Quantitative comparisons: bar-charts

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

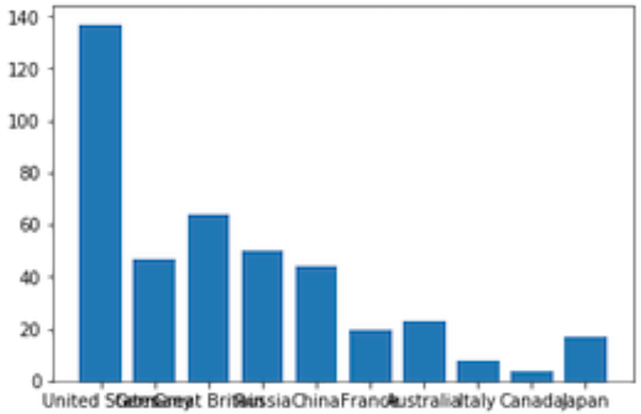
datacamp

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
```

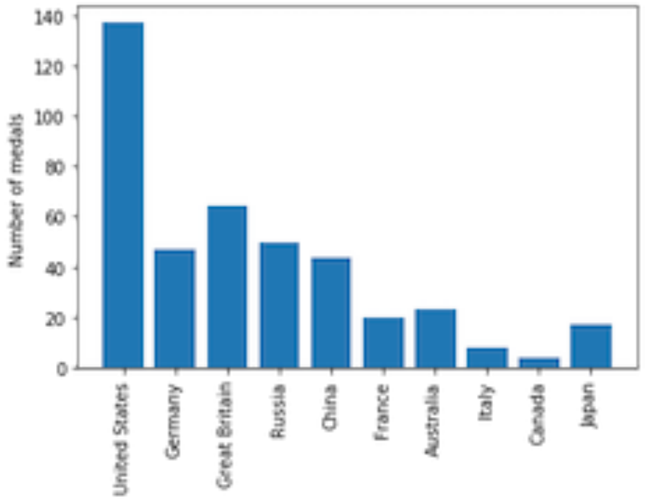
Olympic medals: visualizing the data

```
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

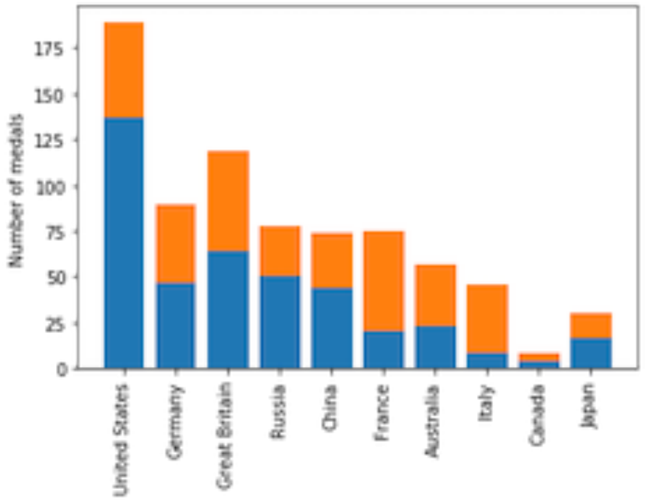
```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])
ax.set_xticklabels(medals.index, rotation=90)
ax.set_ylabel("Number of medals")
plt.show()
```



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()
```

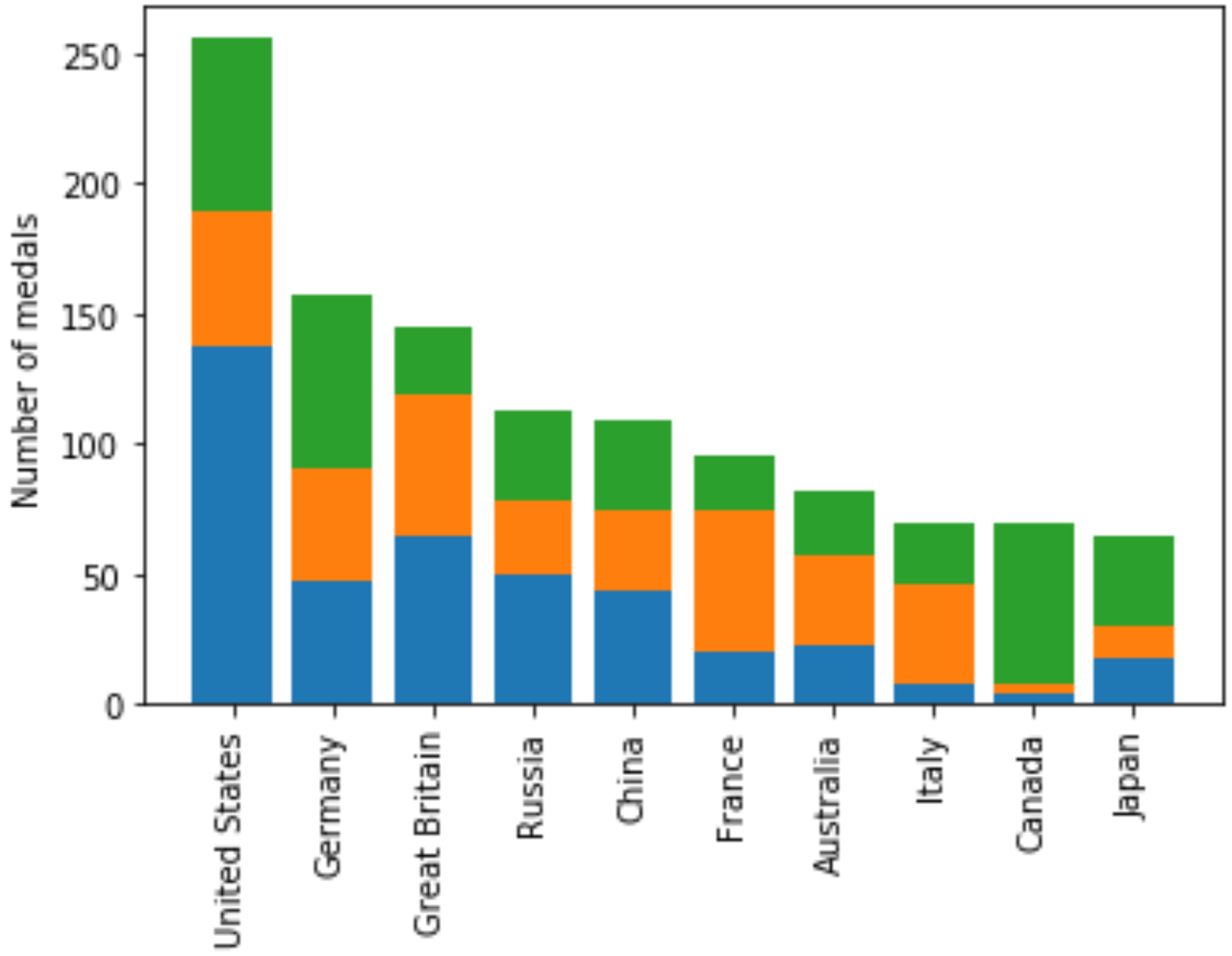
OVERLAP COLUMNS



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()
```

Stacked bar chart



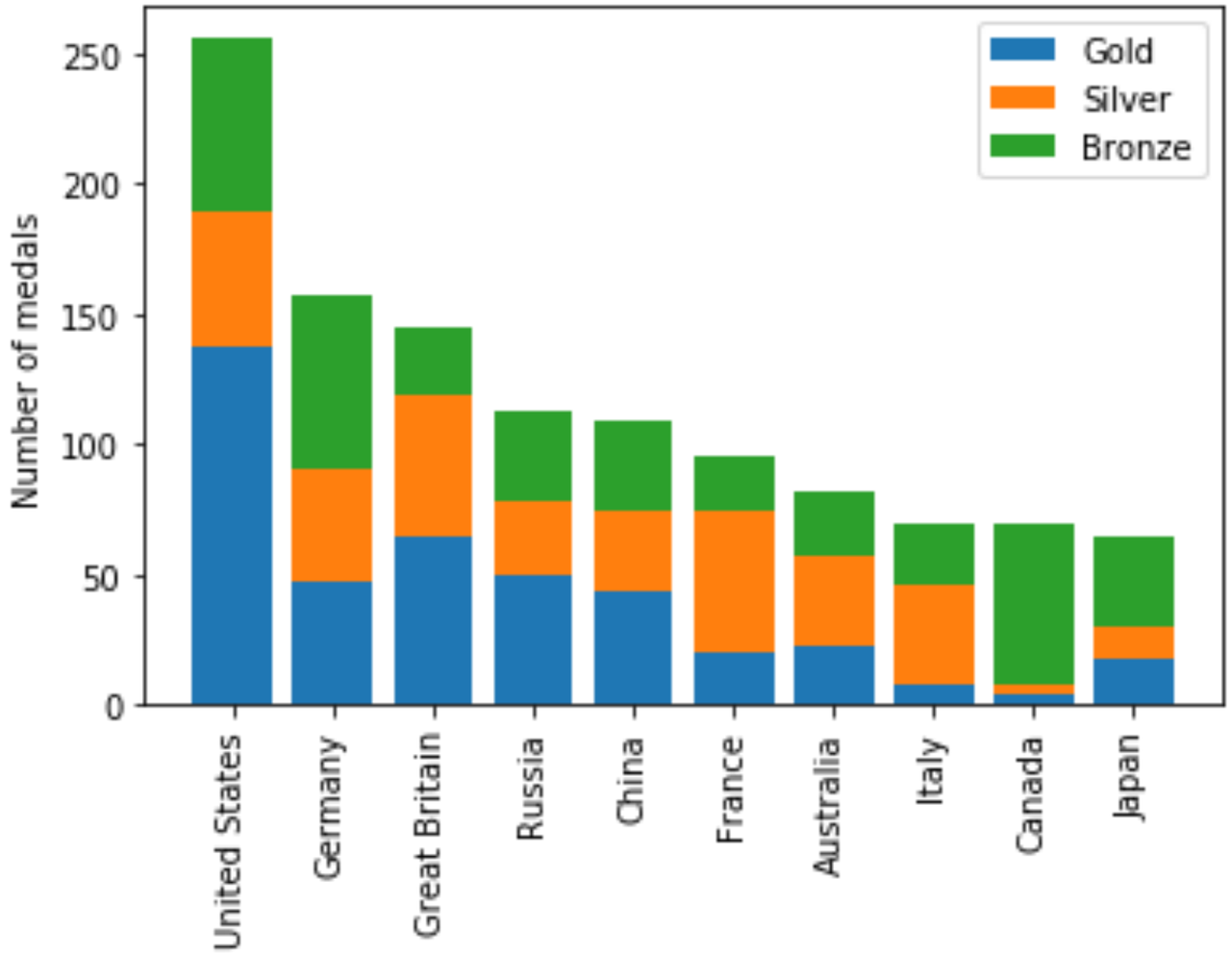
Adding a legend

```
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

Quantitative comparisons: histograms

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

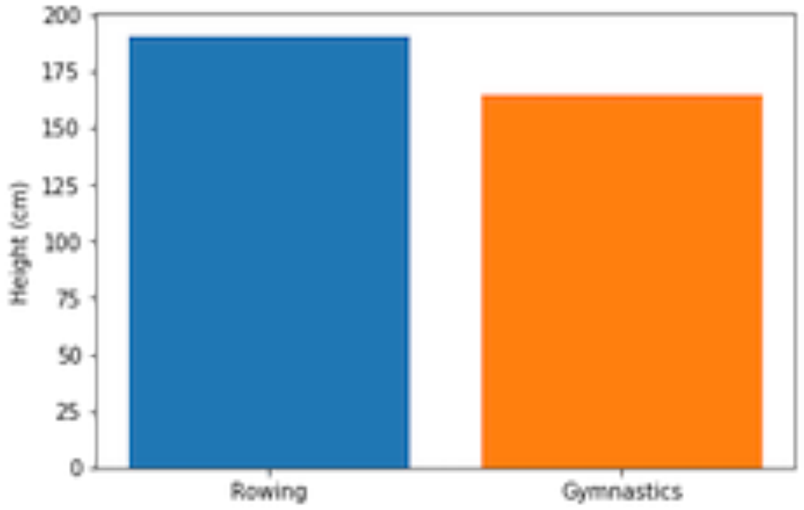
datacamp

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

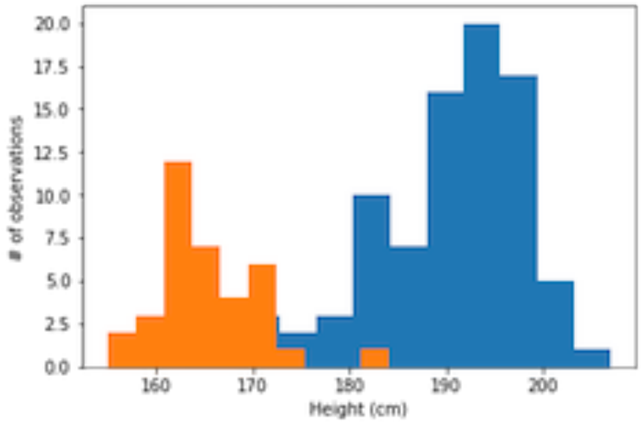
```
fig, ax = plt.subplots()
ax.bar("Rowing", mens_rowing["Height"].mean())
ax.bar("Gymnastics", mens_gymnastics["Height"].mean())
ax.set_ylabel("Height (cm)")
plt.show()
```



Introducing histograms

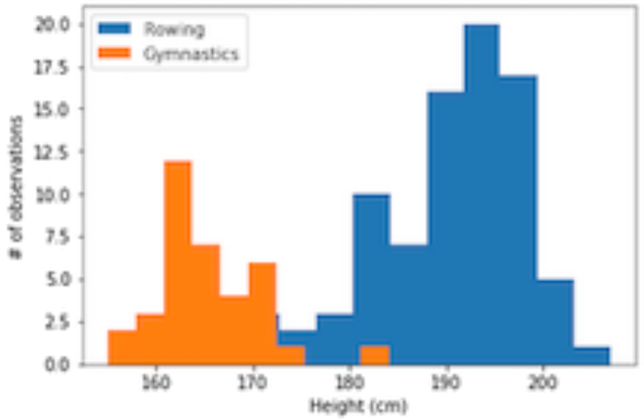
```
fig, ax = plt.subplots()
ax.hist(mens_rowing["Height"])
ax.hist(mens_gymnastic["Height"])
ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
plt.show()
```

HISTOGRAM IS MORE APPROPRIATE



Labels are needed

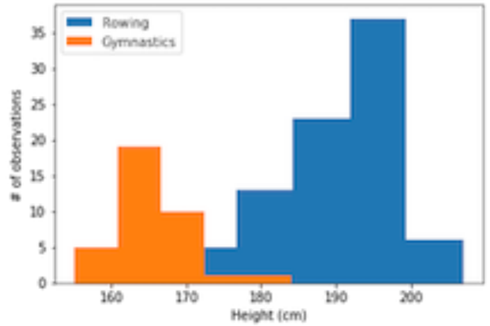
```
ax.hist(mens_rowing["Height"], label="Rowing")
ax.hist(mens_gymnastic["Height"], label="Gymnastics")
ax.set_xlabel("Height (cm)")
ax.set_ylabel("# of observations")
ax.legend()
plt.show()
```



Customizing histograms: setting the number of bins

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

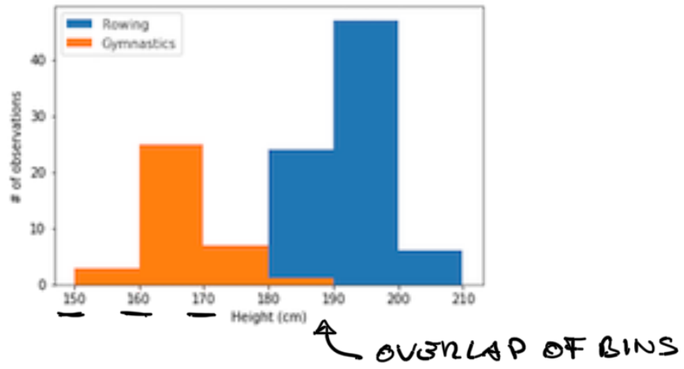
WE CAN DECIDE BINS (BAR) NUMBERS.
DEFAULT IS 10 BINS



Customizing histograms: setting bin boundaries

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

ax.hist(mens_gymnastic["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 LIST OF VALUES DEFINE BOUNDARIES BETWEEN BINS

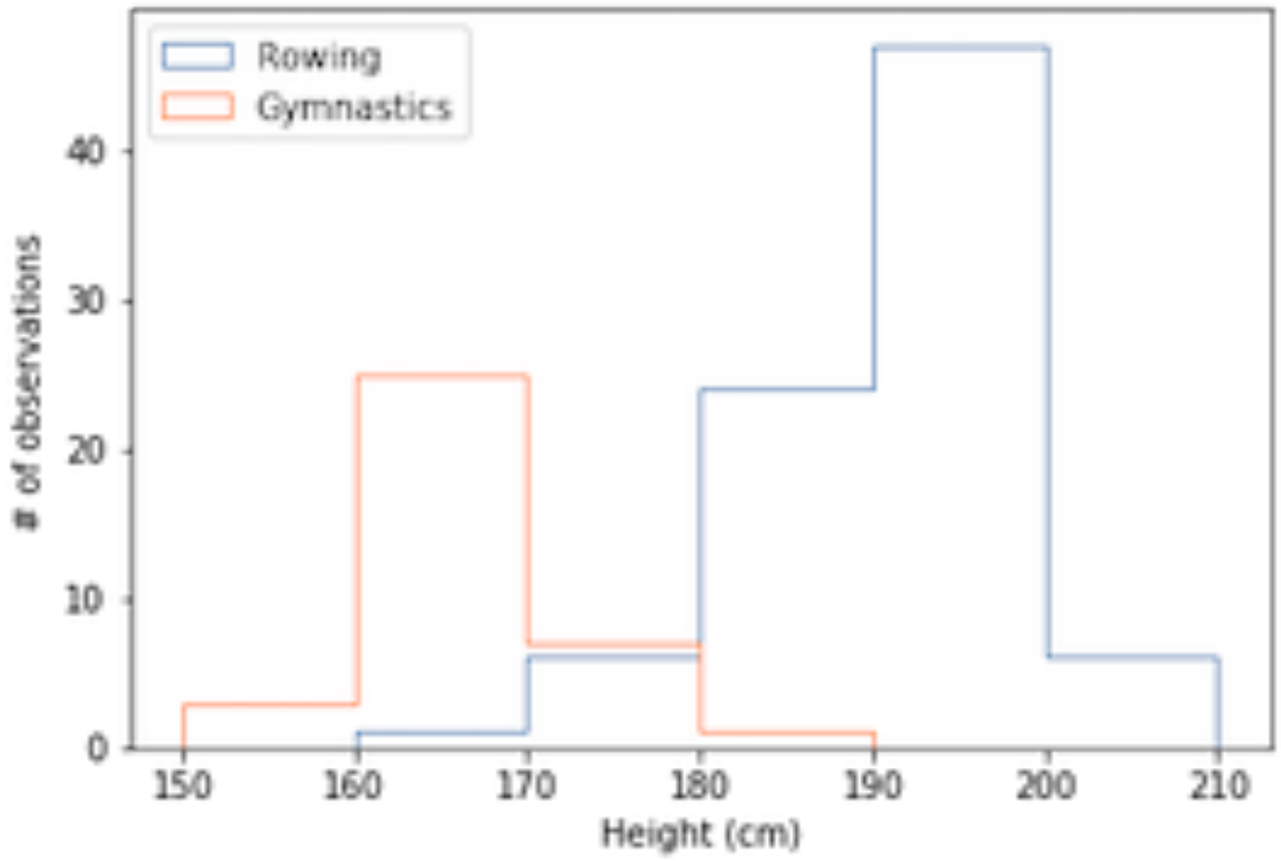


OVERLAP OF BINS

Customizing histograms: transparency

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

ax.hist(mens_gymnastic["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



Create your own histogram!

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Statistical plotting

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

datacamp

Adding error bars to bar charts

ARGUMENT

SUMMARIZES THE DISTRIBUTION OF THE DATA IN ONE NUMB (STANDARD DEVIATION OF THE VALUES IN THIS CASE)

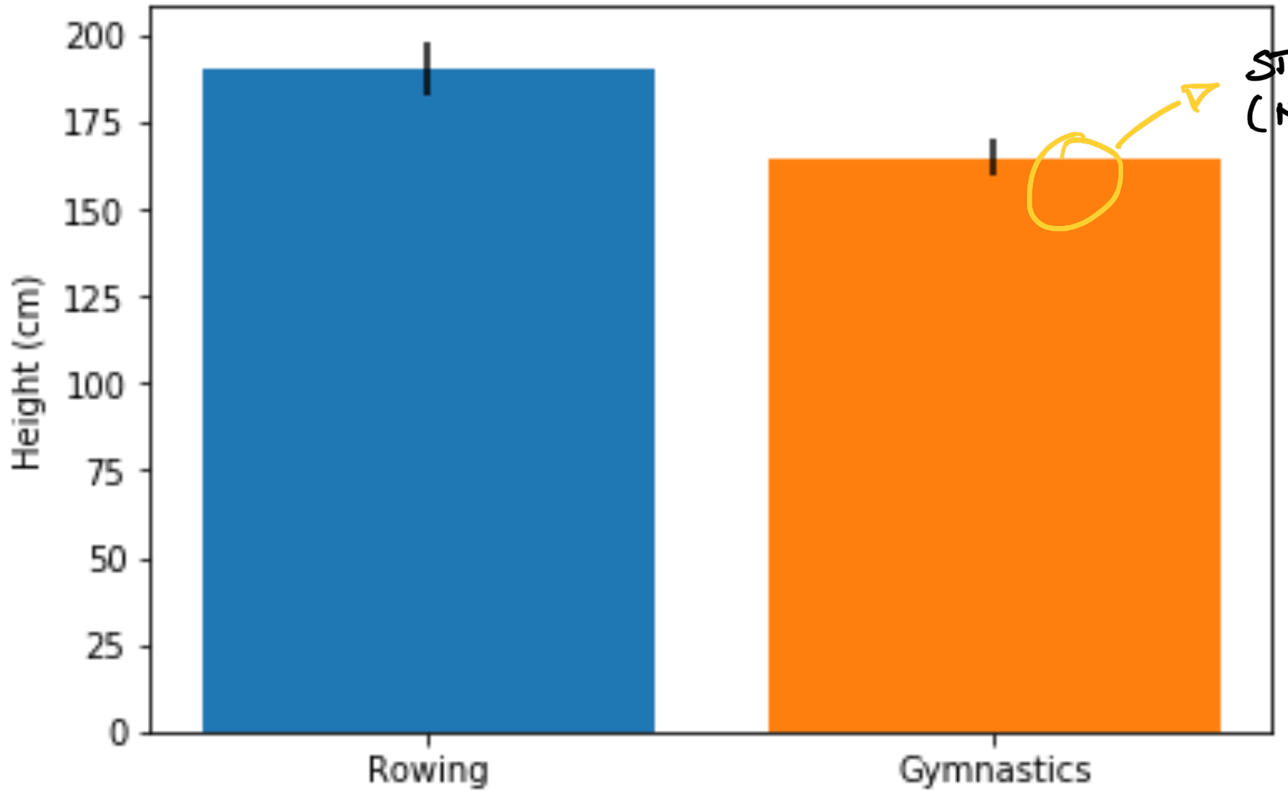
```
fig, ax = plt.subplots()

ax.bar("Rowing",
       mens_rowing["Height"].mean(),
       yerr=mens_rowing["Height"].std())

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

ax.set_ylabel("Height (cm)")

plt.show()
```

X ARGUMENT

Y ARGUMENT

→ FURTHER ARGUMENT DISPLAYED AS A VERTICAL MARK

Error bars in a bar chart



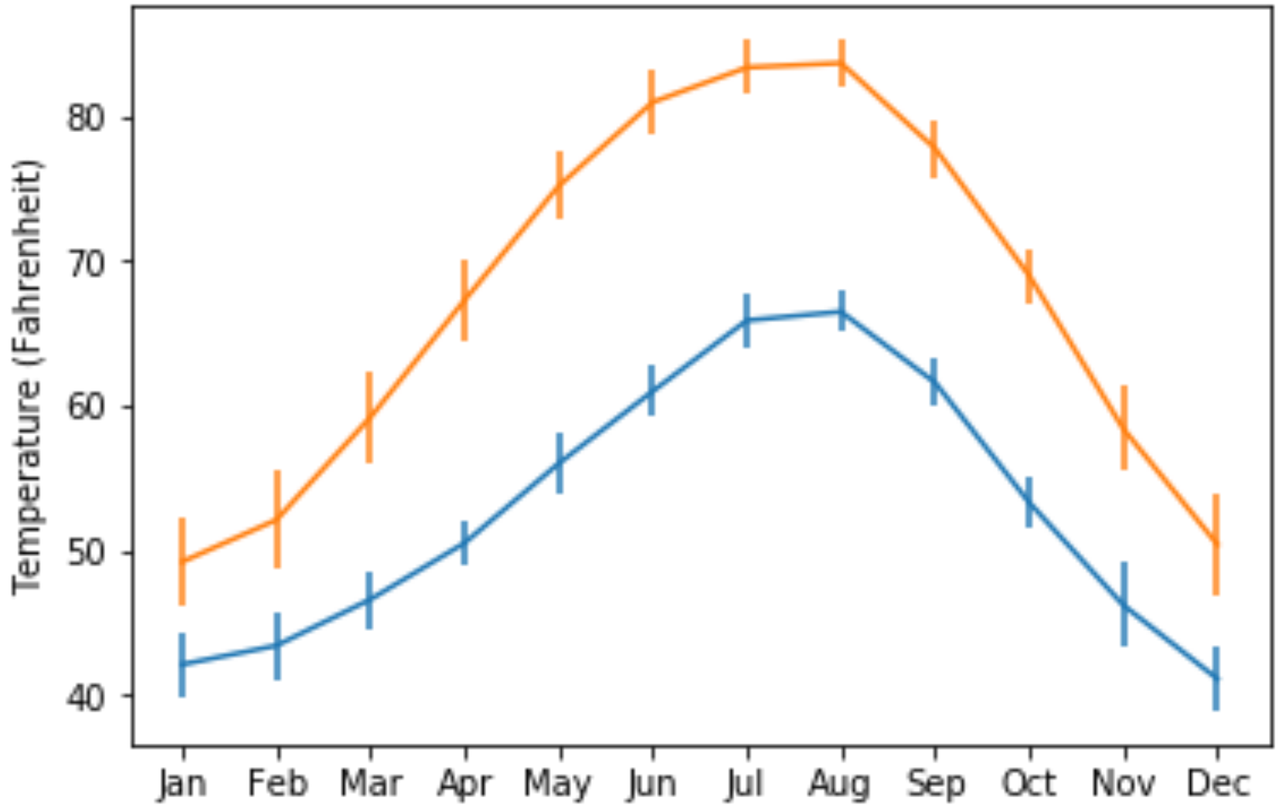
STANDARD DEVIATION
(NOTICE THE HEIGTH OF THE MARK)

MEAN VALUE

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



Adding boxplots

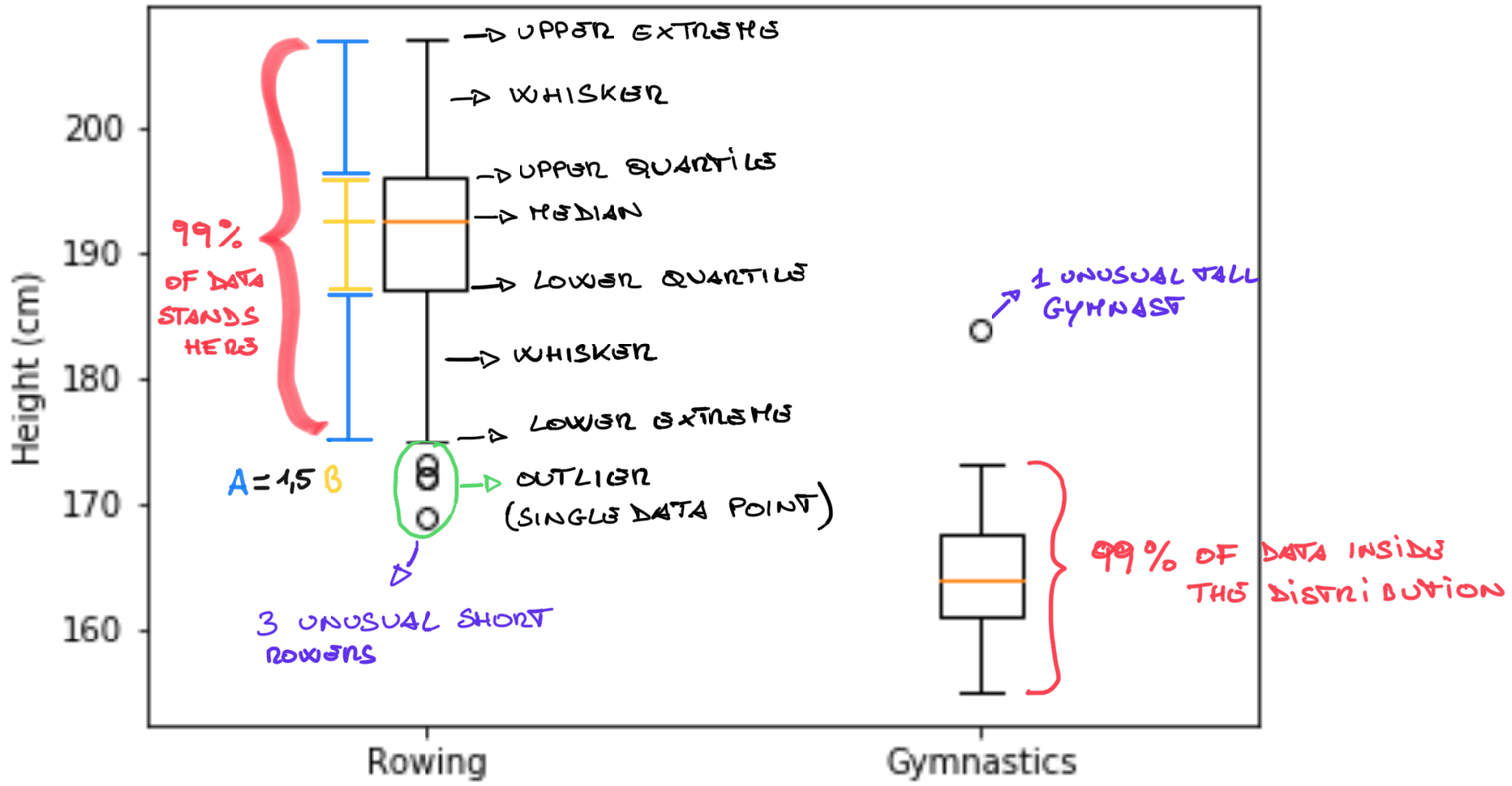
METHOD

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

plt.show()
```

IT'S NOT AWARE OF THE LABEL OF EACH VARIABLE

WE ADD IT SEPARATELY

Interpreting boxplots



Try it yourself!

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Quantitative comparisons: scatter plots

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

Ariel Rokem
Data Scientist

BAR CHARTS SHOW THE VALUES OF ONE VARIABLE ACROSS DIFFERENT CONDITIONS SUCH AS DIFFERENT COUNTRIES.

BI-VARIATE COMPARISON → COMPARE THE VALUES OF DIFFERENT VARIABLES ACROSS OBSERVATIONS. INVOLVE THE VALUES OF TWO DIFFERENT VARIABLES

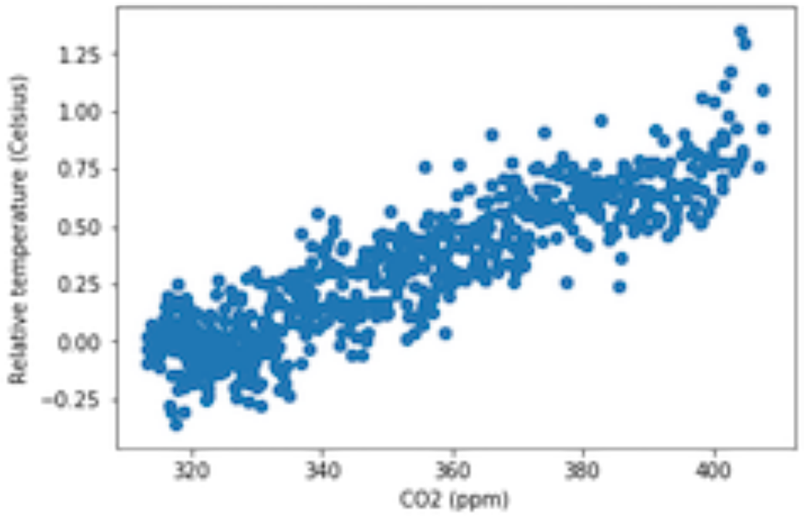
(SCATTER PLOT)

Introducing scatter plots

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

x parameter

y parameter

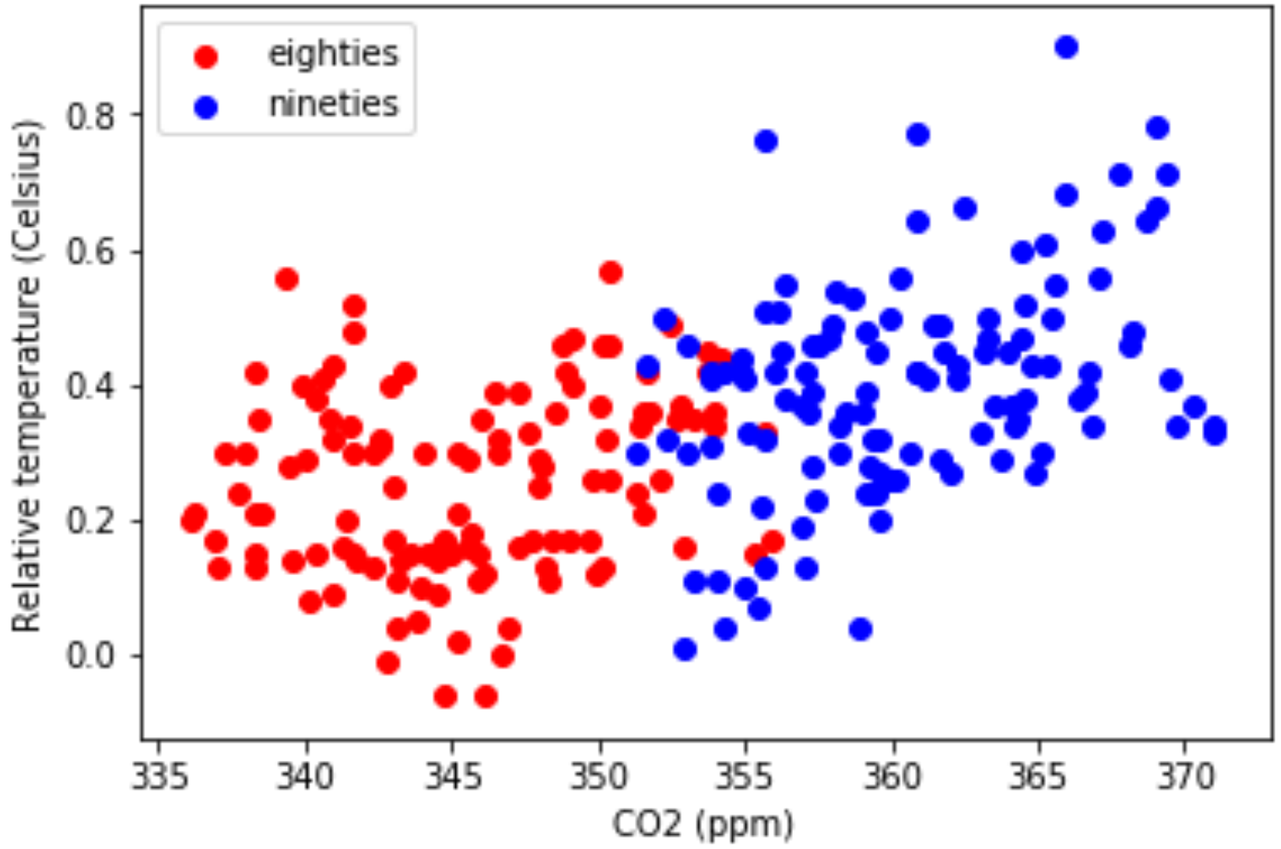


Customizing scatter plots

```
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()
```

TWO BIVARIATE COMPARISONS

Encoding a comparison by color



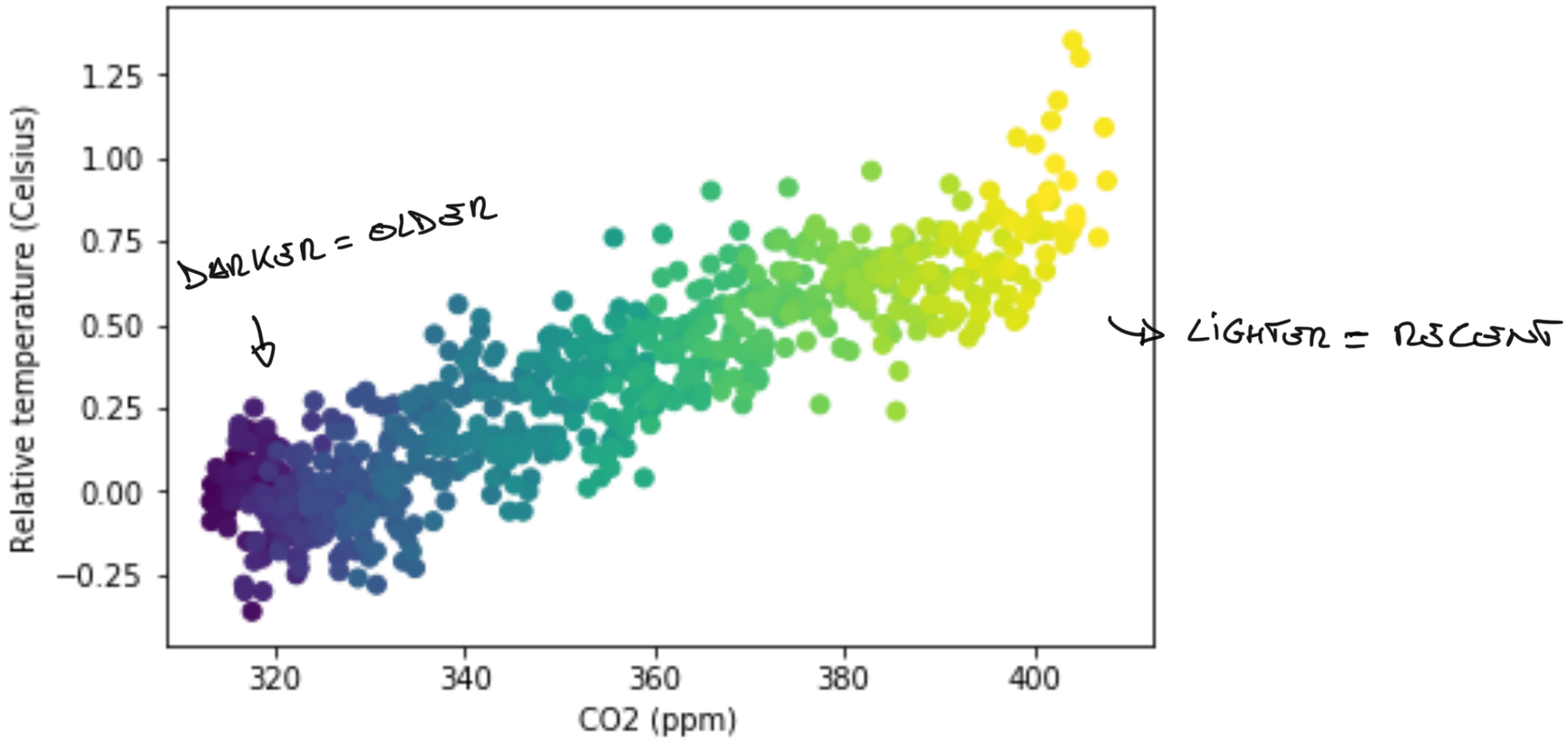
Encoding a third variable by color

```
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()
```

ENCODE TIME AS COLOR

TIME IS STORED IN THE INDEX

Encoding time in color



Practice making your own scatter plots!

INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB