middle East and North Africa	2.707	2.797	2.616	1.377	0.577	0.556	0.173	0.0680	0.029	-
142	Afghanistan	South Asia	1.721	1.775	1.667	0.628	0.000	0.242	0.000	0.0910	0.088	

143 rows × 12 columns

In [12]:
```
df1.index
```

Out[12]:
```
RangeIndex(start=0, stop=143, step=1)
```

In [13]:
```
df1.isnull()
```

Out[13]:

	Country name	Regional indicator	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption	Dystop residu
0	False	False	False	False	False	False	False	False	False	False	False	Fal
1	False	False	False	False	False	False	False	False	False	False	False	Fal
2	False	False	False	False	False	False	False	False	False	False	False	Fal
3	False	False	False	False	False	False	False	False	False	False	False	Fal
4	False	False	False	False	False	False	False	False	False	False	False	Fal
...	...	...	...	...	...	...	...	...	...	...	...	
138	False	False	False	False	False	False	False	False	False	False	False	Fal
139	False	False	False	False	False	False	False	False	False	False	False	Fal
140	False	False	False	False	False	False	False	False	False	False	False	Fal
141	False	False	False	False	False	False	False	False	False	False	False	Fal
142	False	False	False	False	False	False	False	False	False	False	False	Fal

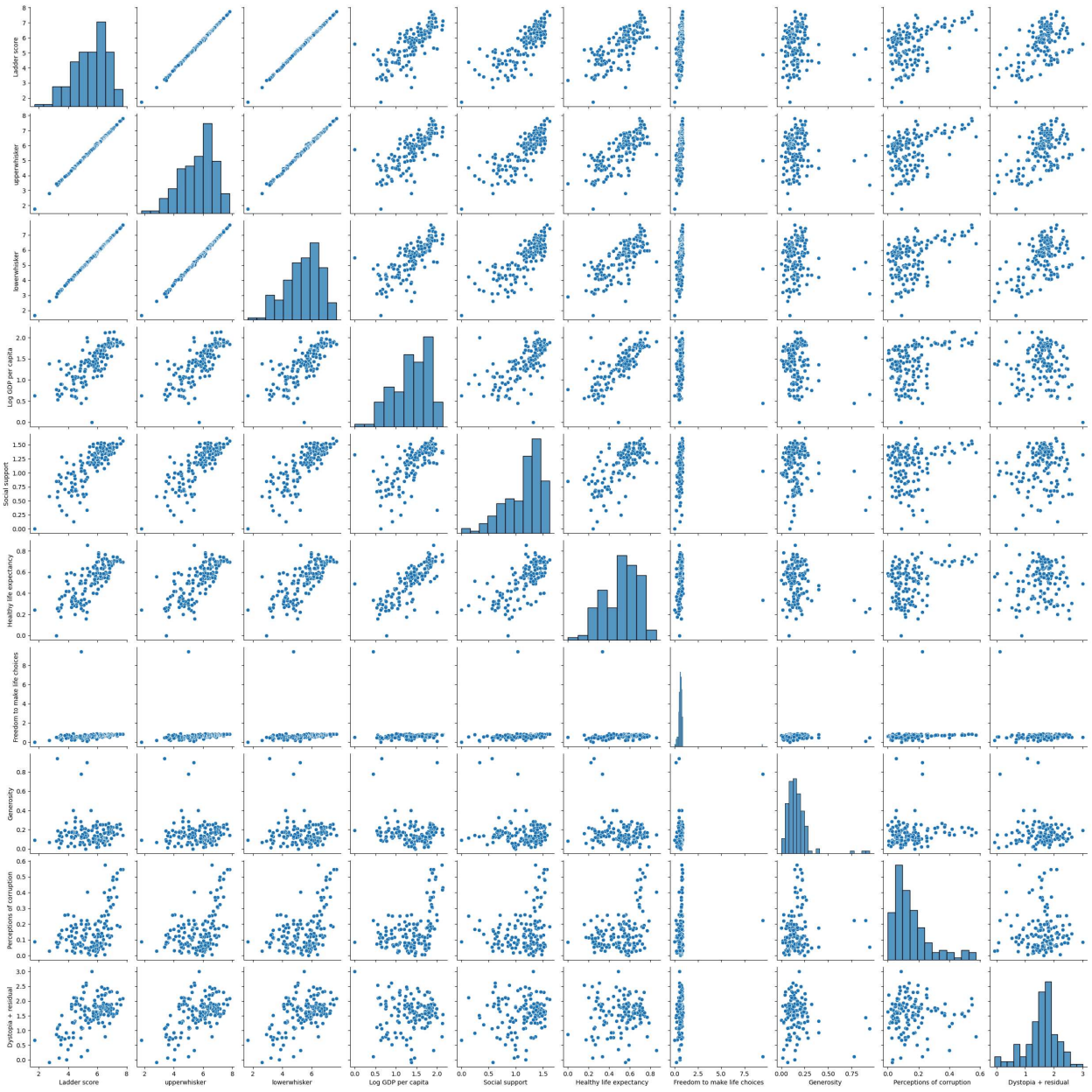
143 rows × 12 columns

Data Visualization Using Seaborn

In [14]:
```
sjh.pairplot(df1)
```

```
C:\Users\SAI\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has chang
ed to tight
  self._figure.tight_layout(*args, **kwargs)
```

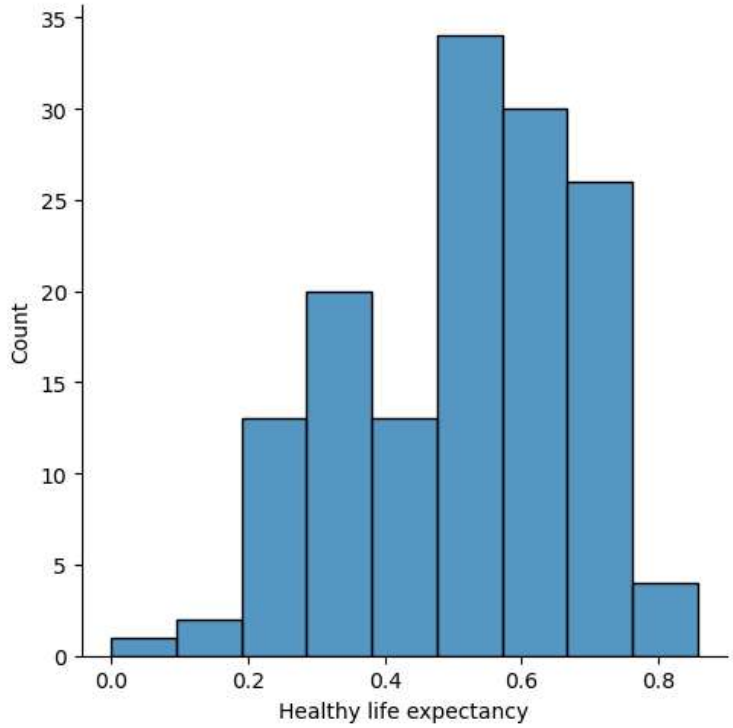
Out[14]:
```
<seaborn.axisgrid.PairGrid at 0x22bf9146390>
```



In [15]:
```
sjh.displot(df1["Healthy life expectancy"])
```

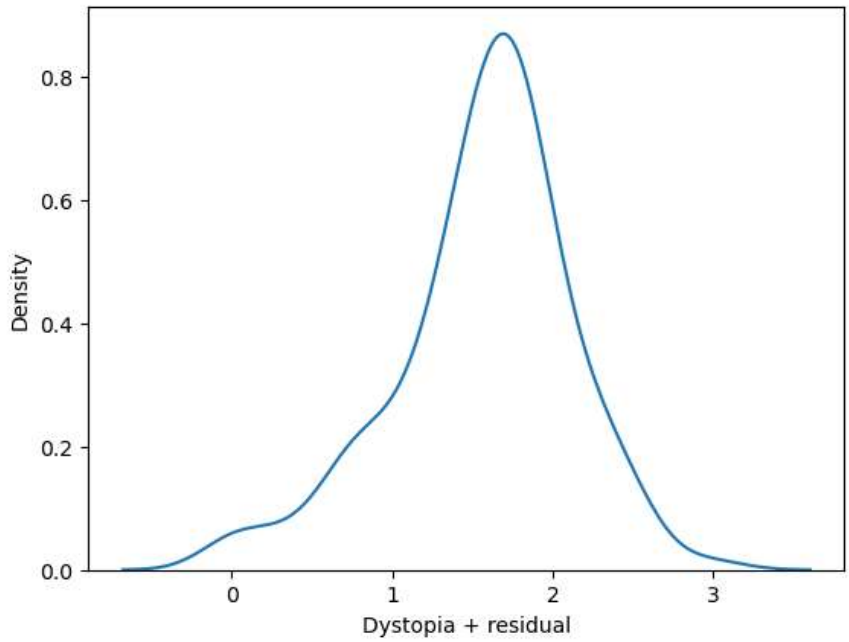
```
C:\Users\SAI\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has chang
ed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[15]:
```
<seaborn.axisgrid.FacetGrid at 0x22bff39efd0>
```



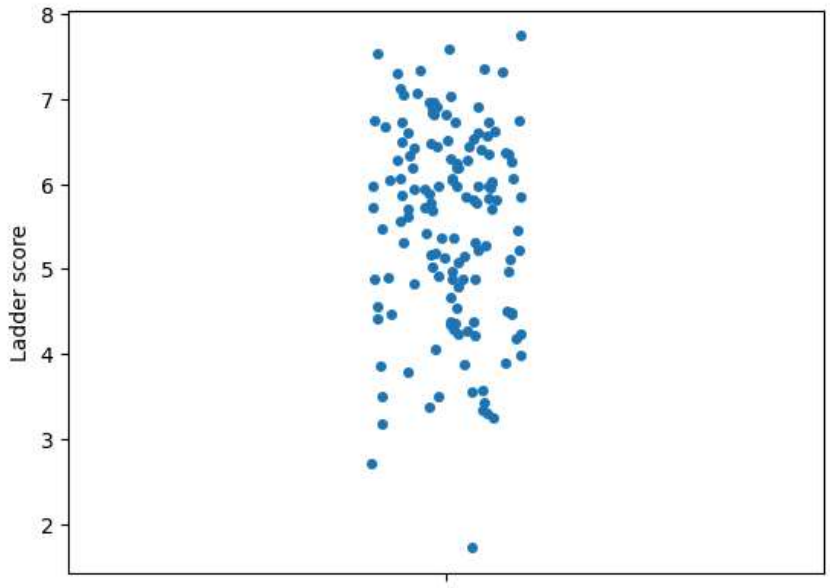
In [16]:
```
sjh.kdeplot(df1["Dystopia + residual"])
```

Out[16]:
```
<Axes: xlabel='Dystopia + residual', ylabel='Density'>
```



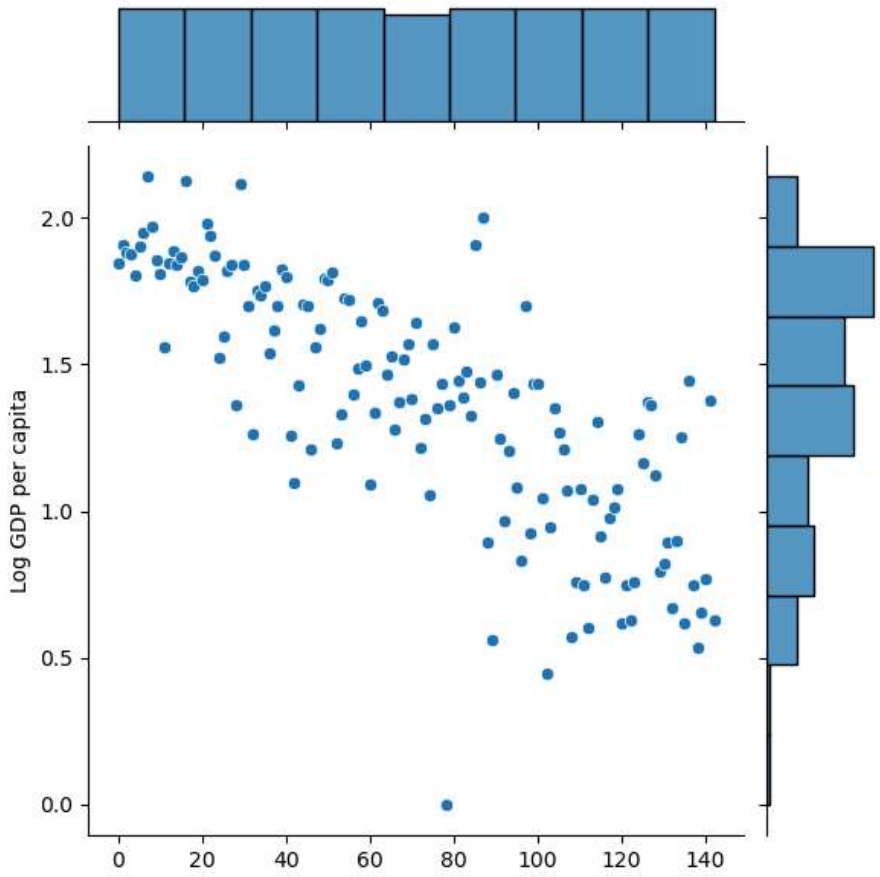
In [17]:
```
sjh.stripplot(df1["Ladder score"])
```

Out[17]: <Axes: ylabel='Ladder score'>



In [19]:
```
sjh.jointplot(df1["Log GDP per capita"])
```

Out[19]: <seaborn.axisgrid.JointGrid at 0x22b8324b910>



Split The Data into X & Y

In [20]:
```
X = df1[["Ladder score","upperwhisker","lowerwhisker","Log GDP per capita","Social support"]]
Y = df1[["Healthy life expectancy","Freedom to make life choices","Generosity","Perceptions of corruption"]]
```

In [21]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.1,random_state=75)
```

In [22]:
```
X_train
```

Out[22]:

	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support
90	5.185	5.283	5.087	1.467	0.990
63	5.934	6.027	5.840	1.684	1.276
137	3.341	3.457	3.226	0.748	0.850
66	5.842	6.059	5.624	1.280	1.324
53	6.043	6.138	5.948	1.331	1.267
...	...	...	...	...	...
34	6.442	6.513	6.370	1.738	1.417
26	6.609	6.685	6.533	1.818	1.348
136	3.383	3.558	3.209	1.445	0.969
19	6.749	6.833	6.665	1.822	1.326
112	4.471	4.630	4.313	0.603	0.805

128 rows × 5 columns

In [23]:
```
X_test
```

Out[23]:

	Ladder score	upperwhisker	lowerwhisker	Log GDP per capita	Social support
119	4.289	4.396	4.182	1.077	0.747
6	7.302	7.389	7.215	1.952	1.517
27	6.594	6.707	6.480	1.842	1.361
85	5.316	5.403	5.229	1.909	1.184
94	5.106	5.243	4.969	1.403	1.038
2	7.525	7.618	7.433	1.881	1.617
67	5.841	5.946	5.736	1.371	1.180
103	4.874	4.996	4.753	0.943	0.856
40	6.324	6.436	6.211	1.800	1.328
80	5.463	5.569	5.357	1.629	1.469
126	3.977	4.066	3.887	1.370	0.996
104	4.873	4.988	4.758	1.350	1.315
28	6.561	6.667	6.455	1.364	1.277
32	6.469	6.599	6.338	1.265	1.080
35	6.421	6.502	6.339	1.766	1.471

In [24]:
```
Y_train
```

Out[24]:

	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
90	0.524	0.680	0.000	0.174
63	0.696	0.337	0.018	0.093
137	0.232	0.487	0.096	0.131
66	0.567	0.647	0.089	0.028
53	0.539	0.843	0.094	0.160
...	...	...	...	...
34	0.639	0.600	0.081	0.175
26	0.727	0.650	0.112	0.281
136	0.241	0.567	0.014	0.082
19	0.672	0.713	0.267	0.351
112	0.199	0.411	0.218	0.113

128 rows × 4 columns

In [25]:
```
Y_test
```

Out[25]:

	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
119	0.360	0.623	0.183	0.028
6	0.704	0.835	0.224	0.484
27	0.511	0.787	0.114	0.188
85	0.857	0.485	0.147	0.402
94	0.344	0.516	0.045	0.100
2	0.718	0.819	0.258	0.182
67	0.662	0.615	0.078	0.029
103	0.288	0.521	0.126	0.060
40	0.720	0.513	0.112	0.074
80	0.567	0.620	0.083	0.006
126	0.488	0.490	0.025	0.259
104	0.513	0.631	0.285	0.025
28	0.599	0.739	0.254	0.073
32	0.549	0.816	0.083	0.253
35	0.729	0.619	0.119	0.177

Checking The Data Types

In [26]:
```
X_train.dtypes
```

Out[26]:
```
Ladder score          float64
upperwhisker          float64
lowerwhisker          float64
Log GDP per capita    float64
Social support        float64
dtype: object
```

In [27]:
```
Y_train.dtypes
```

Out[27]:
```
Healthy life expectancy         float64
Freedom to make life choices    float64
Generosity                      float64
Perceptions of corruption       float64
dtype: object
```

Importing The Random Forest

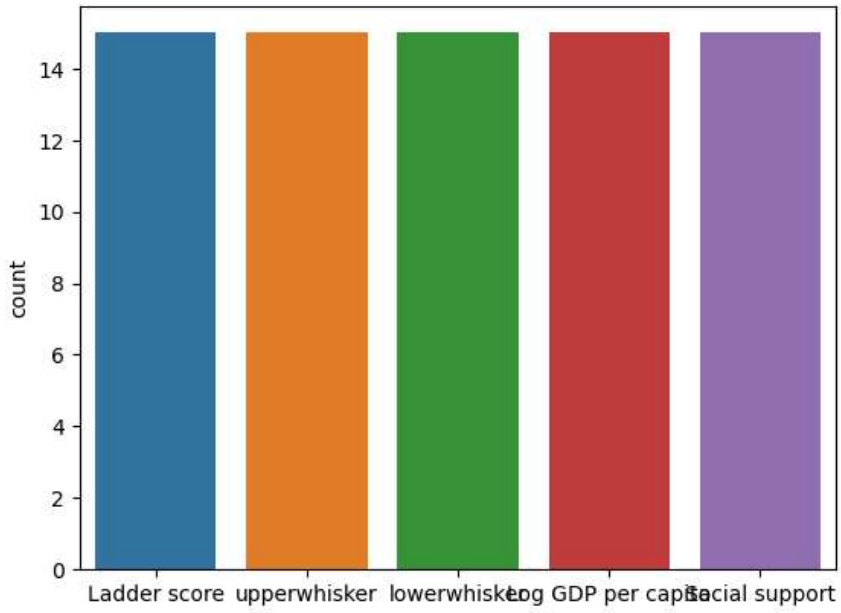
In [38]:
```
from sklearn.ensemble import RandomForestRegressor
rf1=RandomForestRegressor(random_state=75)
rf1.fit(X_train,Y_train)
```

Out[38]:
```
▸ RandomForestRegressor
```

Predictions From Our Model

In [41]:
```
pred1 = rf1.predict(X_test)
sjh.countplot(X_test)
```

Out[41]:
```
<Axes: ylabel='count'>
```



Evaluation Metrics

In [54]:
```
from sklearn import metrics
print('MAE :',metrics.mean_absolute_error(Y_test,pred1))
print('MSE :',metrics.mean_squared_error(Y_test,pred1))
print('RMSE :',nr.sqrt(metrics.mean_squared_error(Y_test,pred1)))
```

```
MAE : 0.08505433333333337
MSE : 0.01421263865666667
RMSE : 0.11921677170879386
```

In [ ]: