Plotting I - due 10/05 *by Clarissa and Charlotte*

Exercise 1

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

In [ ]:
```
import pandas as pd
import numpy as np
import altair as alt

df = pd.read_csv(
    "https://raw.githubusercontent.com/nickeubank/practicaldatascience"
    "/master/Example_Data/Datasaurus.txt",
    delimiter="\t",
)
df
```

Out[ ]:

	example1_x	example1_y	example2_x	example2_y	example3_x	example3_y	exampl
0	32.331110	61.411101	51.203891	83.339777	55.993030	79.277264	55.3
1	53.421463	26.186880	58.974470	85.499818	50.032254	79.013071	51.5
2	63.920202	30.832194	51.872073	85.829738	51.288459	82.435940	46.
3	70.289506	82.533649	48.179931	85.045117	51.170537	79.165294	42.8
4	34.118830	45.734551	41.683200	84.017941	44.377915	78.164628	40.1
...	...	...	...	...	...	...	
137	59.851838	72.958391	50.967748	29.679774	39.921363	19.701850	39.4
138	48.960460	72.629526	91.191054	46.674343	84.794278	55.568650	91.
139	46.844855	36.791714	55.863768	85.336487	55.662959	83.356480	50.(
140	39.963022	42.944915	49.280595	84.048823	50.492248	78.997532	47.9
141	66.704944	32.015095	43.368502	84.332177	51.467101	79.201845	44.

142 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.

In [ ]:

```
line_break = "\n"
for i in range(1,14):
    print(f"Example Dataset {i}")
    print("Mean of x")
    print(df.loc[:, f"example{i}_x"].mean().round(2))
    print("Mean of y")
    print(df.loc[:, f"example{i}_y"].mean().round(2))
    print("Standard deviation of x")
    print(df.loc[:, f"example{i}_x"].std().round(2))
    print("Standard deviation of y")
    print(df.loc[:, f"example{i}_y"].std().round(2))
    print(line_break)
```

```
Example Dataset 1
Mean of x
54.27
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 2
Mean of x
54.27
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 3
Mean of x
54.27
Mean of y
47.84
Standard deviation of x
16.76
Standard deviation of y
```

```
26.93


Example Dataset 4
Mean of x
54.26
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 5
Mean of x
54.26
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.93


Example Dataset 6
Mean of x
54.26
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 7
Mean of x
54.27
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 8
Mean of x
54.27
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.94
```

```
Example Dataset 9
Mean of x
54.27
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 10
Mean of x
54.27
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.93


Example Dataset 11
Mean of x
54.27
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 12
Mean of x
54.27
Mean of y
47.83
Standard deviation of x
16.77
Standard deviation of y
26.94


Example Dataset 13
Mean of x
54.26
Mean of y
47.84
Standard deviation of x
16.77
Standard deviation of y
26.93
```

Exercise 3

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

Based only on the summary statistics calculated (mean and standard deviation), the datasets look very similar. Those summary stats only differ marginally by a few decimals, if at all.

Exercise 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.

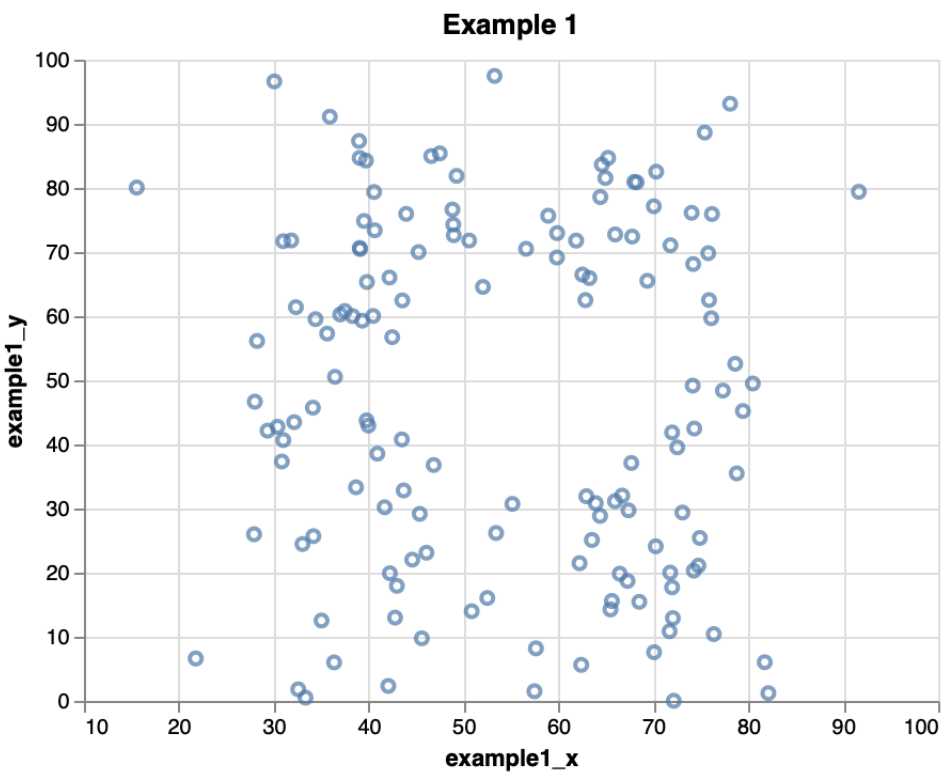
Hint 2: To force Jupyter to display your charts when they're generated within a loop, use the method .display() (e.g. my_chart.display()).

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

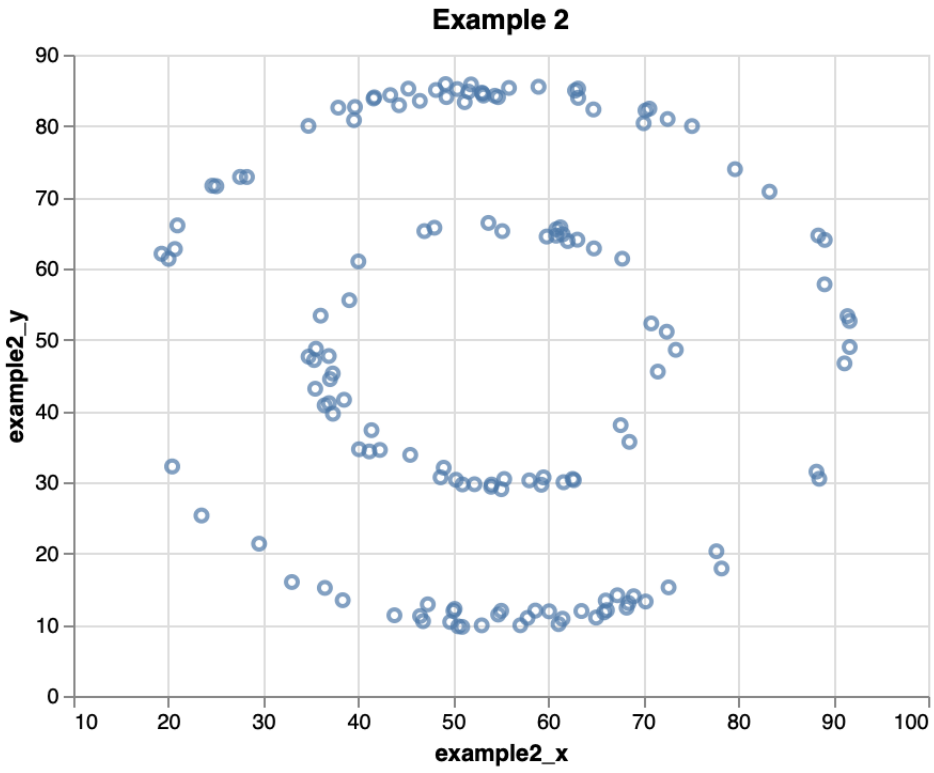
In [ ]:

```
for i in range(1, 14):
    print(f"Visual representation of the Dataset {i}")
    c=(alt.Chart(df, title=f'Example {i}').mark_point().encode(x=alt.X(f"ex
    )
    c.display()
```

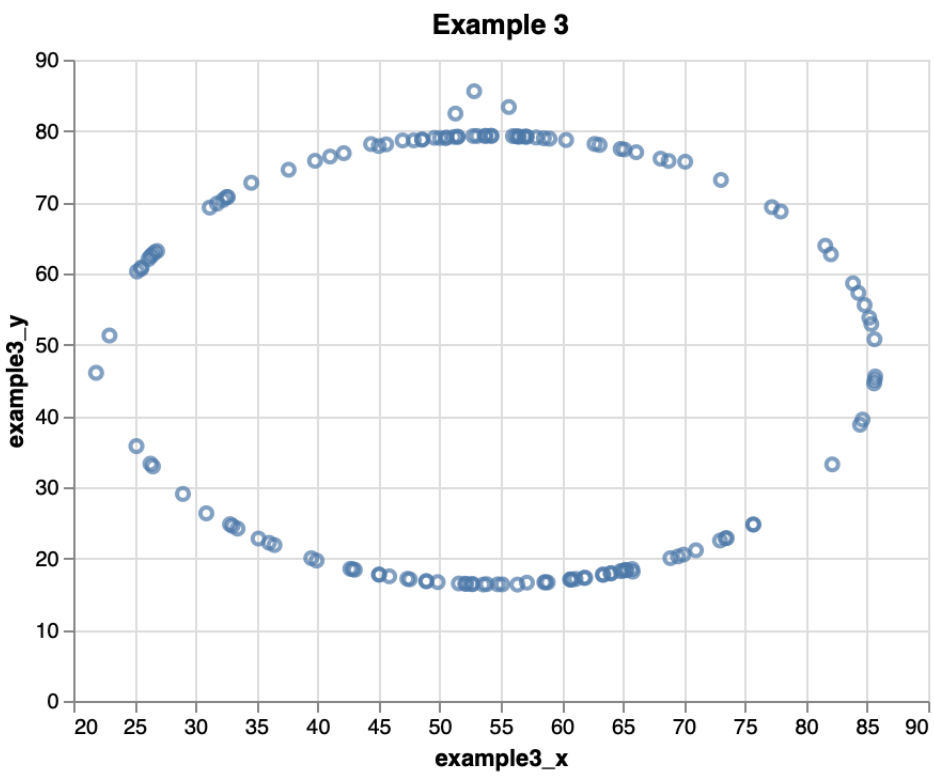
```
Visual representation of the Dataset 1
```



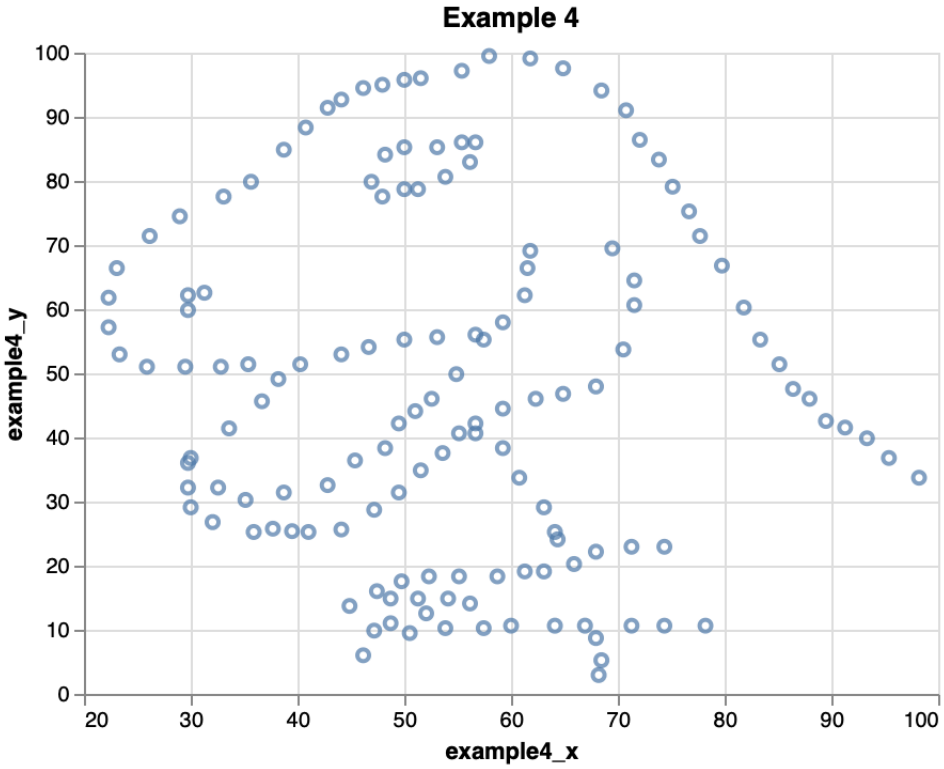
Visual representation of the Dataset 2



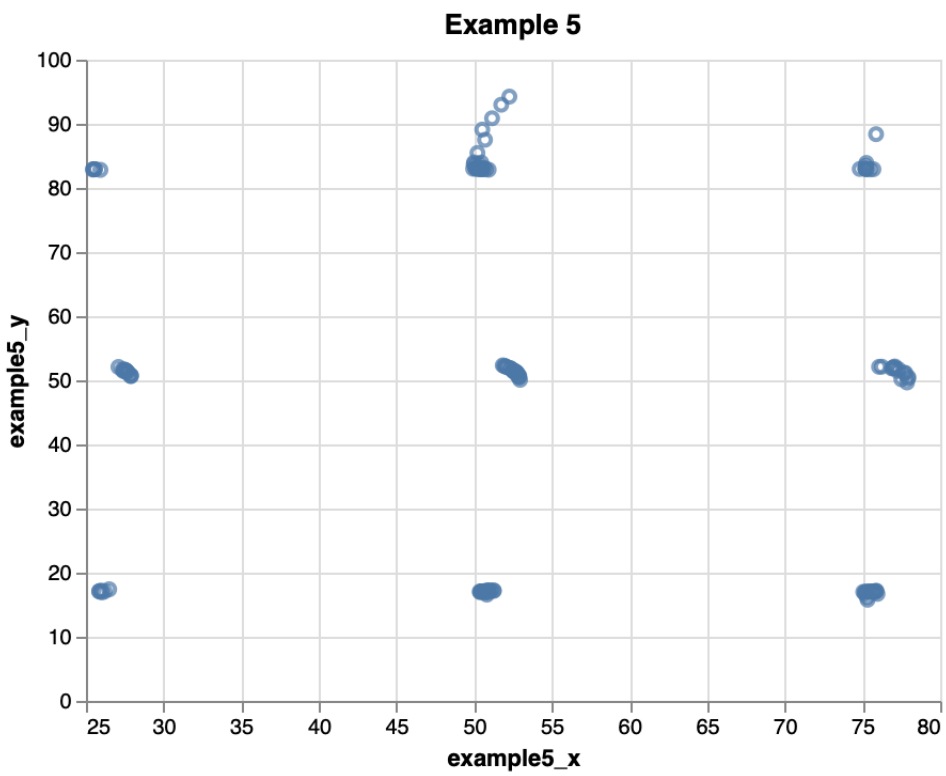
Visual representation of the Dataset 3



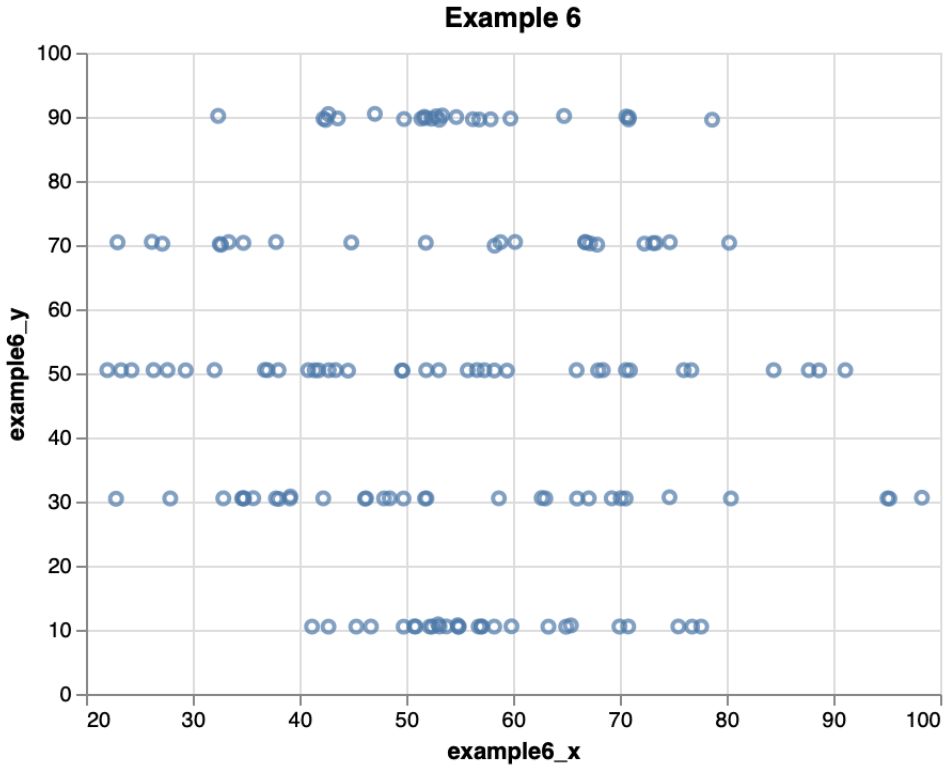
Visual representation of the Dataset 4



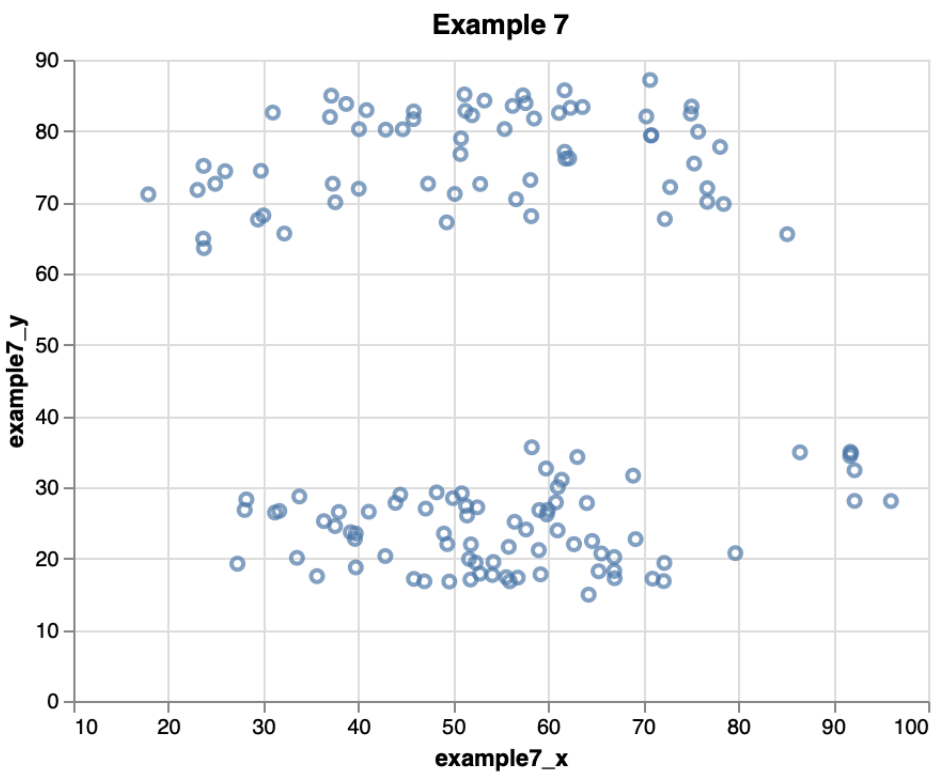
Visual representation of the Dataset 5



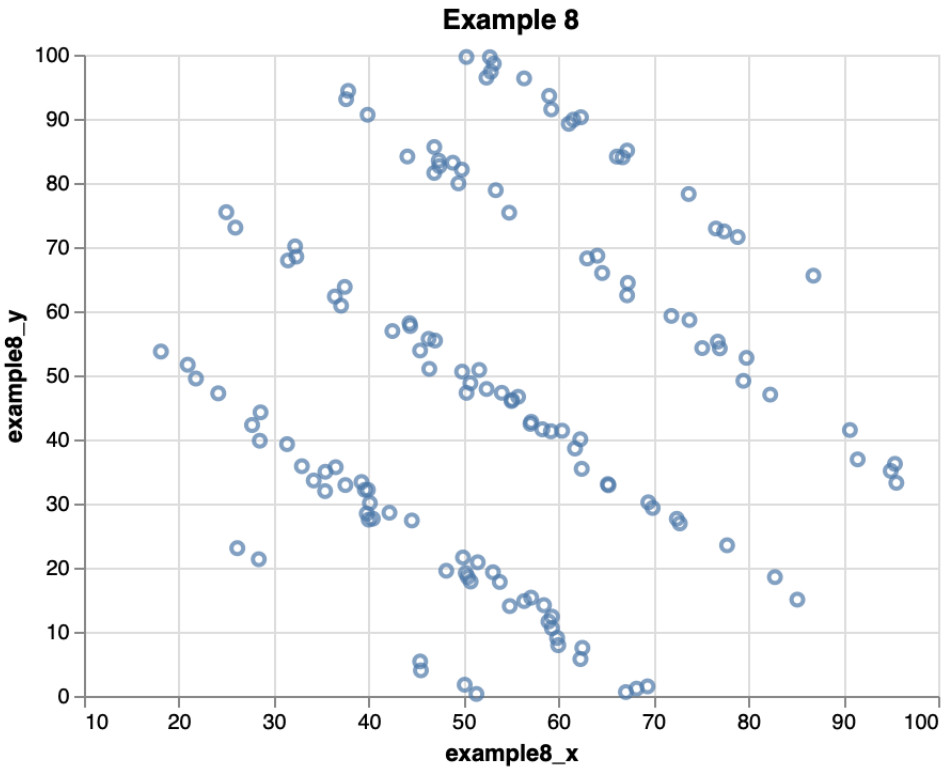
Visual representation of the Dataset 6



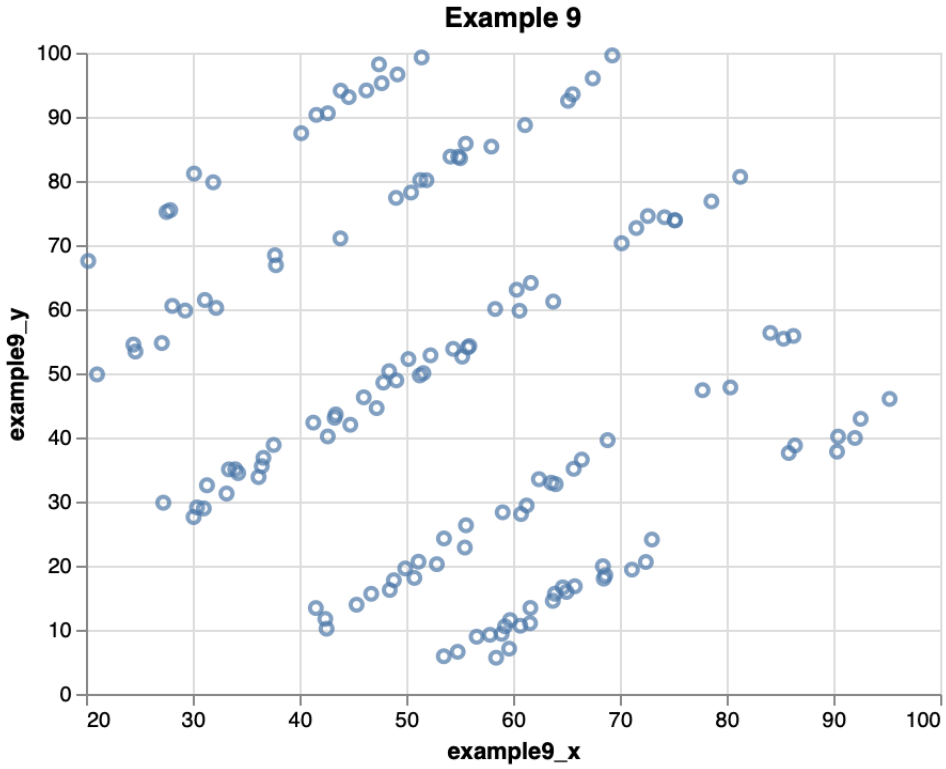
Visual representation of the Dataset 7



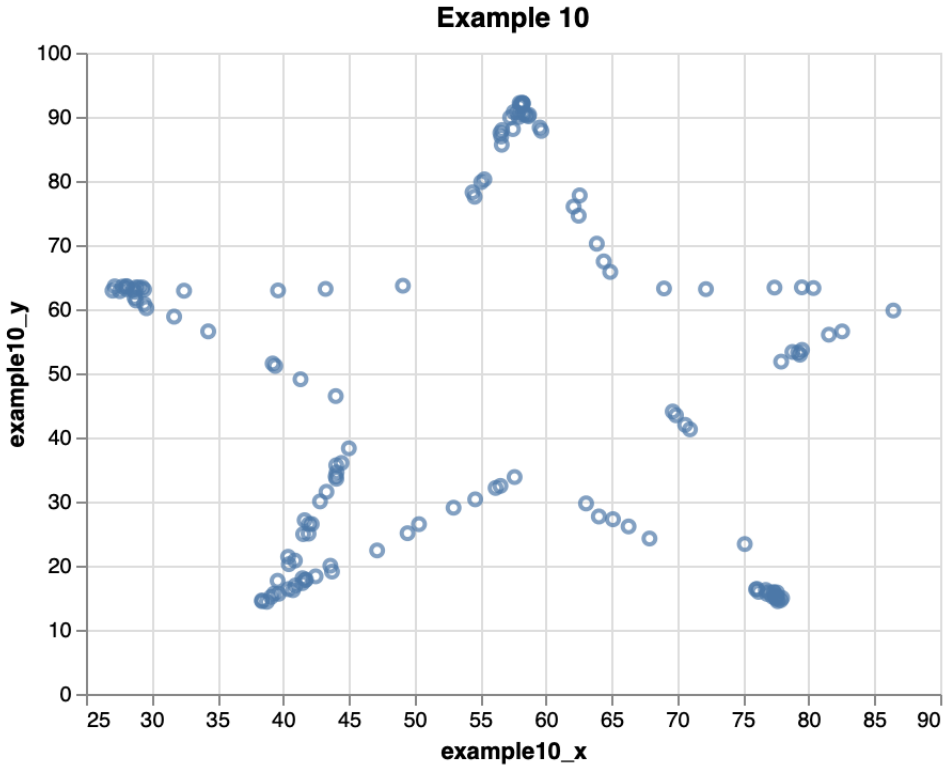
Visual representation of the Dataset 8



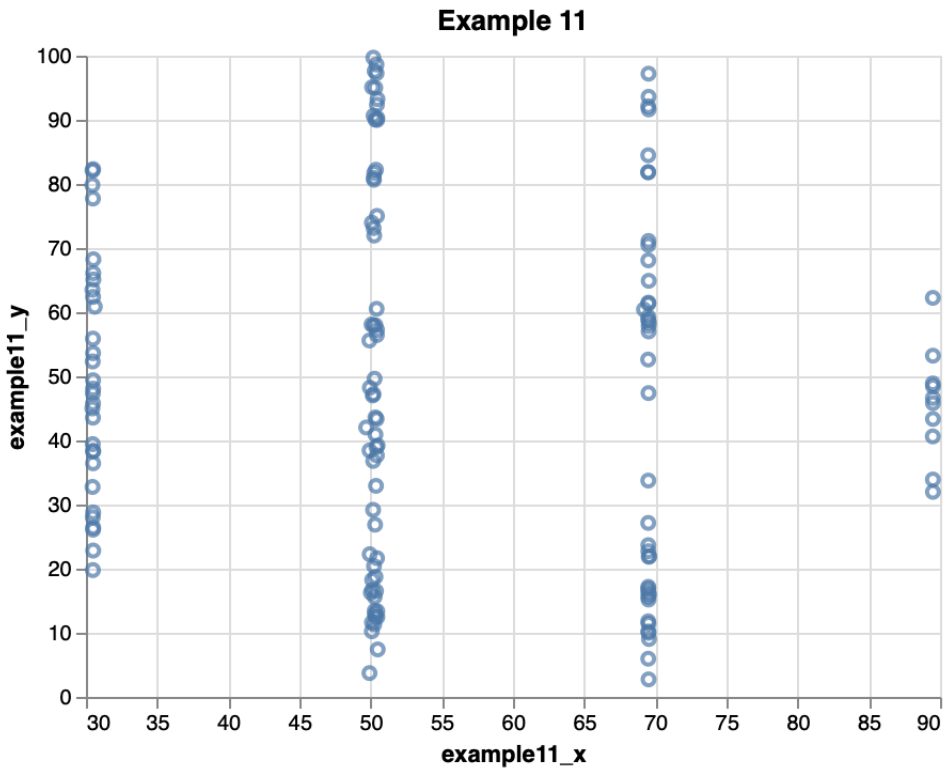
Visual representation of the Dataset 9



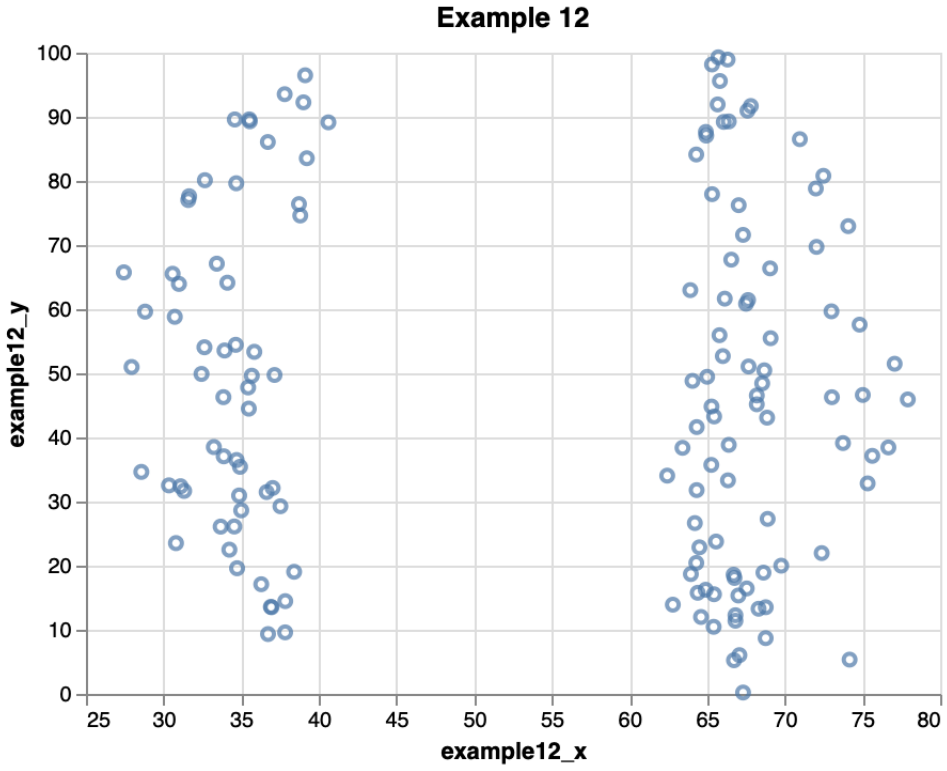
Visual representation of the Dataset 10



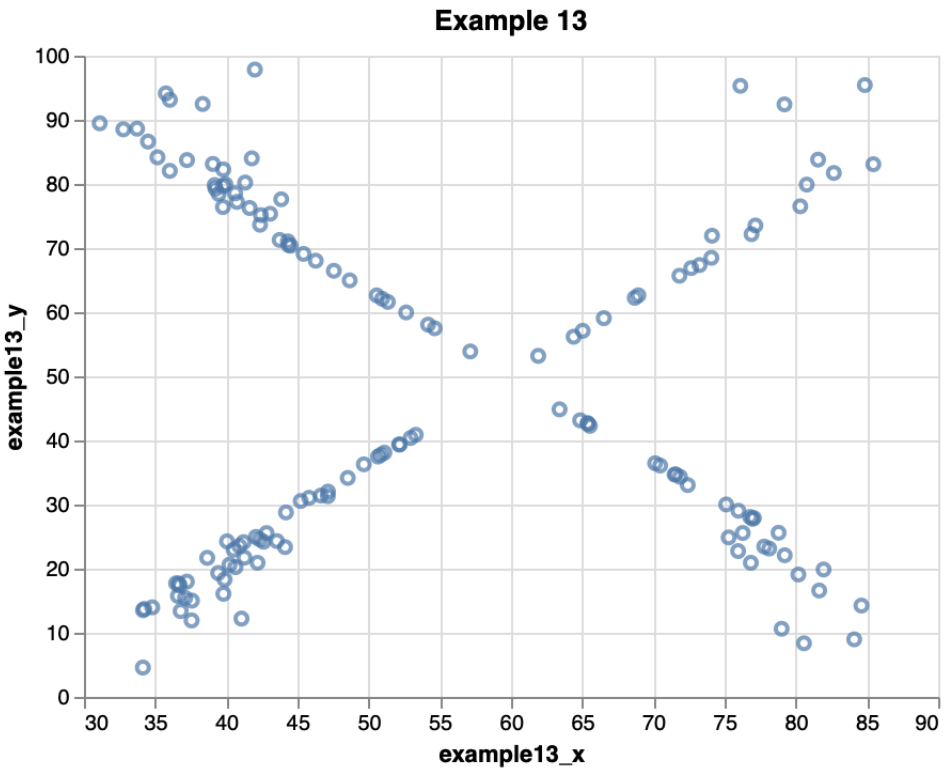
Visual representation of the Dataset 11



Visual representation of the Dataset 12



Visual representation of the Dataset 13



Exercise 5

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

We can conclude that the summary statistics were misleading as in fact the datasets seem to be very different in terms of x and y- value distribution (judging from what we see now in the visual representation).

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/practicaldatascie
world = pd.read_csv(wdi_data)
world.sample(5)
```

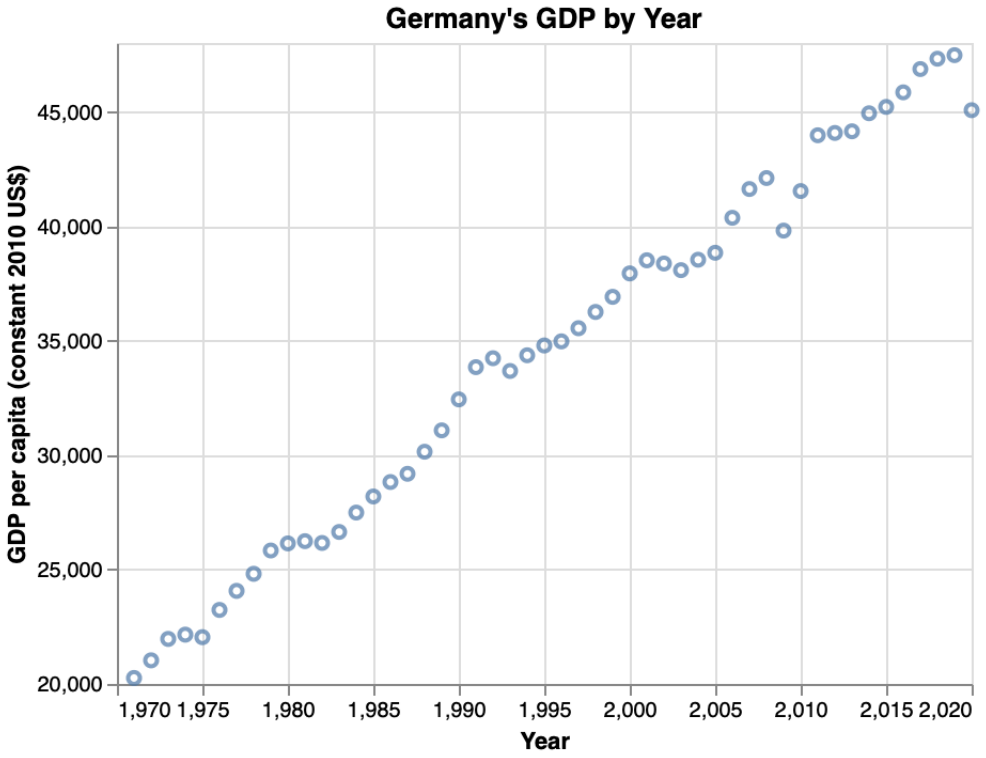
Out[ ]:

	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)	PM pc pop e t exc gu v c
3599	1987	Mexico	MEX	7445.853690	79200081.0	3.880765	NaN	
10571	2019	Philippines	PHL	3340.132670	108116622.0	NaN	NaN	
9356	2014	Botswana	BWA	7864.232154	2088619.0	3.413739	NaN	
5629	1996	United Arab Emirates	ARE	63076.730670	2539121.0	29.856789	NaN	
4306	1990	St. Martin (French part)	MAF	NaN	31522.0	NaN	NaN	

In [ ]:
```
world = world[world.loc[:,'Country Name'] == "Germany"]
alt.Chart(world, title="Germany's GDP by Year").mark_point().encode(
    x=alt.X("Year", scale=alt.Scale(zero=False)),
    y=alt.Y("GDP per capita (constant 2010 US$)", scale=alt.Scale(zero=Fals
)
```

Pick a country

Out[ ]:

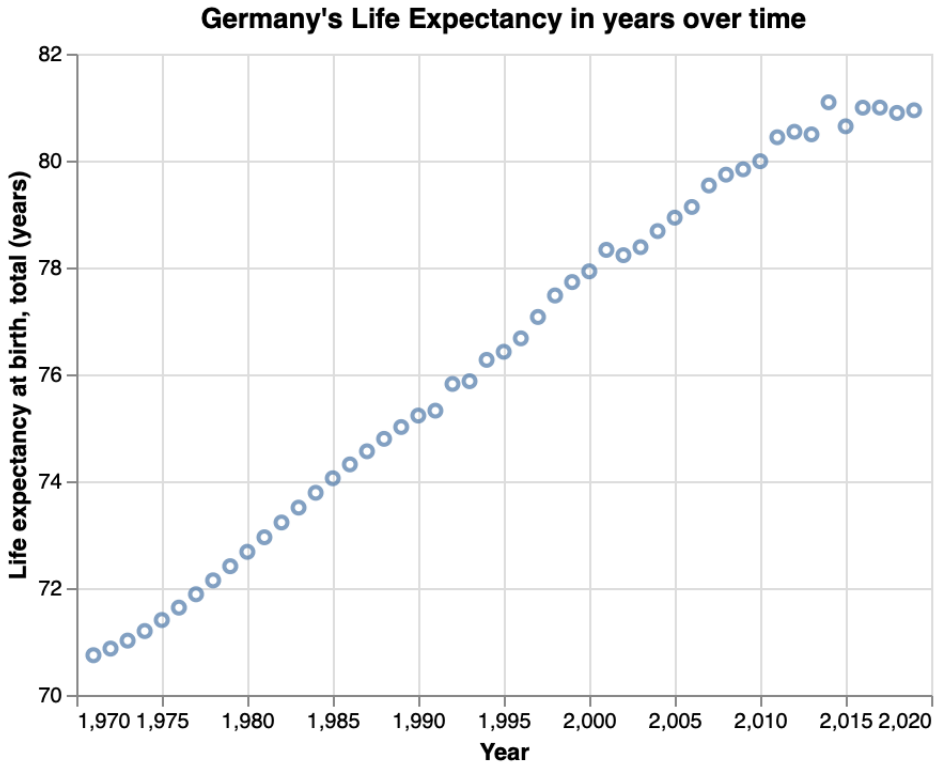


In [ ]:

```
world = world[world.loc[:,'Country Name'] == "Germany"]
alt.Chart(world, title="Germany's Life Expectancy in years over time").mark
    x=alt.X("Year", scale=alt.Scale(zero=False)),
    y=alt.Y("Life expectancy at birth, total (years)", scale=alt.Scale(zero
)
```

Pick a country

Out[ ]:



Exercise 7 & Exercise 8

Now add another series to the plot, so you now have two series on the same plot. Make sure to differentiate them so one can see that they are different series!

Because your two series will probably be on different scales, you can't just layer your plots with the simple + operator. Instead, use alt.layer() method.

Rather than telling you exactly how to do it, however, I'll point you to one of the charts in the Altair Example gallery that has overlapping series with different scales: here. Use your detective skills (and some guess and check work) to figure out how to get it to work!

Give your chart and axes meaningful (and well formatted!) titles.

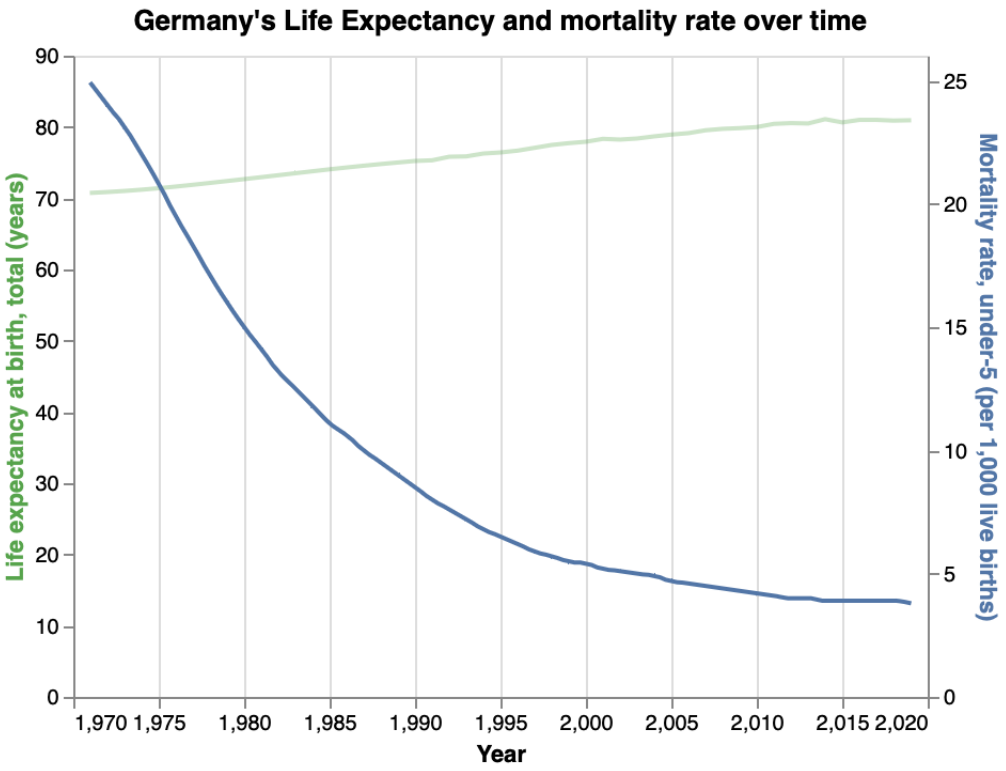
In [ ]:

```
world = world[world.loc[:,'Country Name'] == "Germany"]
base = alt.Chart(world, title="Germany's Life Expectancy and mortality rate
    alt.X('Year', scale=alt.Scale(zero=False), axis=alt.Axis(title='Year'))
)
area = base.mark_line(opacity=0.3, color='#57A44C').encode(
    alt.Y('Life expectancy at birth, total (years)',
          axis=alt.Axis(title='Life expectancy at birth, total (years)', ti
)

line = base.mark_line(stroke='#5276A7', interpolate='monotone').encode(
    alt.Y('Mortality rate, under-5 (per 1,000 live births)',
          axis=alt.Axis(title='Mortality rate, under-5 (per 1,000 live birt
)

alt.layer(area, line).resolve_scale(
    y = 'independent'
)
```

Out[ ]:


Germany's Life Expectancy and mortality rate over time
Life expectancy at birth, total (years)
Mortality rate, under-5 (per 1,000 live births)
Year
0
10
20
30
40
50
60
70
80
90
0
5
10
15
20
25
1,970
1,975
1,980
1,985
1,990
1,995
2,000
2,005
2,010
2,015
2,020