Part - 01

Quantitative comparisons: bar-charts

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

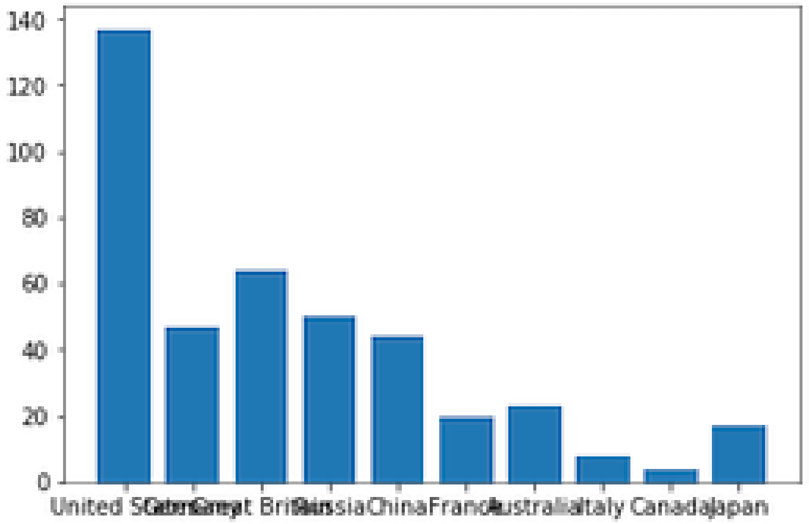
Ariel Rokem
Data Scientist

Olympic medals

```
,Gold, Silver, Bronze
United States, 137, 52, 67
Germany, 47, 43, 67
Great Britain, 64, 55, 26
Russia, 50, 28, 35
China, 44, 30, 35
France, 20, 55, 21
Australia, 23, 34, 25
Italy, 8, 38, 24
Canada, 4, 4, 61
Japan, 17, 13, 34
```

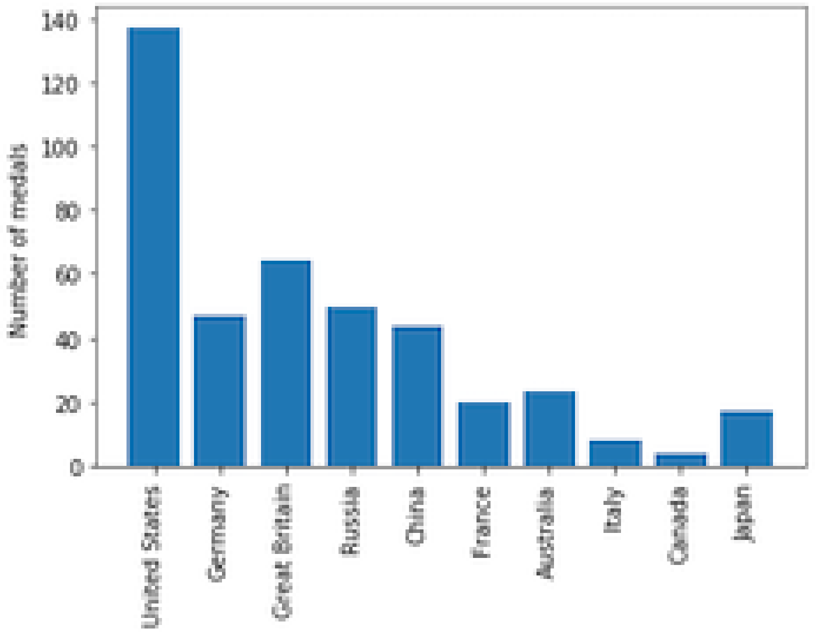
Olympic medals: visualizing the data

```
medals = pd.read_csv('medals_by_country_2016.csv', index_col=0)
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])
plt.show()
```



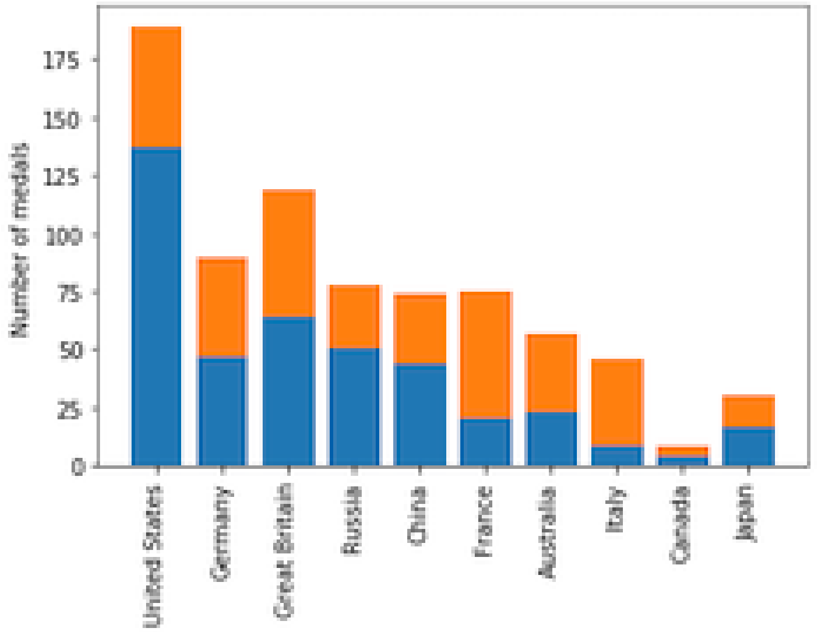
Interlude: rotate the tick labels

```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])
ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
plt.show()
```



Olympic medals: visualizing the other medals

```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])
ax.bar(medals.index, medals["Silver"], bottom=medals["Gold"])
ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
plt.show()
```



Olympic medals: visualizing all three

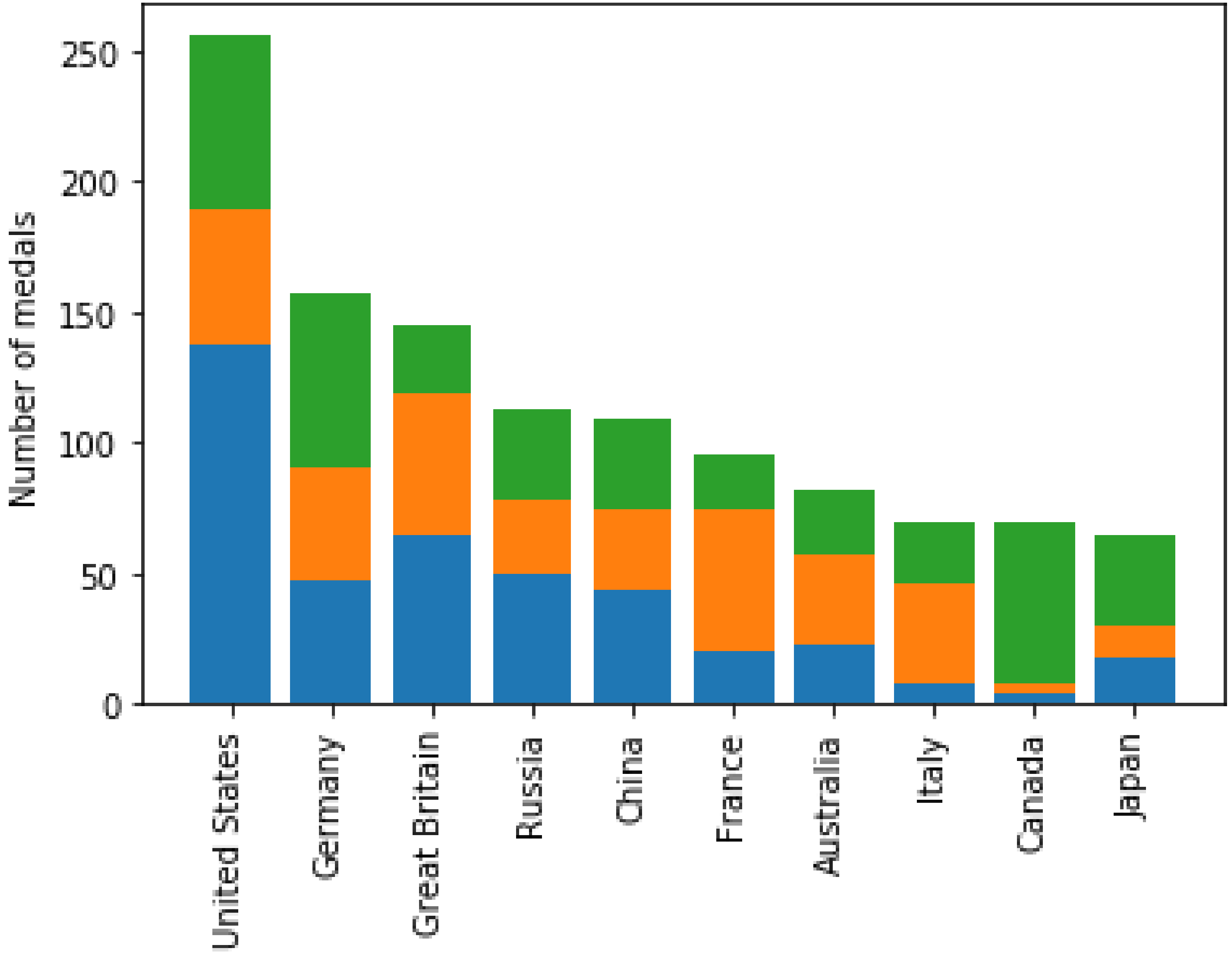
```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])

ax.bar(medals.index, medals["Silver"], bottom=medals["Gold"])
ax.bar(medals.index, medals["Bronze"],
       bottom=medals["Gold"] + medals["Silver"])
ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
plt.show()
```

starts from bottom

it means the silver bar start from height of "Gold"

the bronze bar starts from the height of gold+silver

Stacked bar chart



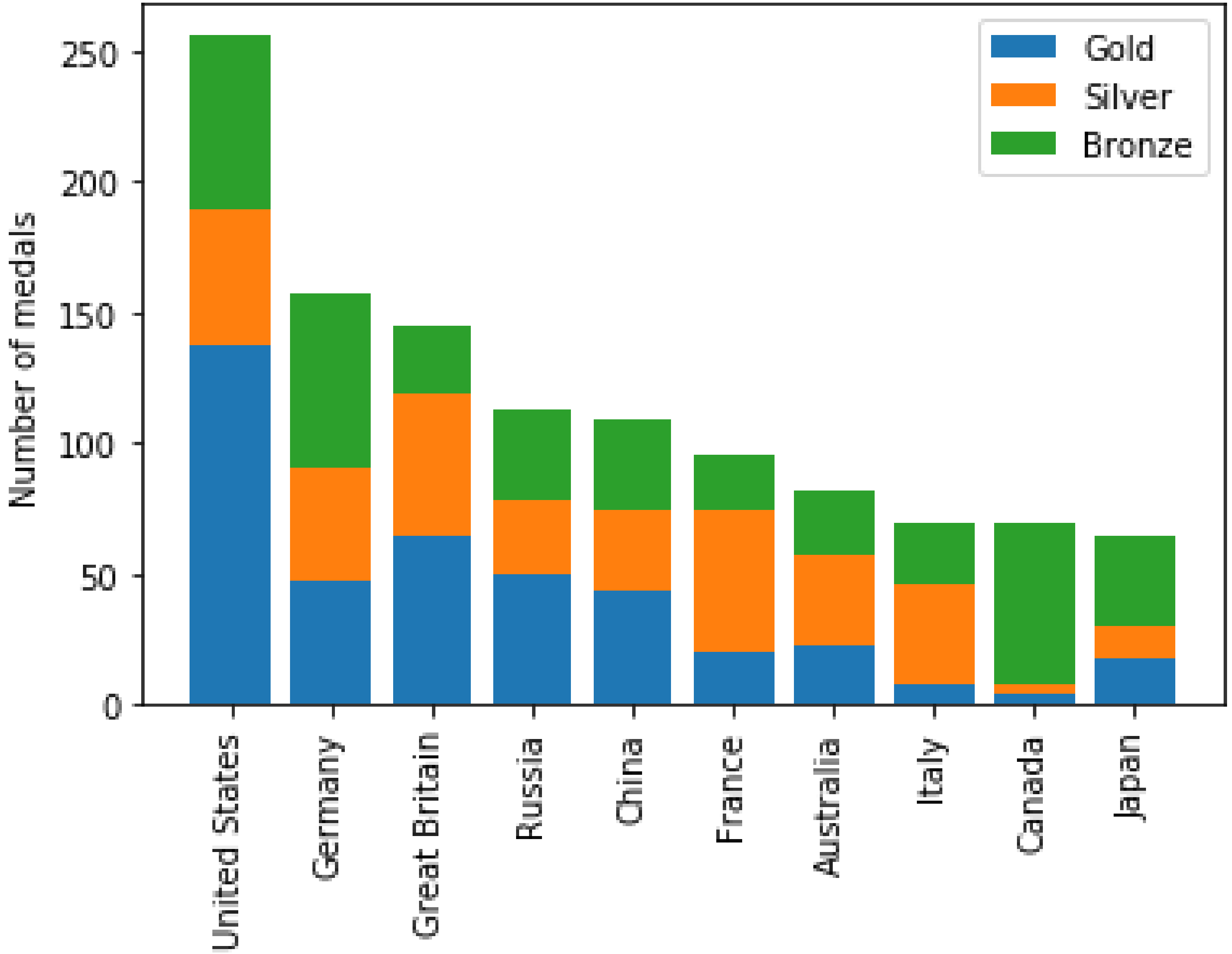
Adding a legend

```
fig, ax = plt.subplots
ax.bar(medals.index, medals["Gold"])
ax.bar(medals.index, medals["Silver"], bottom=medals["Gold"])
ax.bar(medals.index, medals["Bronze"],
       bottom=medals["Gold"] + medals["Silver"])

ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
```

Adding a legend

```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"], label="Gold")
ax.bar(medals.index, medals["Silver"], bottom=medals["Gold"],
       label="Silver")
ax.bar(medals.index, medals["Bronze"],
       bottom=medals["Gold"] + medals["Silver"],
       label="Bronze")

ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
ax.legend()
plt.show()
```

Stacked bar chart with legend



Create a bar chart!

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

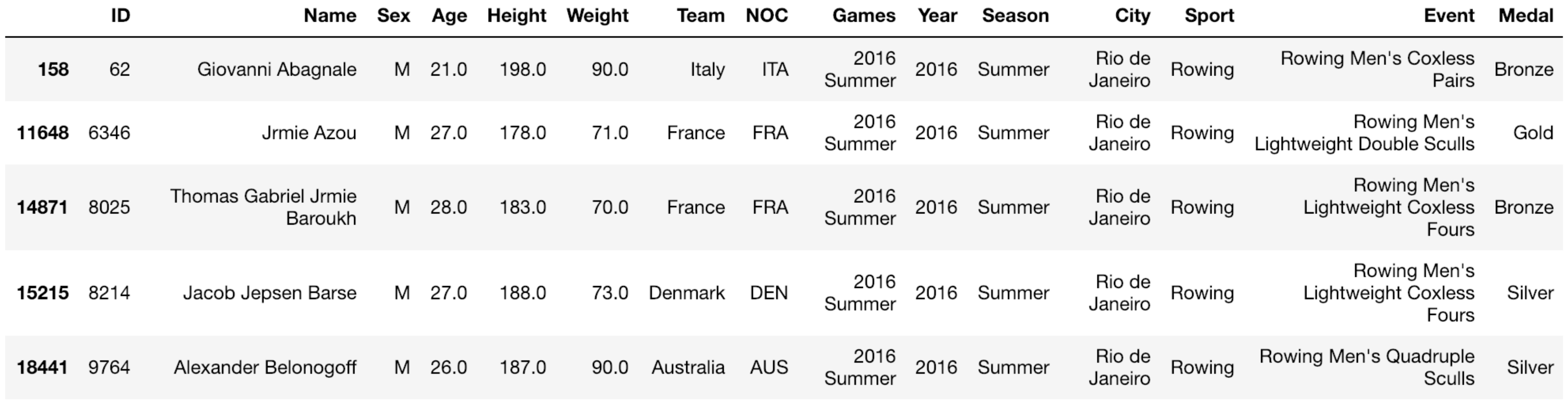
Part-02

Quantitative comparisons: histograms

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

Histograms



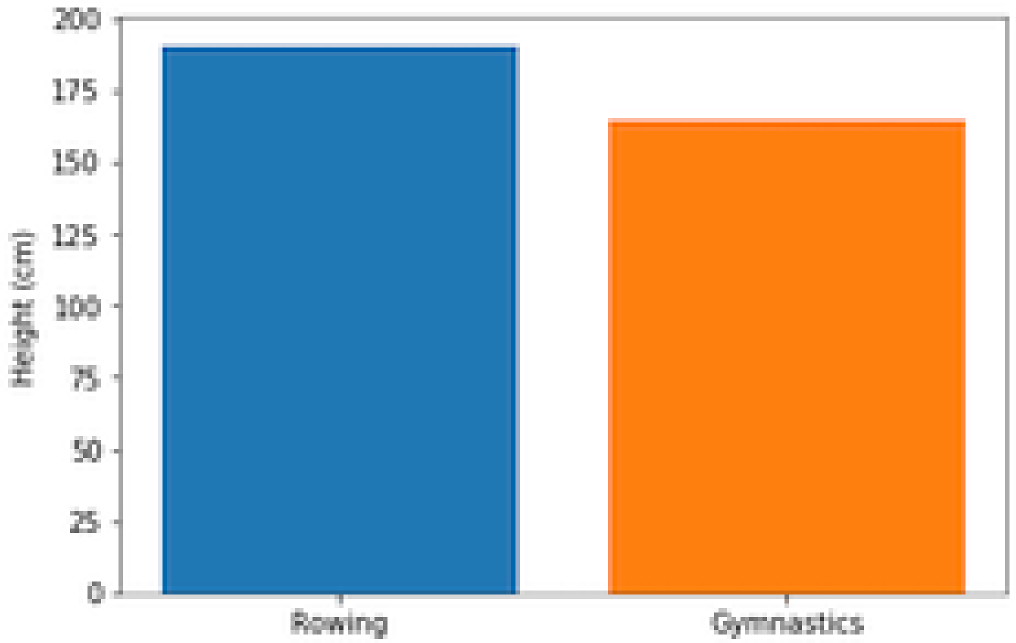
	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
158	62	Giovanni Abagnale	M	21.0	198.0	90.0	Italy	ITA	2016 Summer	2016	Summer	Rio de Janeiro	Rowing	Rowing Men's Coxless Pairs	Bronze
11648	6346	Jrmie Azou	M	27.0	178.0	71.0	France	FRA	2016 Summer	2016	Summer	Rio de Janeiro	Rowing	Rowing Men's Lightweight Double Sculls	Gold
14871	8025	Thomas Gabriel Jrmie Baroukh	M	28.0	183.0	70.0	France	FRA	2016 Summer	2016	Summer	Rio de Janeiro	Rowing	Rowing Men's Lightweight Coxless Fours	Bronze
15215	8214	Jacob Jepsen Barse	M	27.0	188.0	73.0	Denmark	DEN	2016 Summer	2016	Summer	Rio de Janeiro	Rowing	Rowing Men's Lightweight Coxless Fours	Silver
18441	9764	Alexander Belonogoff	M	26.0	187.0	90.0	Australia	AUS	2016 Summer	2016	Summer	Rio de Janeiro	Rowing	Rowing Men's Quadruple Sculls	Silver

A bar chart again

string → value

x y

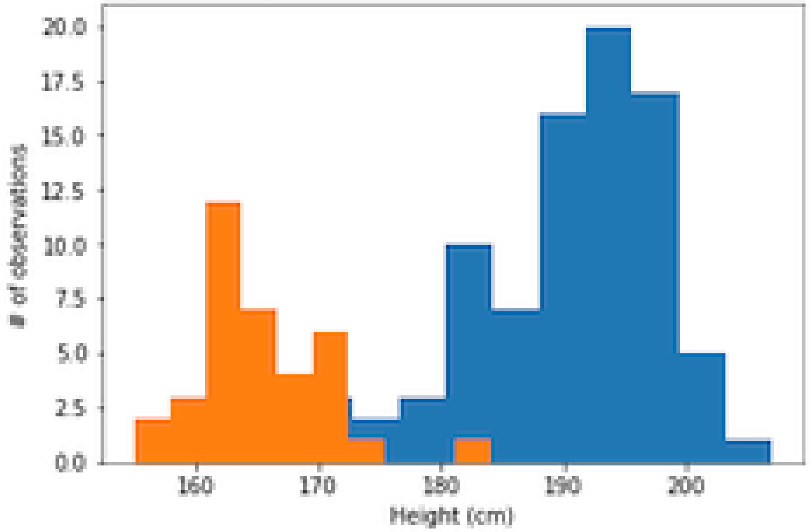
```
fig, ax = plt.subplots()
ax.bar("Rowing", mens_rowing["Height"].mean())
ax.bar("Gymnastics", mens_gymnastics["Height"].mean())
ax.set_ylabel("Height (cm)")
plt.show()
```



Introducing histograms

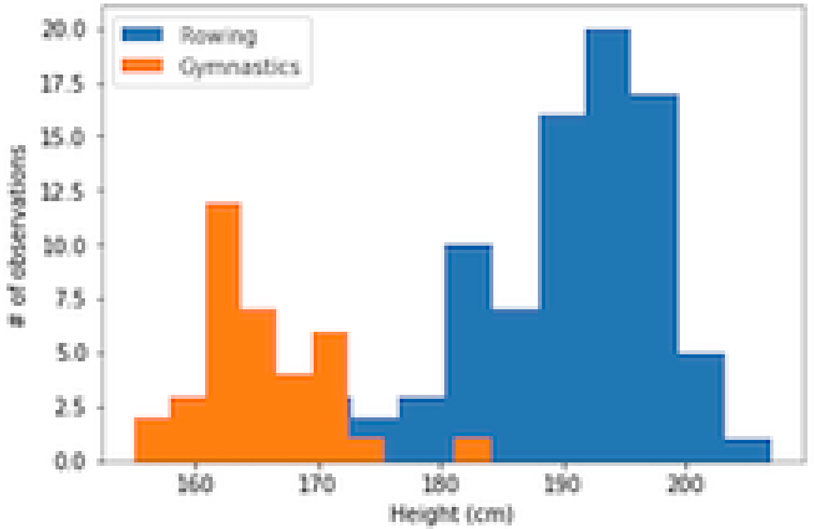
```
fig, ax = plt.subplots()  x
ax.hist(mens_rowing["Height"])  x
ax.hist(mens_gymnastics["Height"])
ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
plt.show()
```

value → cnt



Labels are needed

```
ax.hist(mens_rowing["Height"], label="Rowing")
ax.hist(mens_gymnastics["Height"], label="Gymnastics")
ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
ax.legend()
plt.show()
```



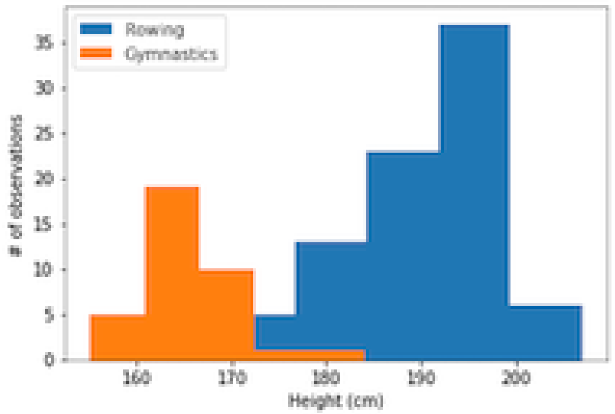
10 levels for each histogram

bins=10(default)

Customizing histograms: setting the number of bins

```
ax.hist(mens_rowing["Height"], label="Rowing", bins=5)
ax.hist(mens_gymnastics["Height"], label="Gymnastics", bins=5)
ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
ax.legend()
plt.show()
```

bins → num of divisions

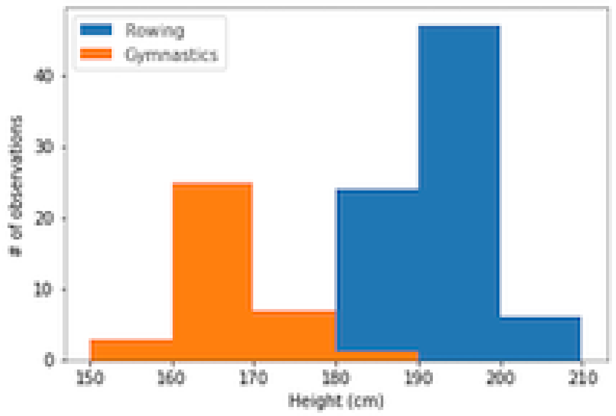


5 levels for both histograms

bins = 5

Customizing histograms: setting bin boundaries

```
ax.hist(mens_rowing["Height"], label="Rowing",
        bins=[150, 160, 170, 180, 190, 200, 210])

ax.hist(mens_gymnastics["Height"], label="Gymnastics",
        bins=[150, 160, 170, 180, 190, 200, 210])

ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
ax.legend()
plt.show()
```

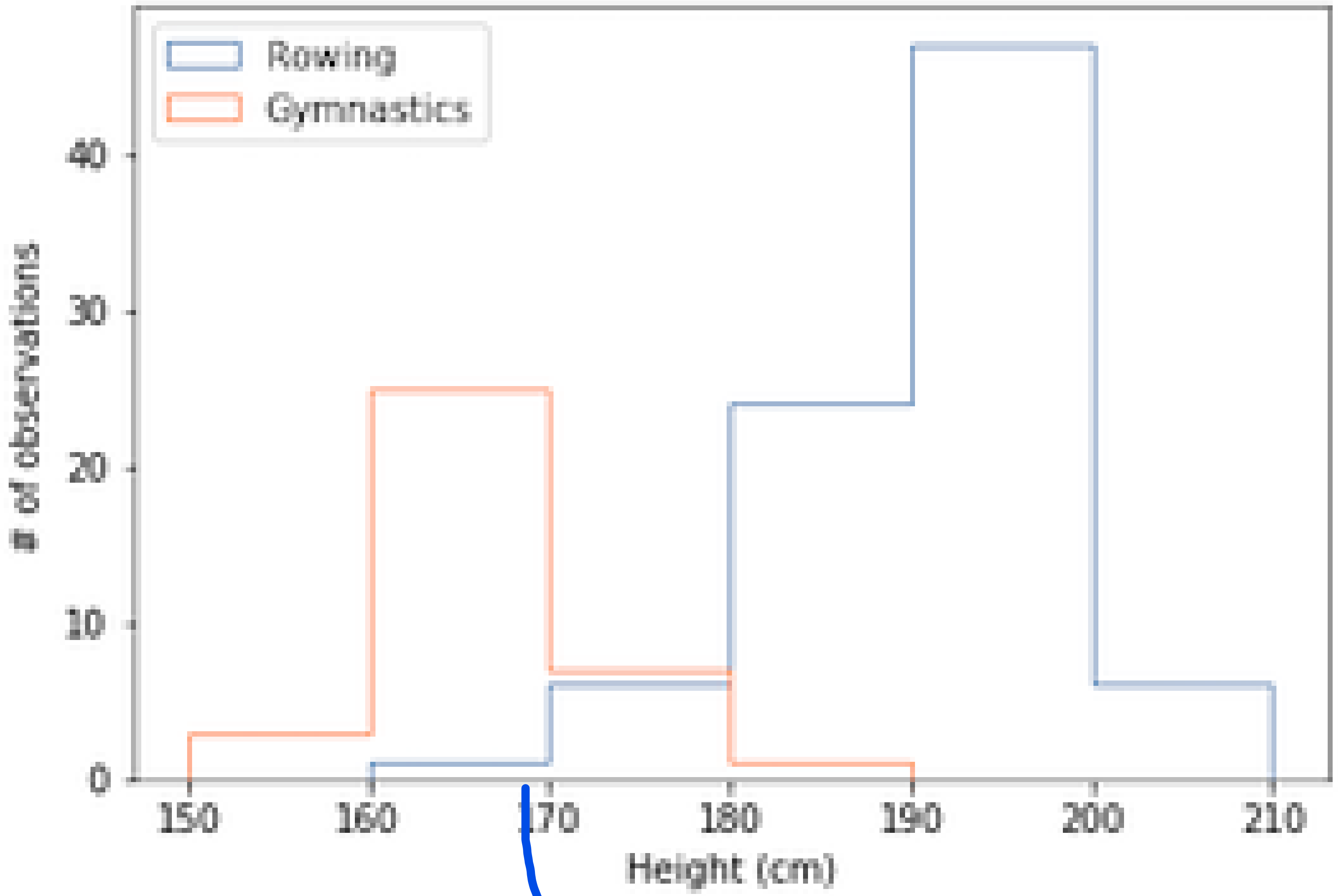


the histogram which is drawn later covers the previous one

Customizing histograms: transparency

```
ax.hist(mens_rowing["Height"], label="Rowing",
        bins=[150, 160, 170, 180, 190, 200, 210],
        histtype="step")

ax.hist(mens_gymnastics["Height"], label="Gymnastics",
        bins=[150, 160, 170, 180, 190, 200, 210],
        histtype="step")

ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
ax.legend()
plt.show()
```

Histogram with a histtype of step



now one does not cover another

Create your own histogram!

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Part - 03

Statistical plotting

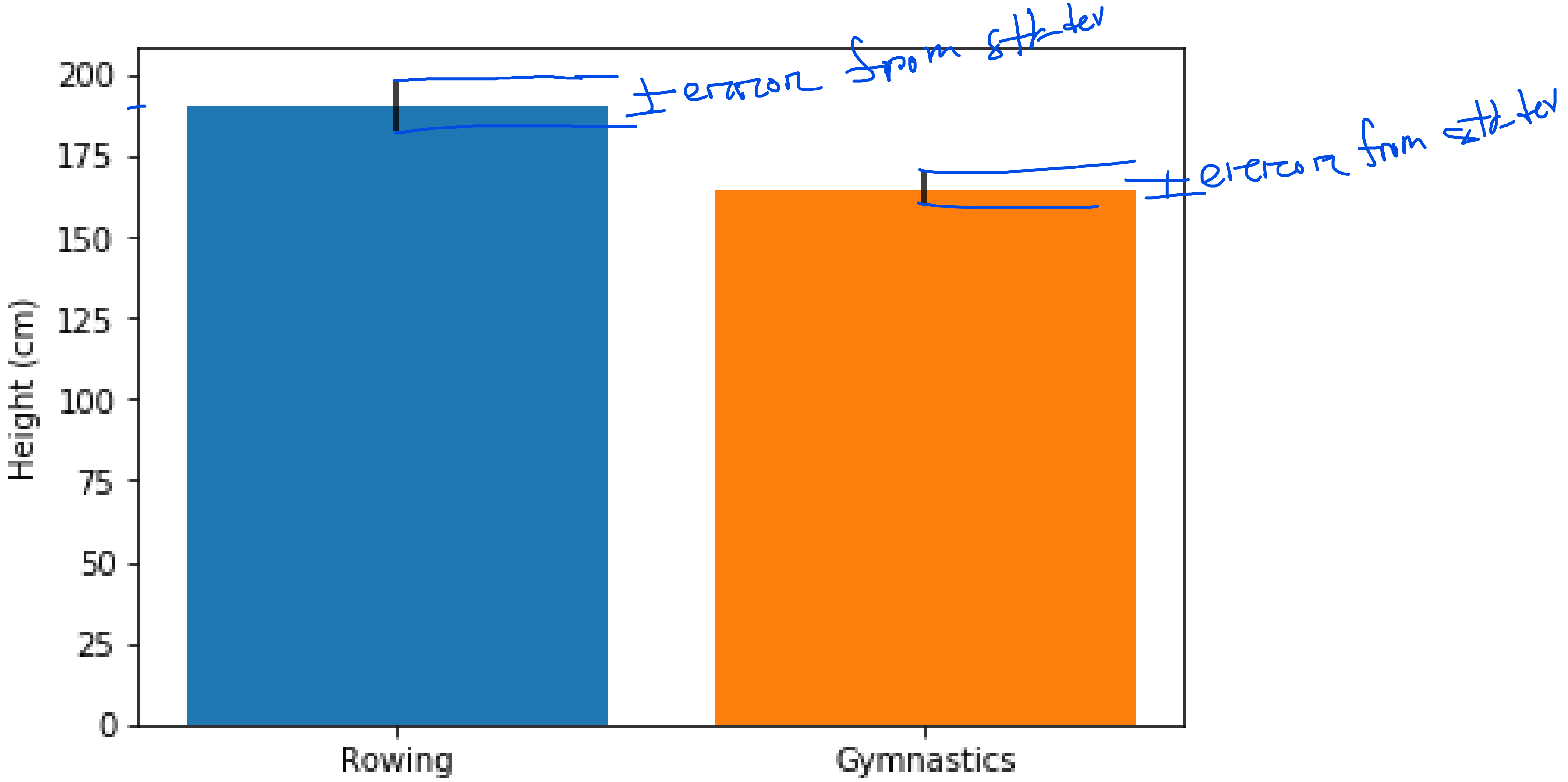
INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

Adding error bars to bar charts

```
fig, ax = plt.subplots()
             x
ax.bar("Rowing",       y
       mens_rowing["Height"].mean(),
       yerr=mens_rowing["Height"].std())  ↳ y- error

ax.bar("Gymnastics",
       mens_gymnastics["Height"].mean(),
       yerr=mens_gymnastics["Height"].std())

ax.set_ylabel("Height (cm)")

plt.show()
```

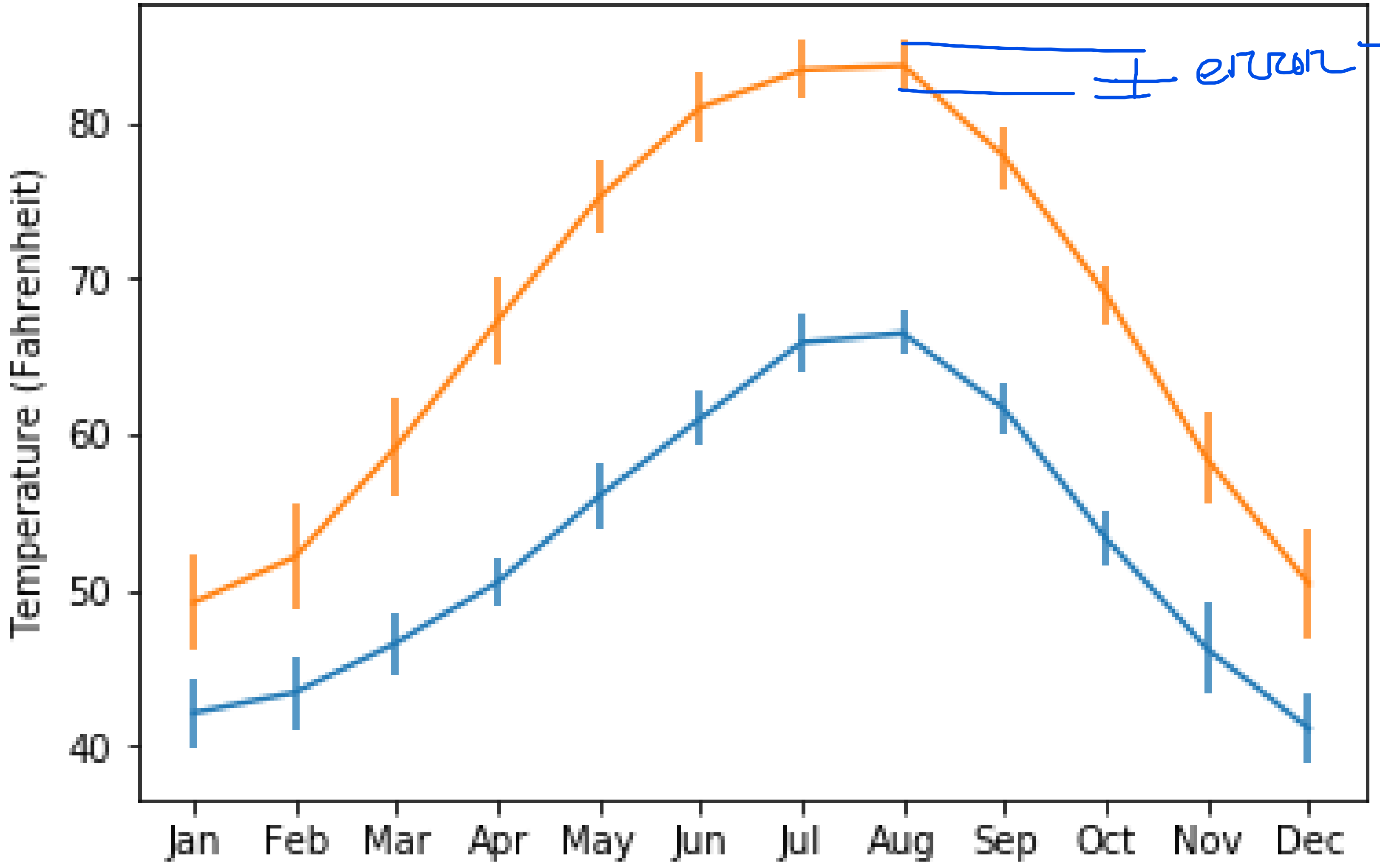
Error bars in a bar chart



Adding error bars to plots

```
fig, ax = plt.subplots()

ax.errorbar(seattle_weather["MONTH"],
            seattle_weather["MLY-TAVG-NORMAL"],
            yerr=seattle_weather["MLY-TAVG-STDDEV"])

ax.errorbar(austin_weather["MONTH"],
            austin_weather["MLY-TAVG-NORMAL"],
            yerr=austin_weather["MLY-TAVG-STDDEV"])

ax.set_ylabel("Temperature (Fahrenheit)")

plt.show()
```

Error bars in plots



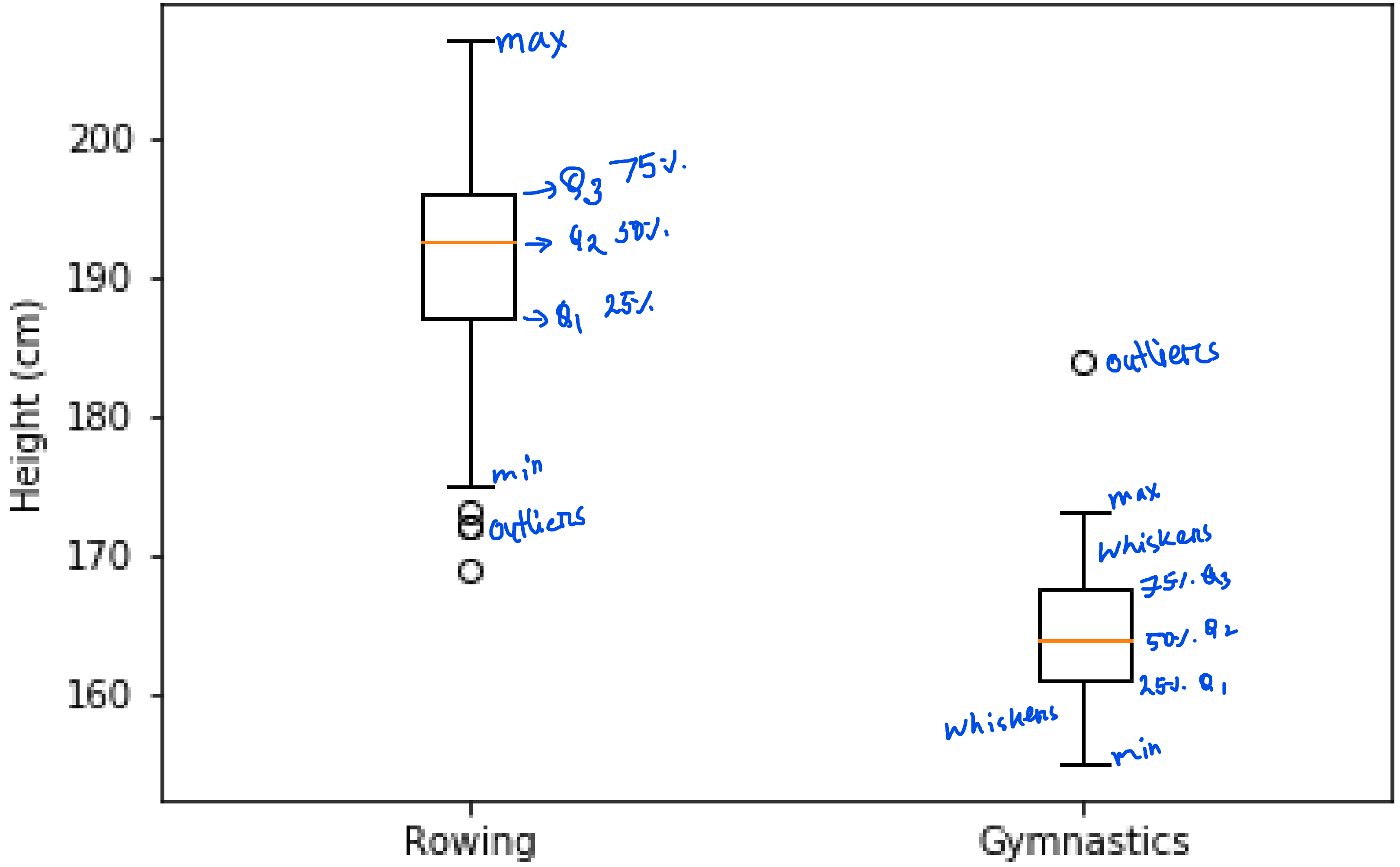
± error from std dev

Adding boxplots

```
fig, ax = plt.subplots()
ax.boxplot([mens_rowing["Height"],
            mens_gymnastics["Height"]])
ax.set_xticklabels(["Rowing", "Gymnastics"])
ax.set_ylabel("Height (cm)")


plt.show()
```

Interpreting boxplots

Outliers : very-small or very_large than what we expect



Try it yourself!

Part - 04

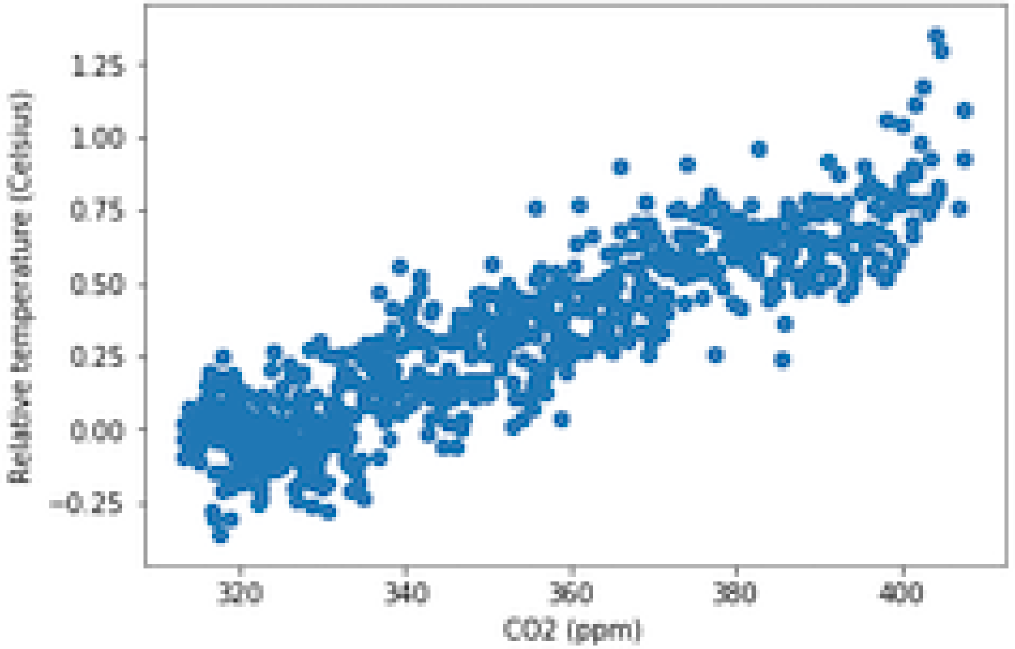
Quantitative comparisons: scatter plots

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

Introducing scatter plots

```
fig, ax = plt.subplots()    x             y
ax.scatter(climate_change["co2"], climate_change["relative_temp"])
ax.set_xlabel("CO2 (ppm)")
ax.set_ylabel("Relative temperature (Celsius)")
plt.show()
```



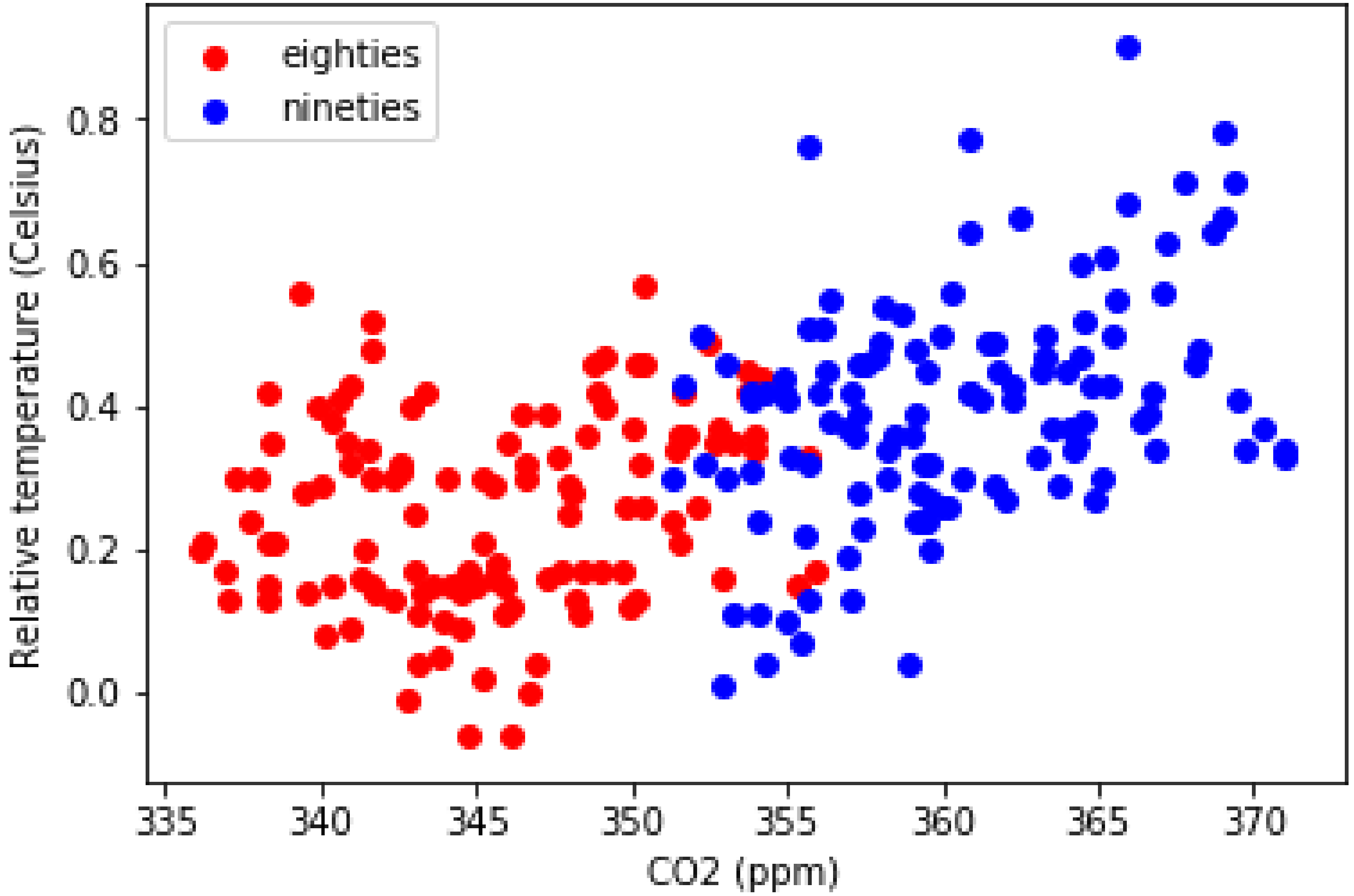
Customizing scatter plots

```
eighties = climate_change["1980-01-01":"1989-12-31"]
nineties = climate_change["1990-01-01":"1999-12-31"]
fig, ax = plt.subplots()
ax.scatter(eighties["co2"], eighties["relative_temp"],
           color="red", label="eighties")
ax.scatter(nineties["co2"], nineties["relative_temp"],
           color="blue", label="nineties")
ax.legend()

ax.set_xlabel("CO2 (ppm)")
ax.set_ylabel("Relative temperature (Celsius)")

plt.show()
```

extracting rows from time-series

It looks for all plot elements (e.g., scatter, line, bar, etc.) that have a label=....
It automatically creates a legend box showing:

Encoding a comparison by color



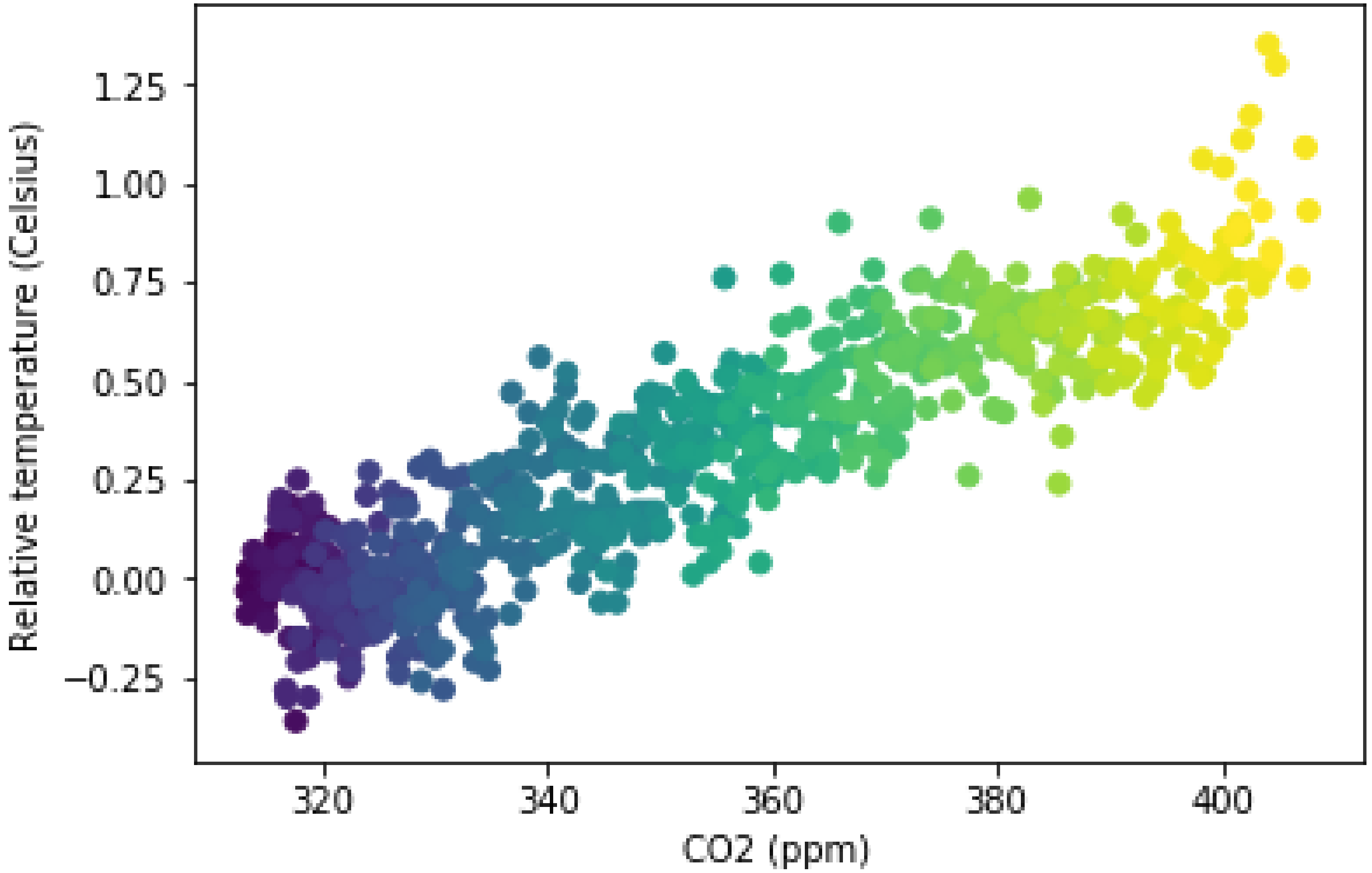
Encoding a third variable by color

```
fig, ax = plt.subplots()
ax.scatter(climate_change["co2"], climate_change["relative_temp"],
           c=climate_change.index)
ax.set_xlabel("CO2 (ppm)")
ax.set_ylabel("Relative temperature (Celsius)")
plt.show()
```

The c parameter in ax.scatter() controls the color of each point in the scatter plot.

Older (smaller) values ->dark purple / blue
Newer (larger) values ->yellow / green

Encoding time in color



Practice making your own scatter plots!