3. Quantitative comparisons and statistical visualizations

```
import pandas as pd

from google.colab import files
myfile = files.upload()
```

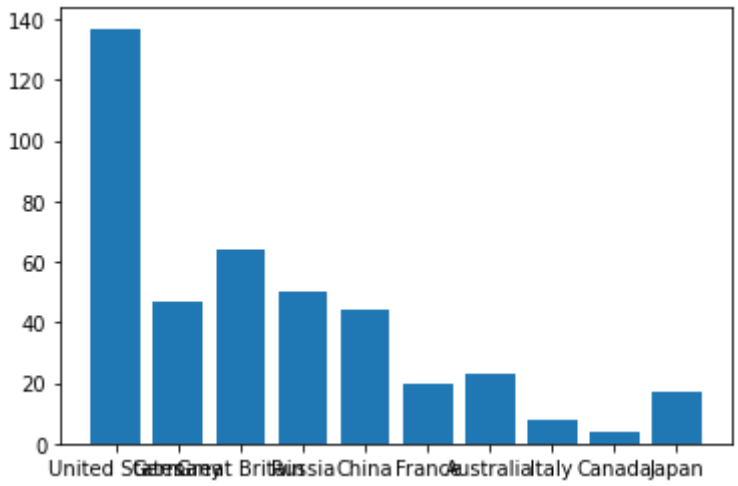
```
파일 선택 선택된 파일 없음
Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving medals_by_country_2016.csv to medals_by_country_2016.csv
```

```
import io


medals=pd.read_csv(io.BytesIO(myfile['medals_by_country_2016.csv']), index_col=0)
```

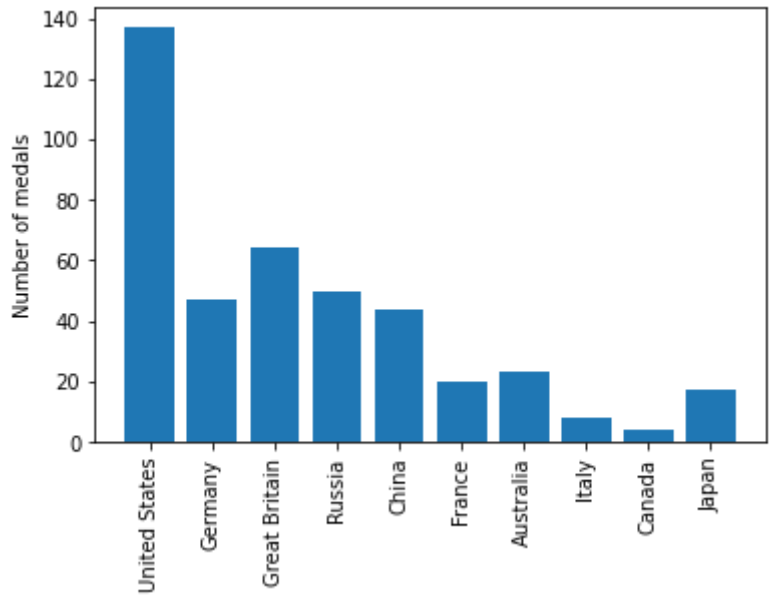
```
import matplotlib.pyplot as plt
```

1) quantitative comparisons: barplot

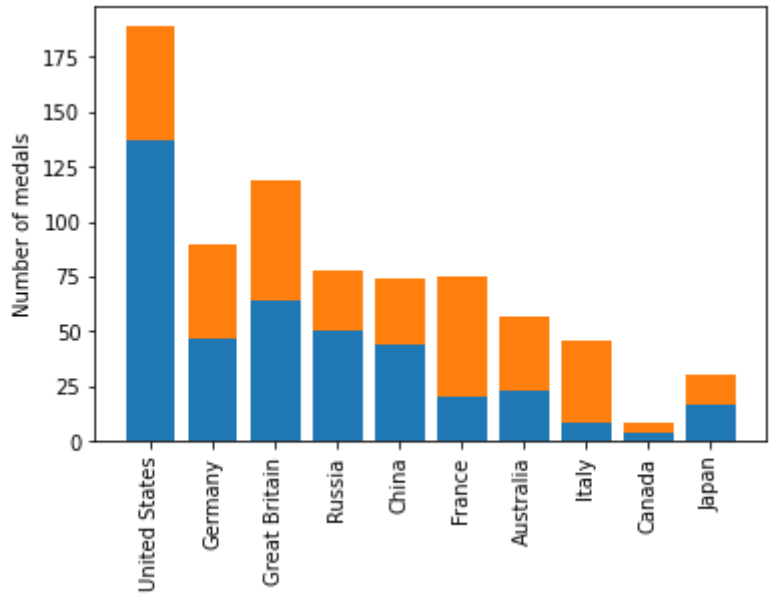
```
fig, ax = plt.subplots()
ax.bar(medals.index, medals["Gold"])
plt.show()
```



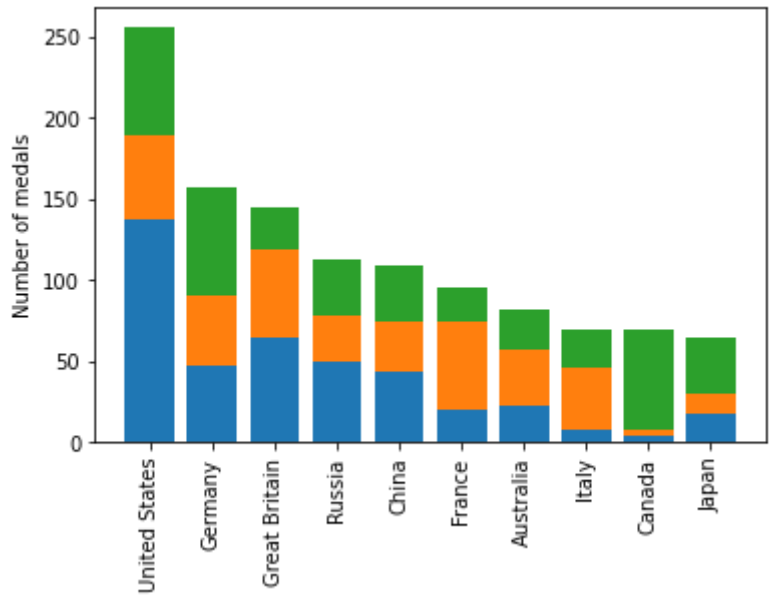

```
#fix overlapping letters

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()
```



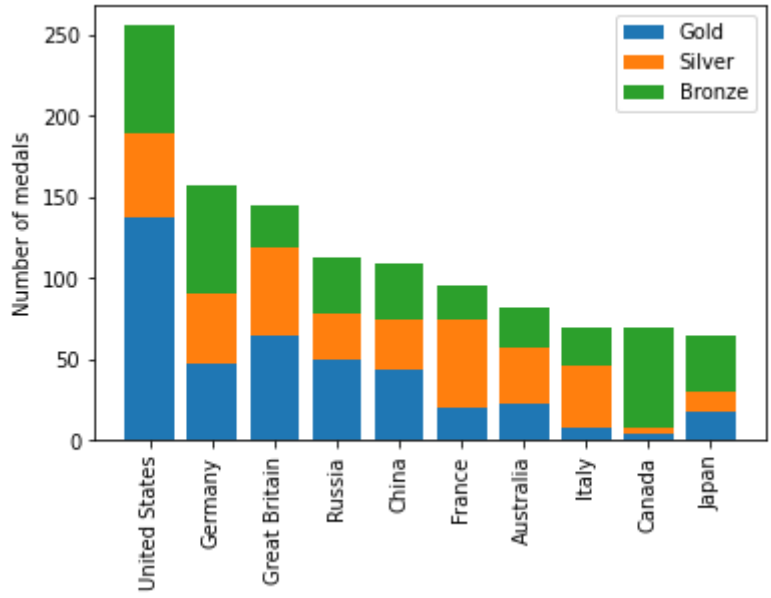

```
#add another information: Gold+Silver

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




```
#add other information: G,S,B

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()
```




```
#add a legend to show what each color means

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() #show which color is which medal
plt.show()
```



2) Quantitative comparisons: histograms

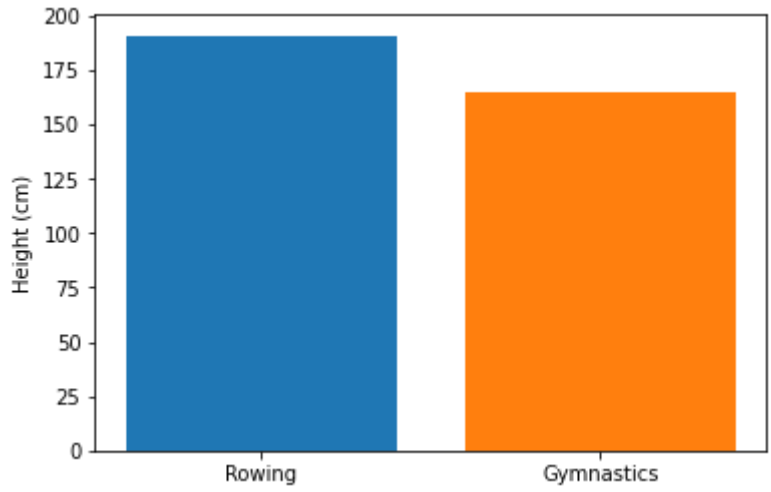
```
import pandas as pd
```

```
from google.colab import files
myfile = files.upload()
```

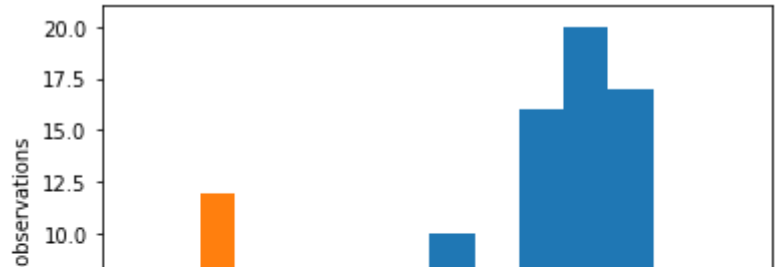
파일 선택 선택된 파일 없음 Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving summer2016.csv to summer2016 (2).csv
```

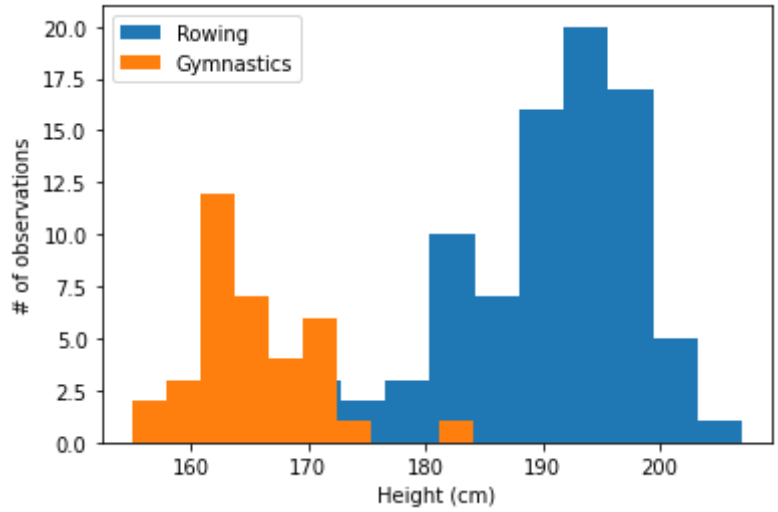
```
import io


summer=pd.read_csv(io.BytesIO(myfile['summer2016.csv']), index_col=0)
mens_rowing=summer[(summer['Sex']=='M') & (summer['Sport']=='Rowing')]
mens_gymnastics = summer[(summer['Sex']=='M') & (summer['Sport']=='Gymnastics')]



#bar charts review

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()
```




```
#histogram

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

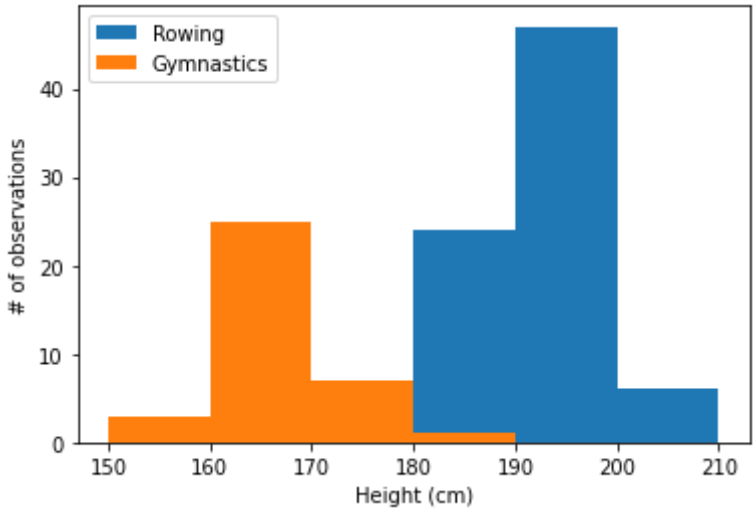



```
#add labels

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




```
#setting the number of bins

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

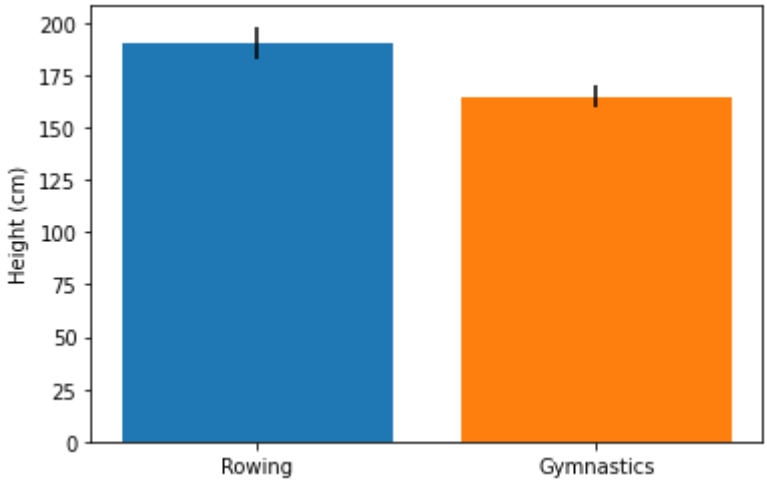

```
#setting bin boundaries

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




```
#transparency

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

3) Statistical plotting

```
#addinng error bars to bar charts

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()
```




```
#adding error bars to line plots

import pandas as pd

from google.colab import files
myfile = files.upload()
```

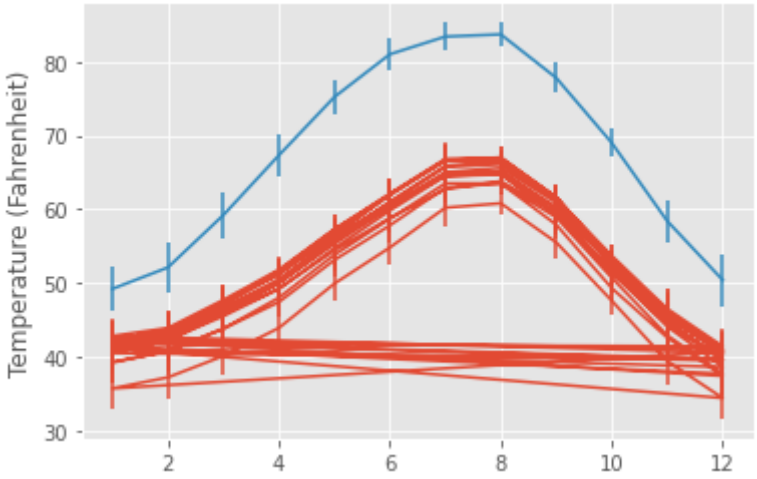
파일 선택 선택된 파일 없음 Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving seattle_weather.csv to seattle_weather (3).csv

```
import io
seattle_weather = pd.read_csv(io.BytesIO(myfile['seattle_weather.csv']))


myfile2 = files.upload()
```

파일 선택 선택된 파일 없음 Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving austin_weather.csv to austin_weather (2).csv
```

```
import io
austin_weather = pd.read_csv(io.BytesIO(myfile2['austin_weather.csv']))


fig, ax = plt.subplots()

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

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

ax.set_ylabel("Temperature (Fahrenheit)")

plt.show()
```




```
#adding boxplots

fig, ax = plt.subplots()

ax.boxplot([mens_rowing["Height"], mens_gymnastics["Height"]]);
ax.set_xticklabels(["Rowing", "Gymnastics"]);
ax.set_ylabel("Height (cm)");
```

4) scatterplots

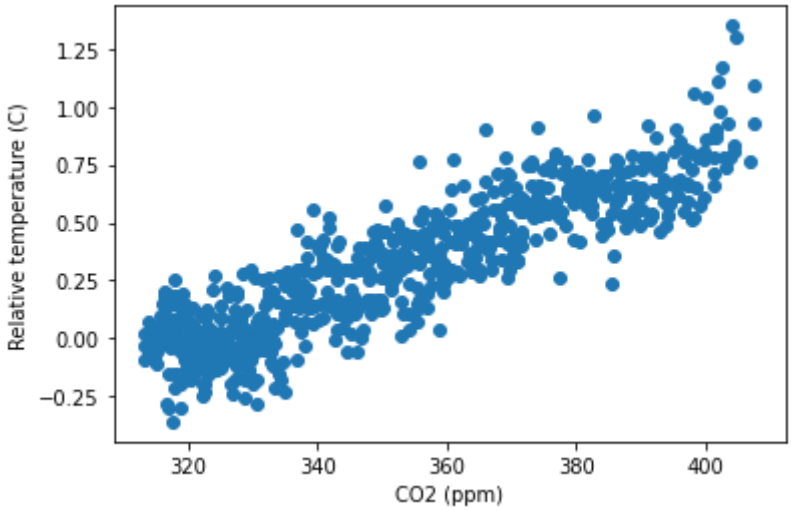
```
myfile = files.upload()
```

파일 선택 선택된 파일 없음 Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving climate_change.csv to climate_change (1).csv
```

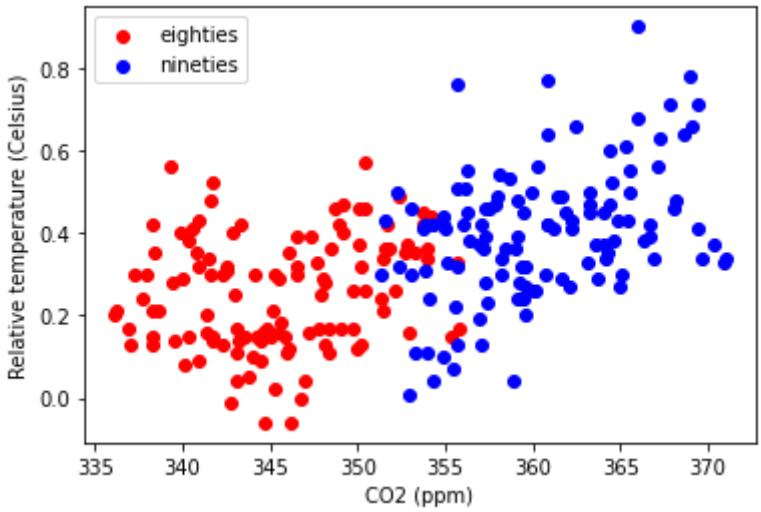
```
import io
climate_change = pd.read_csv(io.BytesIO(myfile['climate_change.csv']), index_col='date')
```

```
#scatterplots

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



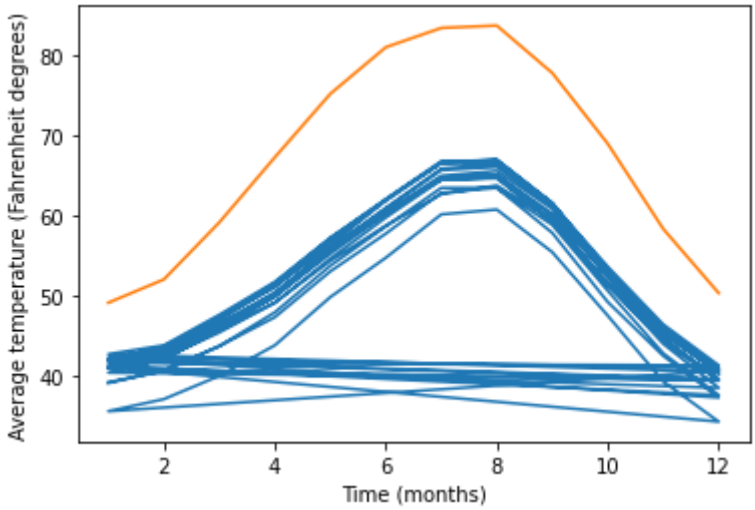

```
#customizing scatterplots

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()
```



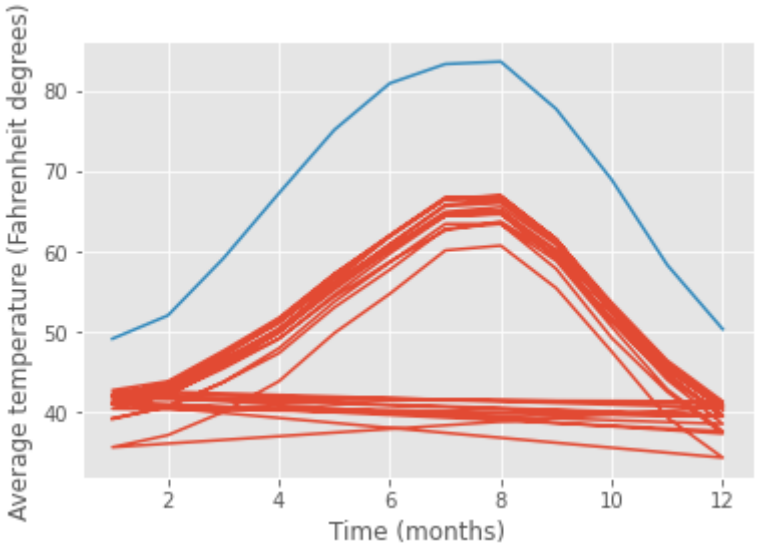
4. Sharing visualizations with others

```
# Changing plot style

fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"], seattle_weather["MLY-TAVG-NORMAL"])
ax.plot(austin_weather["DATE"], austin_weather["MLY-TAVG-NORMAL"])
ax.set_xlabel("Time (months)")
ax.set_ylabel("Average temperature (Fahrenheit degrees)")
plt.show()
```



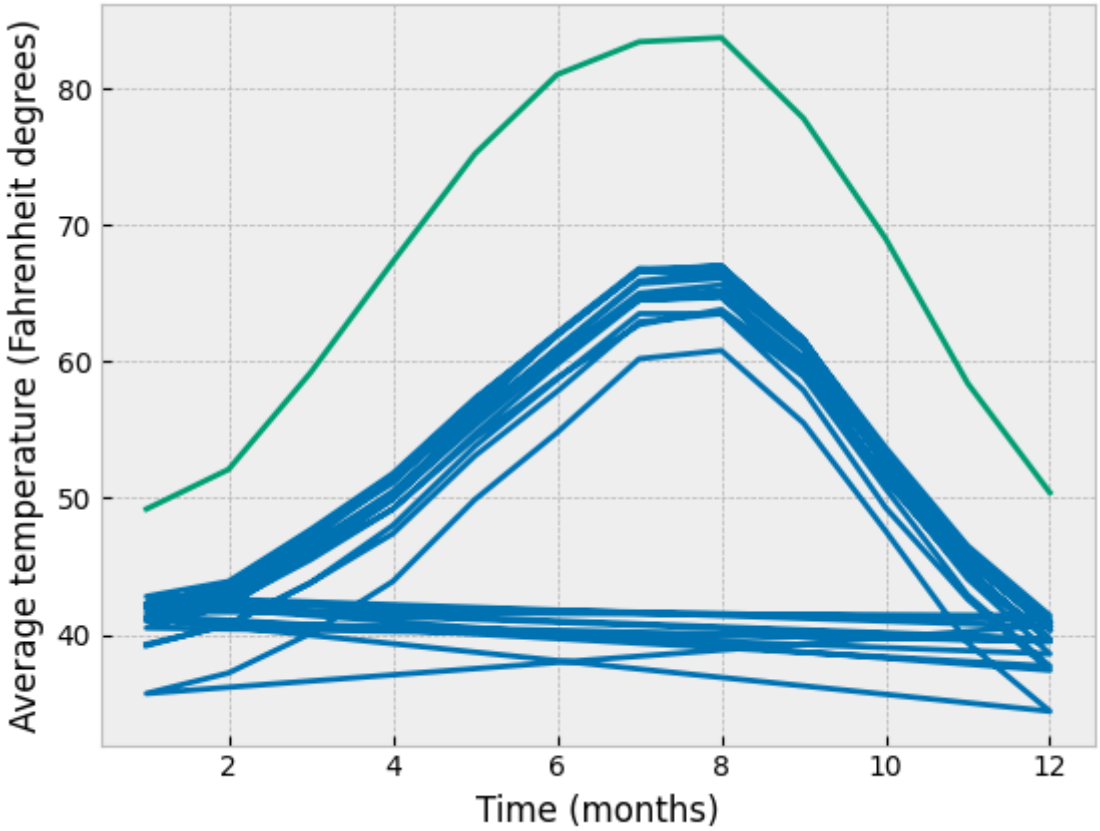

```
plt.style.use("ggplot")
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"], seattle_weather["MLY-TAVG-NORMAL"])
ax.plot(austin_weather["DATE"], austin_weather["MLY-TAVG-NORMAL"])
ax.set_xlabel("Time (months)")
ax.set_ylabel("Average temperature (Fahrenheit degrees)")
plt.show()
```

plt.show()




```
#back to default

plt.style.use("default")


#bmh style

plt.style.use("bmh")
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"], seattle_weather["MLY-TAVG-NORMAL"])
ax.plot(austin_weather["DATE"], austin_weather["MLY-TAVG-NORMAL"])
ax.set_xlabel("Time (months)")
ax.set_ylabel("Average temperature (Fahrenheit degrees)")
plt.show()
```

```
#seaborn styles

plt.style.use("seaborn-colorblind")
fig, ax = plt.subplots()
ax.plot(seattle_weather["DATE"], seattle_weather["MLY-TAVG-NORMAL"])
ax.plot(austin_weather["DATE"], austin_weather["MLY-TAVG-NORMAL"])
ax.set_xlabel("Time (months)")
ax.set_ylabel("Average temperature (Fahrenheit degrees)")
plt.show()
```




```
#Automating figures from data

sports = summer["Sport"].unique()
print(sports)
```

```
['Rowing' 'Taekwondo' 'Handball' 'Wrestling' 'Gymnastics' 'Swimming'
 'Basketball' 'Boxing' 'Volleyball' 'Athletics' 'Rugby Sevens' 'Judo'
 'Rhythmic Gymnastics' 'Weightlifting' 'Equestrianism' 'Badminton'
 'Water Polo' 'Football' 'Fencing' 'Shooting' 'Sailing' 'Beach Volleyball'
 'Canoeing' 'Hockey' 'Cycling' 'Tennis' 'Diving' 'Table Tennis'
 'Triathlon' 'Archery' 'Synchronized Swimming' 'Modern Pentathlon'
 'Trampolining' 'Golf']
```

```
#bar chart of heights for all sports

fig, ax = plt.subplots()
for sport in sports:
    sport_df = summer[summer["Sport"] == sport]
    ax.bar(sport, sport_df["Height"].mean())
```

```
    ax.bar(sport, sport_df["Height"].mean(),
           yerr=sport_df["Height"].std())
ax.set_ylabel("Height (cm)")
ax.set_xticklabels(sports, rotation=90)
plt.show()
```