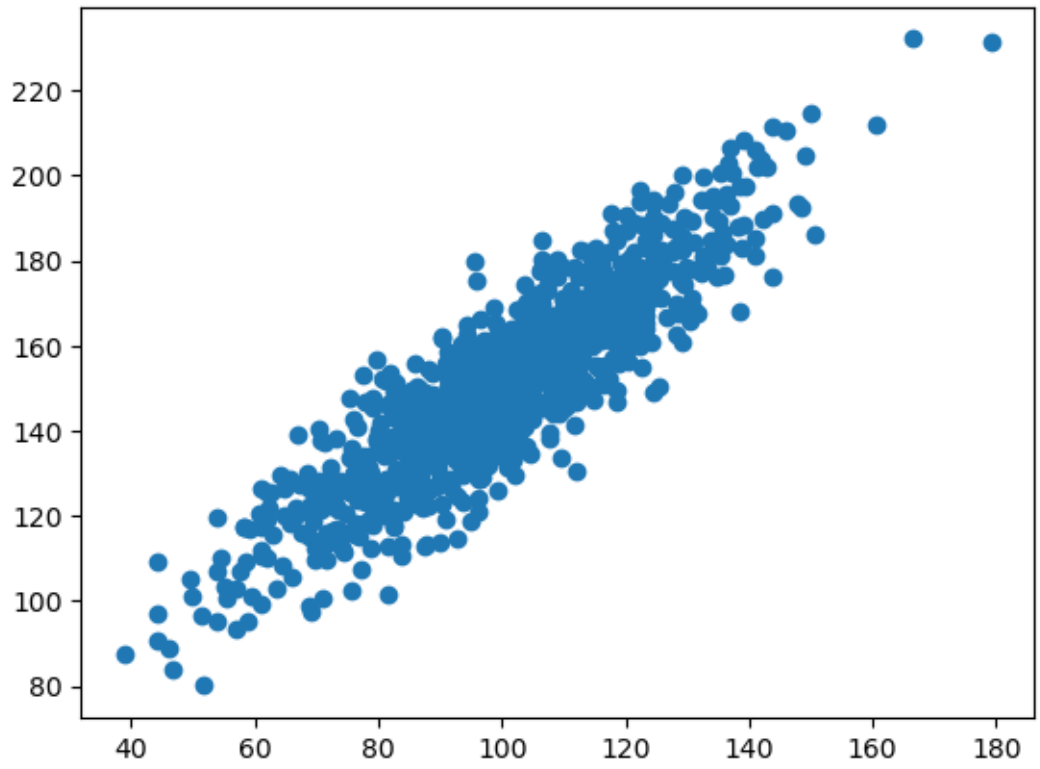
Unit 3

Correlation and Regression activity

This activity required running four programs on a Jupyter notebook. Variables needed to be changed to observe how the change in data points impacts correlation and regression.

Pearson's correlation:

The initial findings of the code presented correlation of 0.888 using the given data1 and data 2.



data1: mean=100.776 stdv=19.620

data2: mean=151.050 stdv=22.358

Covariance: 389.755

Pearsons correlation: 0.888

This shows a strong positive correlation with a positive covariance suggesting both variables are moving in the same direction.

I will first assess both data sets:

Data1 and data2 are both arrays contain a 1000 datapoints each.

In [13]:
```
count = 0
for dat in data1:
    count = count+1
print(f"data1 = {count}")
count = 0
for dat in data2:
    count = count + 1
print(f"data2 = {count}")
```

```
data1 = 1000
data2 = 1000
```

I will combine both arrays into one table, separated into two columns labelled A and B.

In [15]:
```
df = pd.DataFrame({'ColumnA': data1, 'ColumnB': data2})
df
```

Out[15]:

	ColumnA	ColumnB
0	132.486907	180.954546
1	87.764872	113.439787
2	89.436565	144.516408
3	78.540628	125.300304
4	117.308153	152.197387
...	...	...
995	97.671117	149.556946
996	54.454040	110.063221
997	98.607509	139.390919
998	107.077409	163.551160
999	96.260900	160.129156

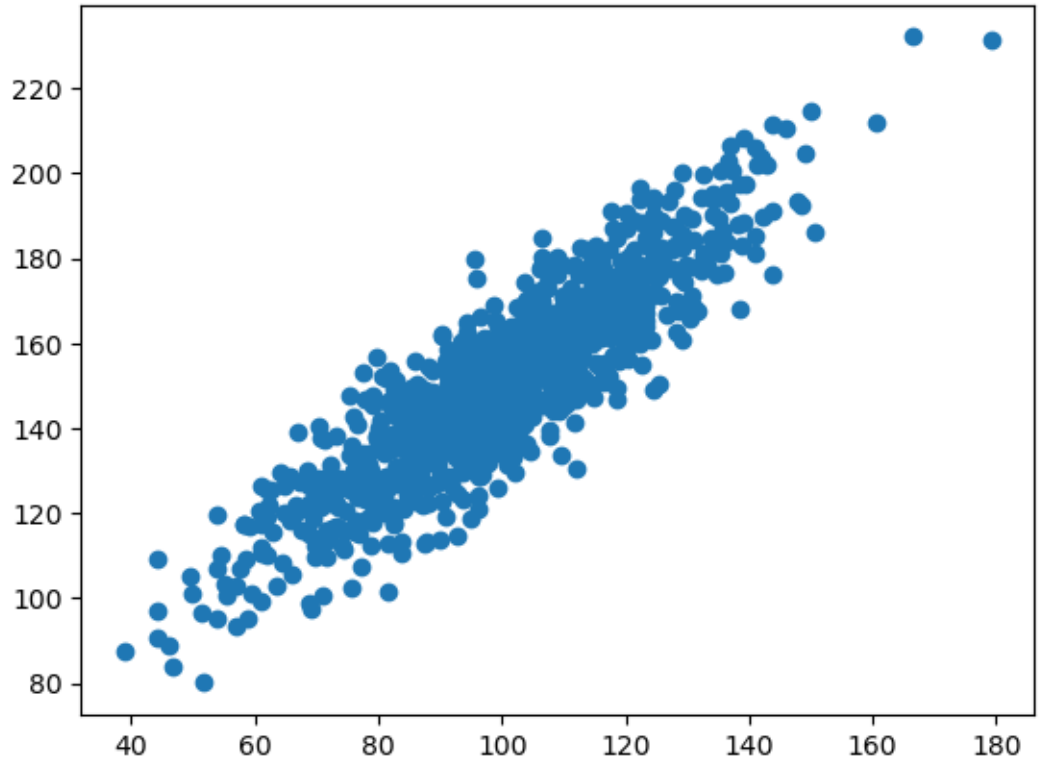
1000 rows × 2 columns

Now I will test to see how changing datapoints will effect the correlation coefficients

I changed one specific datapoint In columnA from 87.764872 to 31.

```
import matplotlib.pyplot as plt
df.loc[df['ColumnA'] == 87.764872, 'ColumnA'] = 31

# calculate covariance matrix
covariance = cov(df['ColumnA'], df['ColumnB'])

# calculate Pearson's correlation
corr, _ = pearsonr(df['ColumnA'], df['ColumnB'])

# plot
plt.scatter(df['ColumnA'], df['ColumnB'])
plt.show()

# summarize
print('data1: mean=%.3f stdv=%.3f' % (mean(df['ColumnA']), std(df['ColumnA'])))
print('data2: mean=%.3f stdv=%.3f' % (mean(df['ColumnB']), std(df['ColumnB'])))
print('Covariance: %.3f' % covariance[0][1])
print('Pearsons correlation: %.3f' % corr)
```

The result was no significant change.



data1: mean=100.776 stdv=19.620

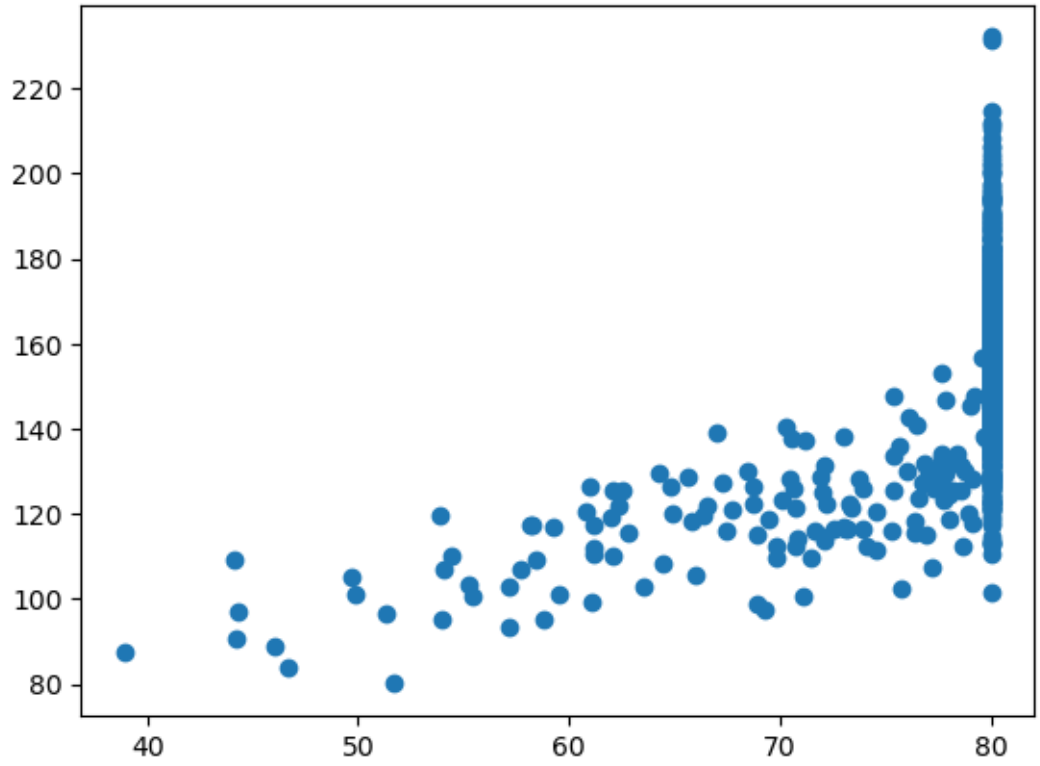
data2: mean=151.050 stdv=22.358

Covariance: 389.755

Pearsons correlation: 0.888

I now changed the code so any value over 80 in columnA will be changed to 80.

```
import matplotlib.pyplot as plt
df.loc[df['ColumnA'] > 80, 'ColumnA'] = 80

# calculate covariance matrix
covariance = cov(df['ColumnA'], df['ColumnB'])

# calculate Pearson's correlation
corr,_ = pearsonr(df['ColumnA'], df['ColumnB'])

# plot
plt.scatter(df['ColumnA'], df['ColumnB'])
plt.show()

# summarize
print('Column A: mean=%.3f stdv=%.3f' % (mean(df['ColumnA']), std(df['ColumnA'])))
print('COlumn B: mean=%.3f stdv=%.3f' % (mean(df['ColumnB']), std(df['ColumnB'])))
print('Covariance: %.3f' % covariance[0][1])
print('Pearsons correlation: %.3f' % corr)
```



Column A: mean=78.447 stdv=5.216

Column B: mean=151.050 stdv=22.358

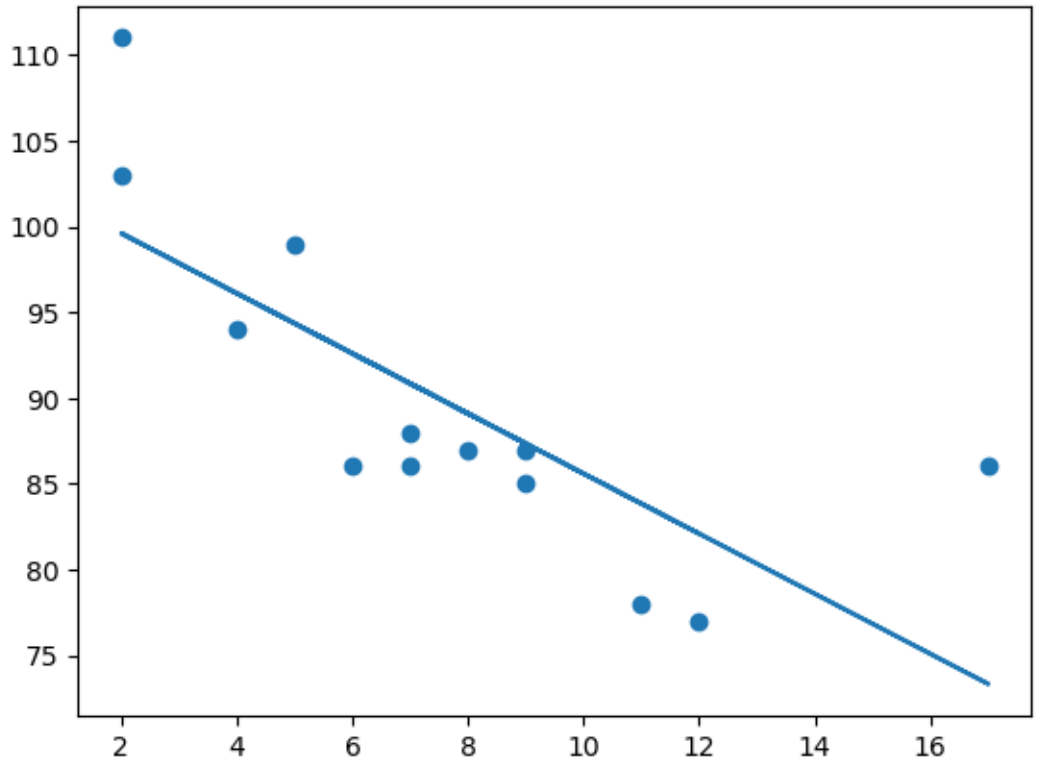
Covariance: 61.518

Pearsons correlation: 0.527

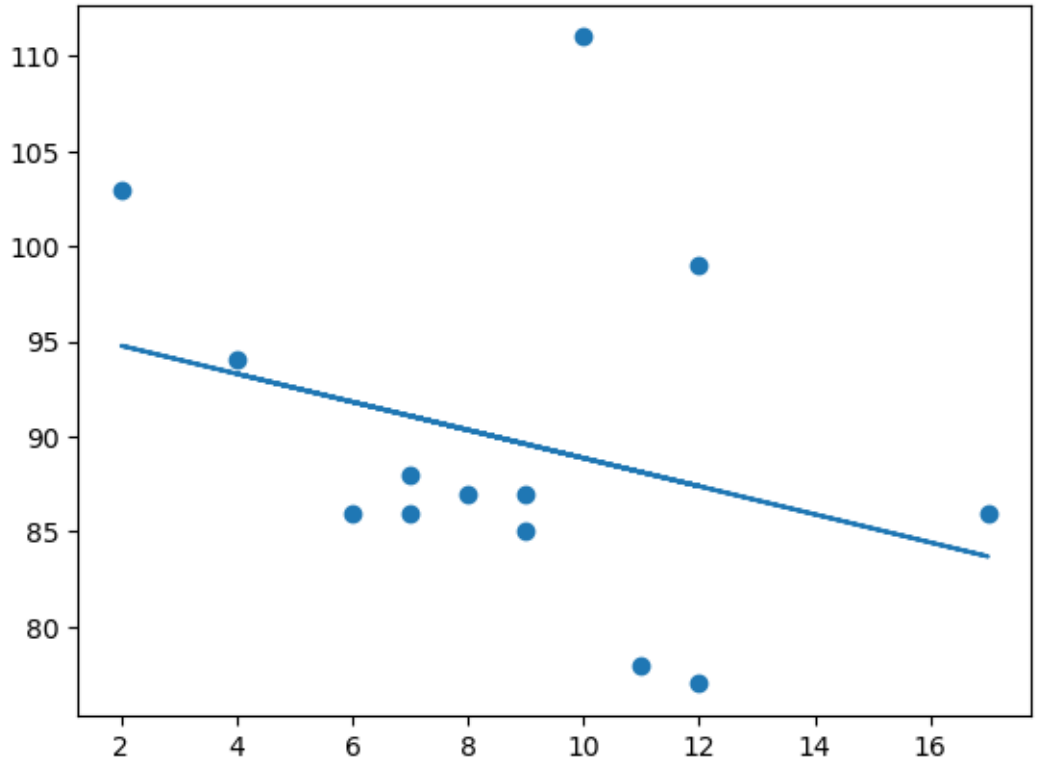
The correlation has dropped to 0.527 showing that the data has a very weak positive correlation. The covariance is still positive suggesting that the variables are moving in the same direction.

Linear regression:

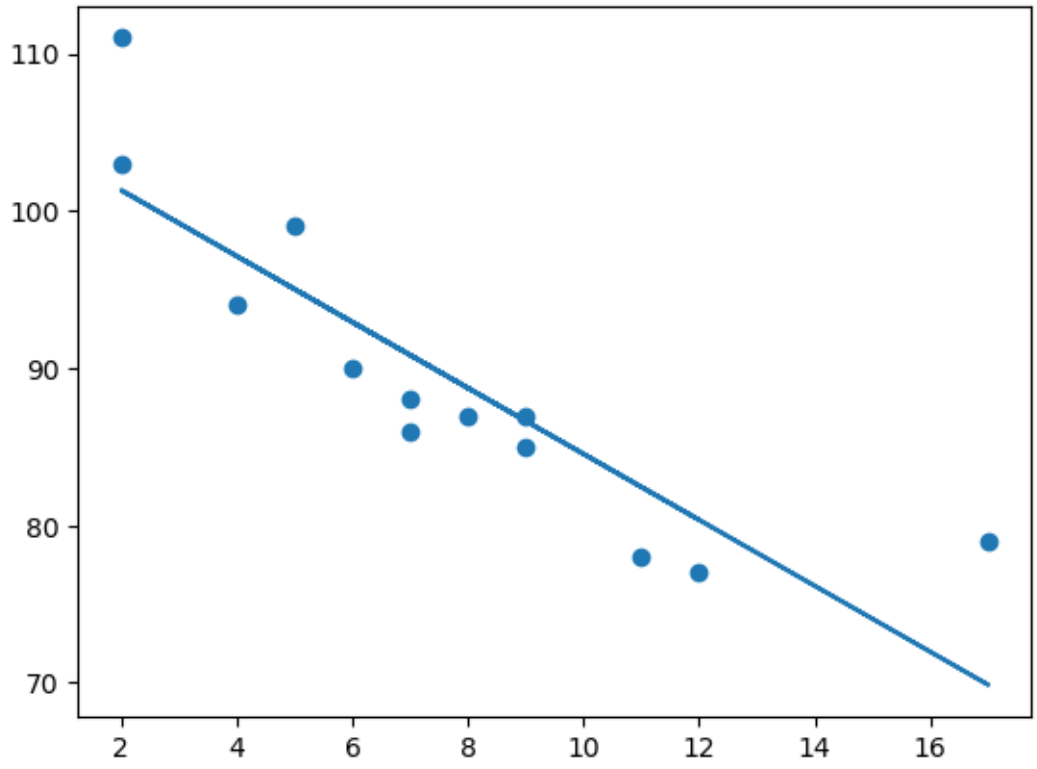
The original Pearson's correlation = -0.759 : strong negative correlation.



After changing a couple of data points the correlation dropped down to -0.295. This is much weaker than the original correlation.



I changed some data points to form a stronger correlation. The result was Pearson's correlation: -0.875



This exercise demonstrated how altering just a few data points can significantly impact the correlation in a scatterplot. Observing how potential outliers influence the overall correlation was insightful, highlighting that in real datasets, outliers can sometimes skew results in a way that could be misleading. This underscores the importance of carefully considering whether to retain or remove outliers, as they might hold valuable insights that are essential for accurate data interpretation.