BOOTCAMP DATA ETL

Make Data Powerful

Summary for today

I - Questions

II - Data Visualization

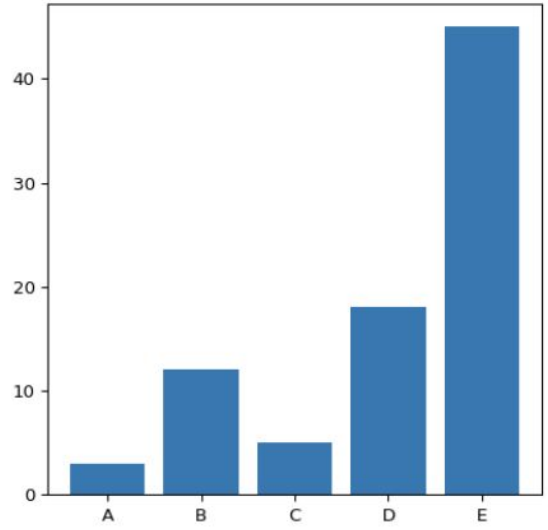
III - Practice

IV - Homework

Different kind of visualizations

Bar plot

A barplot (or barchart) is one of the most common types of graphic. It shows **the relationship between a numerical and a categorical variable**. Each entity of the categoric variable is represented as a bar. The size of the bar represents its numeric value.

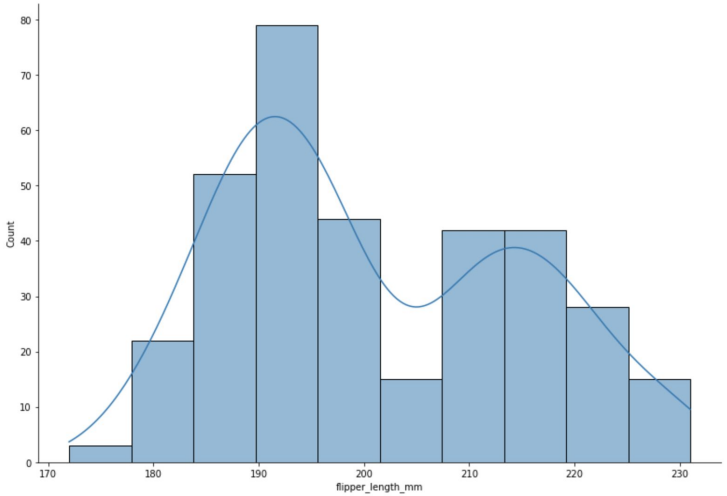



```
import numpy as np
import matplotlib.pyplot as plt

# Make a fake dataset:
height = [3, 12, 5, 18, 45]
bars = ('A', 'B', 'C', 'D', 'E')
y_pos = np.arange(len(bars))

# Create bars
plt.bar(y_pos, height)

# Create names on the x-axis
plt.xticks(y_pos, bars)

# Show graphic
plt.show()
```

Barcharts are often confounded with histograms, which is highly different. (It has only a numerical variable as input and shows its distribution).

Histogram

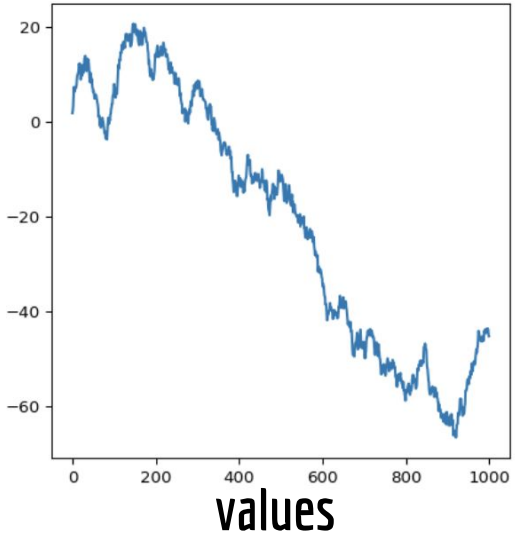
A histogram is an **accurate graphical representation of the distribution of a numeric variable**. It takes as input numeric variables only. The variable is cut into several bins, and the number of observation per bin is represented by the height of the bar.




```
import seaborn as sns

penguins =
sns.load_dataset("penguins")

a = sns.displot(data=penguins,
x="flipper_length_mm", kde=True)

a.fig.set_size_inches(12, 8)

plt.show(a)
```

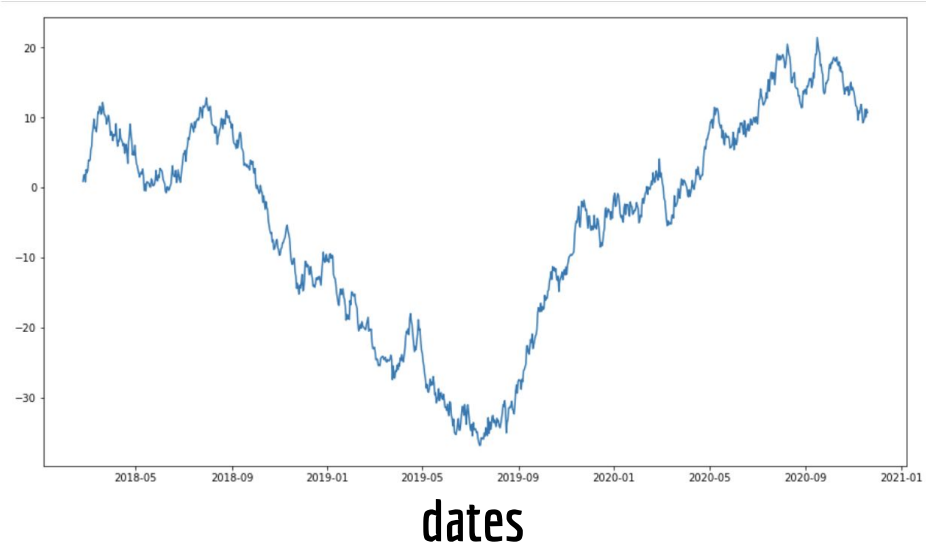
Line chart

A line chart or line graph displays the **evolution of one or several numeric variables**. Data points are connected by straight line segments. A line chart is often used to visualize a trend in data over intervals of time – a time series – thus the line is often drawn chronologically.




```
import matplotlib.pyplot as plt
import numpy as np

# create data
values=np.cumsum(np.random.randn(
1000,1))

# use the plot function
plt.plot(values)

plt.show()
```

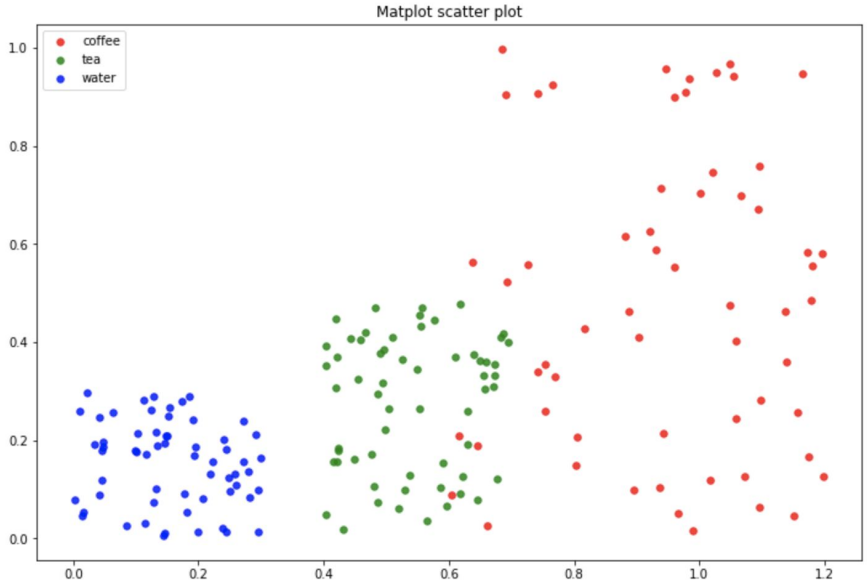
Line chart




```
import matplotlib.pyplot as plt
import numpy as np
from datetime import datetime,
timedelta

# create data
dates = [datetime.now() -
timedelta(number) for number in
range(len(values), 0, -1)]
values=np.cumsum(np.random.randn(
1000,1))


# use the plot function
plt.plot(dates, values)
plt.show()
```

Scatter plot

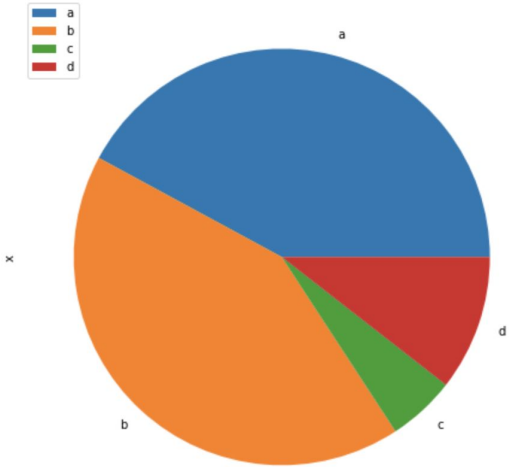
A scatterplot displays the **relationship between 2 numeric variables**. For each data point, the value of its first variable is represented on the X axis, the second on the Y axis.




```
import numpy as np
import matplotlib.pyplot as plt

# Create data
N = 60
g1 = (0.6 + 0.6 * np.random.rand(N),
np.random.rand(N))
g2 = (0.4+0.3 * np.random.rand(N),
0.5*np.random.rand(N))
g3 =
(0.3*np.random.rand(N),0.3*np.random.ran
d(N))

data = (g1, g2, g3)
colors = ("red", "green", "blue")
groups = ("coffee", "tea", "water")

# Create plot
fig = plt.figure(figsize=(12, 8))
for data, color, group in zip(data,
colors, groups):
    x, y = data
    plt.scatter(x, y, alpha=0.8,
c=color, s=30, label=group)

plt.title('Matplot scatter plot')
plt.legend(loc=2)
plt.show()
```

Pie Chart

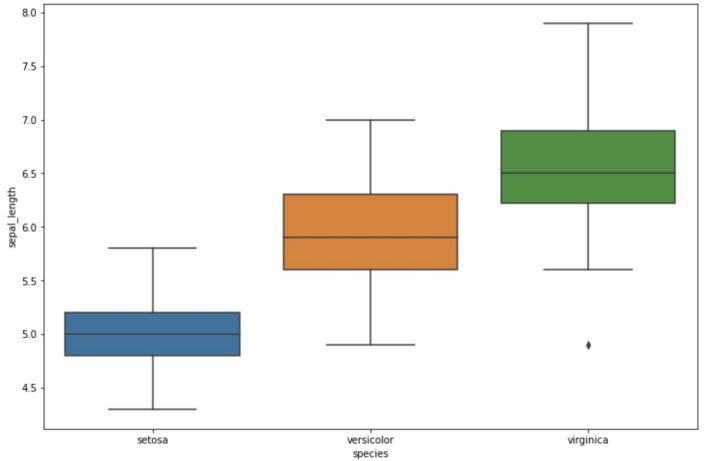
A pie chart is **a circle divided into sectors that each represent a proportion of the whole**. It is often used to show percentage, where the sum of the sectors equals 100%




```
# Pie chart, where the slices will be
ordered and plotted counter-clockwise:
labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
explode = (0, 0.1, 0, 0)  # only
"explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode,
labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio
ensures that pie is drawn as a circle.
plt.title('Pie chart example')
plt.show()
```

Boxplot

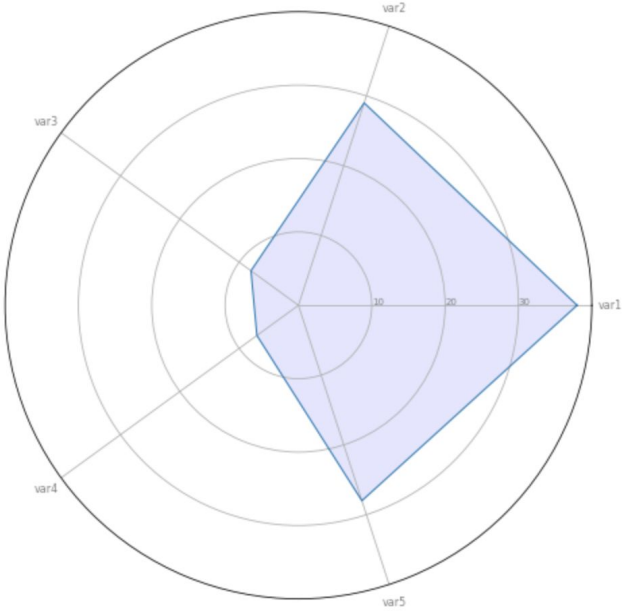
Boxplot gives a **nice summary of one or several numeric variables.** The line that divides the box into 2 parts represents the **median** of the data. The **end of the box shows the upper and lower quartiles**. The **extreme lines shows the highest and lowest value excluding outliers**. Note that boxplot hide the number of values existing behind the v




```
df = sns.load_dataset('iris')
a = sns.boxplot(x=df["species"],
y=df["sepal_length"])
a.figure.set_size_inches(12, 8)
plt.show(a)
```

Spider plot

A Radar chart or Spider plot or Polar chart or Web chart allows to **study the feature of one or several individuals for several numerical variables**.

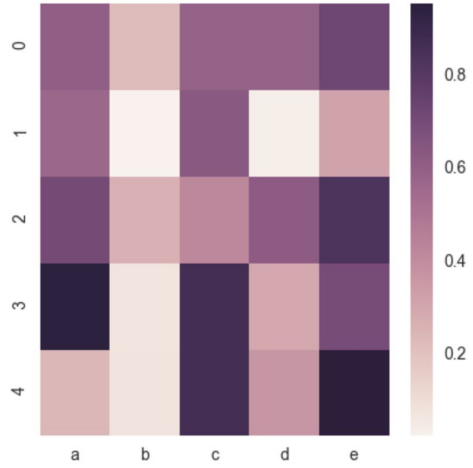



```
from math import pi

df = pd.DataFrame({
'var1': [38, 1.5, 30, 4],
'var2': [29, 10, 9, 34],
'var3': [8, 39, 23, 24],
'var4': [7, 31, 33, 14],
'var5': [28, 15, 32, 14]
})

# number of variable
categories=list(df)
N = len(categories)

# We are going to plot the first line of the data frame.
# But we need to repeat the first value to close the circular
graph:
values= df.loc[0, :].values.tolist()
values += values[:1]
values

# What will be the angle of each axis in the plot? (we divide the
plot / number of variable)
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]

# Initialise the spider plot
plt.figure(figsize=(14, 8))
ax = plt.subplot(111, polar=True)

# Draw one axe per variable + add labels labels yet
plt.xticks(angles[:-1], categories, color='grey', size=8)

# Draw ylabels
ax.set_rlabel_position(0)
plt.yticks([10,20,30], ["10","20","30"], color="grey", size=7)
plt.ylim(0,40)

# Plot data
ax.plot(angles, values, linewidth=1, linestyle='solid')

# Fill area
ax.fill(angles, values, 'b', alpha=0.1)
plt.show()
```

Heat Map

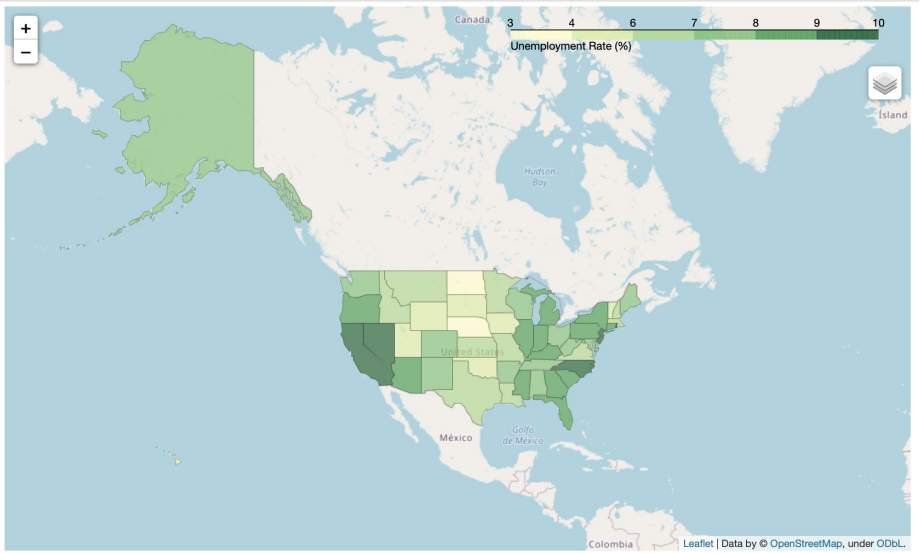
A heat map (or heatmap) is a **graphical representation of data where the individual values contained in a matrix** are represented as colors. It is a bit like looking a data table from above. It is really useful to display a general view of numerical data, not to extract specific data point.

```
import seaborn as sns
import pandas as pd
import numpy as np

# Create a dataset (fake)
df = pd.DataFrame(np.random.random((5,5)),
columns=["a","b","c","d","e"])

# Default heatmap: just a visualization of this square matrix
p1 = sns.heatmap(df)
```



Map plot

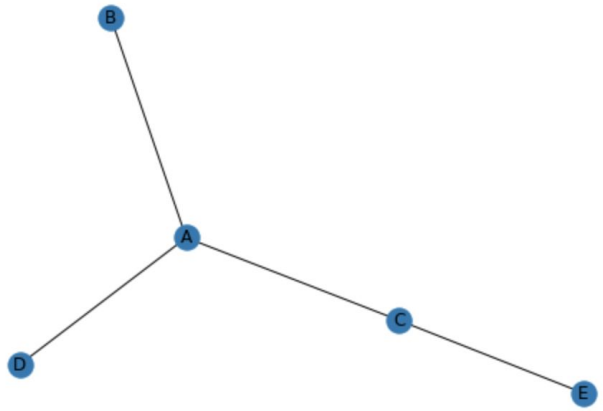
A map plot displays divided geographical areas or regions that are coloured, shaded or patterned in relation to a data variable. It allows to study how a variable evolute along a territory.




```
import folium

url =
'https://raw.githubusercontent.com/python-visu
alization/folium/master/examples/data'
state_geo = f'{url}/us-states.json'
state_unemployment =
f'{url}/US_Unemployment_Oct2012.csv'
state_data = pd.read_csv(state_unemployment)

m = folium.Map(location=[48, -102],
zoom_start=3)

folium.Choropleth(
    geo_data=state_geo,
    name='choropleth',
    data=state_data,
    columns=['State', 'Unemployment'],
    key_on='feature.id',
    fill_color='YlGn',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Unemployment Rate (%)'
).add_to(m)

folium.LayerControl().add_to(m)

m
```

Network chart

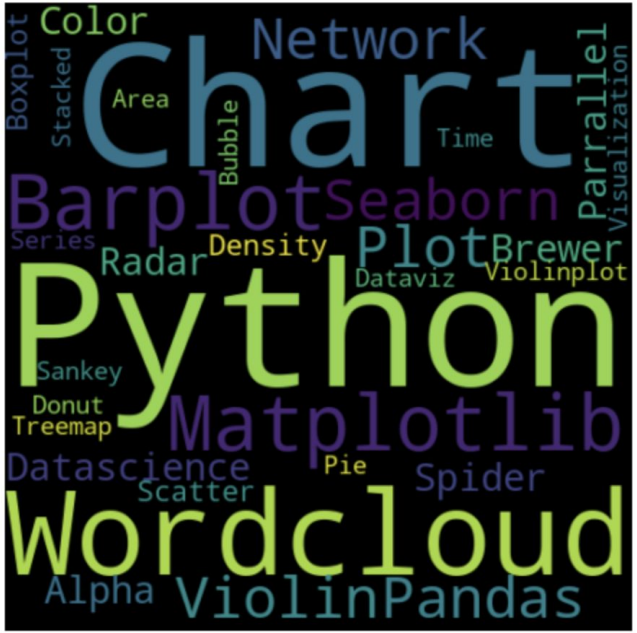
A Network diagrams (or chart, or graph) **show interconnections between a set of entities**. Each entity is represented by a Node (or vertices). Connection between nodes are represented through links (or edges).




```
import networkx as nx

# Build a dataframe with 4 connections
df = pd.DataFrame({ 'from':['A', 'B',
'C','A'], 'to':['D', 'A', 'E','C']})
df

# Build your graph
G=nx.from_pandas_edgelist(df, 'from', 'to')

# Plot it
nx.draw(G, with_labels=True)
plt.show()
```

WordCloud

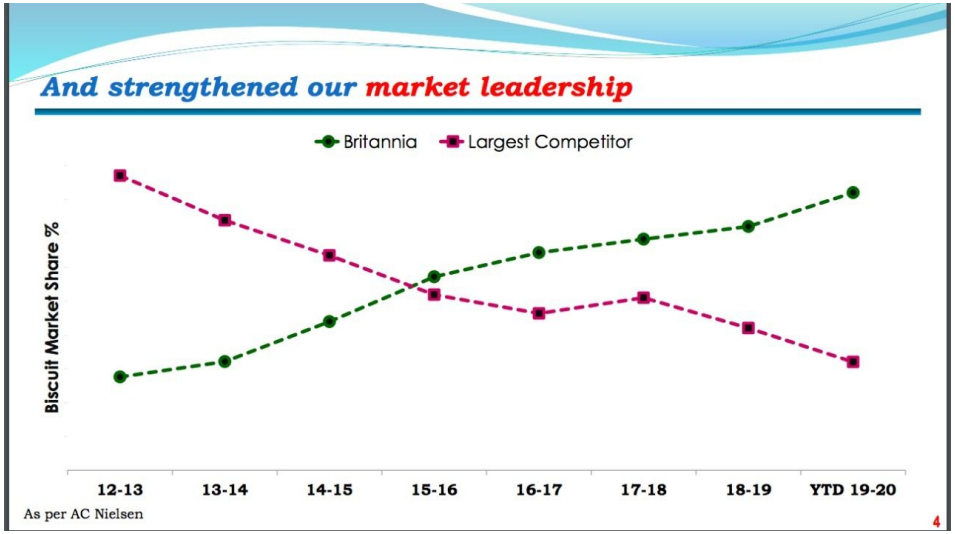
A Wordcloud is a **visual representation of text data**. It displays a list of words, the importance of each being shown with font size or color.




```
from wordcloud import WordCloud

# Create a list of word
text=("Python Python Python Matplotlib
Matplotlib Seaborn Network Plot Violin Chart
Pandas Datascience Wordcloud Spider Radar
Parrallel Alpha Color Brewer Density Scatter
Barplot Barplot Boxplot Violinplot Treemap
Stacked Area Chart Chart Visualization Dataviz
Donut Pie Time-Series Wordcloud Wordcloud
Sankey Bubble")

# Create the wordcloud object
wordcloud = WordCloud(width=480, height=480,
margin=0).generate(text)

# Display the generated image:
plt.figure(figsize=(14, 8))
plt.imshow(wordcloud,
interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

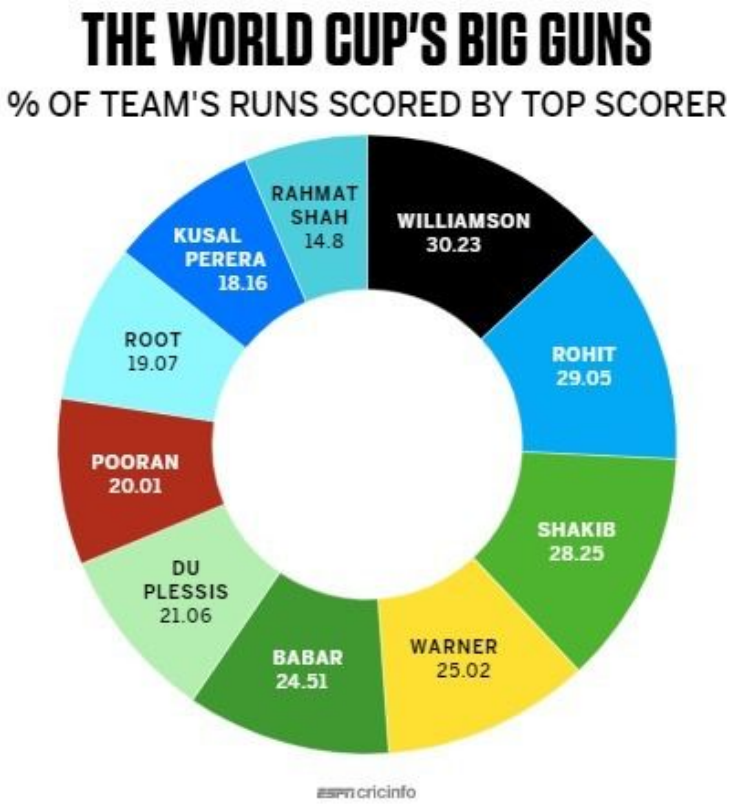
Examples of visualizations

Quizz: Is this a good visualization ?



Where are biscuit market share % values ??
Where is the Y axis ??

Quizz: Is this a good visualization ?

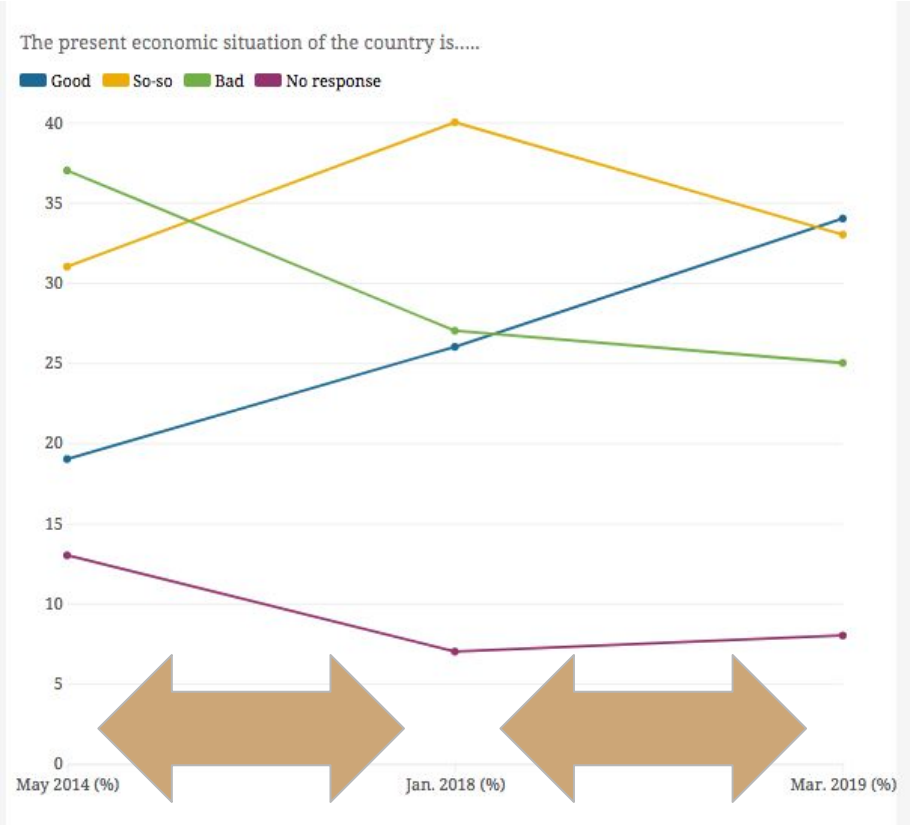


14.8 + 18.16 + 19.07 + 20.01 + 21.06 + 24.51 + 25.02 + 28.25 + 29.05 + 30.23

= 230.16

The data doesn't add up to 100%. The sectors of the pie don't represent mutually exclusive and collectively exhaustive parts of anything. This is absolute nonsense.

Quizz: Is this a good visualization ?



Distance between May 2014 and January 2018 (44 months) is the same as the distance between January 2018 and March 2019 (15 months). It's because Dates was encoded like a categorical variable not a date...

More bad visualization

https://badvisualisations.tumblr.com

Data visualization tools

Python

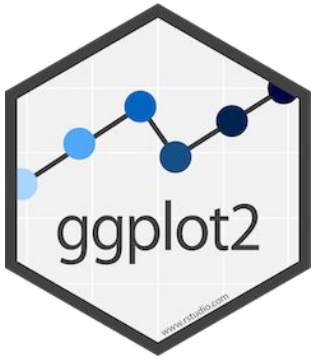
Basic libraries

matplotlib

seaborn

Advanced libraries

plotly



bokeh

Folium

Tools on the shelf

Google Data Studio

Exercises

https://drive.google.com/drive/folders/1Y_XFP42jKsu6lUGnF4g5dIfyPoB60GW5?usp=sharing

Data Analysis on Covid data

Take a notebook and try to analyse data with visualization to show some relevant information about Covid-19.

Instructions:

- You can use all the Python visualization libraries that you want.
- You need to use **at least 3 different kinds of visualization**.
- You need to have **at least 5 plots in your notebook.**
- You need to **explain in few words each visualization** :
  - Why did you choose this kind of visualization ?
  - How is this visualization relevant to understand the dataset ?
- Your submission is **only your notebook** (=> **.ipynb**)
- You need to **submit it before Sunday 12th November 23h59 on Github**

Data to use:

https://raw.githubusercontent.com/owid/covid-19-data/master/public/data/owid-covid-data.csv

Data Description :

https://github.com/owid/covid-19-data/blob/master/public/data/owid-covid-codebook.csv