Simple_Data_Analysis

January 28, 2020

[7]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import scipy

import plotly as px
import plotly.graph_objs as go

from scipy.stats import normaltest
from scipy.stats import shapiro

from IPython.display import Image
```

1 World Happiness report

[2]:
```
df = pd.read_csv('2019.csv')
```

[3]:
```
df.head(10)
```

[3]:
```
   Overall rank Country or region  Score  GDP per capita  Social support  \
0             1           Finland  7.769           1.340           1.587
1             2           Denmark  7.600           1.383           1.573
2             3            Norway  7.554           1.488           1.582
3             4           Iceland  7.494           1.380           1.624
4             5       Netherlands  7.488           1.396           1.522
5             6       Switzerland  7.480           1.452           1.526
6             7            Sweden  7.343           1.387           1.487
7             8       New Zealand  7.307           1.303           1.557
8             9            Canada  7.278           1.365           1.505
9            10           Austria  7.246           1.376           1.475

   Healthy life expectancy  Freedom to make life choices  Generosity  \
0                    0.986                         0.596       0.153
1                    0.996                         0.592       0.252
2                    1.028                         0.603       0.271
3                    1.026                         0.591       0.354
```

```
4                     0.999                        0.557        0.322
5                     1.052                        0.572        0.263
6                     1.009                        0.574        0.267
7                     1.026                        0.585        0.330
8                     1.039                        0.584        0.285
9                     1.016                        0.532        0.244

   Perceptions of corruption
0                      0.393
1                      0.410
2                      0.341
3                      0.118
4                      0.298
5                      0.343
6                      0.373
7                      0.380
8                      0.308
9                      0.226
```

[4]:
```
top10 = df.head(10)
top10 = top10.drop('Overall rank',1)
top10.describe()
```

[4]:
```
           Score  GDP per capita  Social support  Healthy life expectancy  \
count  10.000000       10.000000       10.000000                10.000000
mean    7.455900        1.387000        1.543800                 1.017700
std     0.164015        0.052113        0.048352                 0.020489
min     7.246000        1.303000        1.475000                 0.986000
25%     7.316000        1.367750        1.509250                 1.001500
50%     7.484000        1.381500        1.541500                 1.021000
75%     7.539000        1.393750        1.579750                 1.027500
max     7.769000        1.488000        1.624000                 1.052000

       Freedom to make life choices  Generosity  Perceptions of corruption
count                     10.000000   10.000000                  10.000000
mean                       0.578600    0.274100                   0.319000
std                        0.021093    0.055941                   0.088861
min                        0.532000    0.153000                   0.118000
25%                        0.572500    0.254750                   0.300500
50%                        0.584500    0.269000                   0.342000
75%                        0.591750    0.312750                   0.378250
max                        0.603000    0.354000                   0.410000
```

[5]:
```
df.columns
```

[5]:
```
Index(['Overall rank', 'Country or region', 'Score', 'GDP per capita',
       'Social support', 'Healthy life expectancy',
```

```
       'Freedom to make life choices', 'Generosity',
       'Perceptions of corruption'],
      dtype='object')
```

The rankings in World Happiness Report 2019 use data that come from the Gallup World Poll. The rankings are based on answers to the main life evaluation question asked in the poll. This is called the Cantril ladder: it asks respondents to think of a ladder, with the best possible life for them being a 10, and the worst possible life being a 0. They are then asked to rate their own current lives on that 0 to 10 scale.

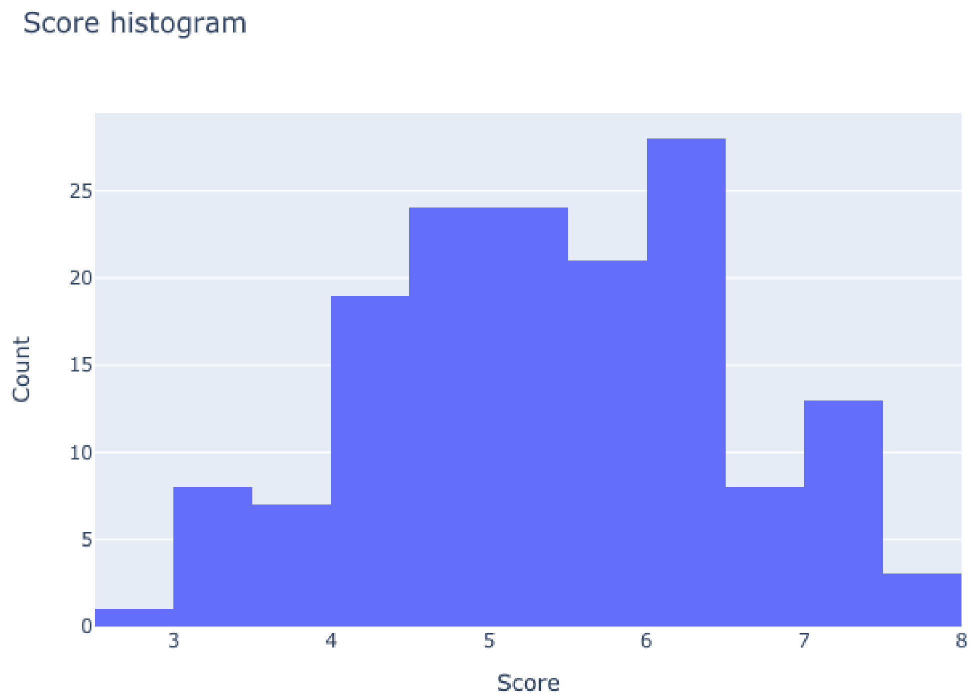
The purpose of this project is to evaluate, which of the six subbars, that is levels of GDP, life expectancy, generosity, social support, freedom, and corruption, have the most influence on the overall score. What is more, correlation between some of the subbars of the report will be computed in order examine additional dependencies between some areas of human life.

[6]:
```
data = df['Score']
data.describe()
```

[6]:
```
count    156.000000
mean       5.407096
std        1.113120
min        2.853000
25%        4.544500
50%        5.379500
75%        6.184500
max        7.769000
Name: Score, dtype: float64
```

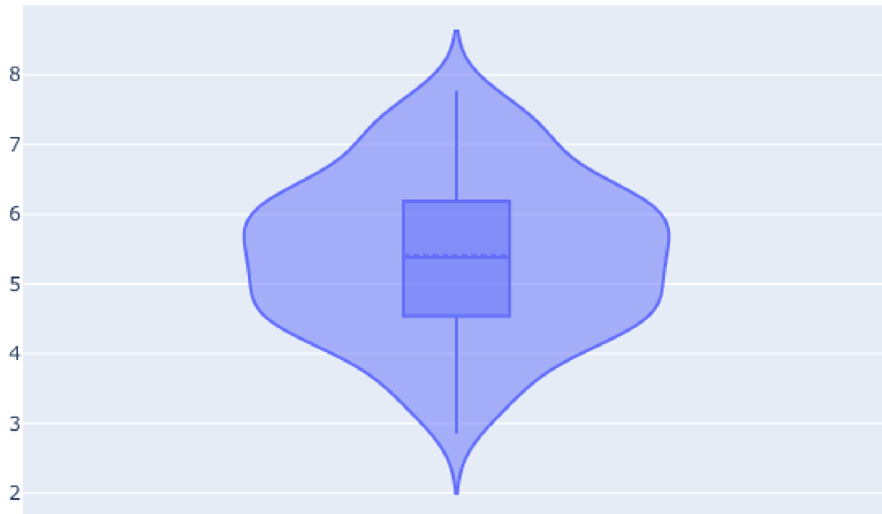
[11]:
```
plot = go.Figure(data=[
    go.Histogram(
        x=data
    )
])

plot.update_layout(
    title_text = "Score histogram",
    xaxis_title_text = 'Score',
    yaxis_title_text = 'Count'
)

#plot.show()
img_bytes = plot.to_image(format='png')
Image(img_bytes)
```

[11]:



[12]:
```
plot = go.Figure()
plot.add_trace(go.Violin(
    y = data,
    x0 ='Score - violin plot',
    box_visible = True,
    meanline_visible = True,
))

#plot.show()
img_bytes = plot.to_image(format='png')
Image(img_bytes)
```

[12]:



As we can see, the scores of countries tend to be concentrated around the average value of the score with few scores much better than average and even less much worse. Testing score for normality is highly aproppiate.

[9]:
```
alpha = 0.05

stat, p = shapiro(df['Score'])
print('Statistics=%.3f, p=%.3f' % (stat, p))

if p > alpha:
    print('Sample seems Gaussian according to Shapiro-Wilk test.')
else:
    print('Sample does not seem Gaussian according to Shapiro-Wilk test.')

stat, p = normaltest(df['Score'])
print('Statistics=%.3f, p=%.3f' % (stat, p))

if p > alpha:
    print("Sample seems Gaussian according to D'Agostino's k^2 test.")
else:
    print("Sample does not seem Gaussian according to D'Agostino's k^2 test.")
```

```
Statistics=0.987, p=0.163
Sample seems Gaussian according to Shapiro-Wilk test.
Statistics=4.465, p=0.107
Sample seems Gaussian according to D'Agostino's k^2 test.
```

According to the results of both the shapiro and D'Agostino' tests the score is normally distributed variable

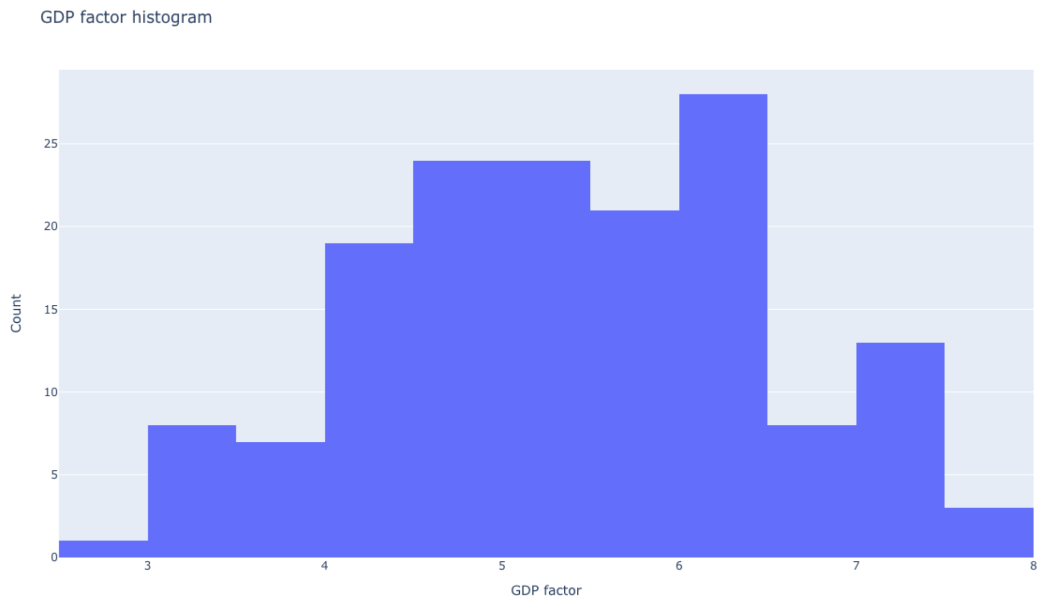
1.1 GDP per capita factor - score

[10]:
```
data = df['GDP per capita']
data.describe()
```

[10]:
```
count    156.000000
mean       0.905147
std        0.398389
min        0.000000
25%        0.602750
50%        0.960000
75%        1.232500
max        1.684000
Name: GDP per capita, dtype: float64
```

[22]:
```
plot = go.Figure(data=[
    go.Histogram(
        x=data
    )
])

plot.update_layout(
    title_text = "GDP factor histogram",
    xaxis_title_text = 'GDP factor',
    yaxis_title_text = 'Count'
)

#plot.show()
img_bytes = plot.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

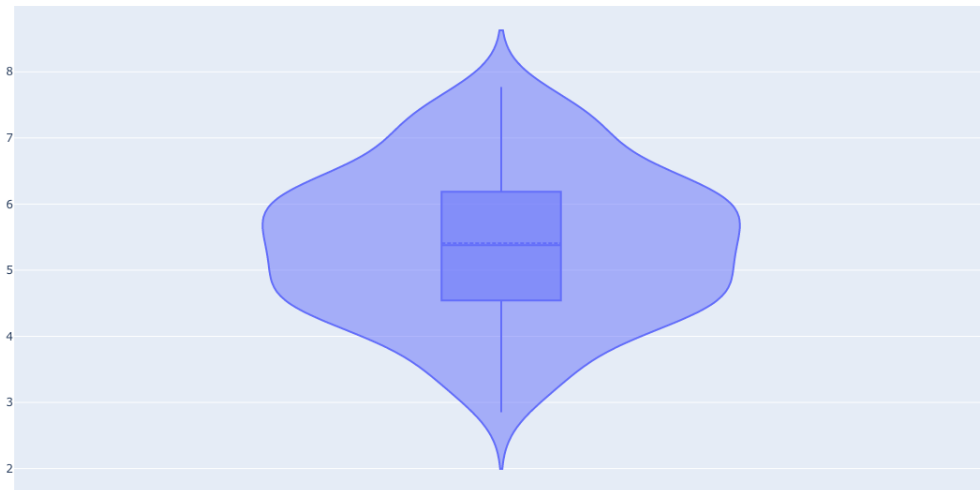
[22]:



[23]:

```
plot = go.Figure()
plot.add_trace(go.Violin(
    y = data,
    x0 ='GDP factor expectancy - violin plot',
    box_visible = True,
    meanline_visible = True
))

#plot.show()
img_bytes = plot.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[23]:



[24]:
```
x_data = df["GDP per capita"]
y_data = df["Score"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between GDP per capita and overall score",
    xaxis_title = "GDP factor",
    yaxis_title = "Score"
)

#fig.show()
img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[24]:

Scatter plot shows signs of considerate amount of correlation between GDP factor and Score. The next step is to compute this correlation.

[14]:
```
print(df['Score'].corr(df['GDP per capita']))
```

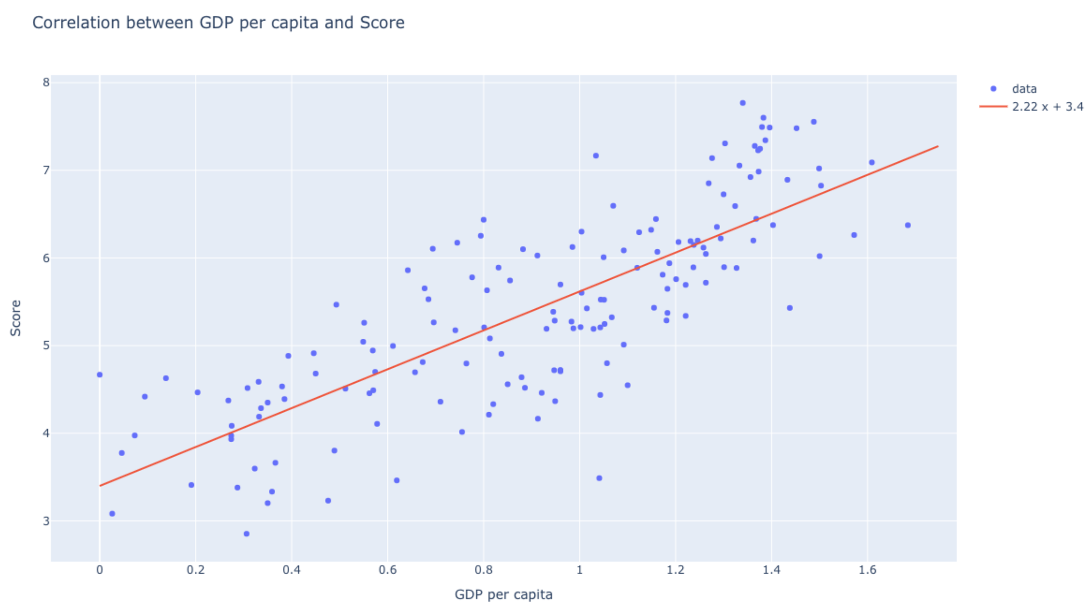
0.7938828678781278

The correlation is quite high. Fitting linear regression seems like a natural choice.

[21]:
```
x_data = df["GDP per capita"]
y_data = df["Score"]


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
 ↪y_data)

xi = [i/400 for i in range(0, 700)]

line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
```

```
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + ' x + ' + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between GDP per capita and Score',
    xaxis_title = "GDP per capita",
    yaxis_title = "Score"

)


fig = go.Figure(data = [scatter1,scatter2], layout = layout)

#fig.show()
img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[21]:



As we can see, polymonial fits the data. People tend to be more happy in the more wealthy countries, which is a quite obvious conclusion.

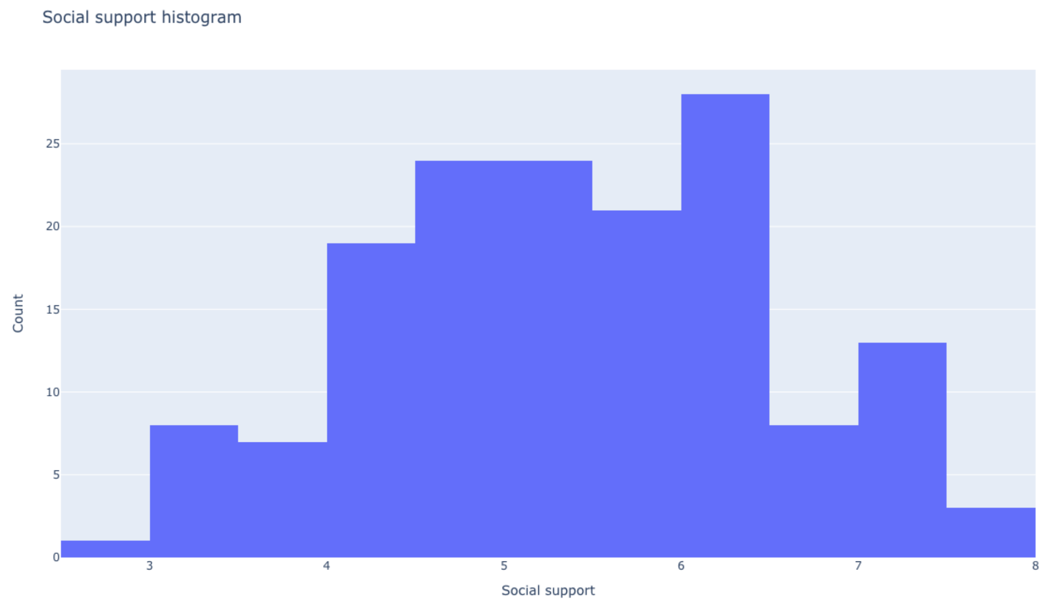
1.1.1 Social support - general score

[16]:
```
data = df['Social support']
data.describe()
```

[16]:
```
count    156.000000
mean       1.208814
std        0.299191
min        0.000000
25%        1.055750
50%        1.271500
75%        1.452500
max        1.624000
Name: Social support, dtype: float64
```

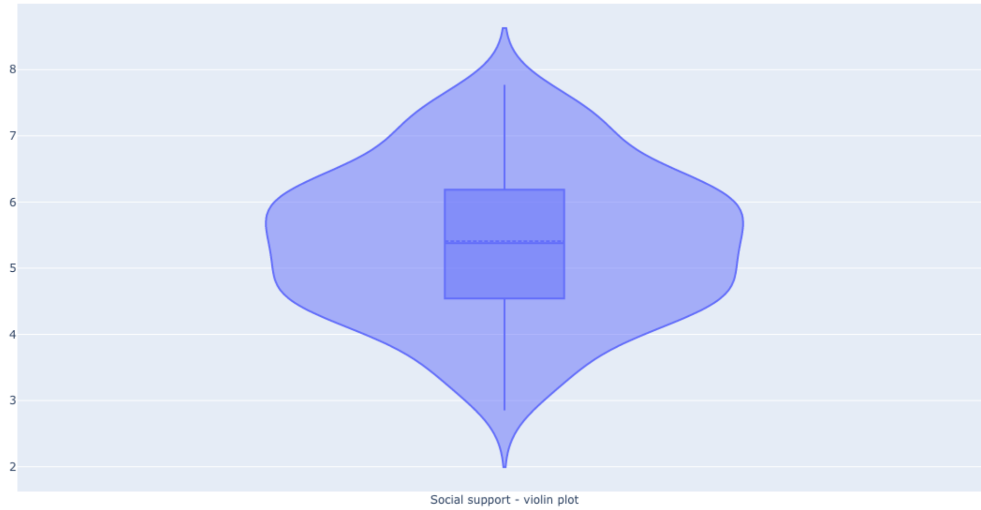
[27]:
```
plot = go.Figure(data=[
    go.Histogram(
        x=data
    )
])

plot.update_layout(
    title_text = "Social support histogram",
    xaxis_title_text = 'Social support',
    yaxis_title_text = 'Count'
)

#plot.show()

img_bytes = plot.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[27]:



[28]:
```
plot = go.Figure()
plot.add_trace(go.Violin(
    y = data,
    x0 ='Social support - violin plot',
    box_visible = True,
    meanline_visible = True
))

#plot.show()

img_bytes = plot.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

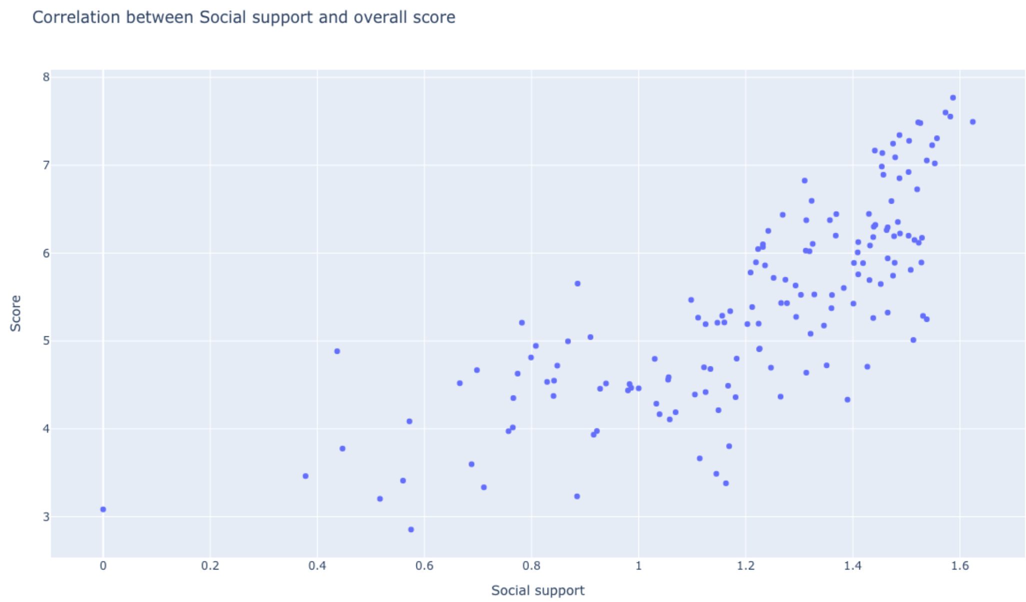
[28]:



1.1.2 Social support score tends to be higher than average for most of the countries, that means that for most of the participants, citizens of the countries in the report tend to think quite well of the social systems introduced by governments.

[29]:
```
x_data = df["Social support"]
y_data = df["Score"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between Social support and overall score",
    xaxis_title = "Social support",
    yaxis_title = "Score"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[29]:



[20]:
```
print(df["Score"].corr(df["Social support"]))
```

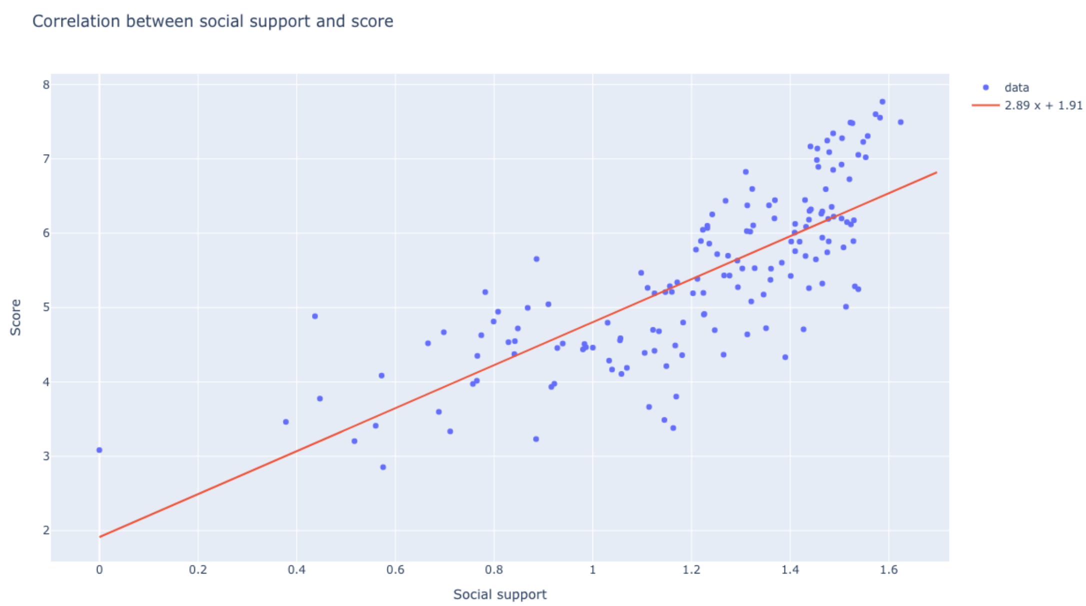
0.7770577880638645

1.1.3 Again correlation between Score and social support is high. It means that most likely linear regression will fit the data quite well.

[30]:
```
x_data = df["Social support"]
y_data = df["Score"]


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
↪y_data)

xi = [i/400 for i in range(0, 680)]

line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)
```

```
scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + ' x + ' + str(round(intercept,2))
)


layout = go.Layout(
    title='Correlation between social support and score',
    xaxis_title = "Social support",
    yaxis_title = "Score"
)

fig = go.Figure(data = [scatter1,scatter2], layout = layout)

#fig.show()
img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[30]:



1.1.4 Most of the data are concentrated in the vicinity of the regression line. The score is possitively influenced by rising level of social care.

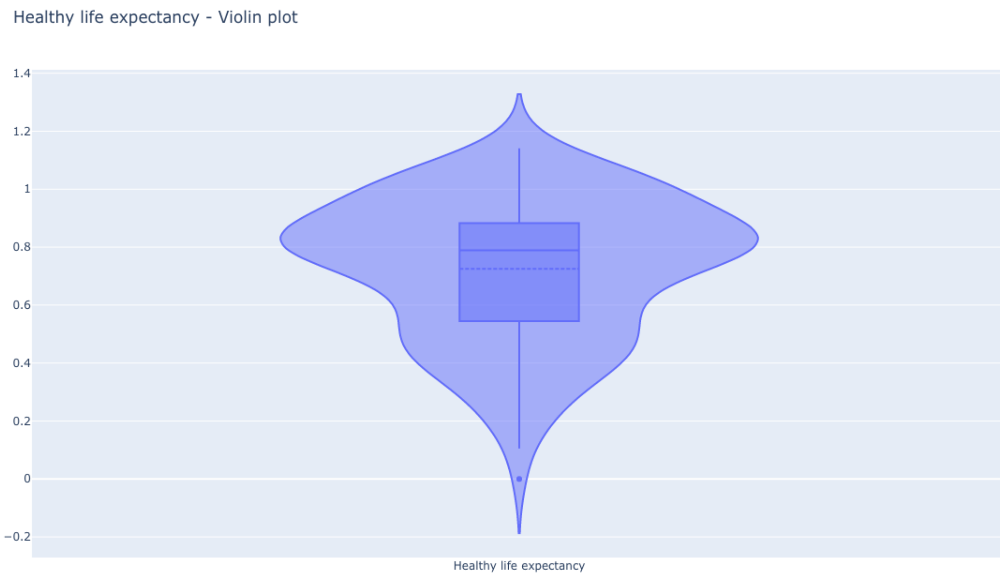
1.2 Healthy life expectancy

[22]:
```
data = df["Healthy life expectancy"]

data.describe()
```

[22]:
```
count    156.000000
mean       0.725244
std        0.242124
min        0.000000
25%        0.547750
50%        0.789000
75%        0.881750
max        1.141000
Name: Healthy life expectancy, dtype: float64
```

[31]:
```
data = df["Healthy life expectancy"]

fig = go.Figure()
fig.add_trace(go.Violin(
    y = data,
    x0 = "Healthy life expectancy",
    box_visible = True,
    meanline_visible = True,
))



fig.update_layout(
    title_text = "Healthy life expectancy - Violin plot"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

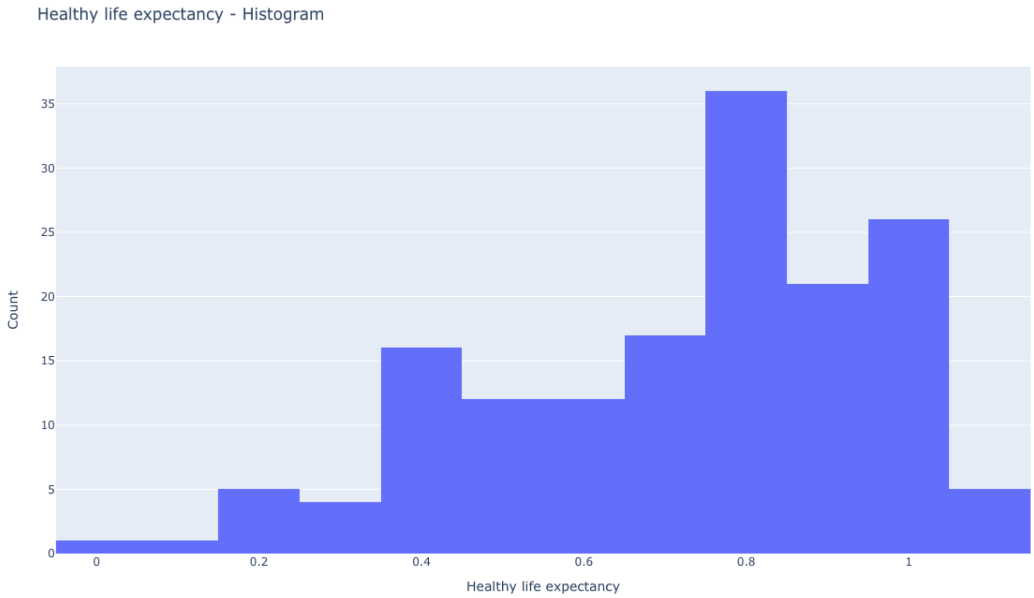
[31]:



[32]:

```
data = df["Healthy life expectancy"]

fig = go.Figure(data=[
    go.Histogram(
        x = data,
    )
])

fig.update_layout(
    title_text = "Healthy life expectancy - Histogram",
    xaxis_title_text = 'Healthy life expectancy',
    yaxis_title_text = 'Count'
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[32]:



[33]:

```python
x_data = df["Healthy life expectancy"]
y_data = df["Score"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between healthy life expectancy and happiness score",
    xaxis_title = "Healthy life expectancy",
    yaxis_title = "Happiness score"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[33]:

[34]:
```
x_data = df['Healthy life expectancy']
y_data = df["Score"]

slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
 ↪y_data)
xi = [i/650 for i in range(0, 900)]
line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + "* x +" + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between Healthy life expectancy and happiness score'
)
```

```
fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[34]:

Correlation coefficient

[27]:
```
df['Score'].corr(df['Healthy life expectancy'])
```

[27]: 0.7798831492425827

As we can see people are more happy the longer they live. This assumption is only natural since we want to expierience as much as we can so the longer we live the more we expirience and the happier we are.

1.3 Freedom to make life choices

[28]:
```
data = df["Freedom to make life choices"]
data.describe()
```

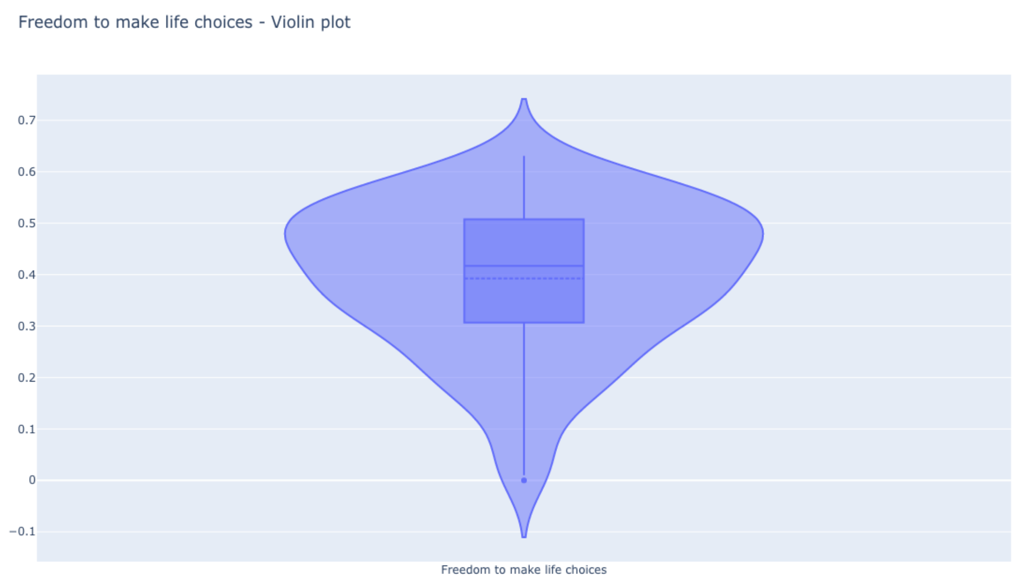
[28]:
```
count    156.000000
mean       0.392571
std        0.143289
```

```
min        0.000000
25%        0.308000
50%        0.417000
75%        0.507250
max        0.631000
Name: Freedom to make life choices, dtype: float64
```

[35]:

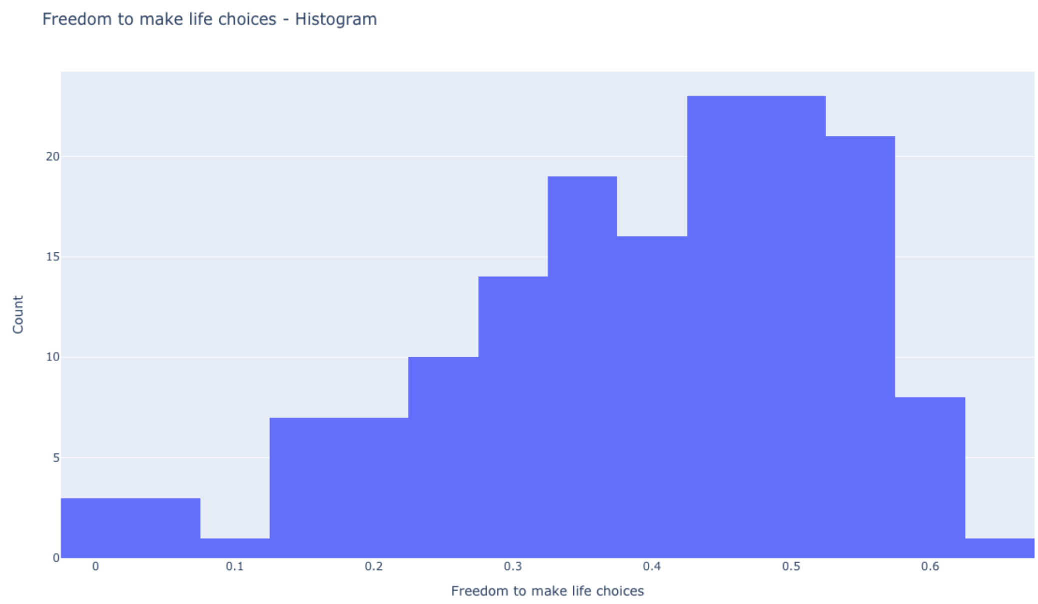
```python
data = df["Freedom to make life choices"]

fig = go.Figure()
fig.add_trace(go.Violin(
    y = data,
    x0 = "Freedom to make life choices",
    box_visible = True,
    meanline_visible = True,
))



fig.update_layout(
    title_text = "Freedom to make life choices - Violin plot"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[35]:



[36]:
```
data = df["Freedom to make life choices"]

fig = go.Figure(data=[
    go.Histogram(
        x = data,
    )
])

fig.update_layout(
    title_text = "Freedom to make life choices - Histogram",
    xaxis_title_text = 'Freedom to make life choices',
    yaxis_title_text = 'Count'
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[36]:



[37]:
```
x_data = df["Freedom to make life choices"]
y_data = df["Score"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
```

```
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between Freedom to make life choices and happiness␣
 ↪score",
    xaxis_title = "Freedom to make life choices",
    yaxis_title = "Happiness score"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[37]:

[38]:
```
x_data = df['Freedom to make life choices']
y_data = df["Score"]



slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
 ↪y_data)
xi = [i/700 for i in range(0, 450)]
line = [slope * i + intercept for i in xi]
```

```
scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + "* x +" + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between Freedom to make life choices and happiness score'
)

fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[38]:

Correlation coefficient

[33]:
```
df['Score'].corr(df['Freedom to make life choices'])
```

[33]:
```
0.5667418257199899
```

Here we can see that freedom to make life choices as a positive impact at happiness score. Comparing it to healthy life expectancy, freedom as less impact on happiness. It aligns well with maslow pyramid of needs as we first want to survive and then we want to satify our social needs.

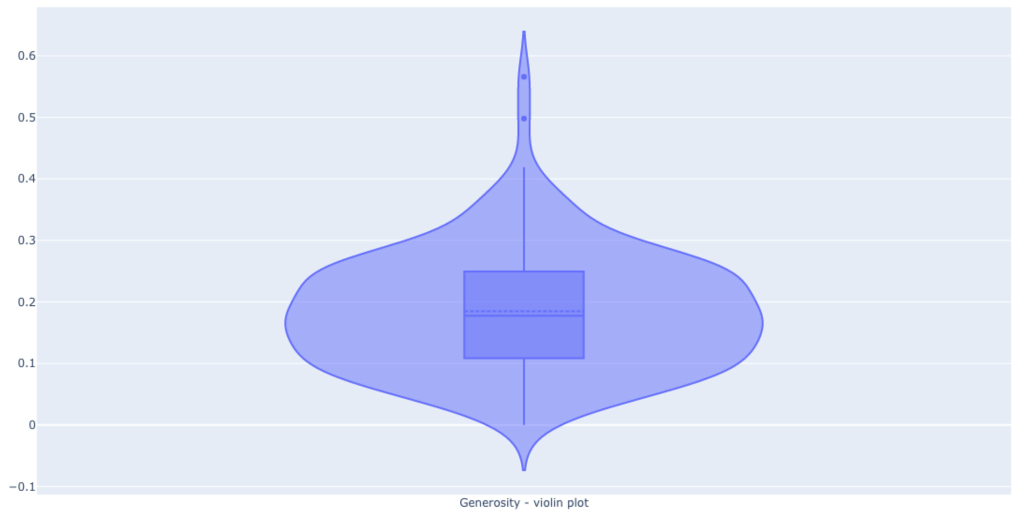
1.4 Generosity

[34]:
```
data = df['Generosity']
data.describe()
```

[34]:
```
count    156.000000
mean       0.184846
std        0.095254
min        0.000000
25%        0.108750
50%        0.177500
75%        0.248250
max        0.566000
Name: Generosity, dtype: float64
```

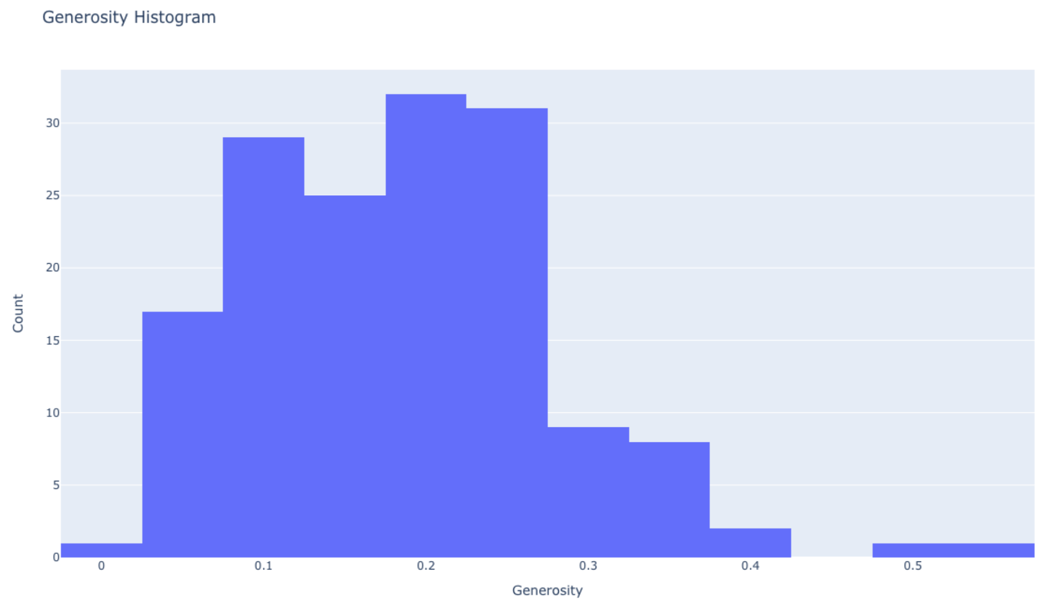
[39]:
```
data = df['Generosity']

fig = go.Figure()
fig.add_trace(go.Violin(
    y = data,
    x0 ='Generosity - violin plot',
    box_visible = True,
    meanline_visible = True,
))

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[39]:



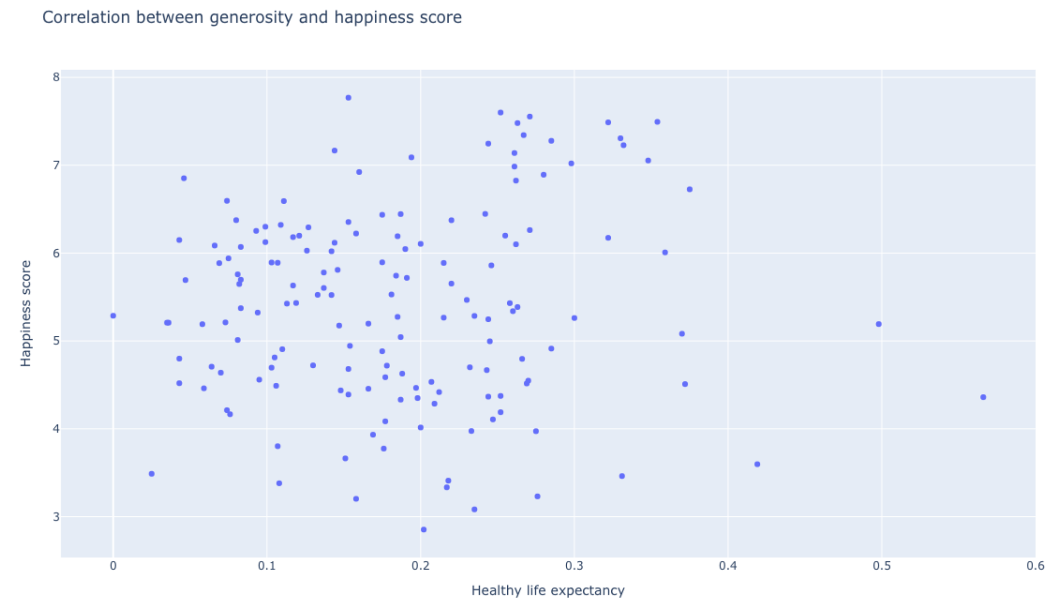
[40]:
```
data = df['Generosity']

fig = go.Figure(go.Histogram(
        x=data,
    ))
fig.update_layout(
    title_text = "Generosity Histogram",
    xaxis_title_text = 'Generosity',
    yaxis_title_text = 'Count'
)
#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[40]:



[41]:
```python
data_x = df['Generosity']
data_y = df['Score']

fig = go.Figure(data = go.Scatter(
    x = data_x,
    y = data_y,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between generosity and happiness score",
    xaxis_title = "Healthy life expectancy",
    yaxis_title = "Happiness score"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[41]:



[42]:
```
data_x = df['Generosity']
data_y = df['Score']


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(data,␣
↪y_data)
xi = [i/600 for i in range(0, 350)]
line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = data_x,
    y = data_y,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope, 2)) + "x + " + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between generosity and happiness score'
```

```
)

fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[42]:

Correlation coefficient:

[39]:
```
df['Score'].corr(data)
```

[39]: 0.07582369490389643

As we can correlation between generosity and final score in ranging is very week. So we can conclude that generosity of inhabitants does not have a big impact on the happiness.

1.5 Perceptions of corruption

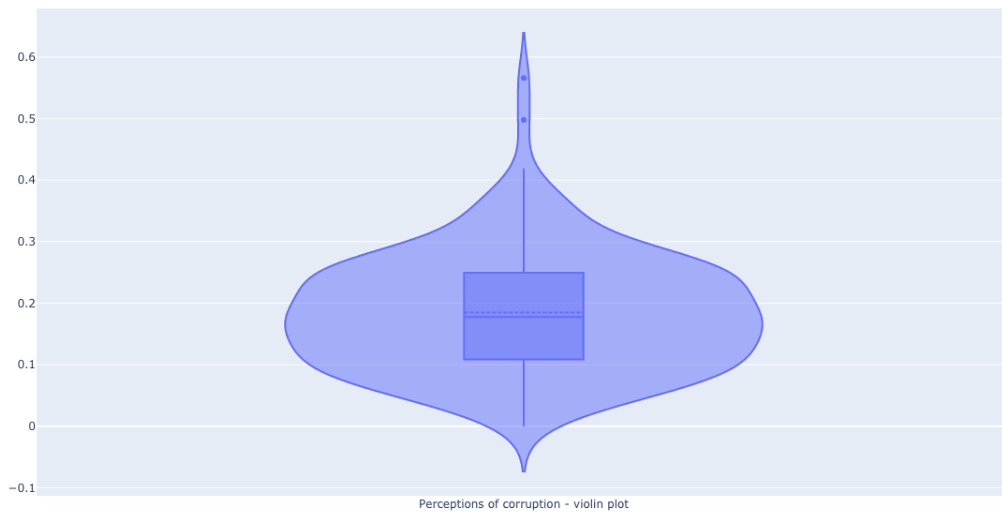
[40]:
```
data = df['Perceptions of corruption']
data.describe()
```

[40]:
```
count    156.000000
mean       0.110603
```

```
std        0.094538
min        0.000000
25%        0.047000
50%        0.085500
75%        0.141250
max        0.453000
Name: Perceptions of corruption, dtype: float64
```

[43]:
```python
fig = go.Figure()
fig.add_trace(go.Violin(
    y = data,
    x0 ='Perceptions of corruption - violin plot',
    box_visible = True,
    meanline_visible = True
))

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

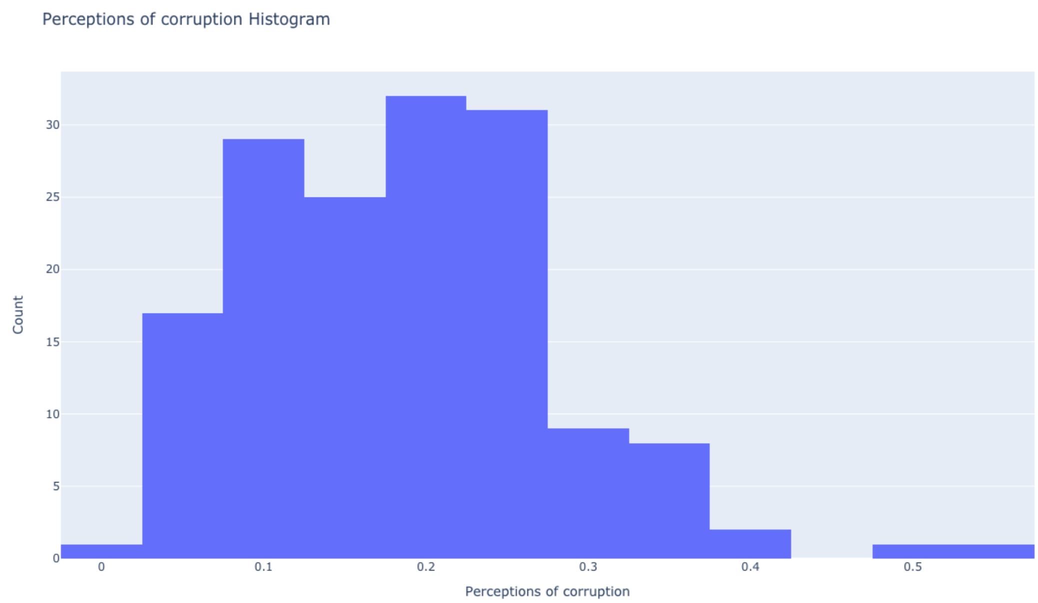
[43]:



[44]:
```python
fig = go.Figure(go.Histogram(
        x=data,
    ))
fig.update_layout(
    title_text = "Perceptions of corruption Histogram",
```

```python
    xaxis_title_text = 'Perceptions of corruption',
    yaxis_title_text = 'Count'
)
#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[44]:

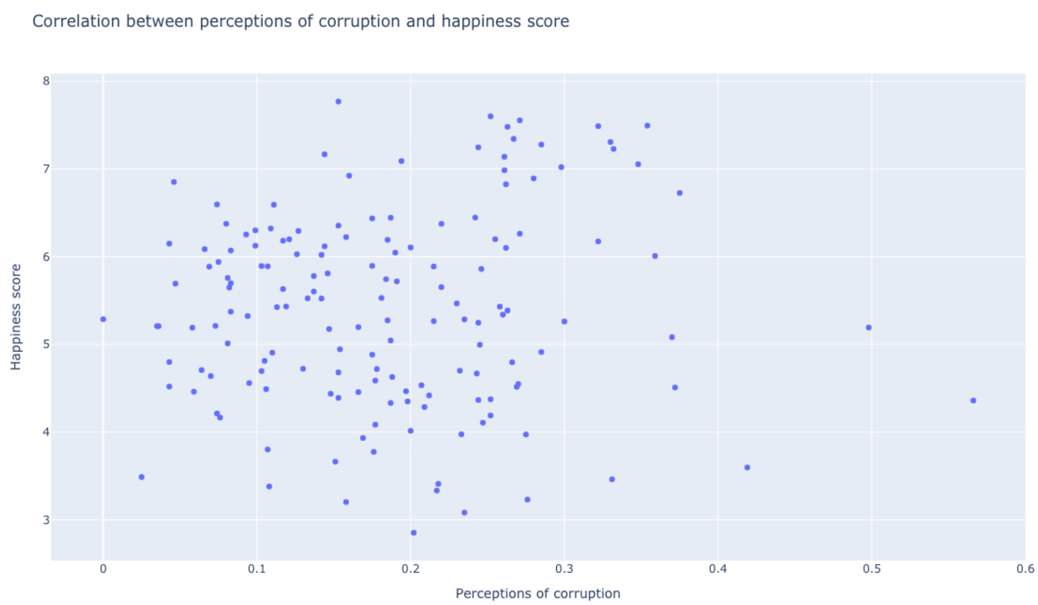


[45]:

```python
data_x = df['Perceptions of corruption']
data_y = df['Score']

fig = go.Figure(data = go.Scatter(
    x = data,
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between perceptions of corruption and happiness score",
    xaxis_title = "Perceptions of corruption",
    yaxis_title = "Happiness score"
)

#fig.show()
```

```
img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[45]:

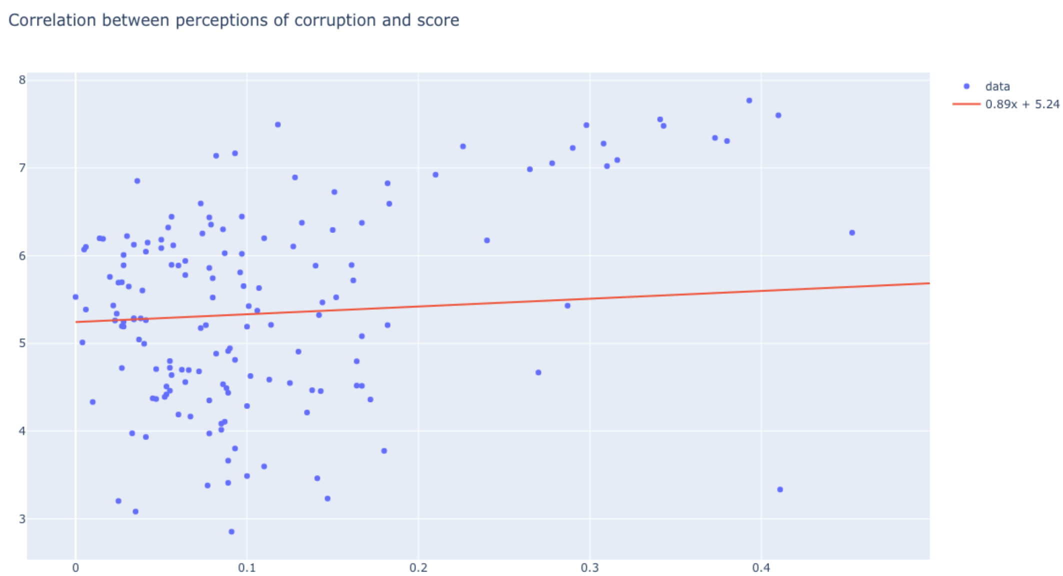


[46]:

```
data_x = df['Perceptions of corruption']
data_y = df['Score']


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(data,␣
↪y_data)
xi = [i/600 for i in range(0, 300)]
line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = data_x,
    y = data_y,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope, 2)) + "x + " + str(round(intercept,2))
)
```

```
layout = go.Layout(
    title='Correlation between perceptions of corruption and score'
)

fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[46]:



Correlation coefficient:

[45]:
```
df['Score'].corr(data)
```

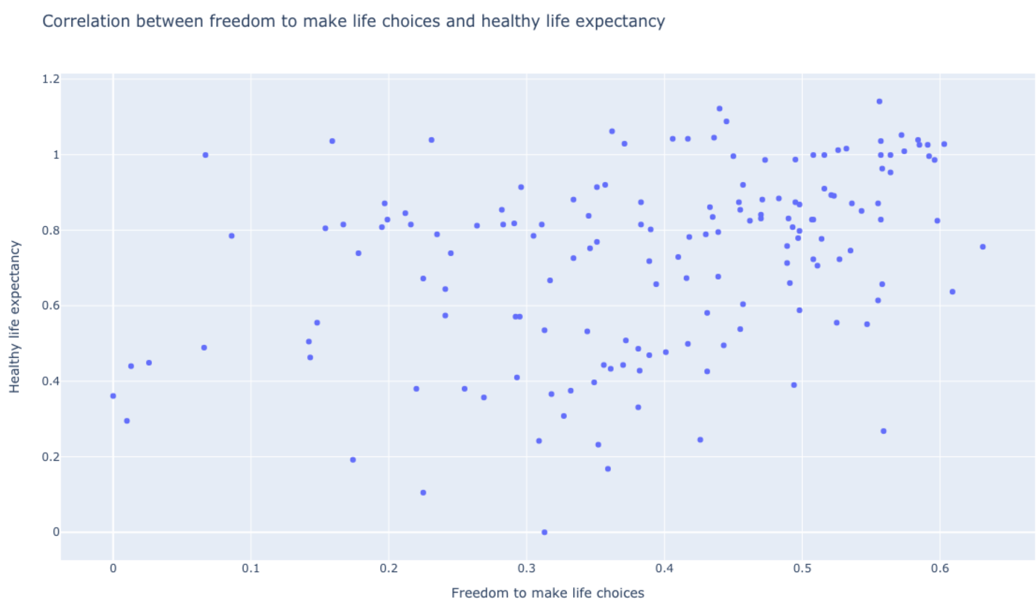
[45]: 0.38561307086647867

As we can correlation between perceptions of corruption and final score in ranging is not strong. So we conclude that it has some input in general score but it is not so significant.

1.6 Correlation between freedom to make life choices and healthy life expectancy

[47]:
```
data_x = df['Freedom to make life choices']
data_y = df['Healthy life expectancy']


fig = go.Figure(data = go.Scatter(
    x = data_x,
    y = data_y,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between freedom to make life choices and healthy life␣
 ↪expectancy",
    xaxis_title = "Freedom to make life choices",
    yaxis_title = "Healthy life expectancy"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

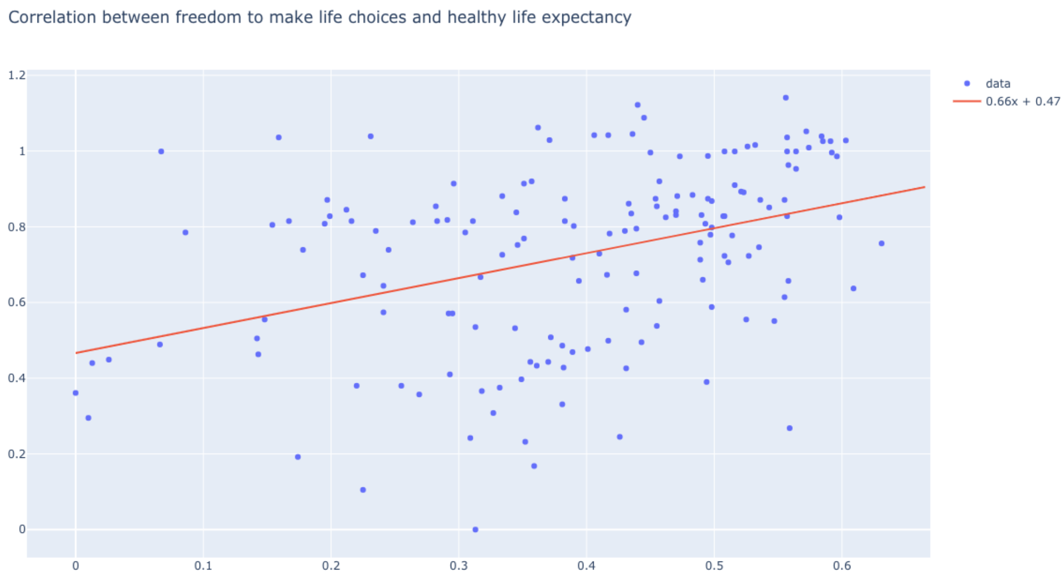
[47]:



[48]:

```
data_x = df['Freedom to make life choices']
data_y = df['Healthy life expectancy']


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(data_x,␣
↪data_y)
xi = [i/600 for i in range(0, 400)]
line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = data_x,
    y = data_y,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope, 2)) + "x + " + str(round(intercept,2))
)

layout = go.Layout(
    title="Correlation between freedom to make life choices and healthy life␣
↪expectancy"
)

fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[48]:



Correlation coefficient:

[48]:
```
data_x.corr(data_y)
```

[48]: 0.3903947764769573

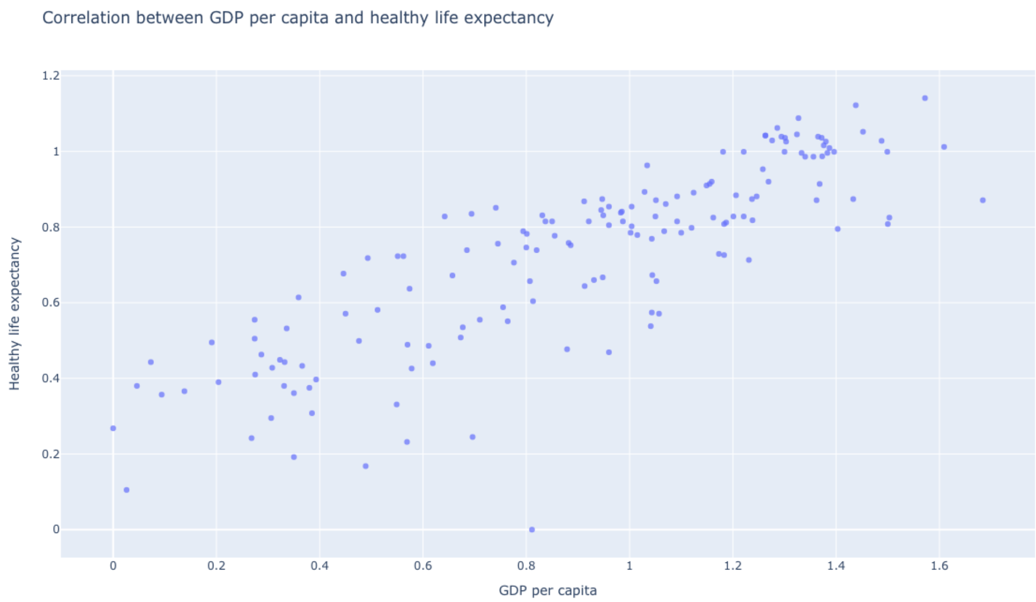
As we can see correlation between freedom to make life choices and healthy life expectancy is not strong. It is understandable because one can expect that there are more important factors which have bigger impact on lifetime.

Correlation between the wealth of the country and the healthy life expectancy of its citizens.

[49]:
```
x_data = df["GDP per capita"]
y_data = df["Healthy life expectancy"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    marker = {
            'opacity': 0.7,
            'colorscale': 'Portland',
        },
    hovertext = df["Country or region"]
))
```

```
fig.update_layout(
    title = "Correlation between GDP per capita and healthy life expectancy",
    xaxis_title = "GDP per capita",
    yaxis_title = "Healthy life expectancy"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[49]:



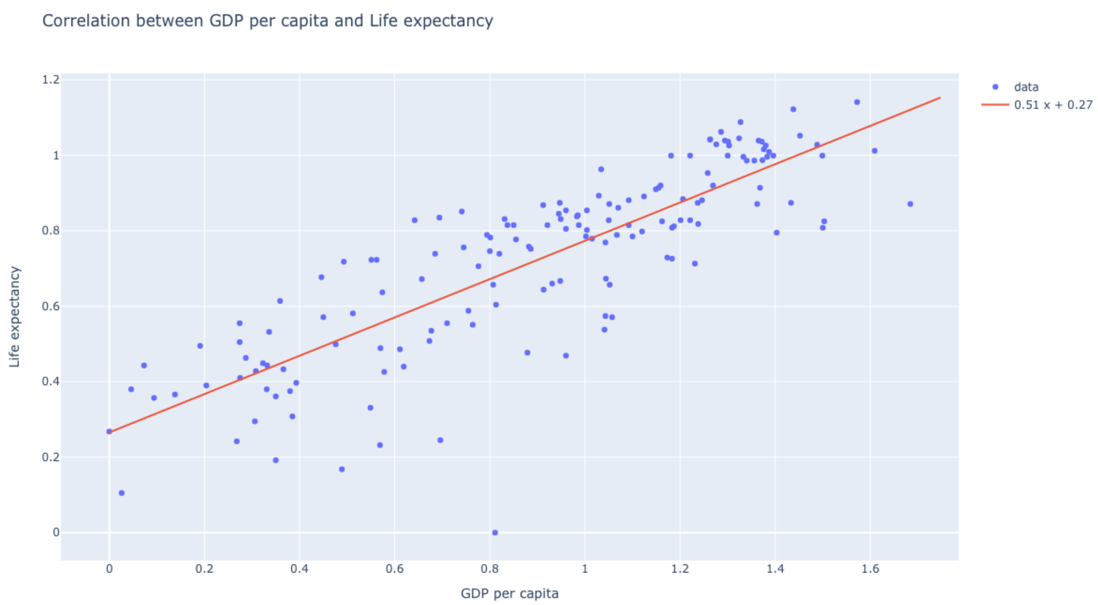
[50]:
```
df['GDP per capita'].corr(df['Healthy life expectancy'])
```

[50]: 0.8354621150416075

[50]:
```
x_data = df["GDP per capita"]
y_data = df["Healthy life expectancy"]


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
↪y_data)

xi = [i/400 for i in range(0, 700)]

line = [slope * i + intercept for i in xi]
```

```
scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + ' x + ' + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between GDP per capita and Life expectancy',
    xaxis_title = "GDP per capita",
    yaxis_title = "Life expectancy"

)

fig = go.Figure(data = [scatter1,scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[50]:

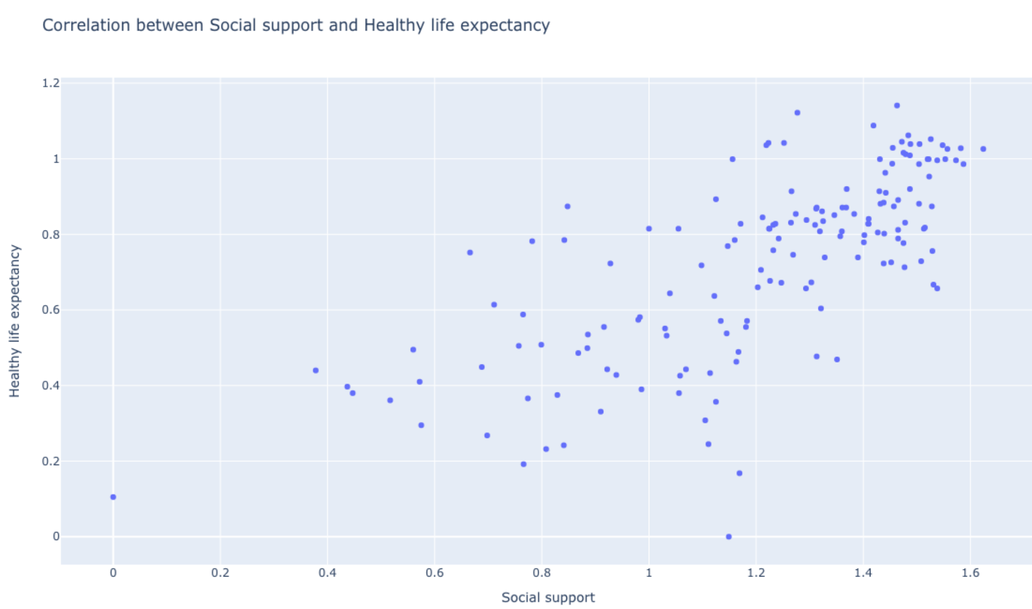


People from wealthy countries are not only happier, but also healthier and live longer as it seems from linear regression. Better healthcare, richer diet and less straining lifestyle are most likely causing such a trend.

1.7 Correlation between social support and healthy life expectancy

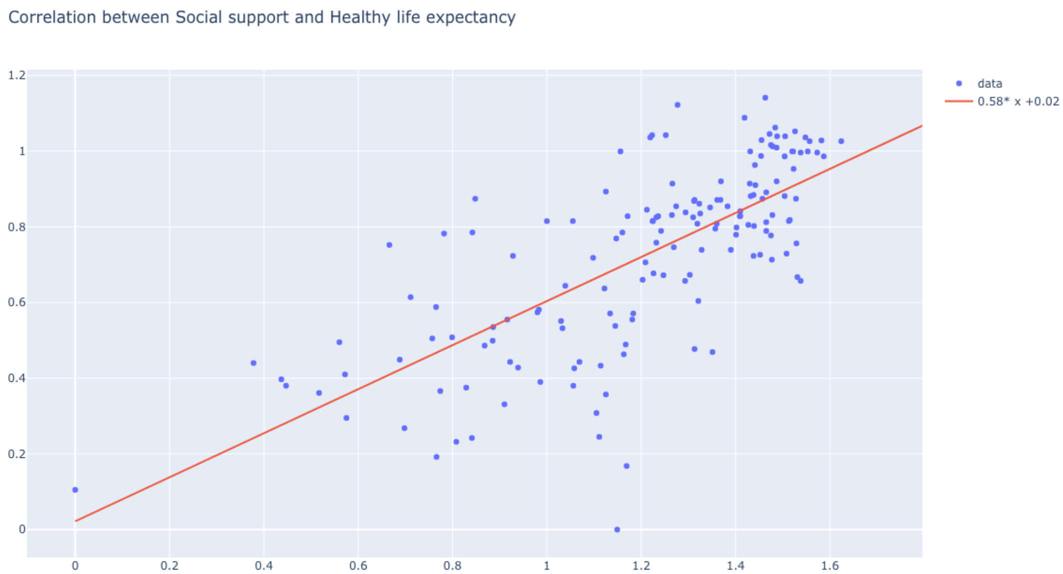
[51]:
```
x_data = df["Social support"]
y_data = df["Healthy life expectancy"]

fig = go.Figure(data = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    hovertext = df["Country or region"]
))

fig.update_layout(
    title = "Correlation between Social support and Healthy life expectancy",
    xaxis_title = "Social support",
    yaxis_title = "Healthy life expectancy"
)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[51]:



[52]:
```
x_data = df['Social support']
y_data = df["Healthy life expectancy"]


slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(x_data,␣
 ↪y_data)
xi = [i/250 for i in range(0, 450)]
line = [slope * i + intercept for i in xi]

scatter1 = go.Scatter(
    x = x_data,
    y = y_data,
    mode = 'markers',
    name = 'data',
    hovertext = df["Country or region"]
)

scatter2 = go.Scatter(
    x = xi,
    y = line,
    mode='lines',
    name= str(round(slope,2)) + "* x +" + str(round(intercept,2))
)

layout = go.Layout(
    title='Correlation between Social support and Healthy life expectancy'
)

fig = go.Figure(data = [scatter1, scatter2], layout = layout)

#fig.show()

img_bytes = fig.to_image(format='png', width=1200, height=700, scale=1)
Image(img_bytes)
```

[52]:



Correlation coefficient:

[54]:
```
df['Social support'].corr(df['Healthy life expectancy'])
```

[54]: 0.7190094590308563

Here we can see that social support has a clear impact on healty life expectancy which is only natural. The more humanitary help Country has the more it values good and healty life.