NYU | TANDON SCHOOL OF ENGINEERING

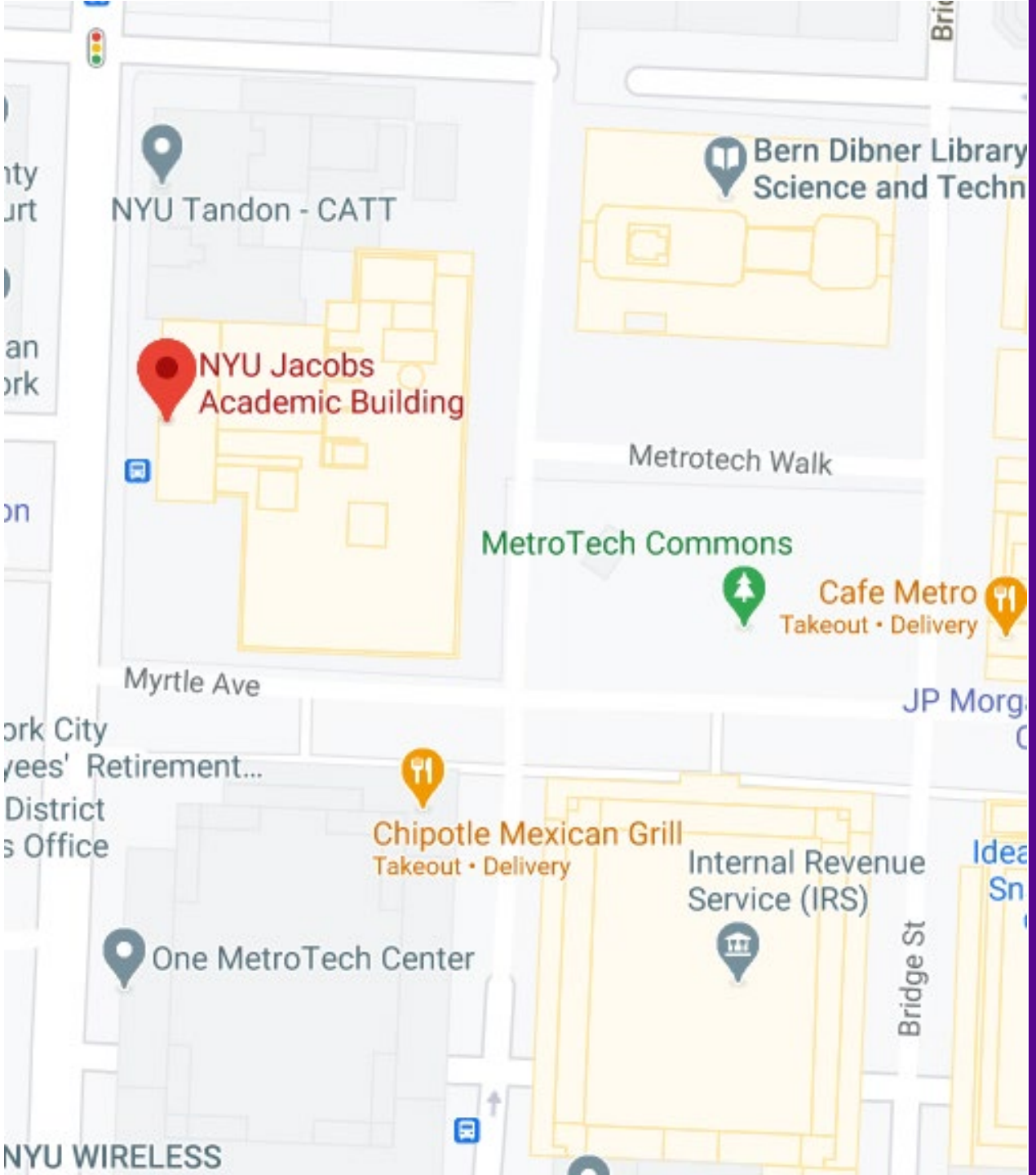
Lecture 4
Charts

Programming for Business Analytics
MG-GY 8401

NYU TANDON SCHOOL OF ENGINEERING

Agenda

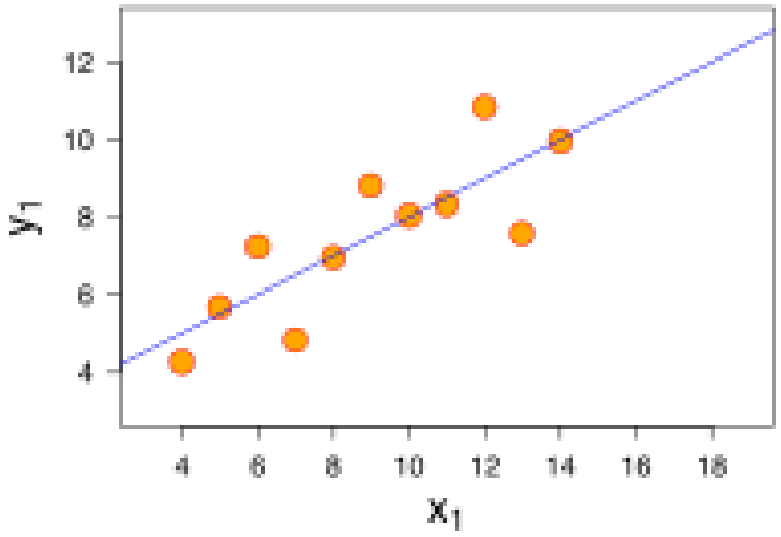
- Charts
  - Figures + Axes
- Tables
  - File Input/Output

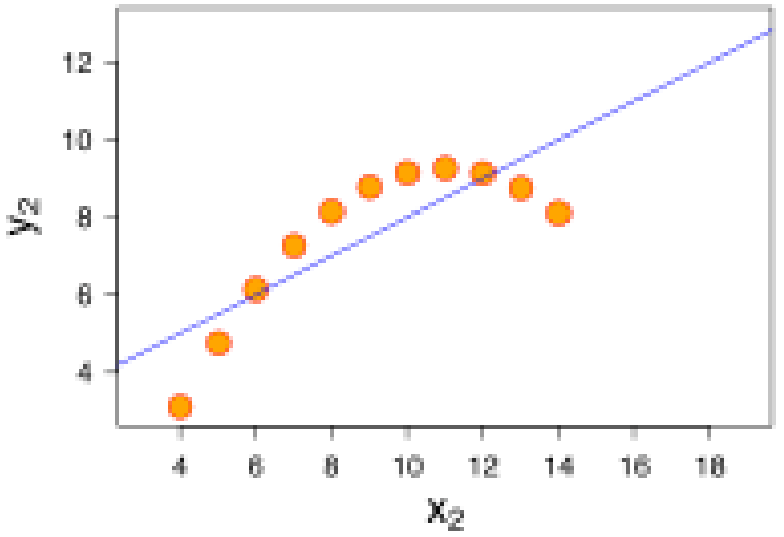


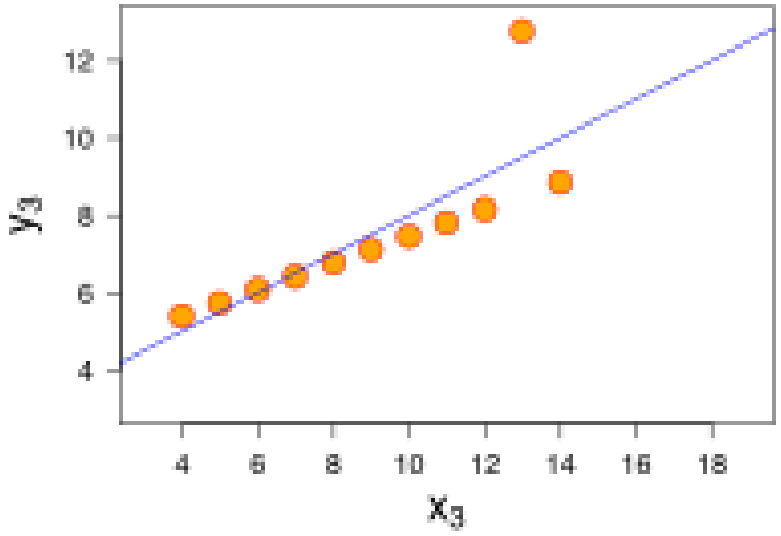
NYU TANDON SCHOOL OF ENGINEERING

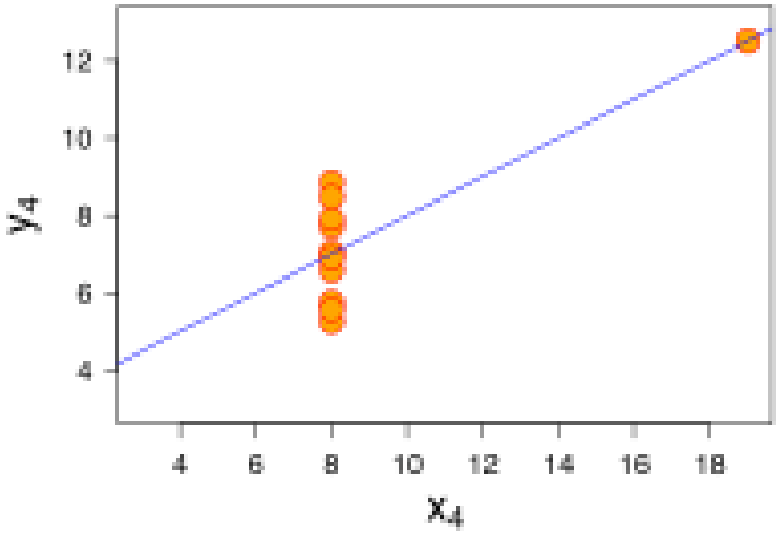
Logistics

- Homework
  - Homework 4
  - Homework 3


12
10
8
6
4
y_1
4 6 8 10 12 14 16 18
x_1

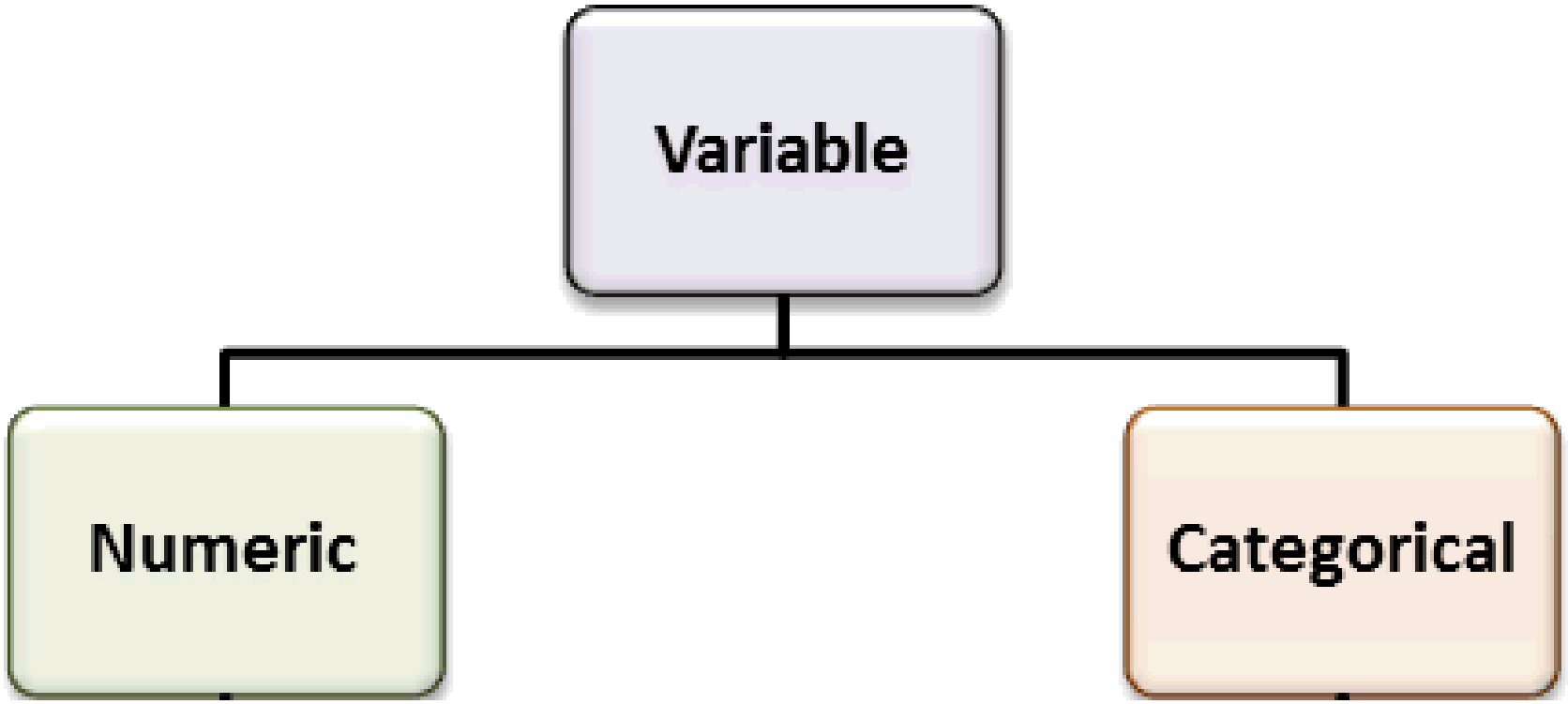

12
10
8
6
4
y_2
4 6 8 10 12 14 16 18
x_2


12
10
8
6
4
y_3
4 6 8 10 12 14 16 18
x_3


12
10
8
6
4
y_4
4 6 8 10 12 14 16 18
x_4

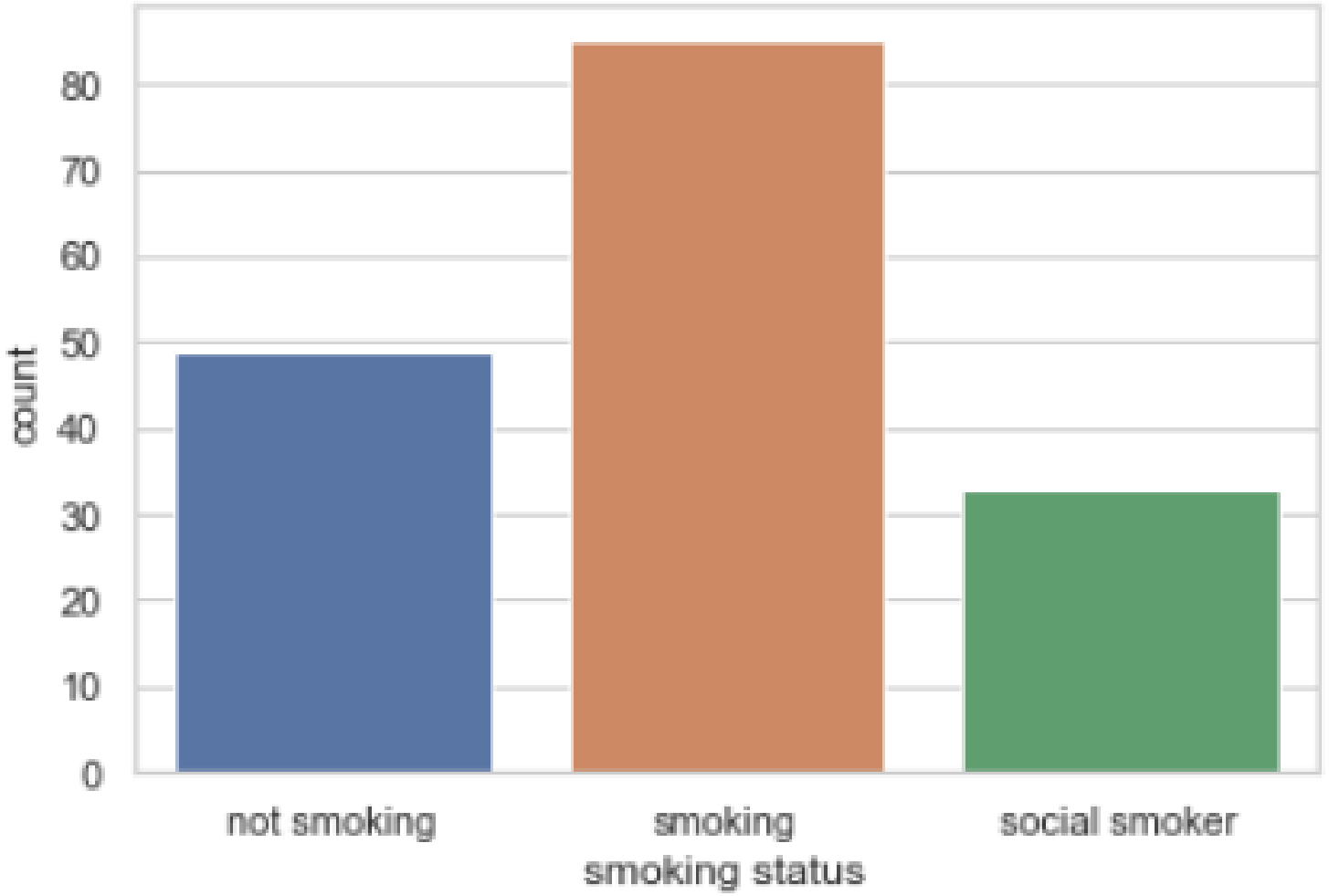
CC BY NC ND

Data Types



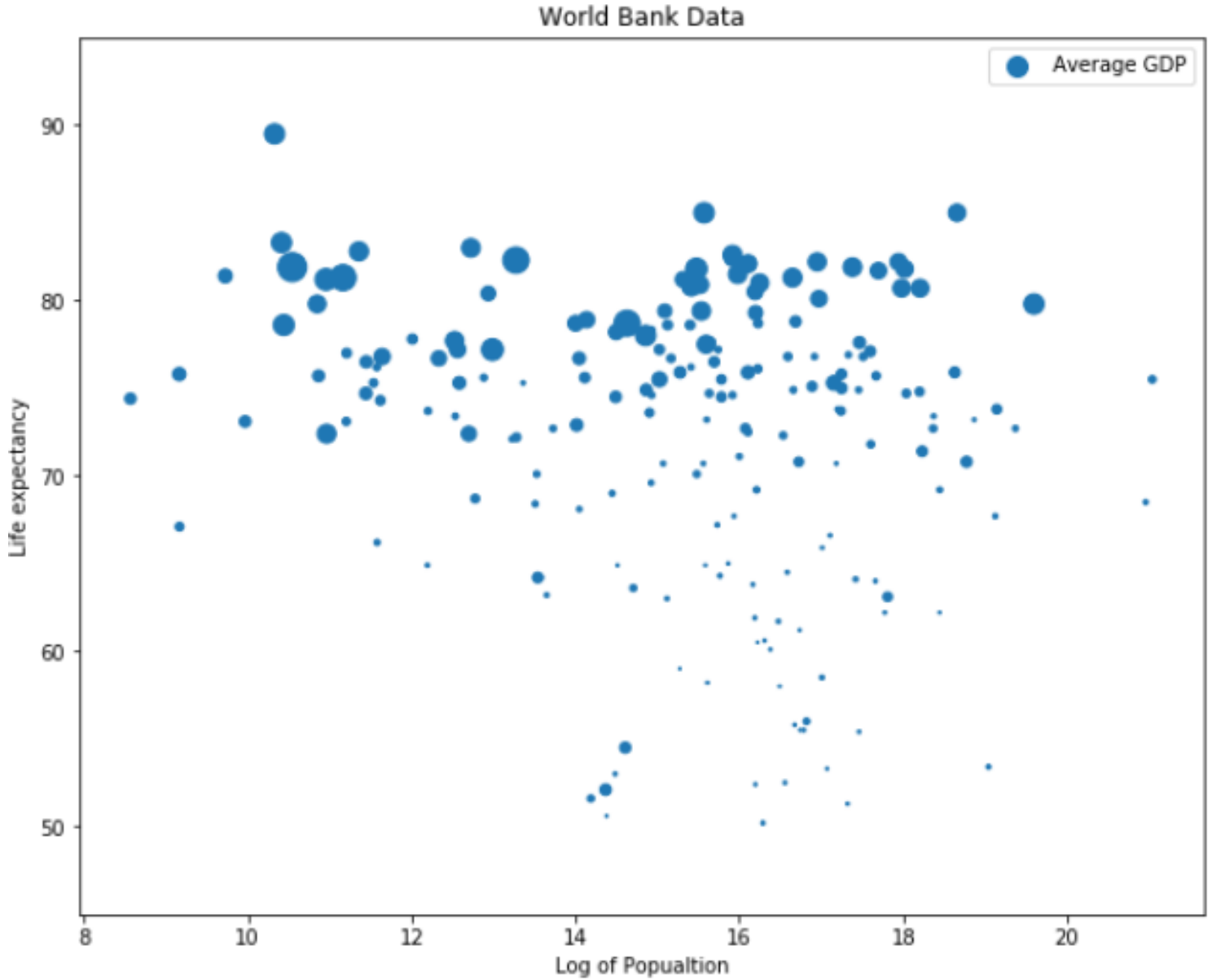
CC BY NC ND

Bar-Chart



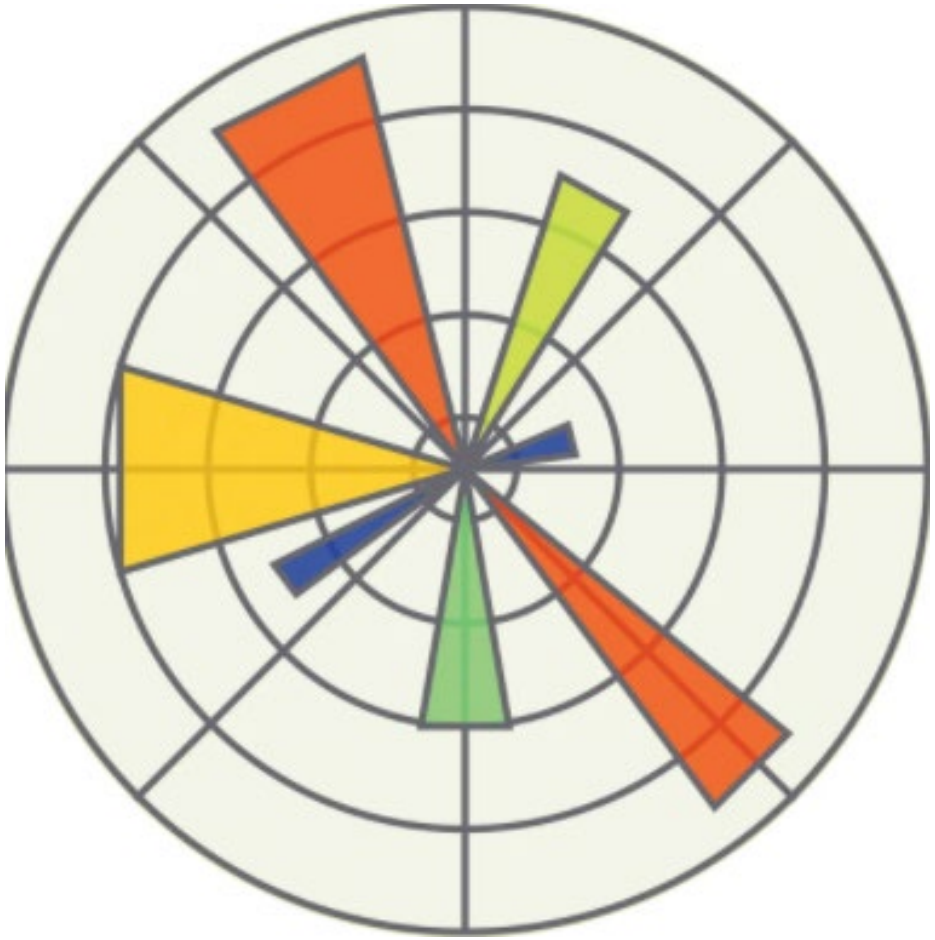
CC BY NC ND

Scatter-Plot



CC BY NC ND

matplotlib



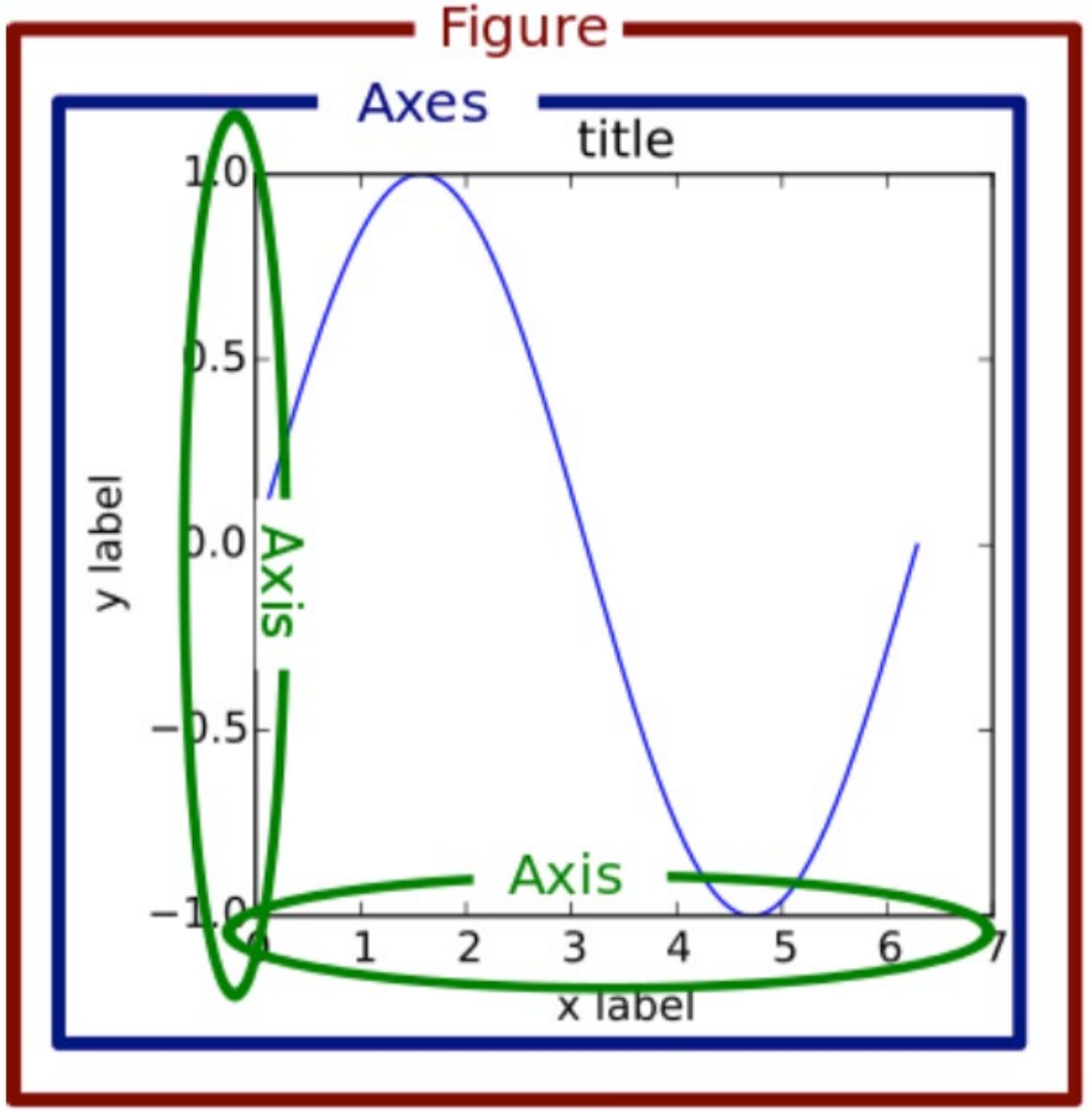
- import matplotlib as mpl
- import matplotlib.pyplot as plt

CC BY NC ND

A ***figure*** holds multiple ***axes***.

An axis contains the information for a chart

- numbers
- tick markers
- tick labels

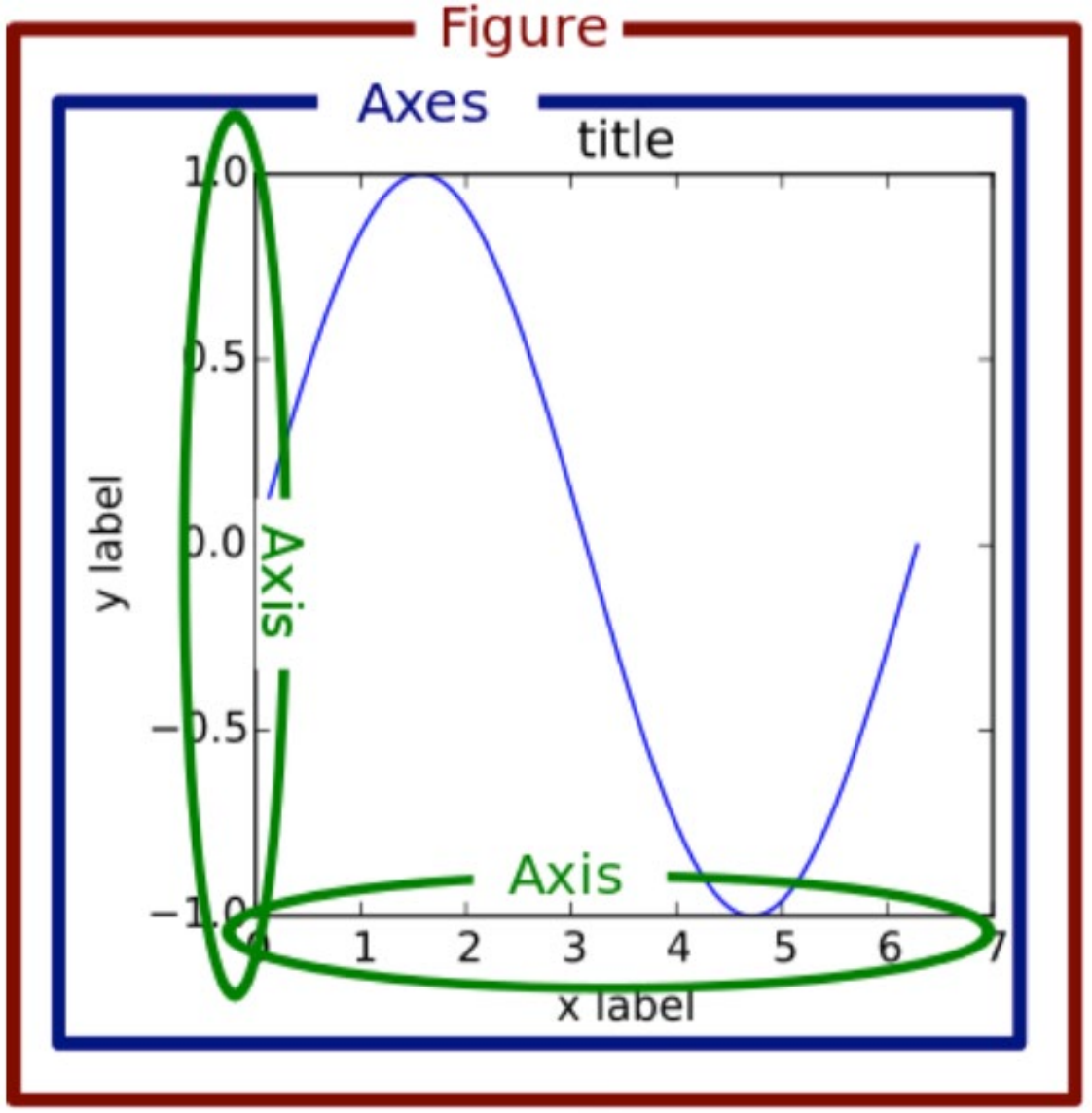


CC BY NC ND

A ***patch*** is a collection of ***artists***.

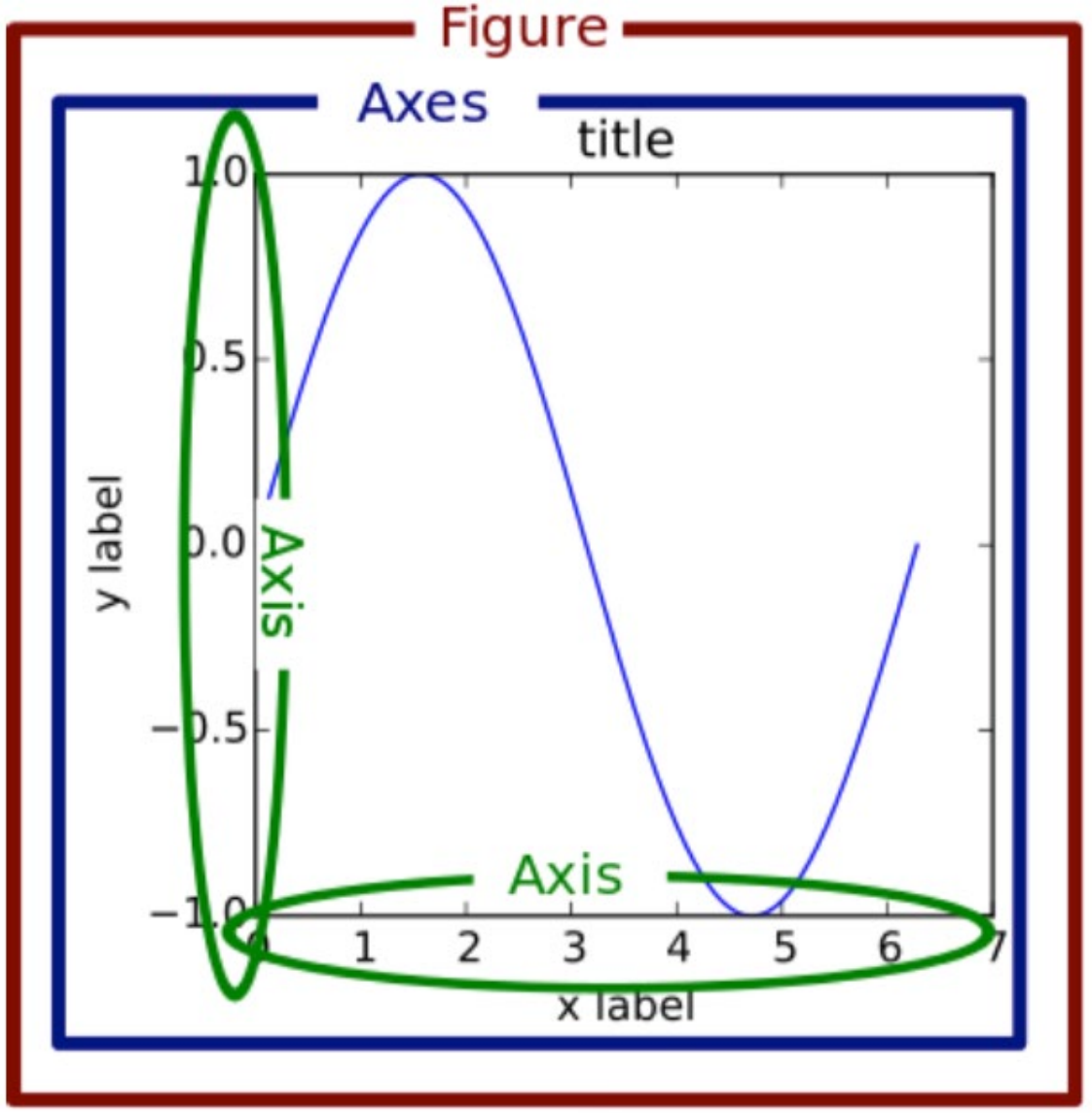
An artist contains the graphics corresponding to the numbers in the axes

- color
- shape
- style



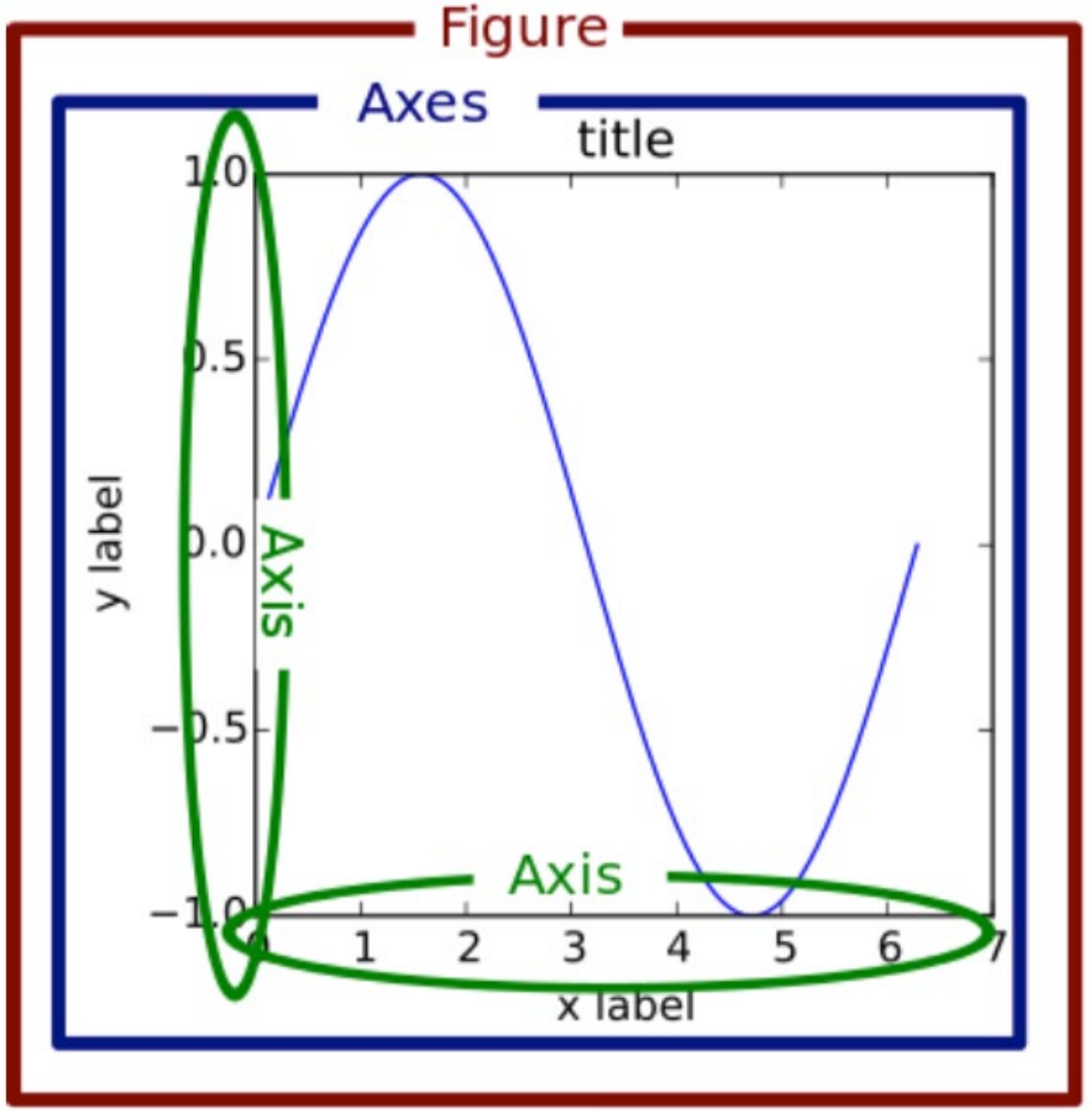
CC BY NC ND

- Many commands change the current figure/axes
- If we have a current figure/axes then we can access them
  - `gca()` function returns the current axes,
  - `sca()` function sets the current axes
  - `cla()` function clears the current axes



CC BY NC ND

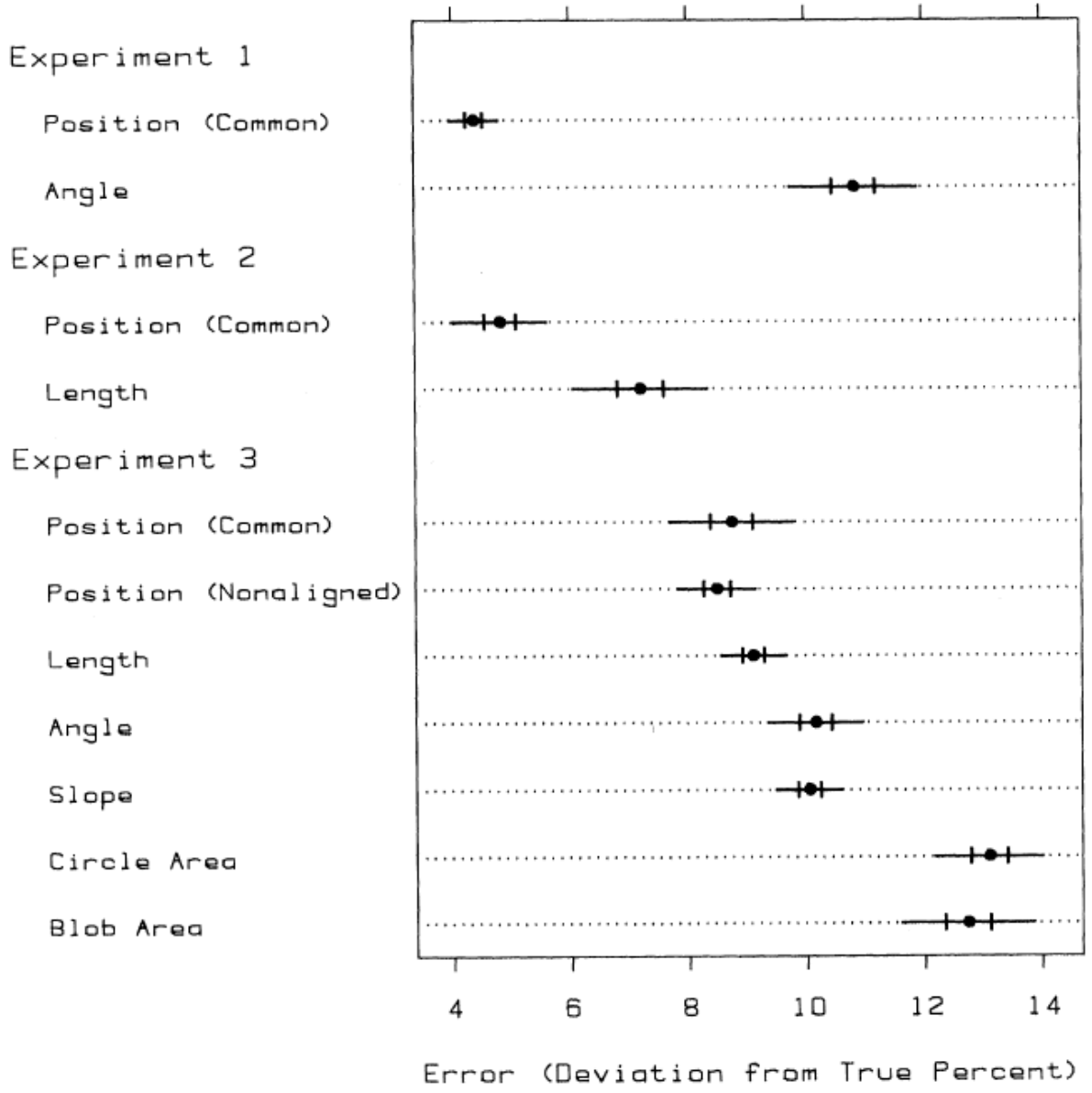
- Many commands change the current figure/axes
- If we have a current figure/axes then we can access them
  - `gcf()` function returns the current figure
  - `figure()` function sets the current figure or creates a new figure
  - `clf()` function clears the current figure



CC BY NC ND

Which aspect of a chart would be most perceptible?

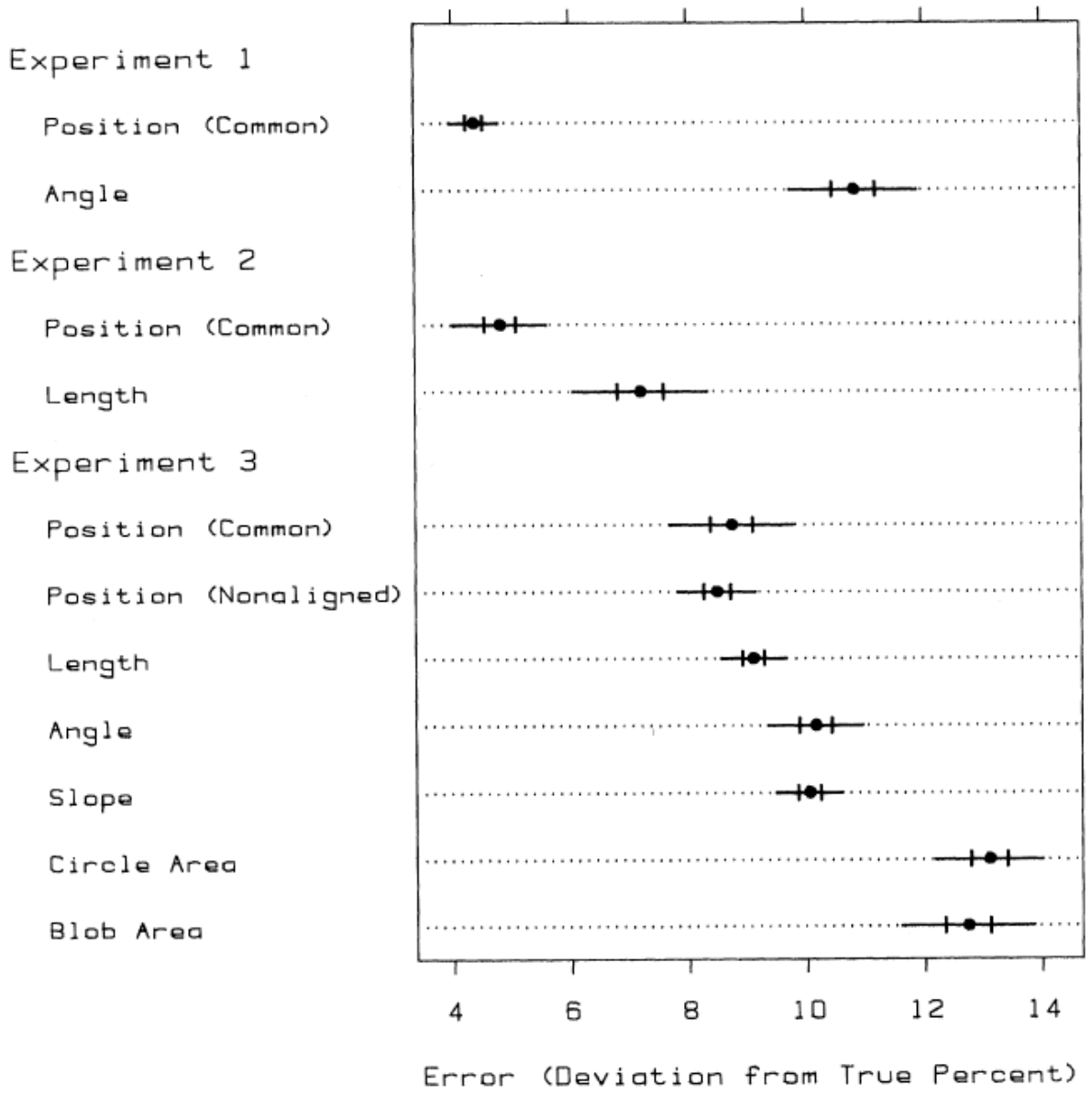
1. Position
2. Length
3. Angle
4. Area
5. Shading



CC BY NC ND

Which aspect of a chart would be most perceptible?

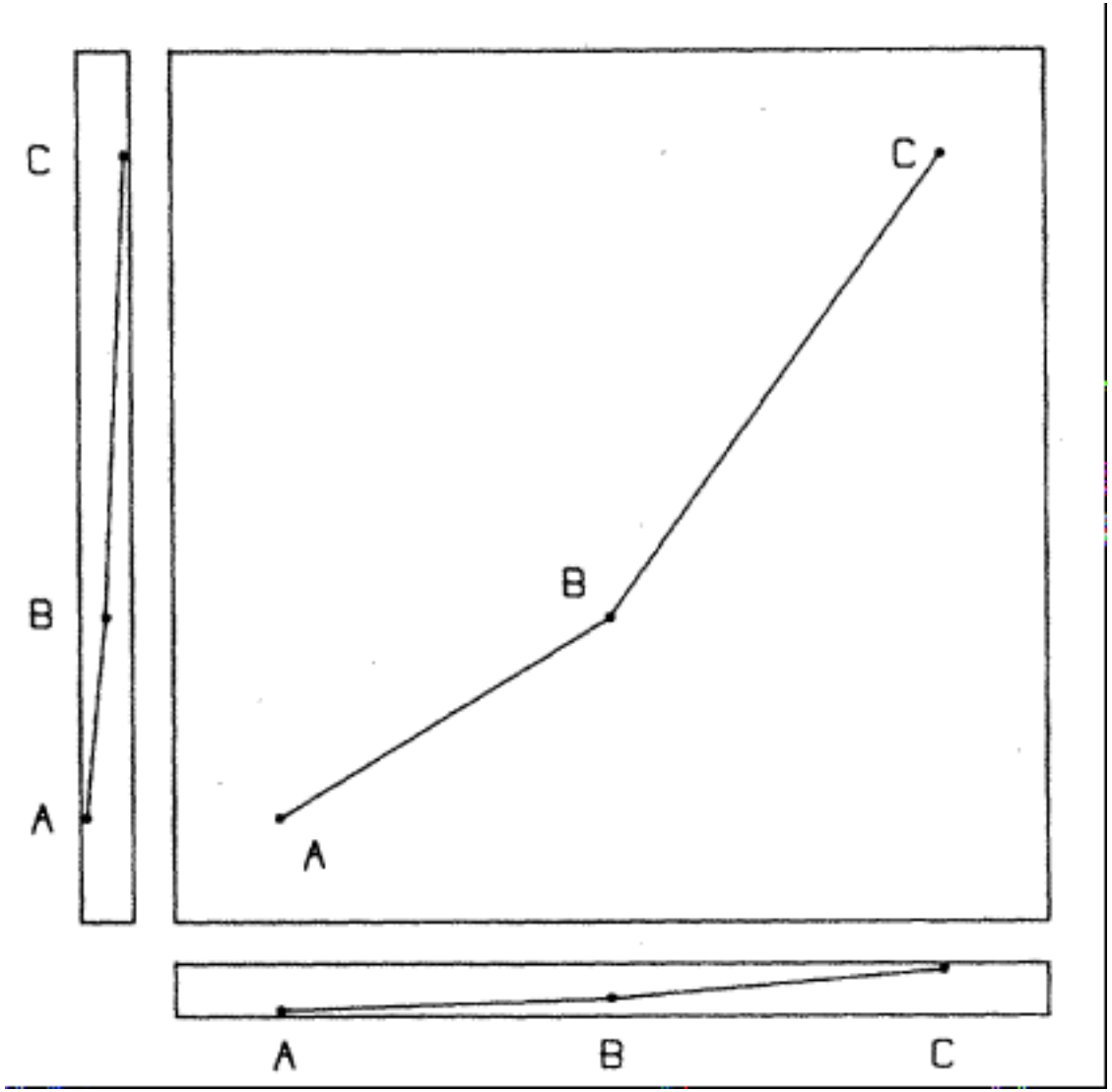
1. Position
2. Length
3. Angle
4. Area
5. Shading



CC BY NC ND

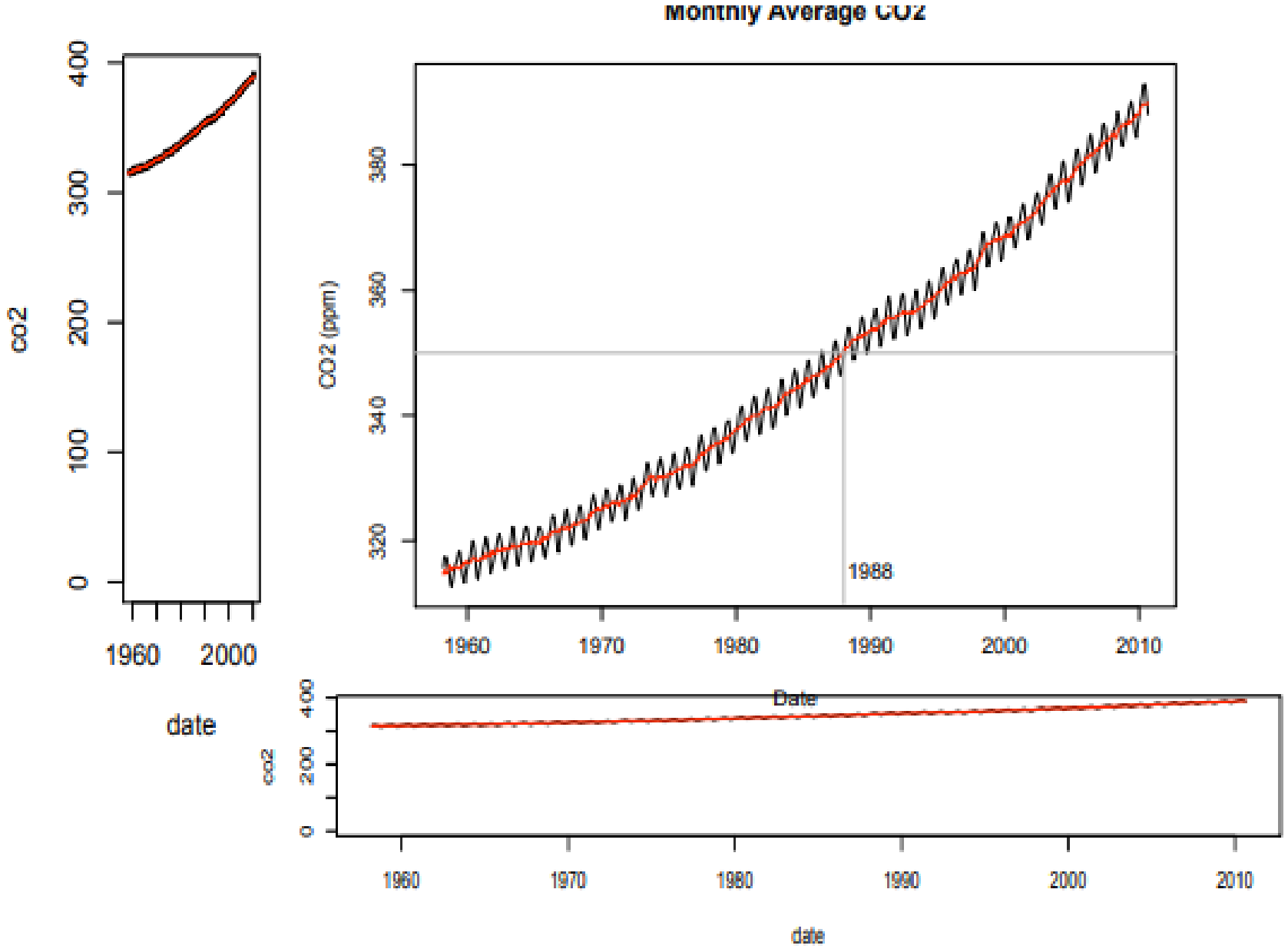
Which aspect of a chart would be most perceptible?

1. Position
2. Length
3. Angle
4. Area
5. Shading



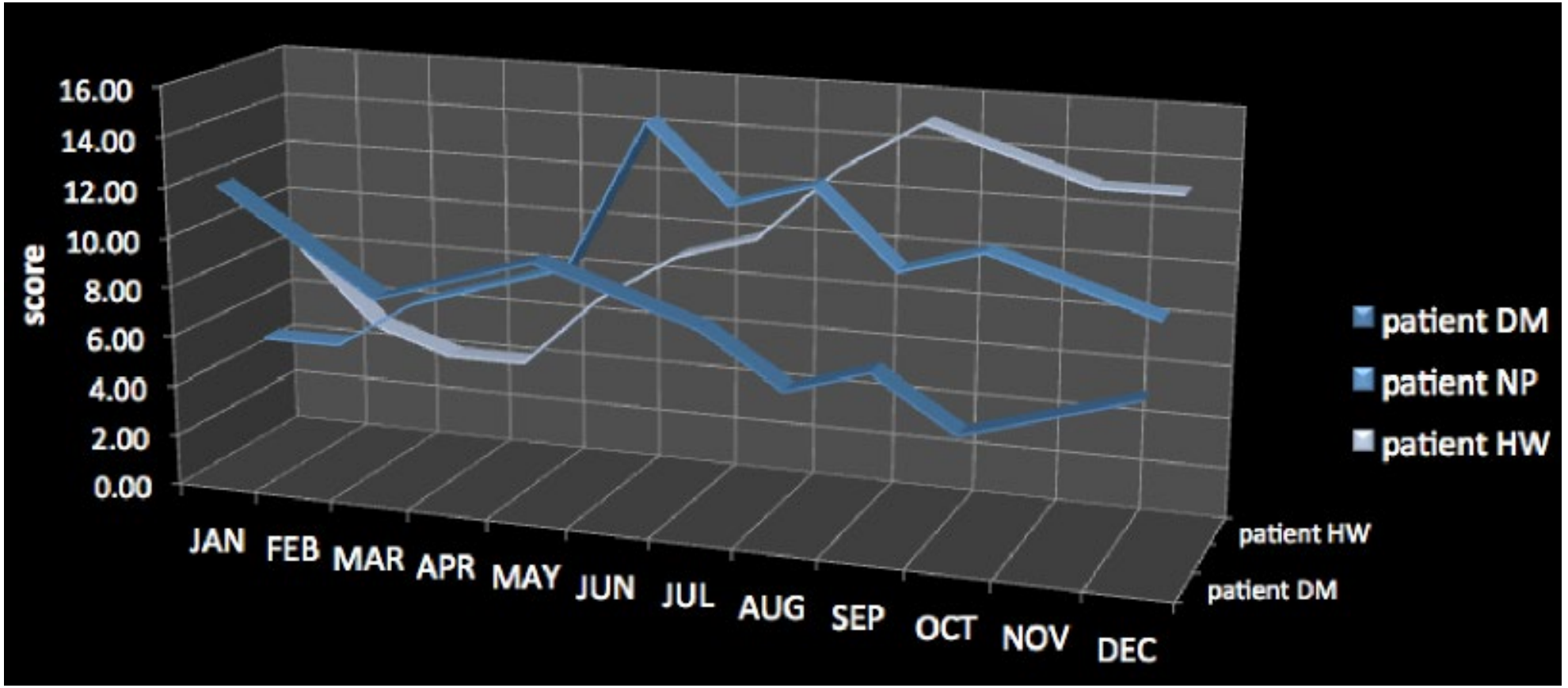
CC BY NC ND

Scale

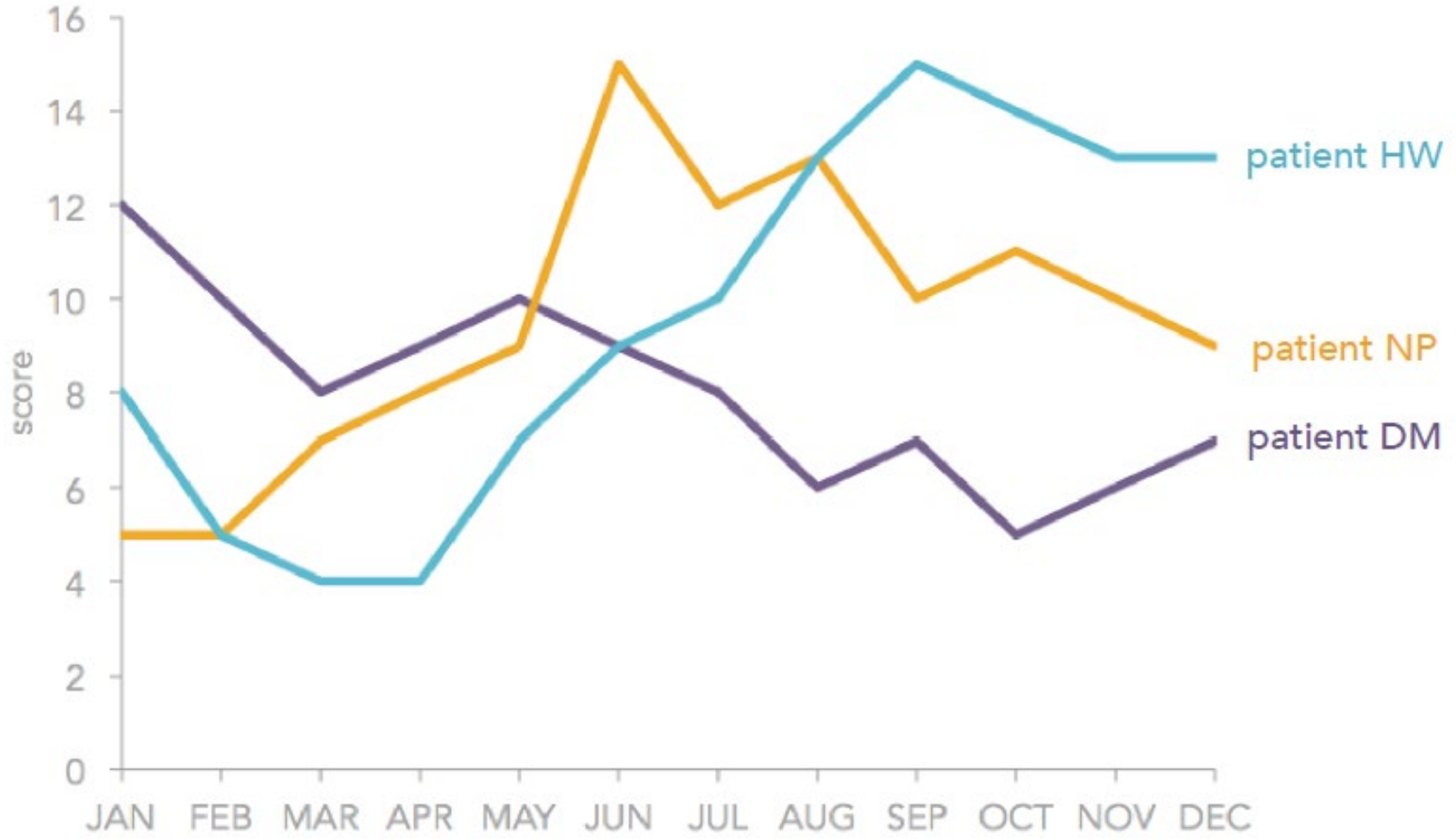


CC BY NC ND

Recommended Practices

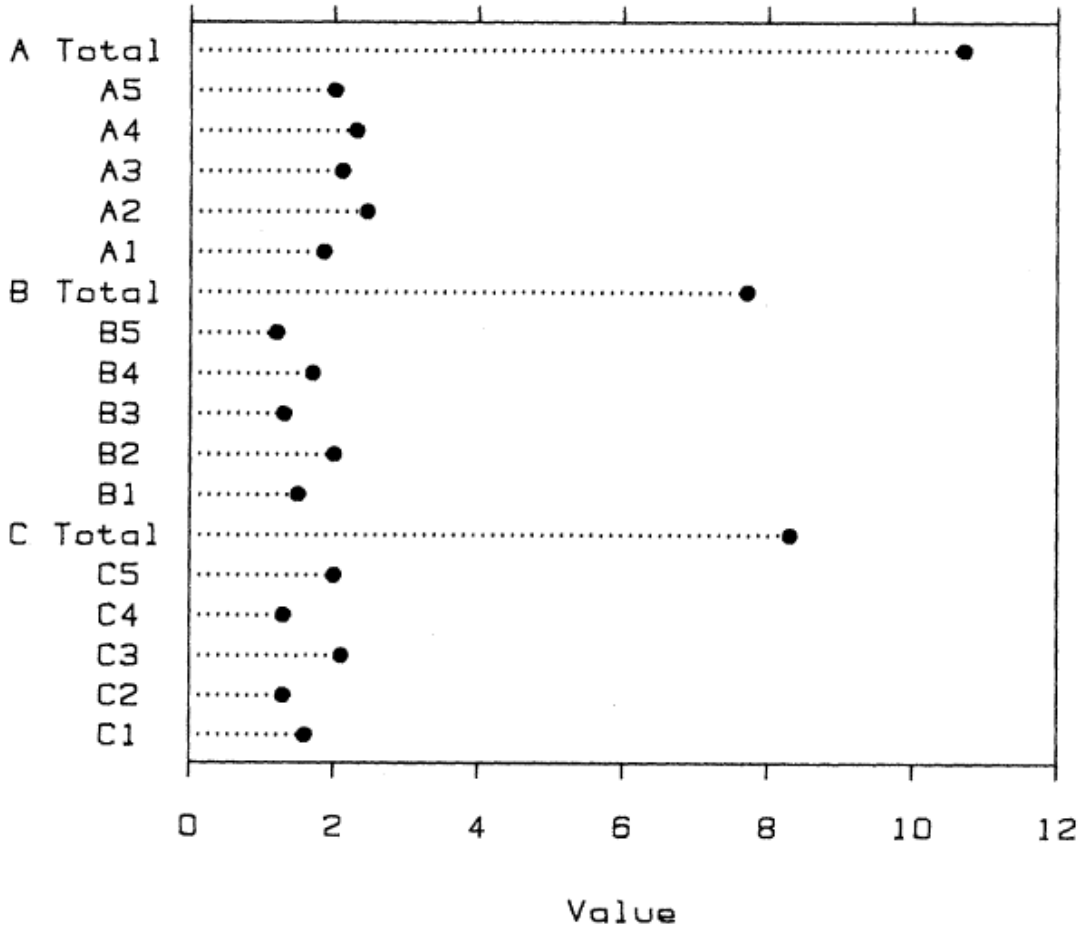


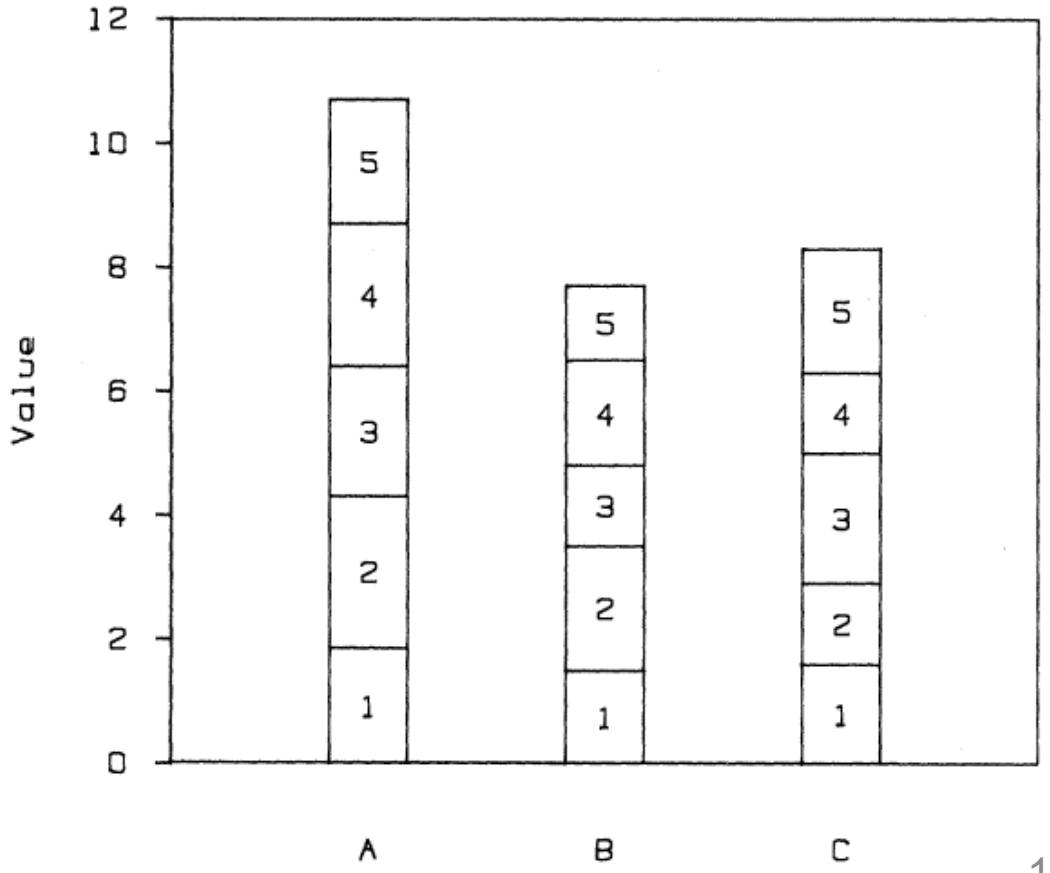
CC BY NC ND


16
14
12
10
8
6
4
2
0
score
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
patient HW
patient NP
patient DM

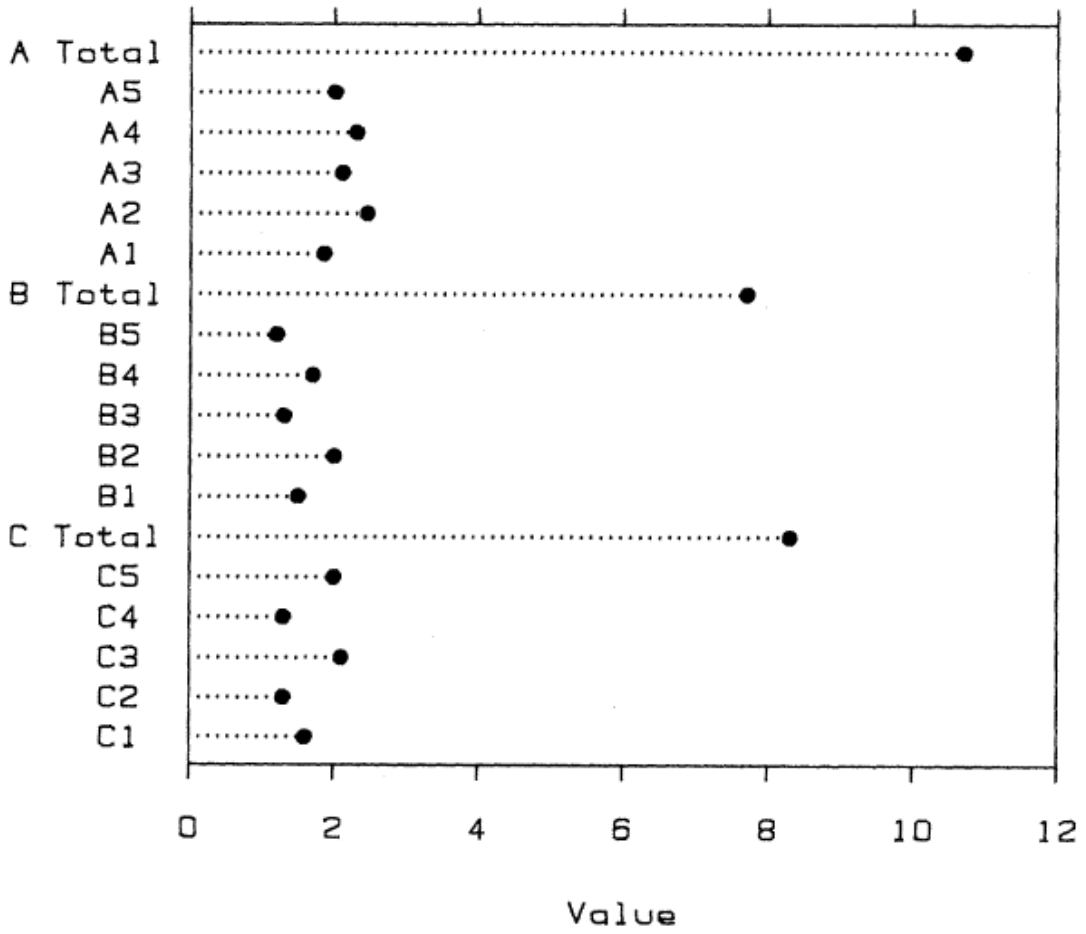
CC BY NC ND

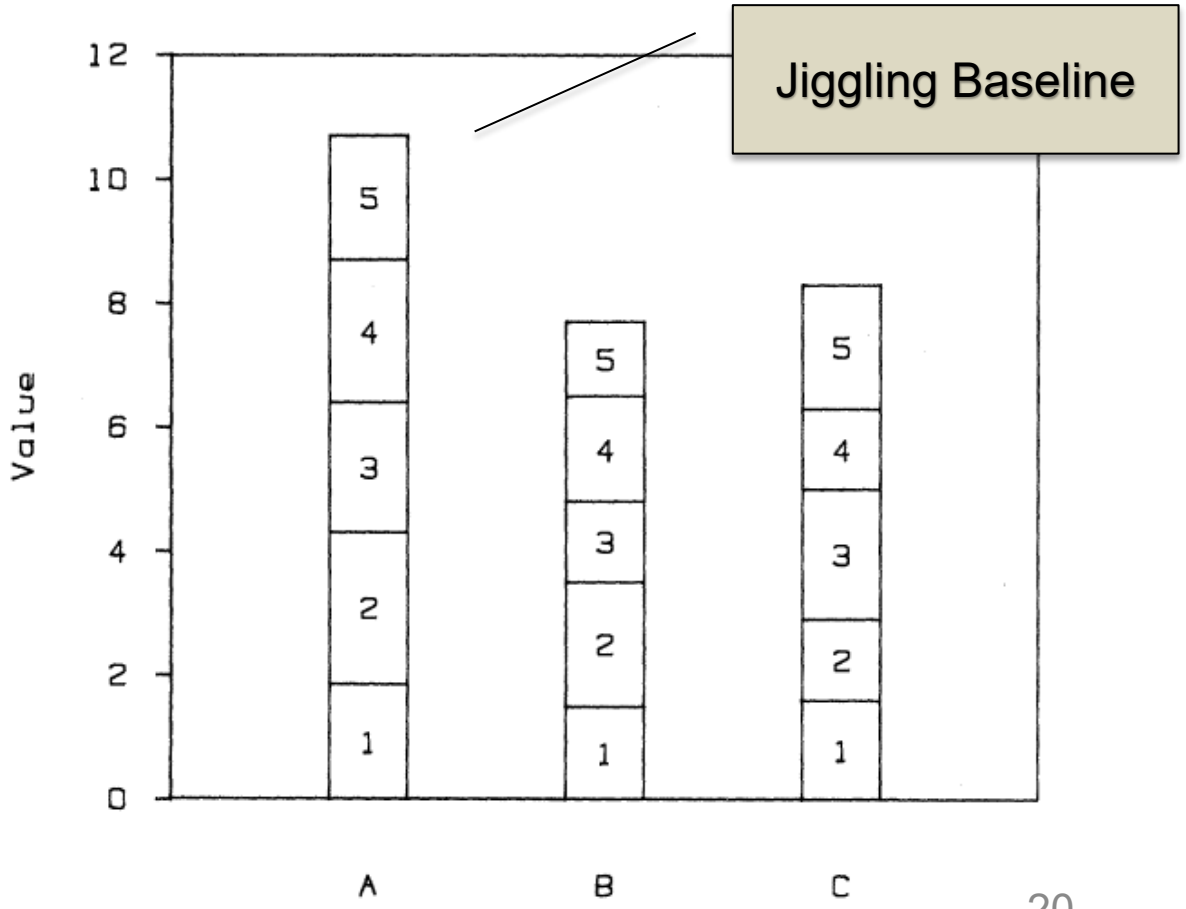
We have two different graphical representations of the same dataset. Which chart is more understandable?





We have two different graphical representations of the same dataset. Which chart is more understandable?





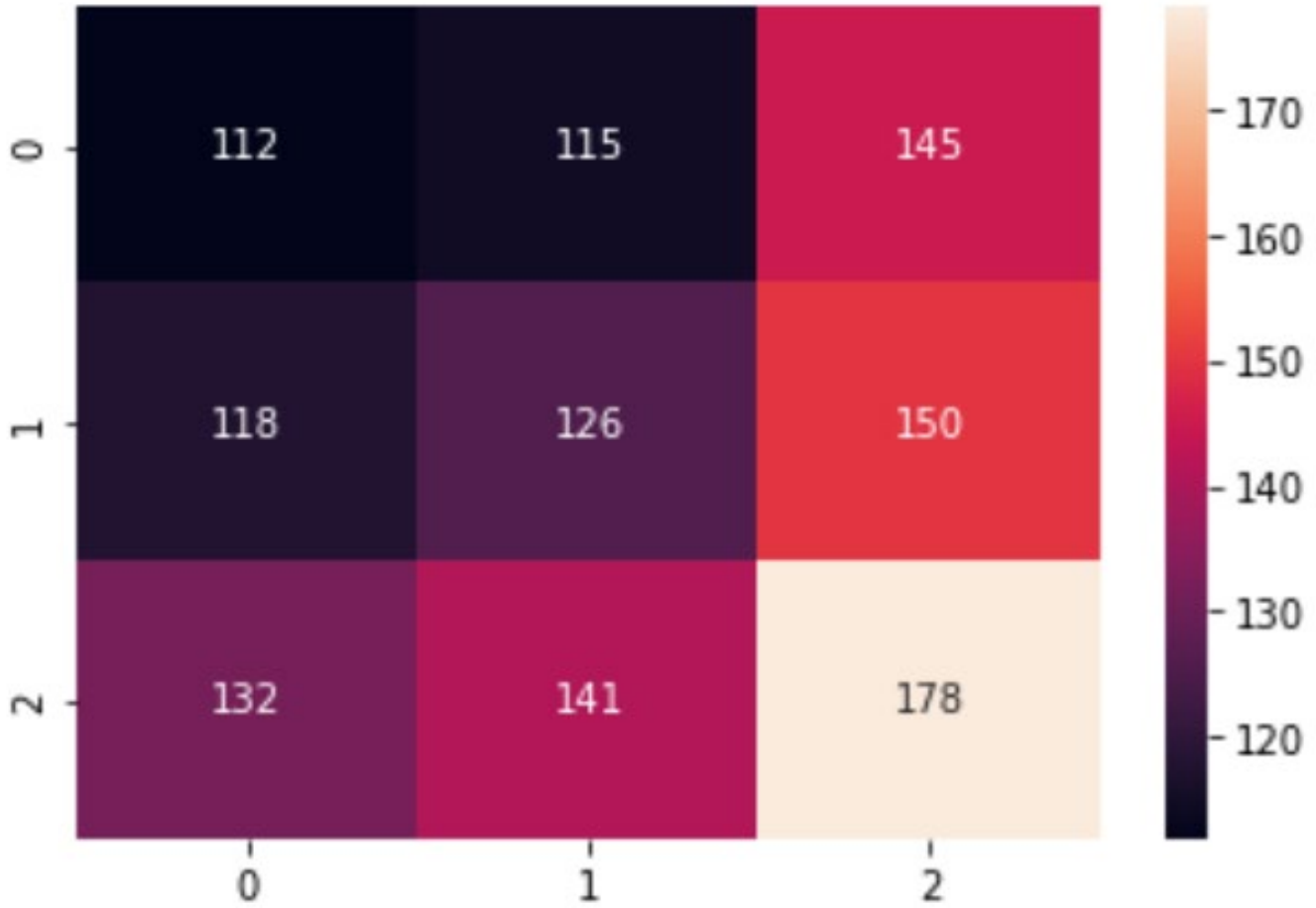
seaborn

- import seaborn
- import seaborn as sns

CC BY NC ND

Heatmap

```
arr = np.array([[112, 115, 145],
                [118, 126, 150],
                [132, 141, 178]])

sns.heatmap(arr, annot=True, fmt="d");
```



BY NC ND

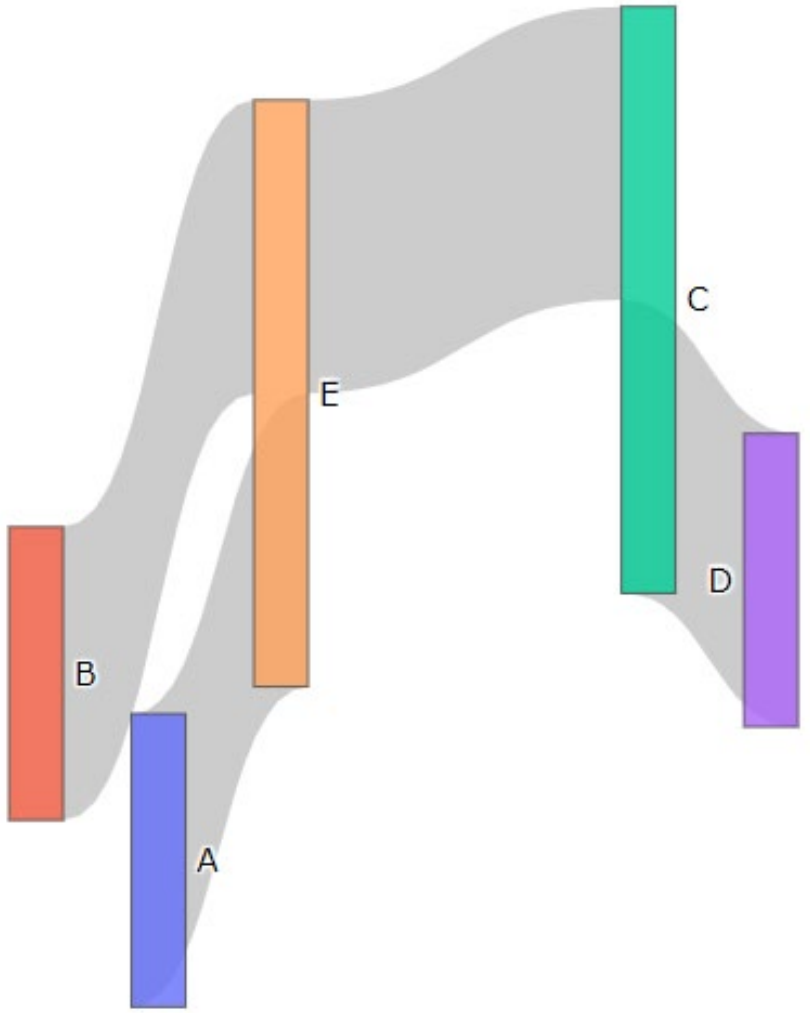
plotly | Dash

- import plotly.graph_objects as go
- import plotly.express as px

CC BY NC ND

Sankey

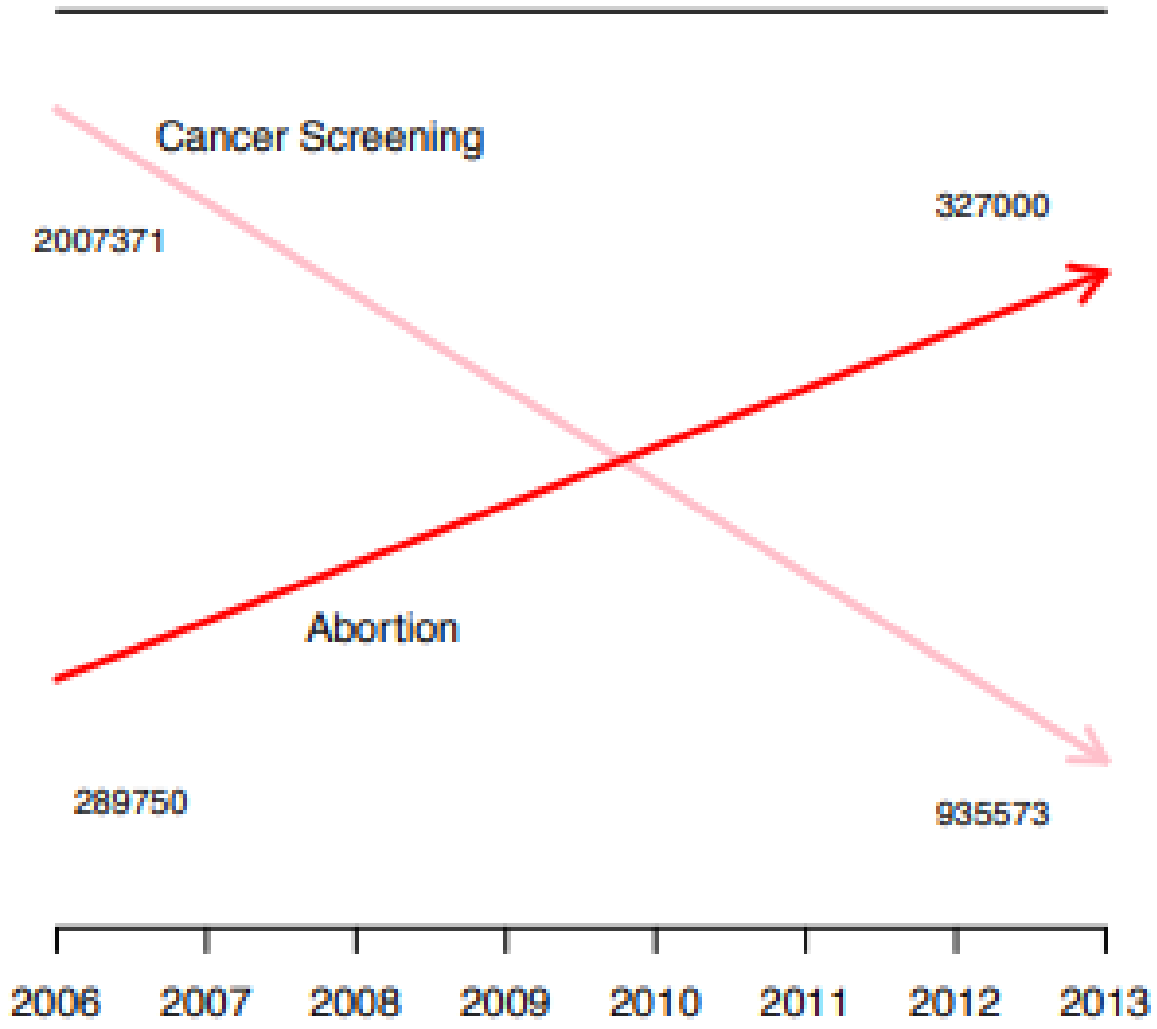
```
fig = go.Figure(go.Sankey(
    node = {
        "label": ["A", "B", "C", "D", "E"],
        "x": [0.2, 0.1, 0.6, 0.7, 0.3],
        "y": [0.7, 0.5, 0.1, 0.4, 0.2]},
    link = {
        "source": [1, 4, 4, 3, 4],
        "target": [4, 0, 2, 2, 3],
        "value": [1, 1, 1, 1, 0]}))

fig.show()
```



CC BY NC ND

When plotting data, which of the following approaches to axis scaling should be avoided?

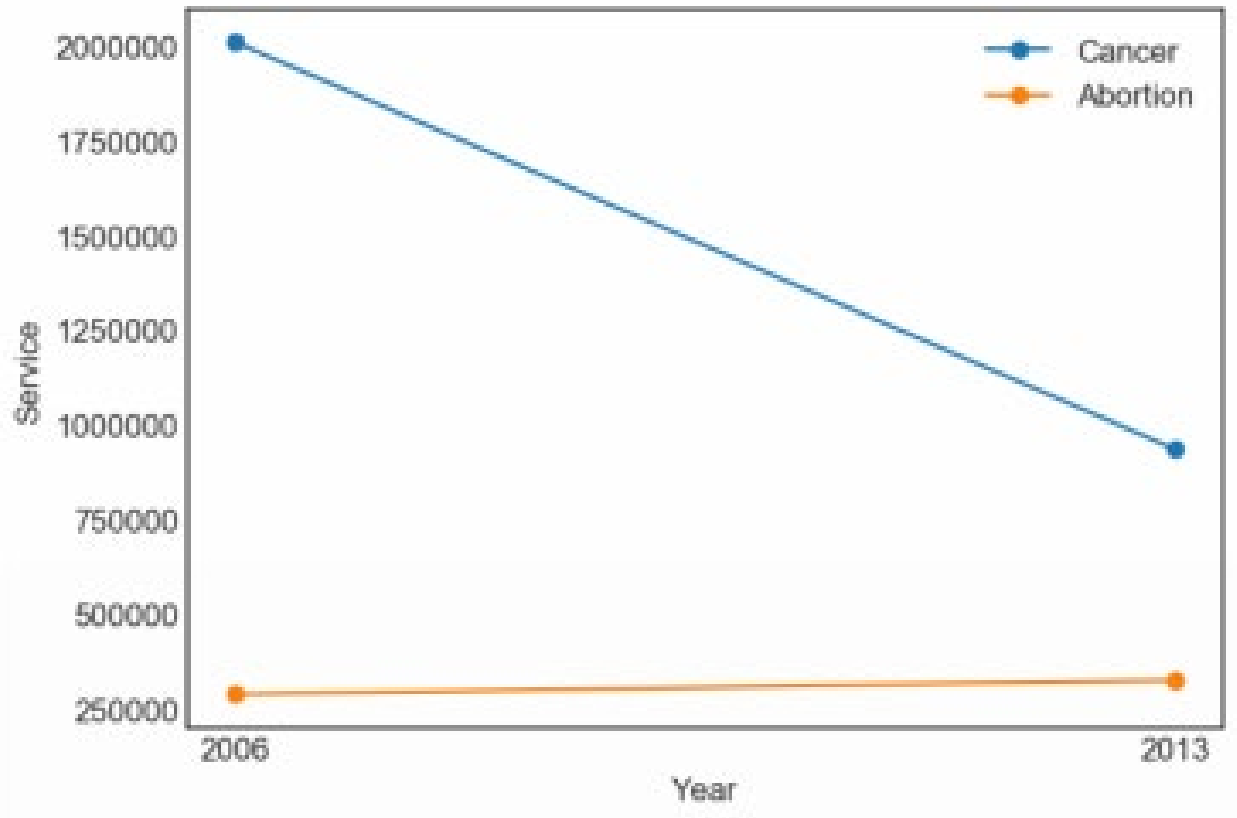
1. Using different scales for variables on the same axis
2. Changing the scale in middle of axis
3. Standardizing the scales for the same axis
4. Maintaining a consistent scale through the axis



CC BY NC ND

When plotting data, which of the following approaches to axis scaling should be avoided?

1. Using different scales for variables on the same axis
2. Changing the scale in middle of axis
3. Standardizing the scales for the same axis
4. Maintaining a consistent scale through the axis

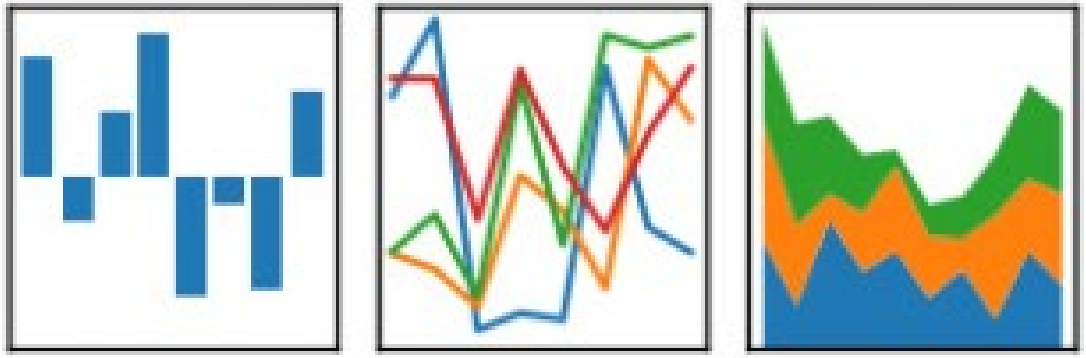


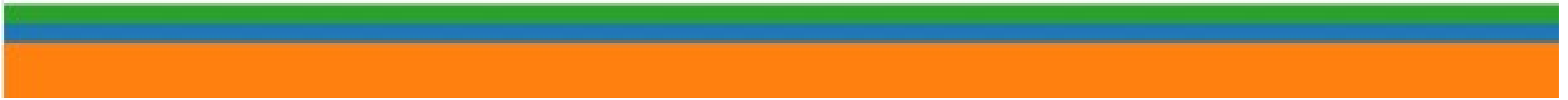
CC BY NC ND

- import pandas as pd

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$





CC BY NC ND

- import pandas as pd

CC BY NC ND

A comma separated value (csv) format can store tabular data

- We separate rows with different lines
- We separate columns with commas
- The first row indicates the headings of the columns

```
year,month,returns
2010,1,-0.5964769750070603
2010,2,0.323102811722204
2010,3,0.5936238389378875
2010,4,0.4837228609905558
2010,5,-0.12064664554679042
2010,6,-0.11388800636514022
2010,7,0.09647915933528232
```

CC BY NC ND

Character	Meaning
`'r'`	open for reading (default)
`'w'`	open for writing, truncating the file first
`'x'`	open for exclusive creation, failing if the file already exists
`'a'`	open for writing, appending to the end of the file if it exists
`'b'`	binary mode
`'t'`	text mode (default)
`'+'`	open a disk file for updating (reading and writing)

CC BY NC ND

We can load the contents of a file in comma separated value (csv) format with read_csv

```
aapl = pd.read_csv("AAPL.csv")
aapl
```

```
year,month,returns
2010,1,-0.596476975007060 3
2010,2,0.323102811722204
2010,3,0.593623838937887 5
2010,4,0.483722860990555 8
2010,5,-0.120646645546790 42
2010,6,-0.113888006365140 22
2010,7,0.096479159335282 32
```

CC BY NC ND

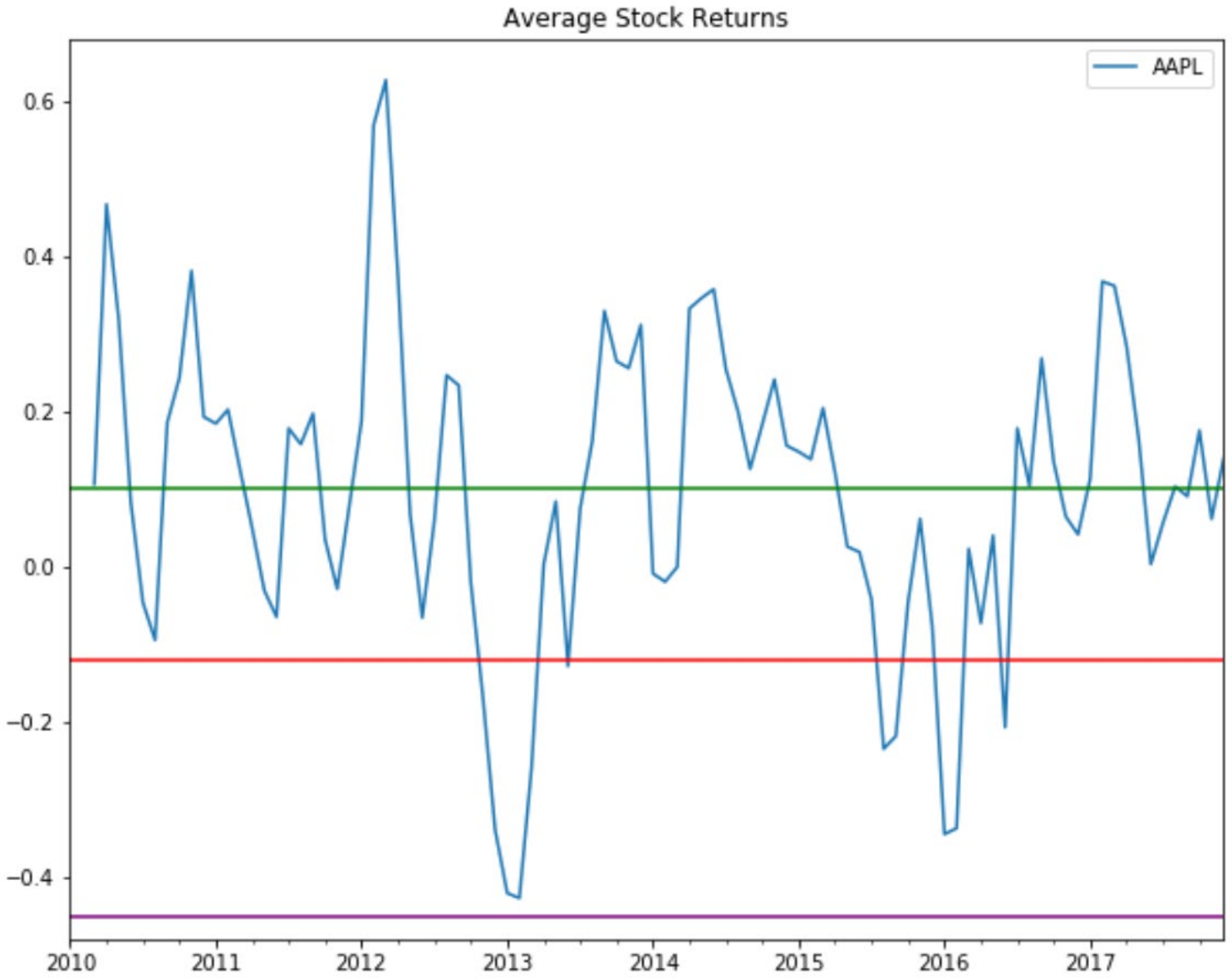
The pandas package supports three containers for storing data

- DataFrame
  - 2-dimensional data
- Series
  - 1-dimensional data
- Index
  - collection of labels.

	year	month	returns
0	2010	1	-0.596477
1	2010	2	0.323103
2	2010	3	0.593624
3	2010	4	0.483723
4	2010	5	-0.120647
...	...	...	...
91	2017	8	0.430815
92	2017	9	-0.316260
93	2017	10	0.412639
94	2017	11	0.088907
95	2017	12	-0.081133

CC BY NC ND

Example



CC BY NC ND

NYU TANDON SCHOOL OF ENGINEERING

Charts

- matplotlib
- seaborn
- plotly

Tables

- pandas

NYU TANDON SCHOOL OF ENGINEERING

References

- McKinney, Python for Data Analysis

(Chapter 5.1-5.2 + 9.1-9.2)

Questions

- Describe the learning objectives.
- Summarize the relevant take-aways.
- Ask about unclear information.