Plotting Exercises, Part 1

> *Bárbara Flores*

Exercise 1

Create a pandas dataframe from the "Datasaurus.txt" file using the code:

Note that the file being downloaded is *not* actually a CSV file. It is tab-delimited, meaning that within each row, columns are separated by tabs rather than commas. We communicate this to pandas with the `delimiter="\t"` option (`"\t"` is how we write a tab, as we will discuss in future lessons).

In [ ]:
```
import pandas as pd
import numpy as np
import altair as alt
import seaborn.objects as so


pd.set_option("mode.copy_on_write", True)

# Download the data
datasaurus = pd.read_csv(
    "https://raw.githubusercontent.com/nickeubank/practicaldatascience/maste
    "/Example_Data/Datasaurus.txt",
    delimiter="\t",
)

datasaurus.sample(5)
```

Out[ ]:

	example1_x	example1_y	example2_x	example2_y	example3_x	example3_y	ex
105	36.355695	6.005849	68.453046	13.035529	69.513583	20.279358	
116	30.846919	37.341640	63.497323	11.914069	51.566995	16.485714	
110	48.823197	76.635331	52.992029	9.906668	64.010241	17.924478	
73	28.003324	46.672192	53.174656	84.260568	56.475383	79.167840	
38	67.759796	72.421202	40.074667	34.623685	33.446183	24.190643	

5 rows × 26 columns

Exercise 2

This dataset actually contains 13 separate example datasets, each with two variables named `example[number]_x` and `example[number]_y`.

In order to get a better sense of what these datasets look like, write a loop that iterates over each example dataset (numbered 1 to 13) and print out the mean and standard deviation for `example[number]_x` and `example[number]_y` for each dataset.

For example, the first iteration of this loop might return something like:

```
Example Dataset 1:
Mean x: 23.12321978429576,
Mean y: 98.23980921730972,
Std Dev x: 21.2389710287,
Std Dev y: 32.2389081209832,
Correlation: 0.73892819281
```

(Though you shouldn't get those specific values)

In [ ]:
```python
for i in range(13):
    x = datasaurus[datasaurus.columns[i * 2]]
    y = datasaurus[datasaurus.columns[i * 2 + 1]]
    x_mean = x.mean()
    y_mean = y.mean()
    x_std = x.std()
    y_std = y.std()
    corr = x.corr(y)
    print(f"Example Dataset {i+1}:")
    print(f"Mean x: {x_mean:4f},")
    print(f"Mean y: {y_mean:4f},")
    print(f"Std Dev x: {x_std:4f},")
    print(f"Std Dev y: {y_std:4f},")
    print(f"Correlation: {corr:4f}\n")
```

Example Dataset 1:
Mean x: 54.266100,
Mean y: 47.834721,
Std Dev x: 16.769825,
Std Dev y: 26.939743,
Correlation: -0.064128

Example Dataset 2:
Mean x: 54.268730,
Mean y: 47.830823,
Std Dev x: 16.769239,
Std Dev y: 26.935727,
Correlation: -0.068586

Example Dataset 3:
Mean x: 54.267320,
Mean y: 47.837717,
Std Dev x: 16.760013,
Std Dev y: 26.930036,
Correlation: -0.068343

Example Dataset 4:
Mean x: 54.263273,
Mean y: 47.832253,
Std Dev x: 16.765142,
Std Dev y: 26.935403,
Correlation: -0.064472

Example Dataset 5:
Mean x: 54.260303,
Mean y: 47.839829,
Std Dev x: 16.767735,
Std Dev y: 26.930192,
Correlation: -0.060341

Example Dataset 6:
Mean x: 54.261442,
Mean y: 47.830252,
Std Dev x: 16.765898,
Std Dev y: 26.939876,
Correlation: -0.061715

Example Dataset 7:
Mean x: 54.268805,
Mean y: 47.835450,
Std Dev x: 16.766704,
Std Dev y: 26.939998,
Correlation: -0.068504

Example Dataset 8:
Mean x: 54.267849,
Mean y: 47.835896,
Std Dev x: 16.766759,
Std Dev y: 26.936105,
Correlation: -0.068980

```
Example Dataset 9:
Mean x: 54.265882,
Mean y: 47.831496,
Std Dev x: 16.768853,
Std Dev y: 26.938608,
Correlation: -0.068609

Example Dataset 10:
Mean x: 54.267341,
Mean y: 47.839545,
Std Dev x: 16.768959,
Std Dev y: 26.930275,
Correlation: -0.062961

Example Dataset 11:
Mean x: 54.269927,
Mean y: 47.836988,
Std Dev x: 16.769959,
Std Dev y: 26.937684,
Correlation: -0.069446

Example Dataset 12:
Mean x: 54.266916,
Mean y: 47.831602,
Std Dev x: 16.770000,
Std Dev y: 26.937902,
Correlation: -0.066575

Example Dataset 13:
Mean x: 54.260150,
Mean y: 47.839717,
Std Dev x: 16.769958,
Std Dev y: 26.930002,
Correlation: -0.065583
```

Exercise 3

Based only on these results, discuss what might you conclude about these example datasets with your partner. Write down your thoughts.

> *When analyzing the statistics at first glance, they seem to be quite similar. The mean of x is around 54.26, and the mean of Y is 47.8. The same holds for the standard deviation; among the different examples, the data appears to be quite similar.*
>
> *It's probably necessary to visualize the data to observe if there are different patterns.*

Execise 4

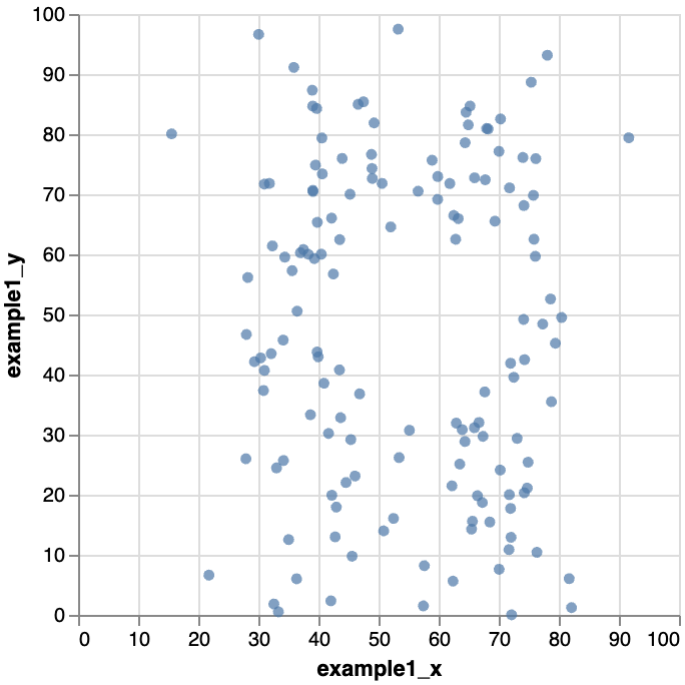
Write a loop that iterates over these example datasets, and using Altair library, plot a simple scatter plot of each dataset with the `x` variable on the x-axis and the `y` variable on the y-axis.

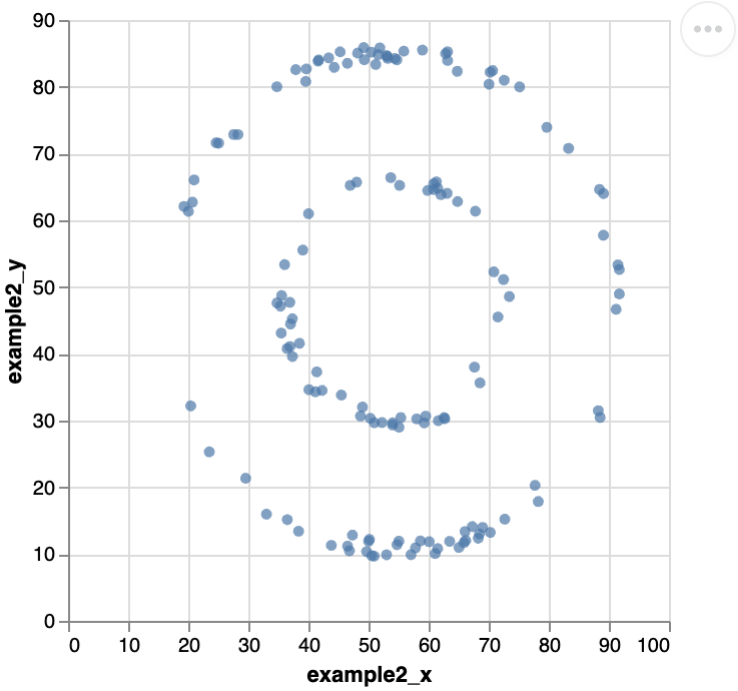
Hint: When writing this type of code, it is often best to start by writing code to do what you want for the first iteration of the loop. Once you have code that works for the first example dataset, then write the full loop around it.

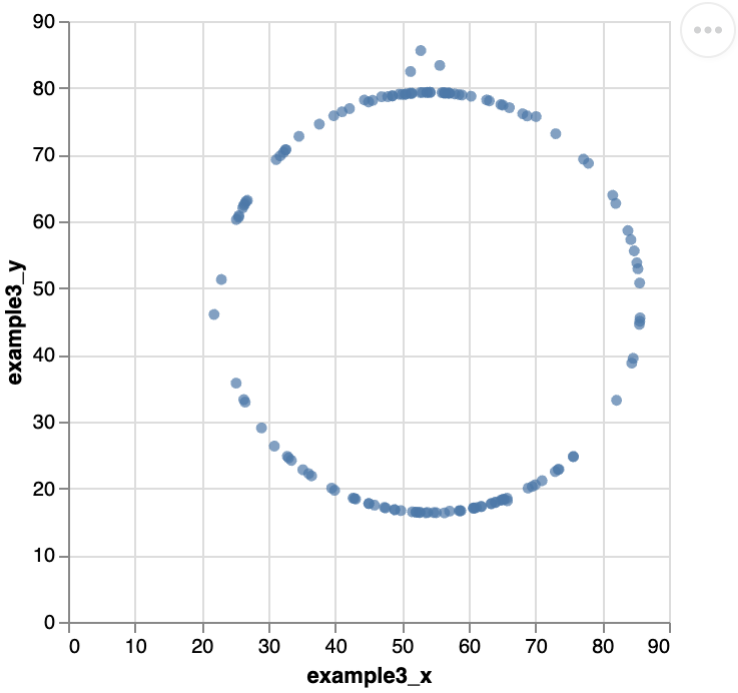
Hint 2: To force Jupyter to display your charts when they're generated within a loop, use the method `.show()` (e.g. `my_chart.show()`).

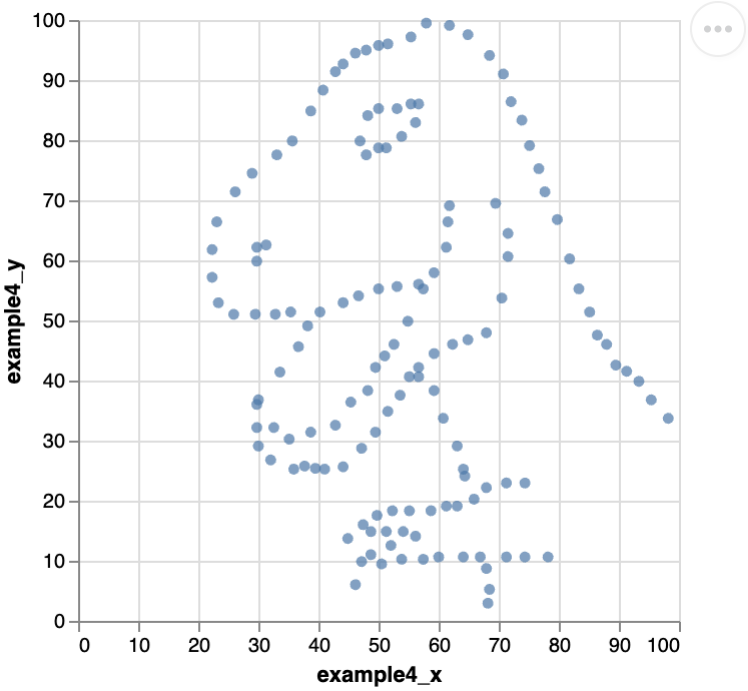
Hint 3: You will need to change the range of the axes to make the plots look good!

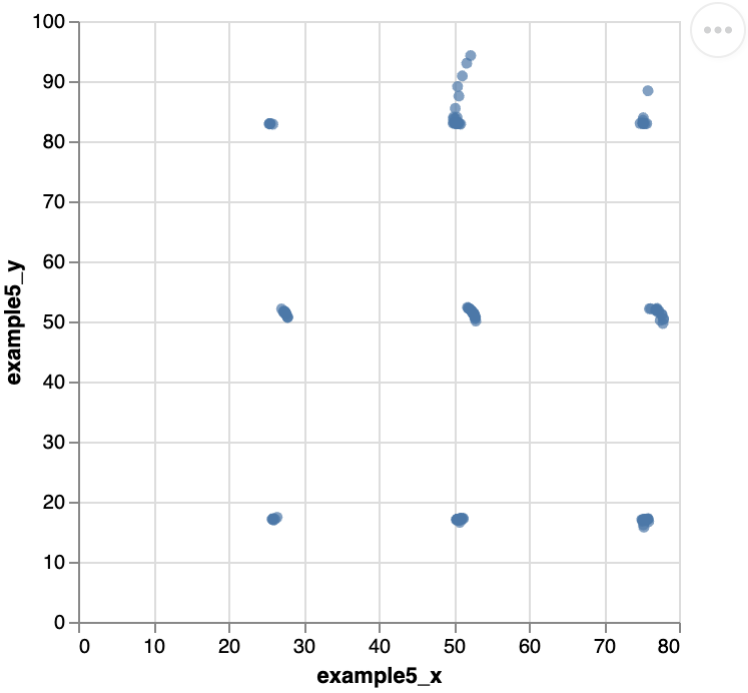
In [ ]:
```
for i in range(13):
    df = datasaurus[datasaurus.columns[i * 2 : i * 2 + 2]]
    x_name = df.columns[0]
    y_name = df.columns[1]
    chart = alt.Chart(df).mark_circle().encode(x=x_name, y=y_name)

    chart.display()
```

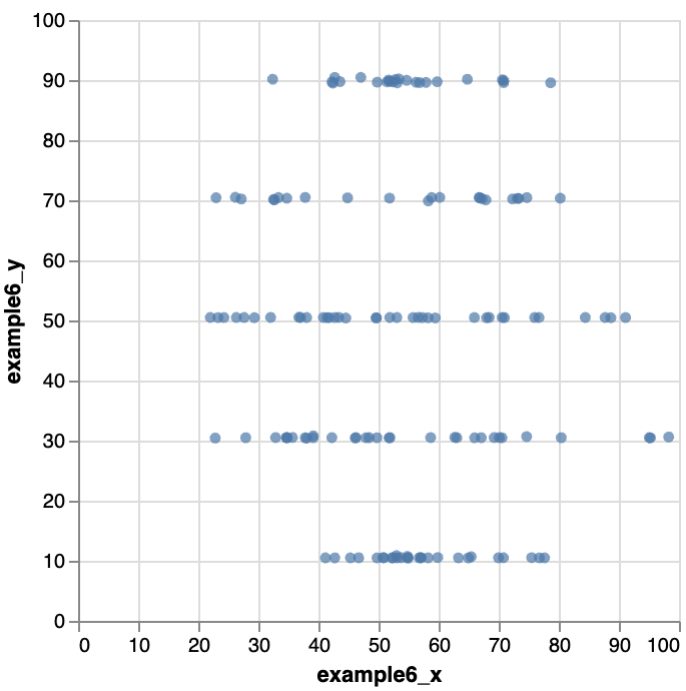


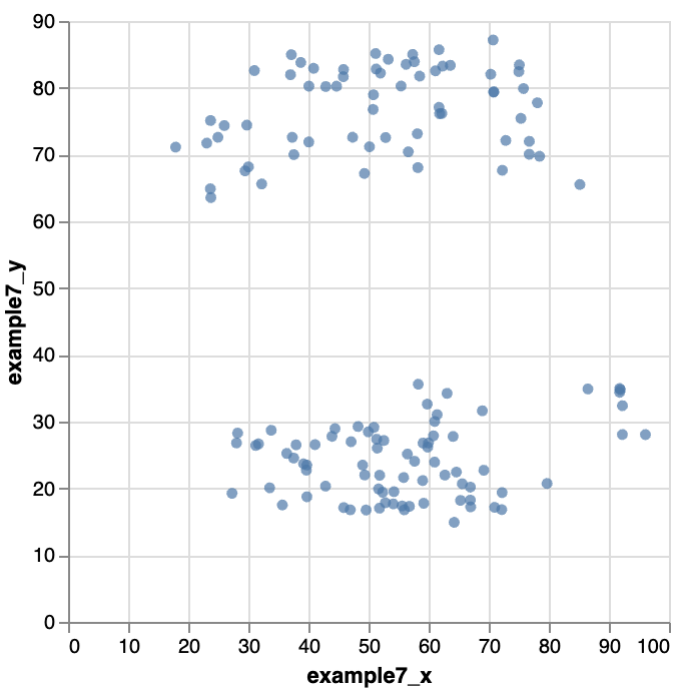

example2_y
example2_x
0
10
20
30
40
50
60
70
80
90
100

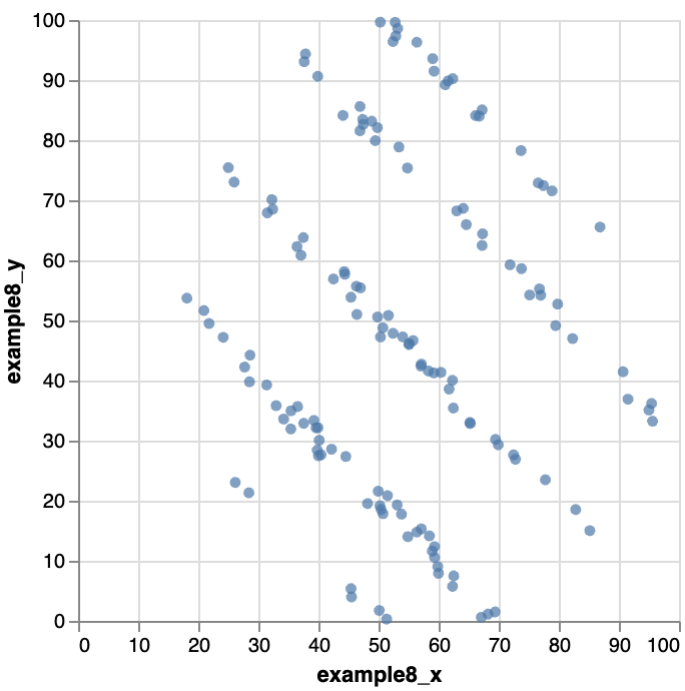

example3_y
example3_x
0
10
20
30
40
50
60
70
80
90

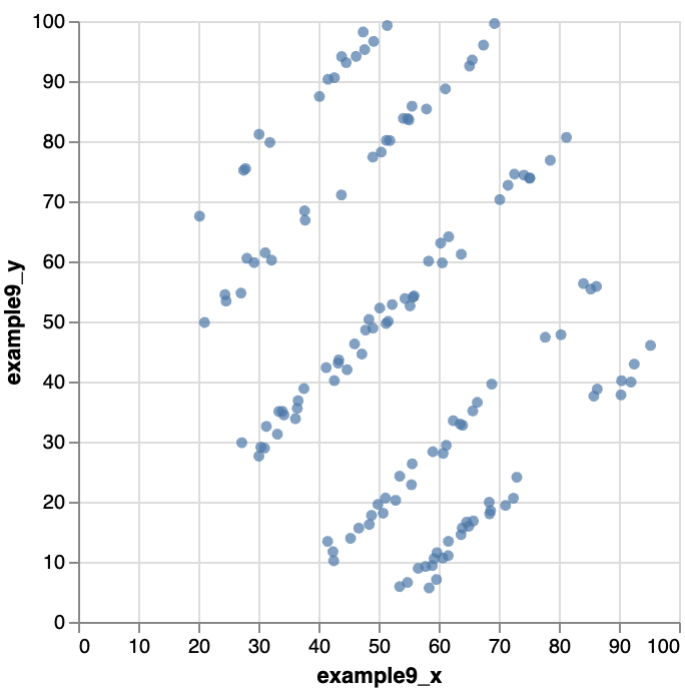

example4_y
example4_x
0
10
20
30
40
50
60
70
80
90
100

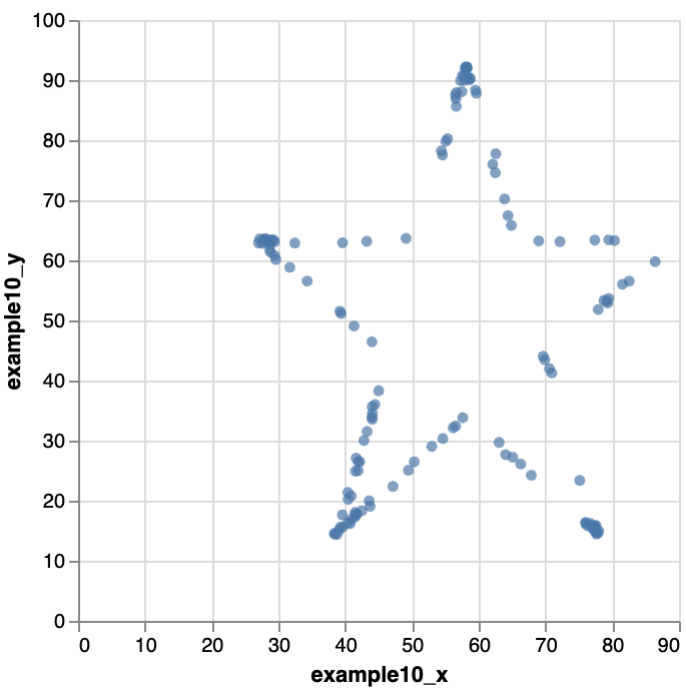

example5_y
example5_x
0
10
20
30
40
50
60
70
80
90
100

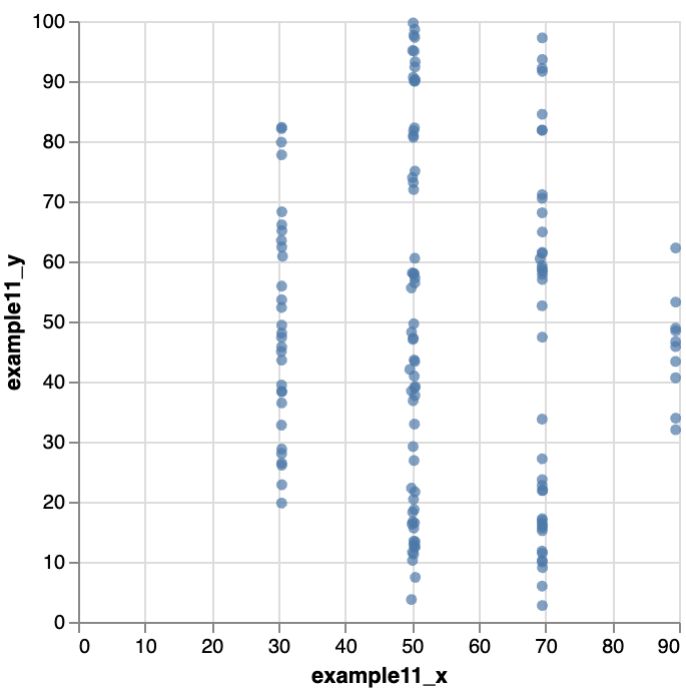

example6_y
example6_x
0
10
20
30
40
50
60
70
80
90
100

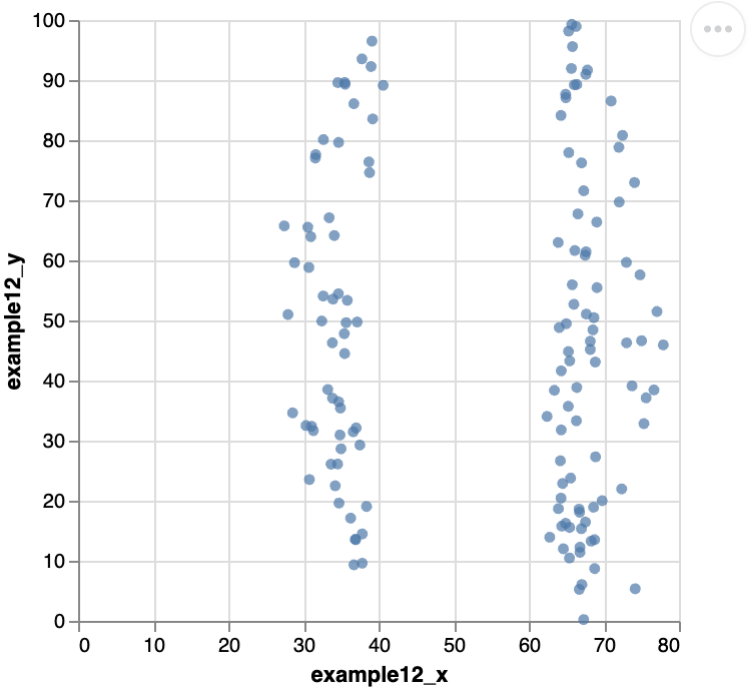

example7_y
example7_x
0
10
20
30
40
50
60
70
80
90
100

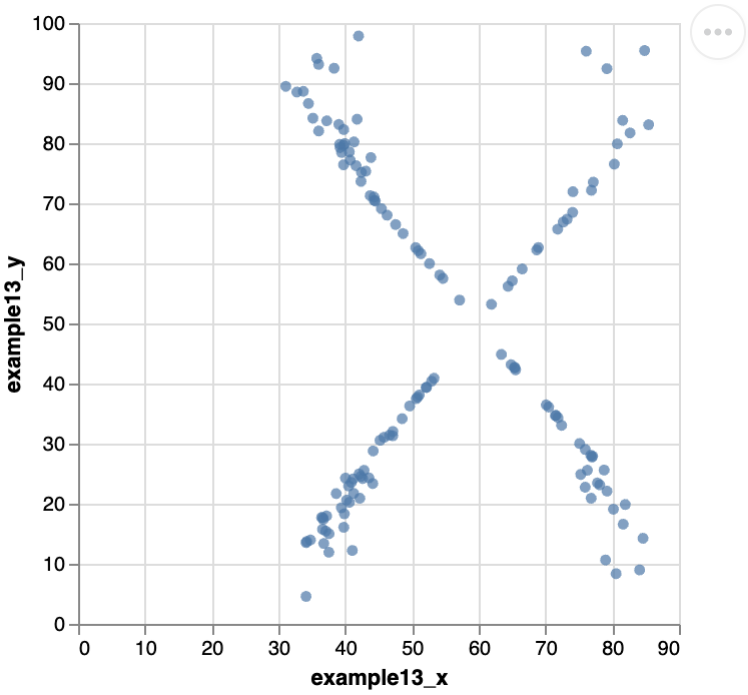

example8_y
example8_x
0
10
20
30
40
50
60
70
80
90
100


example9_y
example9_x
0
10
20
30
40
50
60
70
80
90
100


example10_y
example10_x
0
10
20
30
40
50
60
70
80
90
100


example11_y
example11_x
0
10
20
30
40
50
60
70
80
90
100





Exercise 5

Review you plots. How does your impression of how these datasets differ from what you wrote down in Exercise 3?

> *At first glance, when we only analyze the statistics such as mean, standard deviation, and correlation, all datasets appear to be very similar. However, when plotted, we realize that they are distributed in very different ways, information that we couldn't identify before graphing.*

> *This is a small example of the powerful tool that data visualization represents*

Economic Development and... Your Choice!

Exercise 6

Load the World Development Indicator data used in the plotting reading. Rather than picking a single year, pick a single country and look at how GDP per capita and one of the other variables in that dataset have evolved together over time.

Make any adjustments to the functional forms of your variables and/or axes needed to make the figure legible.

In [ ]:
```
wdi_data = (
    "https://raw.githubusercontent.com/nickeubank/"
    "practicaldatascience/master/Example_Data/wdi_plotting.csv"
)
world = pd.read_csv(wdi_data)
```

In [ ]:
```
world.sample(3)
```

Out[ ]:

	Year	Country Name	Country Code	GDP per capita (constant 2010 US$)	Population, total	CO2 emissions (metric tons per capita)	Mortality rate attributed to household and ambient air pollution, age-standardized (per 100,000 population)	P p po e
3970	1989	Ethiopia	ETH	209.628861	46272308.0	0.061100	NaN	
4918	1993	Nigeria	NGA	1437.278325	102700751.0	0.821902	NaN	
3994	1989	Iceland	ISL	NaN	252852.0	7.526826	NaN	

In [ ]:
```
world.info()
```

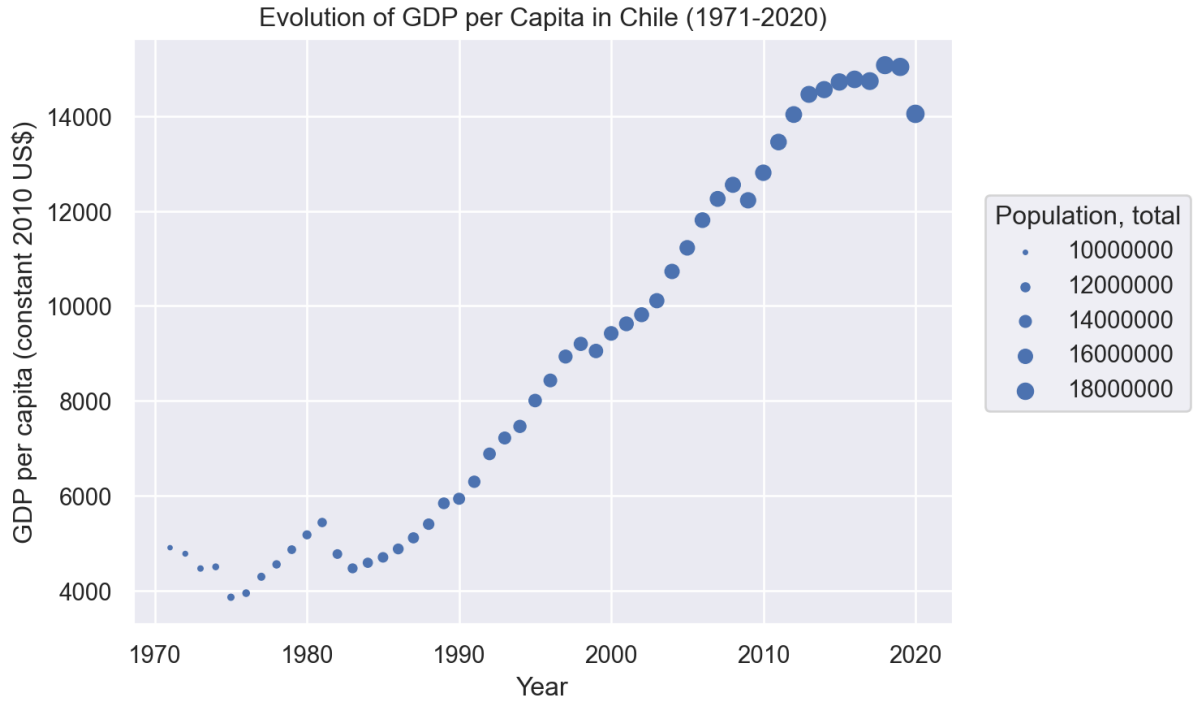
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10850 entries, 0 to 10849
Data columns (total 11 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   Year
10850 non-null  int64
 1   Country Name
10850 non-null  object
 2   Country Code
10850 non-null  object
 3   GDP per capita (constant 2010 US$)
8468 non-null   float64
 4   Population, total
10819 non-null  float64
 5   CO2 emissions (metric tons per capita)
8745 non-null   float64
 6   Mortality rate attributed to household and ambient air pollution, age-s
tandardized (per 100,000 population)  183 non-null    float64
 7   PM2.5 air pollution, population exposed to levels exceeding WHO guideli
ne value (% of total)                 2328 non-null   float64
 8   Life expectancy at birth, total (years)
9642 non-null   float64
 9   Mortality rate, under-5 (per 1,000 live births)
9072 non-null   float64
 10  Literacy rate, youth female (% of females ages 15-24)
918 non-null    float64
dtypes: float64(8), int64(1), object(2)
memory usage: 932.6+ KB
```

> *For this exercise, I will select my country, Chile, and we will graph GDP per capita (constant 2010 US$) and its evolution between the years 1971 and 2020. Additionally, we will incorporate the variable "Population, total" as the size of the points on the graph.*

In [ ]:
```
chile = world[world["Country Name"] == "Chile"]
```

In [ ]:
```
(
    so.Plot(
        chile,
        x="Year",
        y="GDP per capita (constant 2010 US$)",
        pointsize="Population, total",
    )
    .add(so.Dot())
    .label(title="Evolution of GDP per Capita in Chile (1971-2020)")
)
```

Out[ ]:

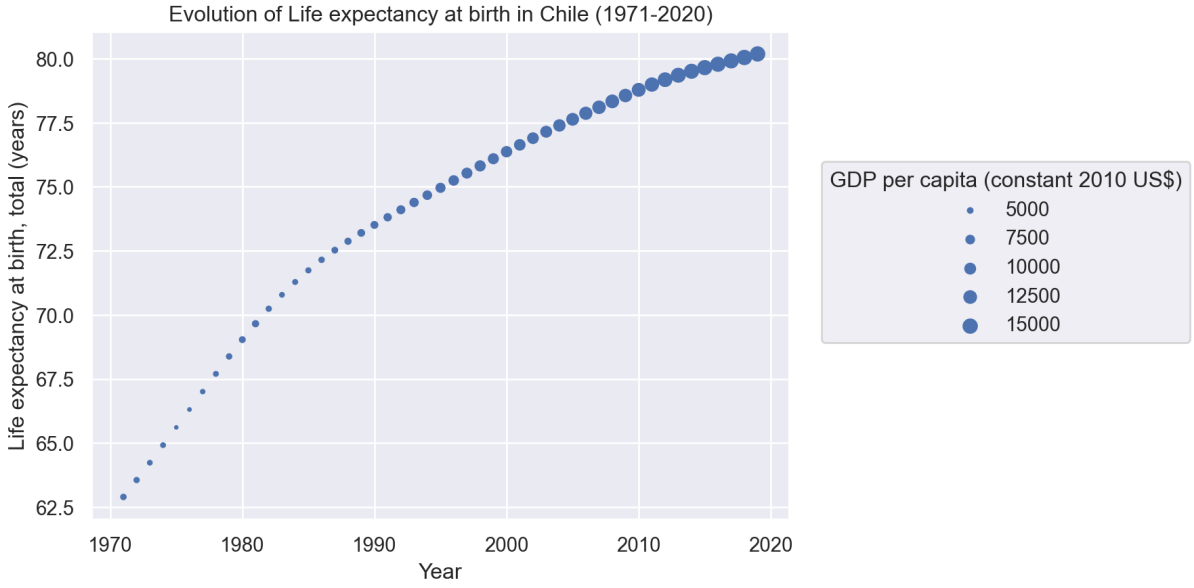


> *Another interesting variable to assess over time is the evolution of Life Expectancy at Birth, total (years). This will allow us to analyze how it relates to GDP per capita (constant 2010 US$)*

In [ ]:

```
(
    so.Plot(
        chile,
        x="Year",
        y="Life expectancy at birth, total (years)",  # GDP per capita (cons
        pointsize="GDP per capita (constant 2010 US$)",
    )
    .add(so.Dot())
    .label(title="Evolution of Life expectancy at birth in Chile (1971-2020)
)
```

Out[ ]:



> *Based on the previous graphs, we can observe that from 1971 to 2020, Chile has experienced a relatively constant evolution of its GDP per Capita, with some declines in certain crisis periods. In the recent period, we can see the effects of the 2008 crisis and the impact of the 2020 pandemic, in addition to the effects of the so-called 'social outbreak' in Chile. This was an event of protests and disturbances that took place mainly between October 2019 and March 2020, and it also affected the country's economy.*
>
> *In addition to the evolution of GDP per Capita, we can observe that Life Expectancy at Birth has consistently increased, going from around 63 years in 1971 to surpassing 80 years in 2020.*
>
> *At first glance, it can be seen that these two variables are related to the level of development of the country. The sustained increase in life expectancy could suggest an improvement in living conditions and healthcare services over time, which is often associated with higher economic development*

Exercise 7

Now add a second series. Facet your plot so that the two subplots are positioned so that they are effectively sharing the same time axes (e.g., if you draw a line up from 2010 on one plot, you get to 2010 on the other).

Rather than telling you exactly how to do it, however, I'll point you to the seaborn tutorial. It has examples that don't do exactly what you want, but should be close enough you can guess-and-check to the solution you want!

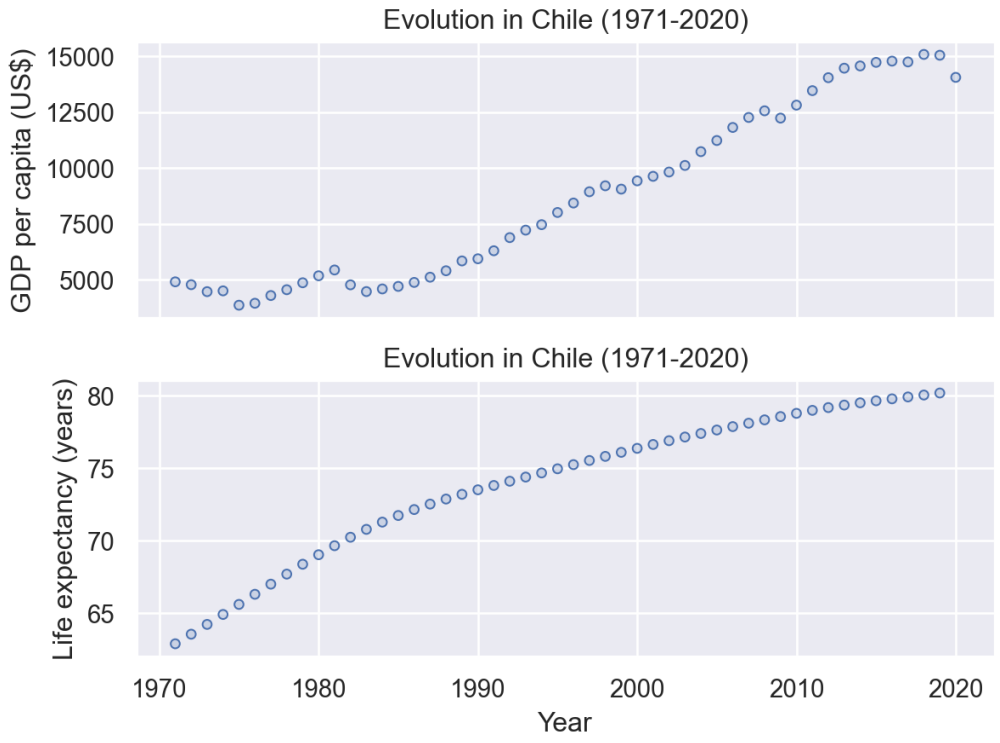
Use your detective skills (and some guess and check work) to figure out how to get it to work!

> *"In the following graph, you can observe both variables analyzed previously, GDP per capita (constant 2010 US dollars), and Life expectancy at birth, total (years). This facilitates the visualization of how both variables behave over time*

In [ ]:
```
chile.rename(
    columns={
        "GDP per capita (constant 2010 US$)": "GDP per capita (US$)",
        "Life expectancy at birth, total (years)": "Life expectancy (years)"
    },
    inplace=True,
)
```

In [ ]:
```
(
    so.Plot(chile, x="Year")
    .pair(
        y=[
            "GDP per capita (US$)",
            "Life expectancy (years)",
        ]
    )
    .add(so.Dots())
    .label(title="Evolution in Chile (1971-2020)")
)
```

Out[ ]:

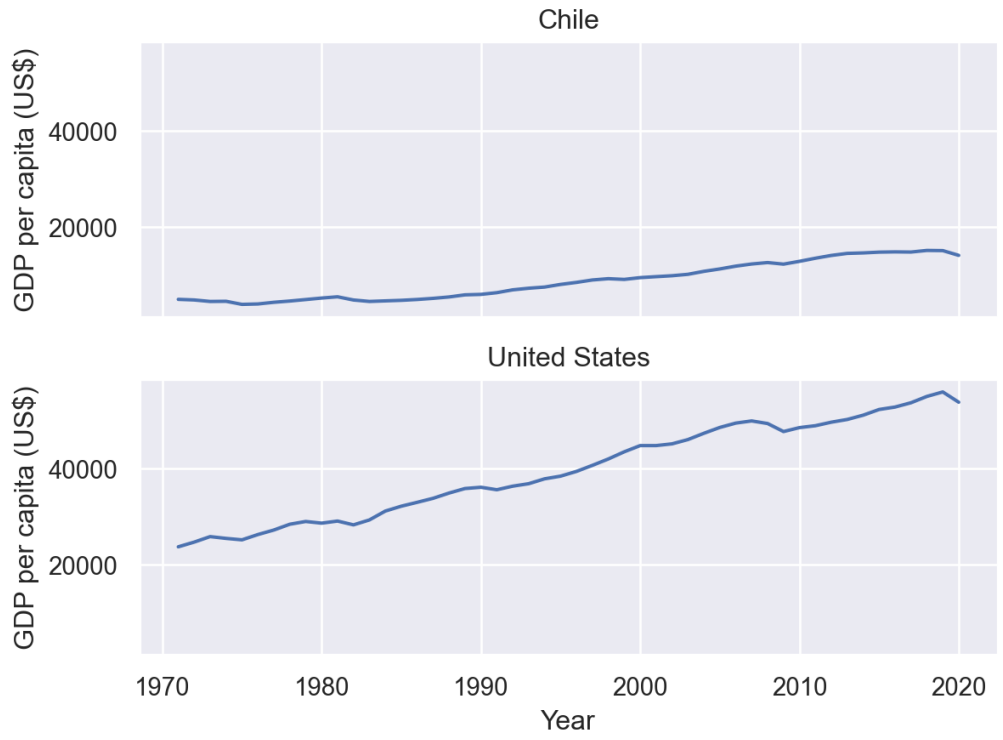


> *On the other hand, if we wanted to compare the evolution of GDP per capita (constant 2010 US dollars) over time in Chile with another country, such as the United States, we could visualize it using the 'facet' function.*

In [ ]:
```
chile_usa = world[(world["Country Name"] == "Chile") | (world["Country Code"

chile_usa.rename(
    columns={
        "GDP per capita (constant 2010 US$)": "GDP per capita (US$)",
        "Life expectancy at birth, total (years)": "Life expectancy (years)"
    },
    inplace=True,
)
```

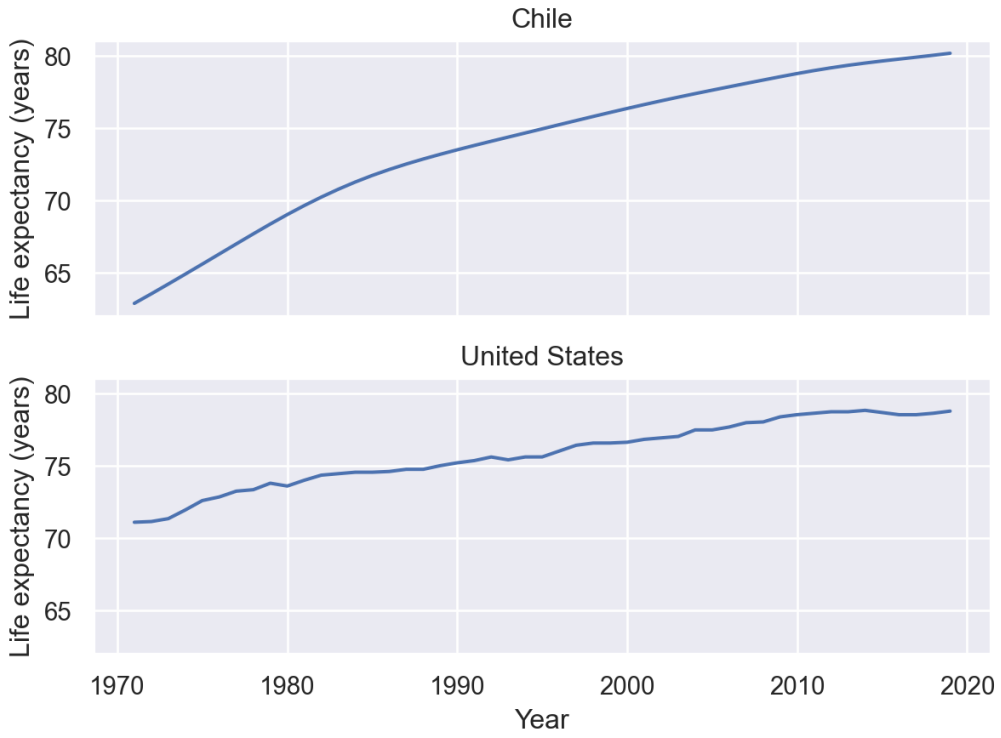
In [ ]:
```
(
    so.Plot(chile_usa, x="Year", y="GDP per capita (US$)")
    .facet(row="Country Name", wrap=3)
    .add(so.Line())
)
```

Out[ ]:



In [ ]:
```
(
    so.Plot(chile_usa, x="Year", y="Life expectancy (years)")
    .facet(row="Country Name", wrap=3)
    .add(so.Line())
)
```

Out[ ]:



> *After analyzing the previous graphs, we can draw the following conclusions:*
>
> *1. As mentioned earlier, there appears to be a correlation between the variables "Life expectancy at birth, total (years)" and "GDP per capita (constant 2010 US$)," suggesting a relationship between economic prosperity and life expectancy.*
>
> *2. When comparing Chile with the United States, we observe that both countries have seen growth in both indicators over the last 50 years. However, the growth in GDP per capita in the United States has been significantly higher, with a decrease in 2020, similar to Chile, due to the COVID-19 pandemic.*
>
> *3. Despite Chile having a lower GDP per capita than the United States, it achieves a higher life expectancy in 2020. This suggests that other factors influence this variable, and once a certain threshold of poverty and development is surpassed, other factors besides economic development need to be considered.*