# РК1

# Студент:Чжан Аньци

# Группа:ИУ5И-22М

# номер по списку группы(Вариант):17

# Задач:

1.для произвольной колонки данных построить гистограмму 2.Задача №17. Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием преобразования Йео-Джонсона (Yeo-Johnson transformation). 3.Задача №37. Для набора данных проведите процедуру отбора признаков (feature selection). Используйте класс SelectPercentile для 5% лучших признаков, и метод, основанный на взаимной информации.

# Подключим все необходимые библиотеки:

In [1]:

```
import pandas as pd
from pandas import DataFrame
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
from matplotlib import rcParams
import plotly.graph_objects as go
import plotly.express as px
from plotly.colors import n_colors
import numpy as np
import datetime as dt
import plotly.express as px
import seaborn as sns
import scipy.stats as stats
%matplotlib inline
```

# Загрузим непосредственно данные:

In [2]:

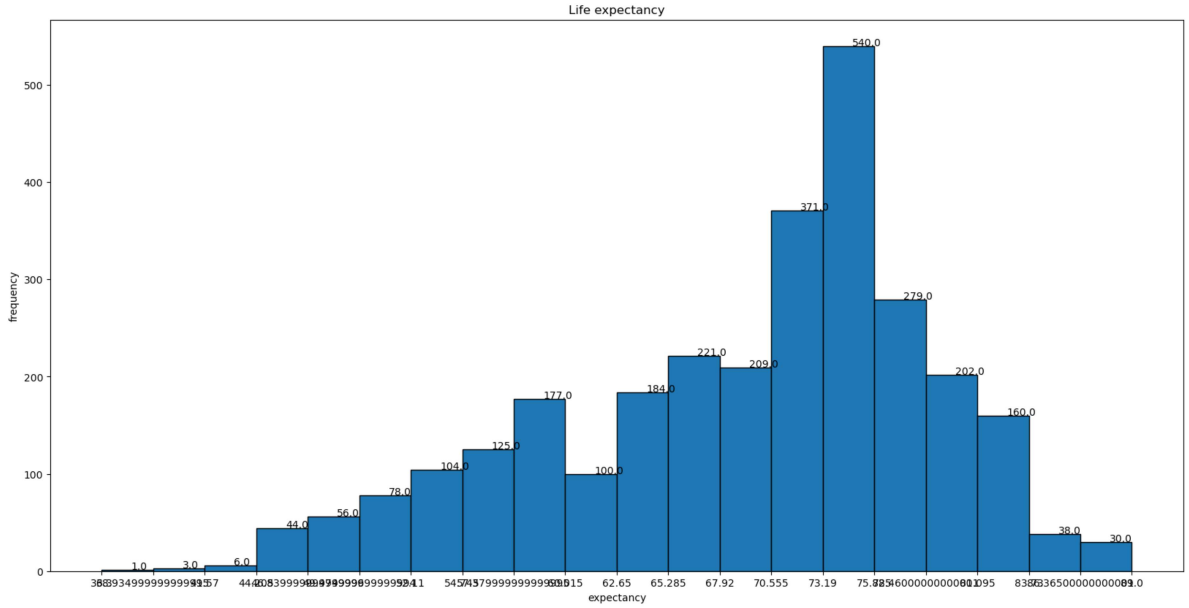
```
data = pd.read_csv("Life Expectancy Data.csv")
data.info()
```

```
 5   infant deaths                    2938 non-null   int64
 6   Alcohol                          2744 non-null   float64
 7   percentage expenditure           2938 non-null   float64
 8   Hepatitis B                      2385 non-null   float64
 9   Measles                          2938 non-null   int64
 10   BMI                             2904 non-null   float64
 11  under-five deaths                2938 non-null   int64
 12  Polio                            2919 non-null   float64
 13  Total expenditure                2712 non-null   float64
 14  Diphtheria                       2919 non-null   float64
 15   HIV/AIDS                        2938 non-null   float64
 16  GDP                              2490 non-null   float64
 17  Population                       2286 non-null   float64
 18   thinness  1-19 years            2904 non-null   float64
 19   thinness 5-9 years             2904 non-null   float64
 20  Income composition of resources  2771 non-null   float64
 21  Schooling                        2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
```

# для произвольной колонки данных построить гистограмму

In [3]:

```
Life_expectancy= data['Life expectancy ']
plt.figure(figsize=(20, 10), dpi=100)
nums,bins,patches = plt.hist(Life_expectancy,bins=20,edgecolor='k')
plt.xticks(bins,bins)
for num,bin in zip(nums,bins):
    plt.annotate(num,xy=(bin,num),xytext=(bin+1.5,num+0.5))
plt.title("Life expectancy")
plt.xlabel("expectancy")
plt.ylabel("frequency")
plt.show()
```

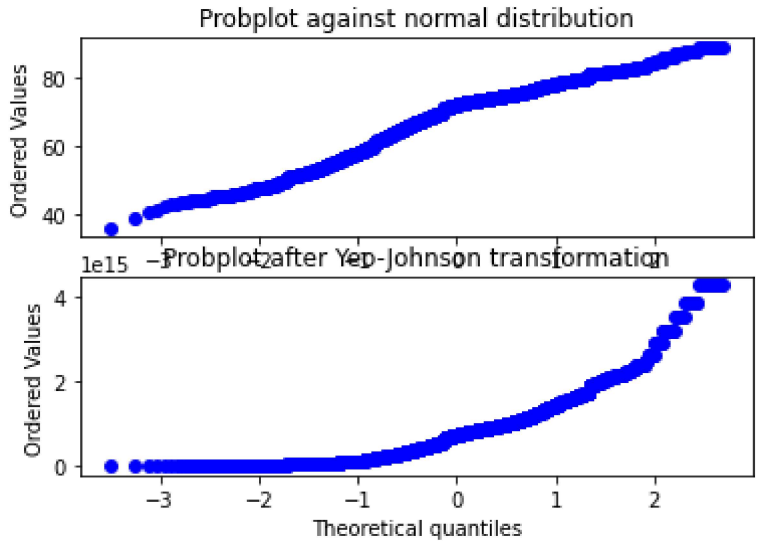



# Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием преобразования Йео-Джонсона (Yeo-Johnson transformation).

In [4]:

```
from scipy import stats
import matplotlib.pyplot as plt
import scipy

fig = plt.figure()
ax1 = fig.add_subplot(211)
x = Life_expectancy
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
ax1.set_xlabel('')
ax1.set_title('Probplot against normal distribution')
ax2 = fig.add_subplot(212)
xt, lmbda = stats.yeojohnson(x)
prob = stats.probplot(xt, dist=stats.norm, plot=ax2)
ax2.set_title('Probplot after Yeo-Johnson transformation')
plt.show()
```




# Для набора данных проведите процедуру отбора признаков (feature selection). Используйте класс SelectPercentile для 5% лучших признаков, и метод, основанный на взаимной информации.

Сначала проводится корреляционный анализ.

In [59]:

```
data = pd.read_csv("Life Expectancy Data.csv")
data1=data[:16]
data1
```

Out[59]:

| | Country | Year | Status | Life expectancy | Adult Mortality | infant deaths | Alcohol | percentage expenditure | Hepatitis B |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Afghanistan | 2015 | Developing | 65.0 | 263.0 | 62 | 0.01 | 71.279624 | 65.0 |
| 1 | Afghanistan | 2014 | Developing | 59.9 | 271.0 | 64 | 0.01 | 73.523582 | 62.0 |
| 2 | Afghanistan | 2013 | Developing | 59.9 | 268.0 | 66 | 0.01 | 73.219243 | 64.0 |
| 3 | Afghanistan | 2012 | Developing | 59.5 | 272.0 | 69 | 0.01 | 78.184215 | 67.0 |
| 4 | Afghanistan | 2011 | Developing | 59.2 | 275.0 | 71 | 0.01 | 7.097109 | 68.0 |
| 5 | Afghanistan | 2010 | Developing | 58.8 | 279.0 | 74 | 0.01 | 79.679367 | 66.0 |
| 6 | Afghanistan | 2009 | Developing | 58.6 | 281.0 | 77 | 0.01 | 56.762217 | 63.0 |
| 7 | Afghanistan | 2008 | Developing | 58.1 | 287.0 | 80 | 0.03 | 25.873925 | 64.0 |
| 8 | Afghanistan | 2007 | Developing | 57.5 | 295.0 | 82 | 0.02 | 10.910156 | 63.0 |
| 9 | Afghanistan | 2006 | Developing | 57.3 | 295.0 | 84 | 0.03 | 17.171518 | 64.0 |
| 10 | Afghanistan | 2005 | Developing | 57.3 | 291.0 | 85 | 0.02 | 1.388648 | 66.0 |
| 11 | Afghanistan | 2004 | Developing | 57.0 | 293.0 | 87 | 0.02 | 15.296066 | 67.0 |
| 12 | Afghanistan | 2003 | Developing | 56.7 | 295.0 | 87 | 0.01 | 11.089053 | 65.0 |
| 13 | Afghanistan | 2002 | Developing | 56.2 | 3.0 | 88 | 0.01 | 16.887351 | 64.0 |
| 14 | Afghanistan | 2001 | Developing | 55.3 | 316.0 | 88 | 0.01 | 10.574728 | 63.0 |
| 15 | Afghanistan | 2000 | Developing | 54.8 | 321.0 | 88 | 0.01 | 10.424960 | 62.0 |

16 rows × 22 columns

In [61]:

```
df=data1.iloc[:,3:7]
df
```

Out[61]:

| | Life expectancy | Adult Mortality | infant deaths | Alcohol |
|---|---|---|---|---|
| **0** | 65.0 | 263.0 | 62 | 0.01 |
| **1** | 59.9 | 271.0 | 64 | 0.01 |
| **2** | 59.9 | 268.0 | 66 | 0.01 |
| **3** | 59.5 | 272.0 | 69 | 0.01 |
| **4** | 59.2 | 275.0 | 71 | 0.01 |
| **5** | 58.8 | 279.0 | 74 | 0.01 |
| **6** | 58.6 | 281.0 | 77 | 0.01 |
| **7** | 58.1 | 287.0 | 80 | 0.03 |
| **8** | 57.5 | 295.0 | 82 | 0.02 |
| **9** | 57.3 | 295.0 | 84 | 0.03 |
| **10** | 57.3 | 291.0 | 85 | 0.02 |
| **11** | 57.0 | 293.0 | 87 | 0.02 |
| **12** | 56.7 | 295.0 | 87 | 0.01 |
| **13** | 56.2 | 3.0 | 88 | 0.01 |
| **14** | 55.3 | 316.0 | 88 | 0.01 |
| **15** | 54.8 | 321.0 | 88 | 0.01 |

In [67]:

```
X=df.iloc[:,0]
Y=df.iloc[:,1]
result1 = np.corrcoef(X, Y)
result1
```

Out[67]:

```
array([[1.        , 0.02675163],
       [0.02675163, 1.        ]])
```

In [69]:

```
result2 = np.corrcoef(df, rowvar=False)
result2
```

Out[69]:

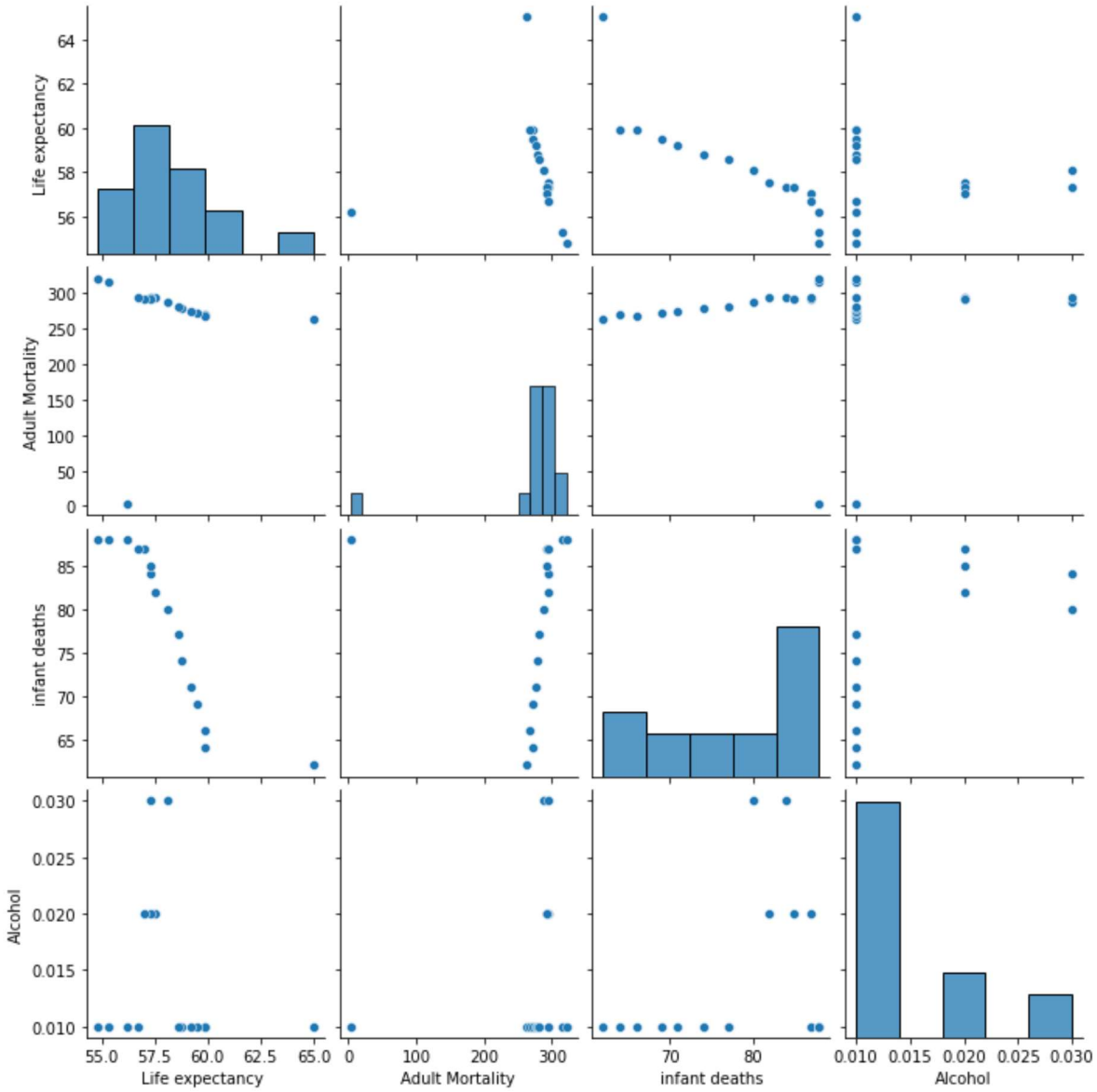
```
array([[ 1.        ,  0.02675163, -0.8887607 , -0.18310044],
       [ 0.02675163,  1.        , -0.07956915,  0.20097813],
       [-0.8887607 , -0.07956915,  1.        ,  0.33600743],
       [-0.18310044,  0.20097813,  0.33600743,  1.        ]])
```

In [71]:

```
sns.pairplot(df)
```

Out[71]:

<seaborn.axisgrid.PairGrid at 0x1b646e1a700>

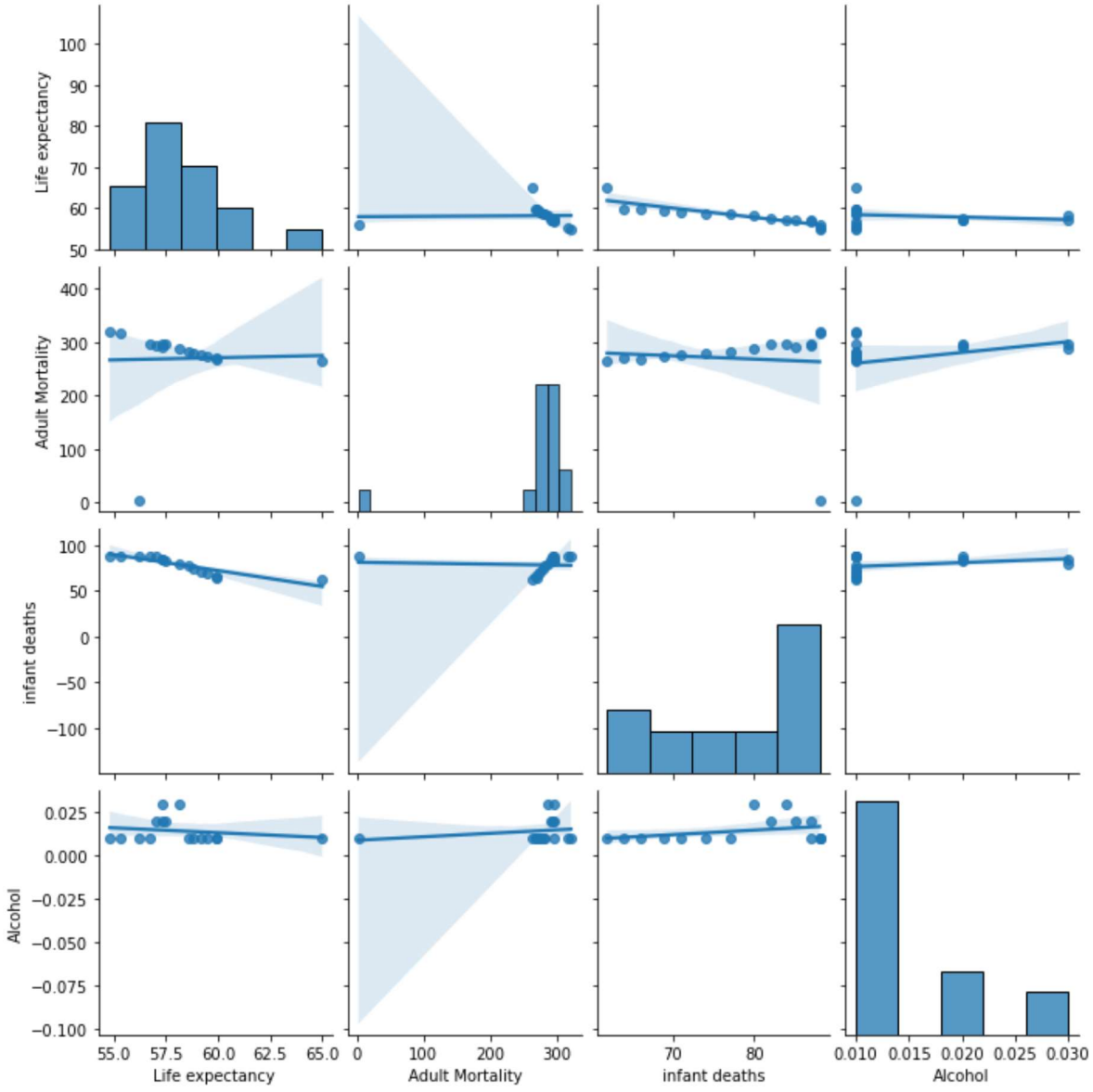


In [72]:

```
sns.pairplot(df,kind='reg',diag_kind='hist')
```

Out[72]:

<seaborn.axisgrid.PairGrid at 0x1b6476a4970>

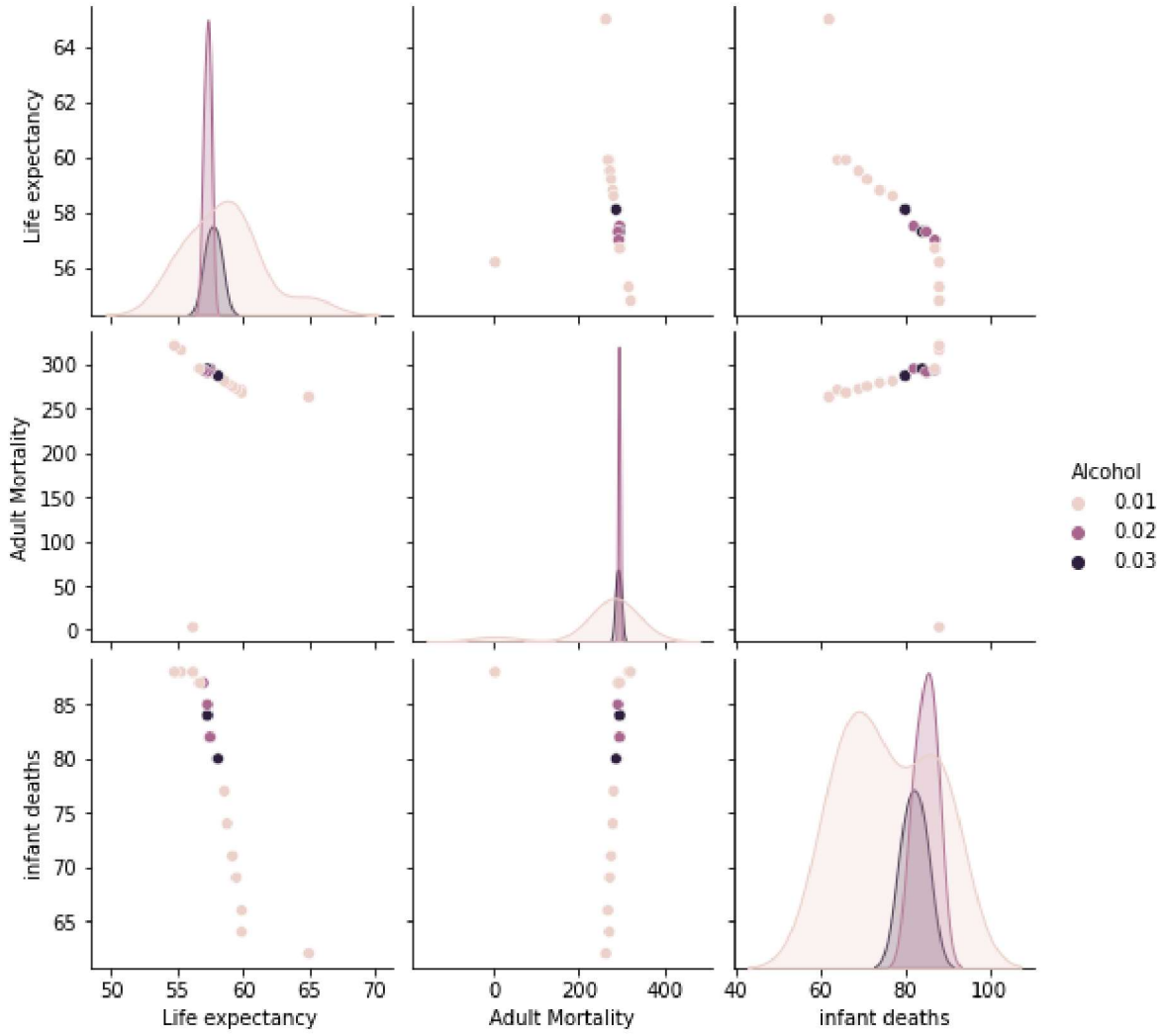


In [73]:

```
sns.pairplot(df,hue='Alcohol')
```

Out[73]:

<seaborn.axisgrid.PairGrid at 0x1b644d26fa0>

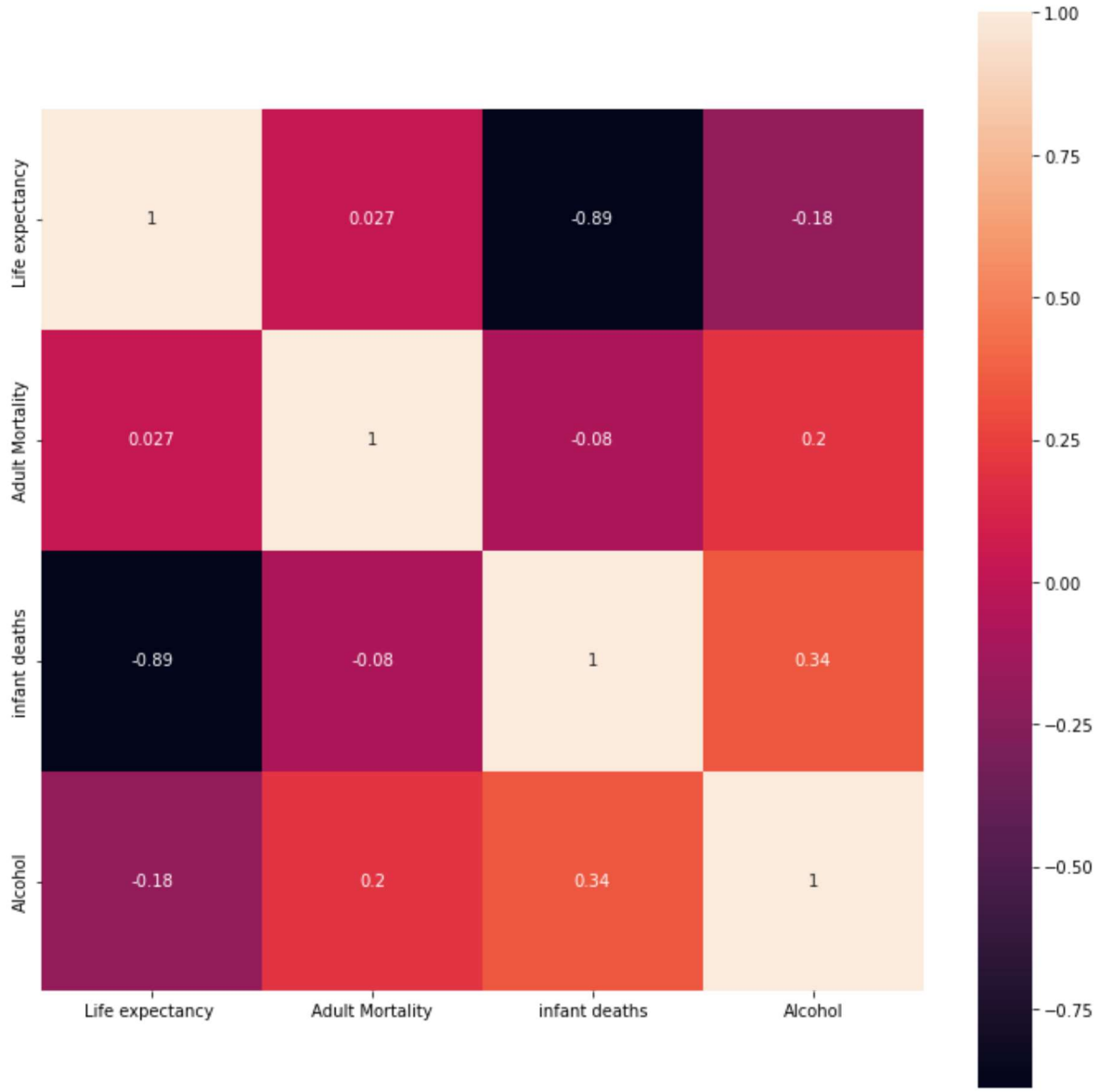


In [74]:

```
import matplotlib.pyplot as plt
figure, ax = plt.subplots(figsize=(12, 12))
sns.heatmap(df.corr(), square=True, annot=True, ax=ax)
```

Out[74]:

<AxesSubplot:>



In [77]:

```
np.percentile(df, 5, axis=None, out=None, overwrite_input=False, interpolation='linear', keepdims=
```

Out[77]:

```
array([[0.01]])
```

```
р и м е р  в ы б о р а  х а р а к т е р и с т и к (SelectPercentile)
```

In [78]:

```
from sklearn.datasets import load_digits
from sklearn.feature_selection import SelectPercentile, chi2

X, y = load_digits(return_X_y=True)
X.shape
```

Out[78]:

```
(1797, 64)
```

In [79]:

```
X_new = SelectPercentile(chi2, percentile=5).fit_transform(X, y)
X_new.shape
```

Out[79]:

```
(1797, 4)
```