Writing your report

A (very brief) introduction to scientific writing

Flavio Giobergia

Data science lab: process and methods

Scientific writing (in a slide)

- Scientific writing is the technical writing used to communicate your work to others
- Scientific communication *requires* clarity and concision
- Scientific writing should address a research question, hypotheses, experiments, results and discussion

Scientific writing (in a slide)

- Scientific writing is the technical writing used to communicate your work to others
- Scientific communication *requires* clarity and concision
- Scientific writing should address a research question, hypotheses, experiments, results and discussion

Technical writing for communication

- Technical writing is the writing found in:
  - Textbooks
  - Scientific papers
  - Technical reports

- Communicating your work means:
  - Stating a *question* and giving an *answer*
  - *Explaining* the rationale behind the answer
  - Giving the means for (independent) *replication*

Scientific writing (in a slide)

- Scientific writing is the technical writing used to communicate your work to others
- Scientific communication *requires* clarity and concision
- Scientific writing should address a research question, hypotheses, experiments, results and discussion

Clarity & concision

- Clarity
  - *Does it make sense to the reader?*
  - Use precise words and sentences
    - There should be no room for ambiguities
    - Be objective!

- Concision
  - A wordy sentence is a confusing sentence
  - A picture is *sometimes* worth a thousand words
  - Meeting page quotas is not beneficial to anyone

Scientific writing (in a slide)

- Scientific writing is the technical writing used to communicate your work to others
- Scientific communication *requires* clarity and concision
- Scientific writing should address a research question, hypotheses, experiments, results and discussion

Some guidelines

- Support everything with evidence
- Distinguish fact from possibility
- Thoroughly understand your sources
  - And make sure your sources are peer reviewed!
  - Google Scholar can help you find and navigate sources
- Know your audience
- Never make your readers work harder than they have to

Based on "Scientific Writing, Style & Structure", University Writing Center, Texas A&M University

Writing your report

- Structure
  - Paragraphs
  - Sections (IMRaD!)
- Writing choices
  - Words
  - Sentences
- Supports
  - Lists
  - Images
  - Tables
  - ~~Code~~

Writing your report

- Structure
  - Paragraphs
  - Sections (IMRaD!)
- Writing choices
  - Words
  - Sentences
- Supports
  - Lists
  - Images
  - Tables
  - ~~Code~~

Structure

- Paragraphs
  - One paragraph ⇔ One important concept
    - 1:N and N:1 are not effective!
- Sections – IMRaD!
  - Introduction
    - Present your problem
  - Methods
    - Present your solution
  - Results, and
    - Apply your solution to your problem
  - Discussion
    - Did that work?
- + Abstract

Introduction (Problem overview)

- What problem do you have?
- Explore the data
  - What's interesting?
  - What's worth mentioning?
  - What requires careful handling?
- Visual aids may be particularly useful here
  - Data distributions
  - Visualization of some points
  - Summary tables

Method (Proposed approach)

- How do you propose you solve your problem?

- Keep it structured
  - Preprocessing
    - What steps did you take to prepare the data? Why?
  - Model selection
    - What models did you use? Why?
  - Hyperparameters tuning
    - Which hyperparameters did you focus on?
    - How did you tune them?

Results

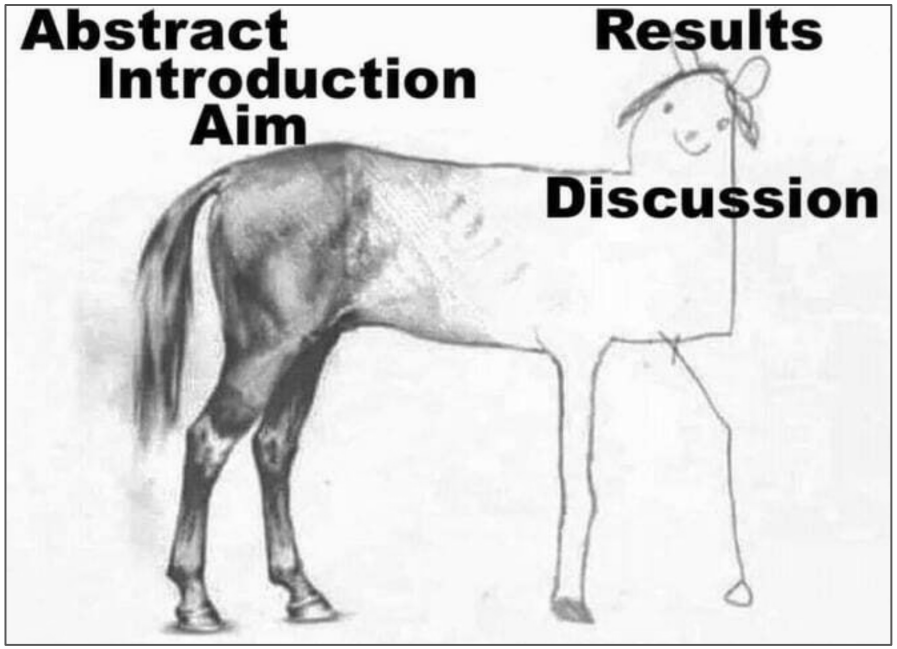
- What happens when you apply your solution to your problem?
- What configurations of *Algorithms* × *Parameters* did you select?
- Let's talk performance:
  - Validation performance
  - Public score performance
- How good is your solution?
  - Vs. random guess?
  - Vs. a naïve solution?
  - Vs. others in the leaderboard?

Discussion

- What conclusions can you draw based on what happened when you applied your solution to your problem?
- What went well?
- What could you improve?
  - Other possible approaches
  - Limitations found
- Considerations on the problem

Abstract

- 2-3 sentences that describe your work
- Gives an idea of what's in the paper
- Should be self-contained
- (hopefully) appealing
  - But avoid clickbaits!



Writing your report

- Structure
  - Paragraphs
  - Sections (IMRaD!)
- Writing choices
  - Words
  - Sentences
- Supports
  - Lists
  - Images
  - Tables
  - ~~Code~~

Writing choices

- Avoid needless complexity
  - No redundancy and gratuitous verbosity
  - Keep sentences short
    - One sentence ⇔ one clause (ideally!)
    - Reduce compound sentences
    - Resort to lists, images, tables
- Use formal English
  - No contracted forms
    - Wouldn't, didn't, it'll, … → would not, did not, it will, …
  - No informal terms (or slang!)
    - Tons of, totally, … → large quantities of, completely, …
- Passive vs Active voice

Writing your report

- Structure
  - Paragraphs
  - Sections (IMRaD!)
- Writing choices
  - Words
  - Sentences
- Supports
  - Lists
  - Images
  - Tables
  - ~~Code~~

Lists

- Lists are a great way to:
  1. Write less
  2. Help you convey ideas more easily
  3. Make the reading experience more pleasing

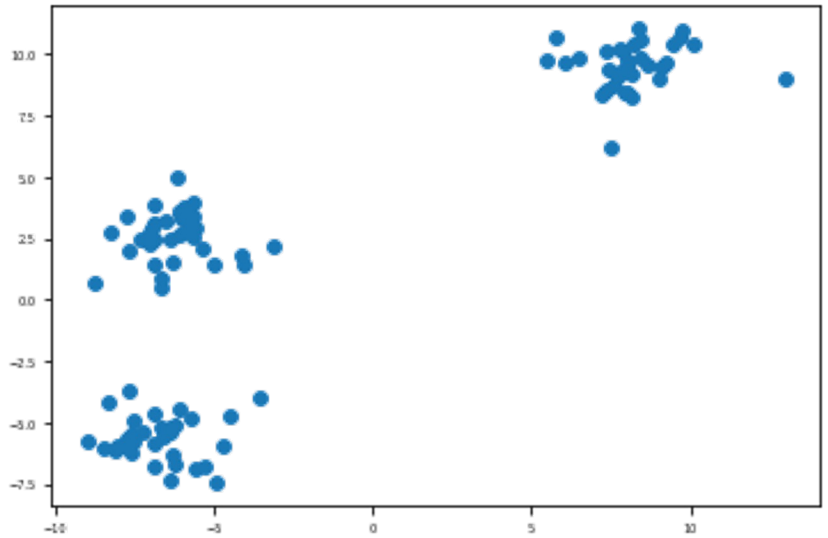
As opposed to:

- A support, such as a list, can be anything that can help you achieve one (or more) of the following three goals. First, it lets you write less words, because it is structured in a way that forces the adoption of few (or no) words. Second, it helps convey an idea more easily, because sometimes giving a few key concepts and letting the reader figure out the rest is better than explaining every single aspect of something in an overly verbose (and a bit patronizing) way. Finally, it makes the reading experience more pleasing, avoiding walls of text that would otherwise bore the reader greatly. This is a self-evident slide, by the way.

Images

- Images (e.g. plots, diagrams) are great, if presented correctly
  - But awful, otherwise
- For example, the following plot would be perfectly fine…
  - … except for a few things

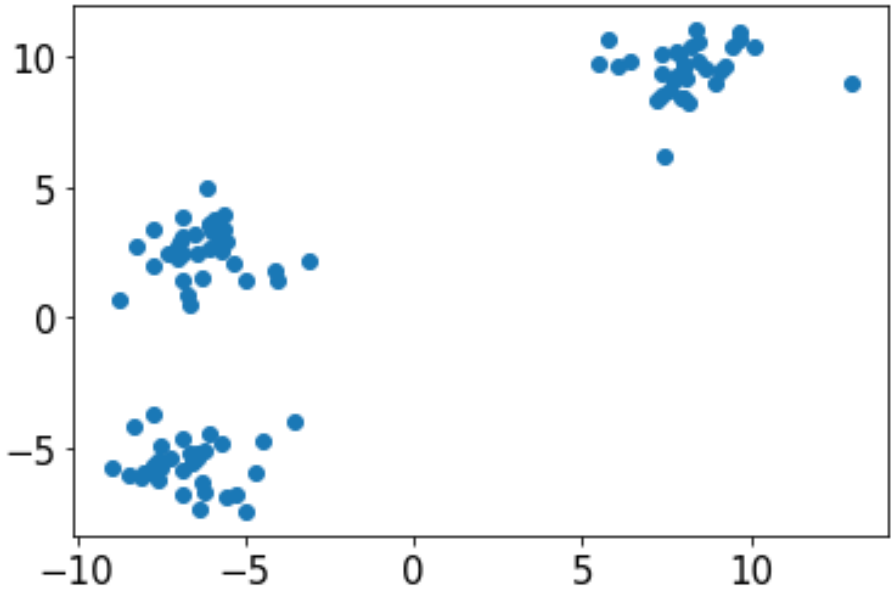
```
X = ... # data (x1, x2)
y = ... # get cluster id

fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
```



Fundamentals of Data Visualization
https://clauswilke.com/dataviz/

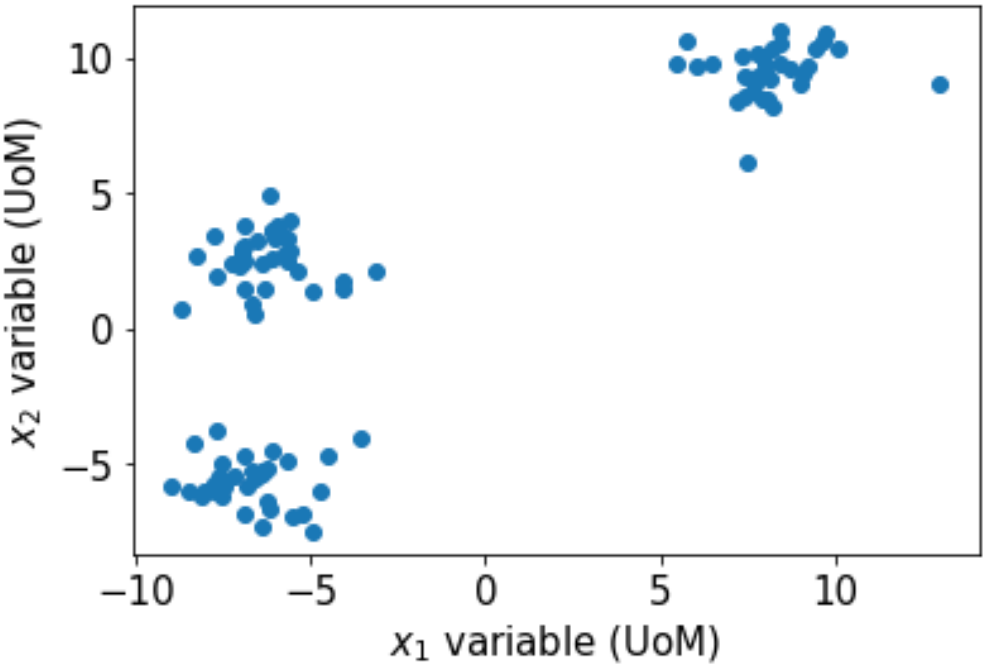
Step 1: make it readable

```
import matplotlib as mpl

mpl.rcParams["font.size"] = 14

X = ... # data (x1, x2)
y = ... # get cluster id

fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])
```



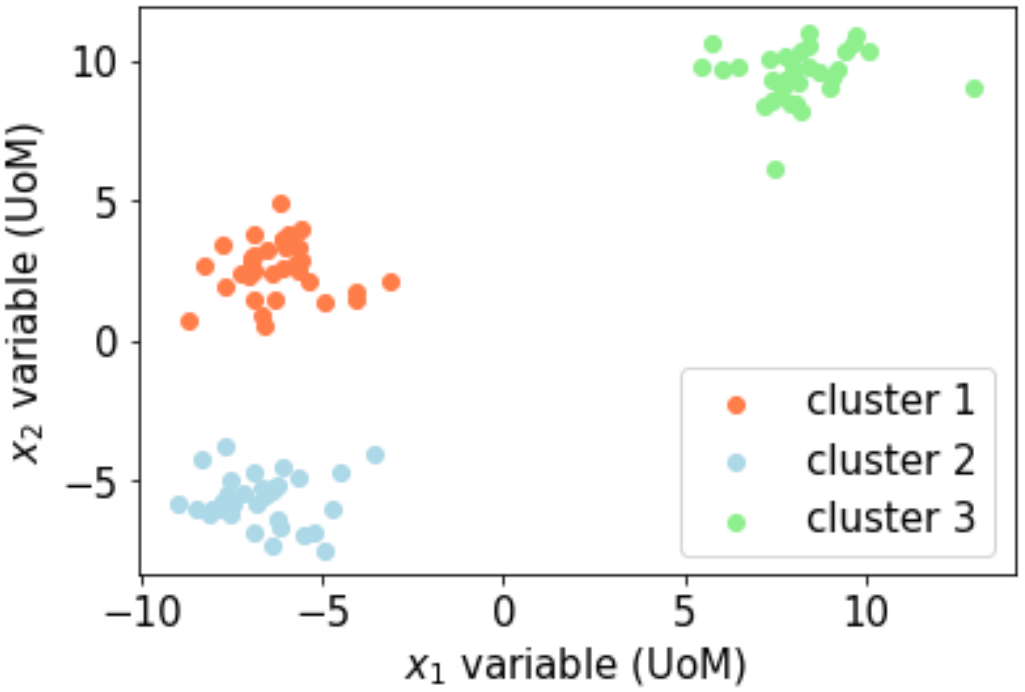
Step 2: labels and units of measure

```
import matplotlib as mpl

mpl.rcParams["font.size"] = 14

X = ... # data (x1, x2)
y = ... # get cluster id

fig, ax = plt.subplots()
ax.scatter(X[:,0], X[:,1])

ax.set_xlabel("$x_1$ variable (UoM)")
ax.set_ylabel("$x_2$ variable (UoM)")
```



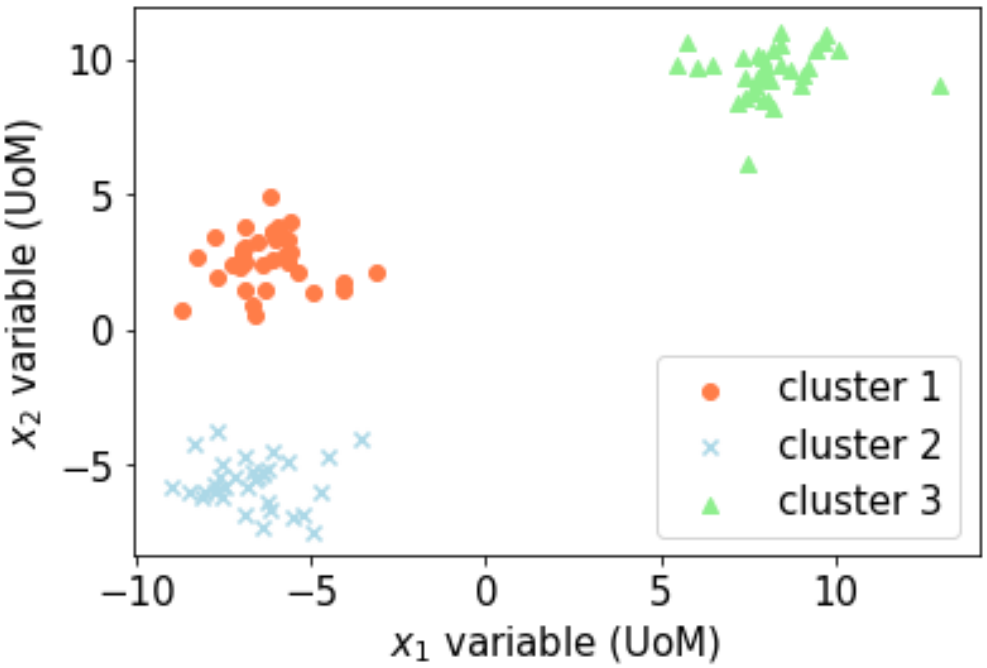
Step 3: add some colors!

```
import matplotlib as mpl

mpl.rcParams["font.size"] = 14

X = ... # data (x1, x2)
y = ... # get cluster id

clusters = [("cluster1", "coral"),
            ("cluster2", "lightblue"),
            ("cluster3", "lightgreen")]

fig, ax = plt.subplots()
for val, (name, c) in enumerate(clusters):
    ax.scatter(X[y==val,0],
               X[y==val,1],
               c=c,
               label=name)
ax.set_xlabel("$x_1$ variable (UoM)")
ax.set_ylabel("$x_2$ variable (UoM)")
ax.legend()
```



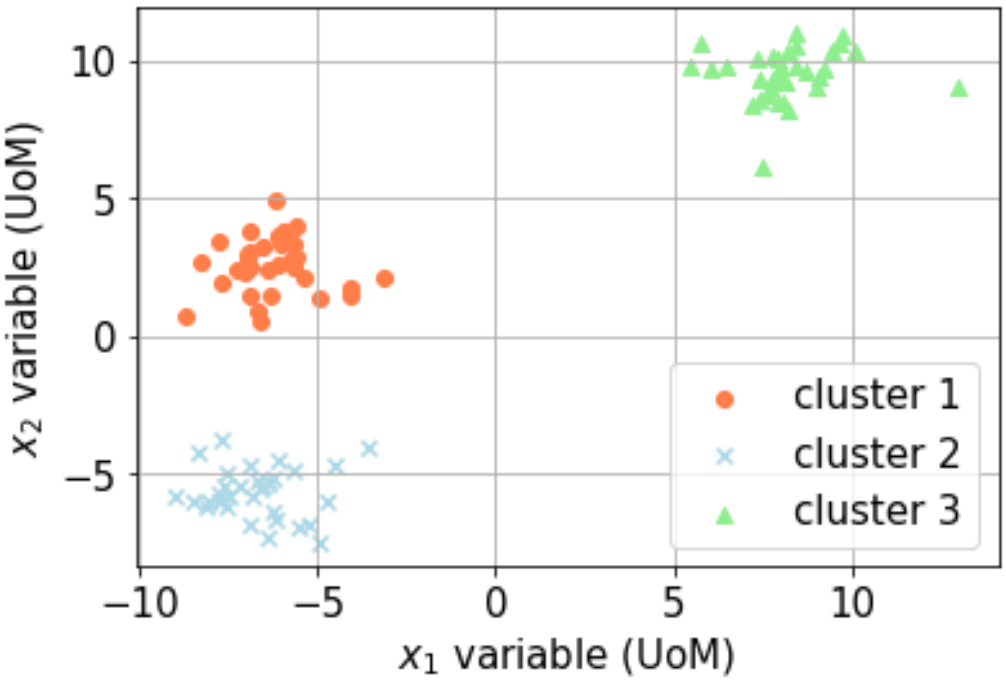
Step 4: go color blind & B/W friendly

```
import matplotlib as mpl

mpl.rcParams["font.size"] = 14

X = ... # data (x1, x2)
y = ... # get cluster id

clusters = [("cluster1", "coral", "o"),
            ("cluster2", "lightblue", "x"),
            ("cluster3", "lightgreen", "^")]

fig, ax = plt.subplots()
for val, (name, c, m) in enumerate(clusters):
    ax.scatter(X[y==val,0],
               X[y==val,1],
               c=c,
               label=name,
               marker=m)
ax.set_xlabel("$x_1$ variable (UoM)")
ax.set_ylabel("$x_2$ variable (UoM)")
ax.legend()
```



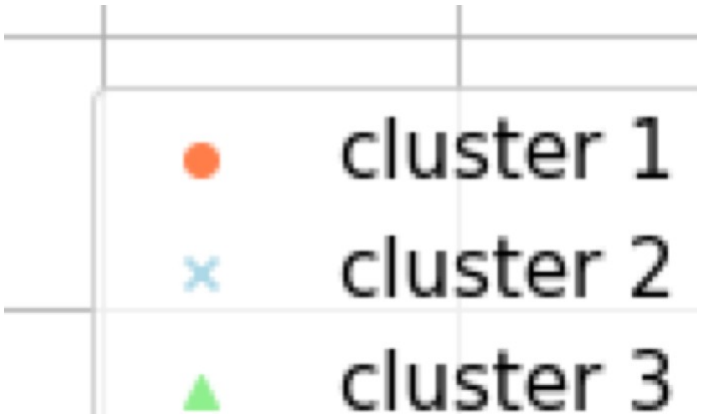
Protip 1: add a grid

```
import matplotlib as mpl

mpl.rcParams["font.size"] = 14

X = ... # data (x1, x2)
y = ... # get cluster id

clusters = [("cluster1", "coral", "o"),
            ("cluster2", "lightblue", "x"),
            ("cluster3", "lightgreen", "^")]

fig, ax = plt.subplots()
for val, (name, c, m) in enumerate(clusters):
    ax.scatter(X[y==val,0],
               X[y==val,1],
               c=c,
               label=name,
               marker=m)
ax.set_xlabel("$x_1$ variable (UoM)")
ax.set_ylabel("$x_2$ variable (UoM)")
ax.legend()
ax.grid()
```

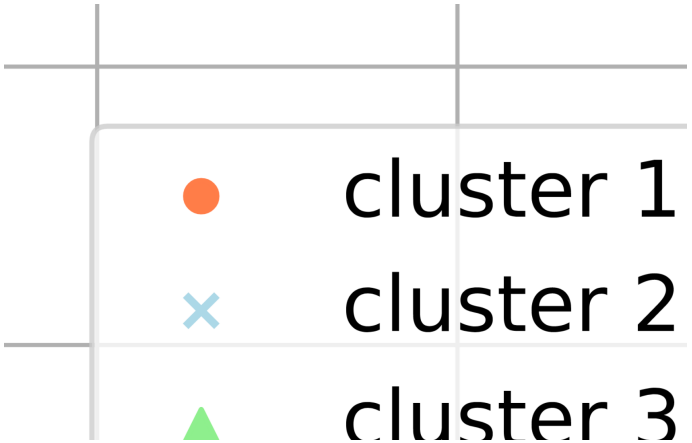


Protip 2: export vector images

bitmap



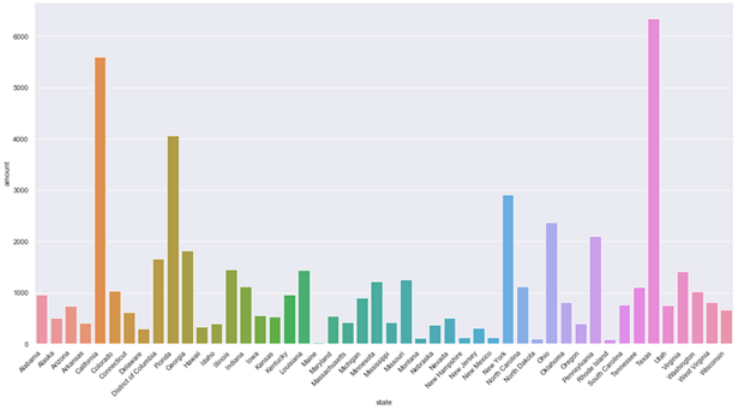
vector

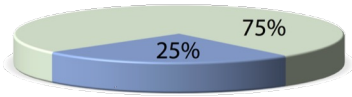


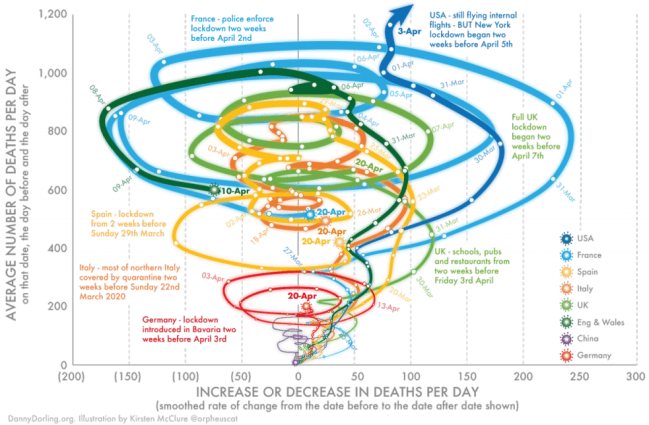

```
fig.savefig("file.pdf", bbox_inches="tight")
```

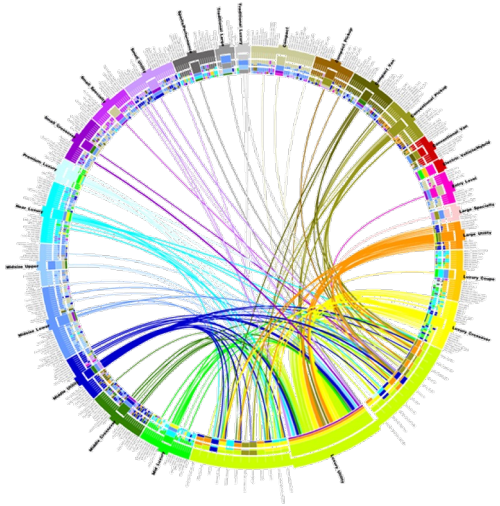
Bad visualizations

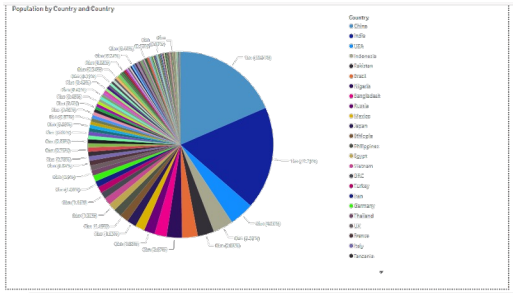
- Avoid "bad" visualizations:
  - Pie charts
  - Gratuitous 3D plots
  - Unnecessary information
  - Overcrowded plots
  - …

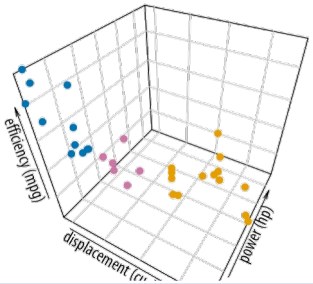








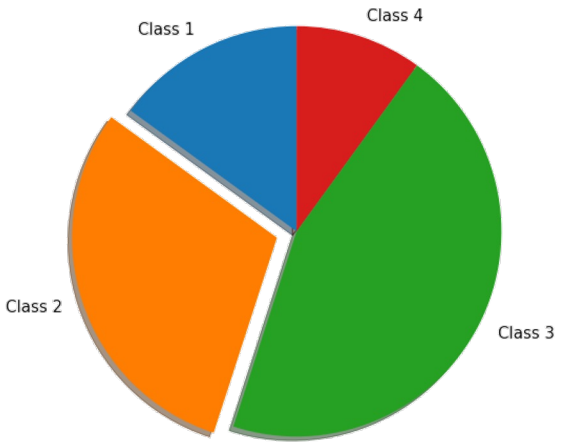




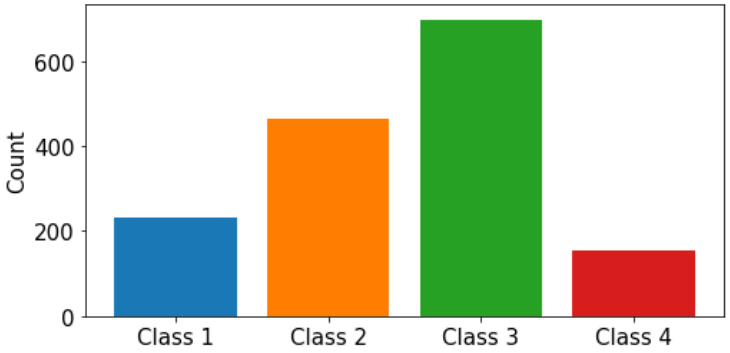
"Cosmetic decoration, which frequently distorts the data, will never salvage an underlying lack of content."
— Edward Tufte

Alternatives

- Pie charts?
  - 🤮



- Bar charts



- Tables

Class	Count
Class 1	232
Class 2	465
Class 3	698
Class 4	155

Tables

- We use tables:
  - If we want to show accurate comparisons
  - If we want to show data that is
    - Long
    - Multidimensional
    - Hierarchical
  - When it makes sense!

Category	Class	Cardinality	Color	SVM		Random Forest	
				Precision	Recall	Precision	Recall
Even	Class 2	465	Orange	0.911	0.943	0.812	0.849
	Class 4	155	Red	0.823	0.955	0.88	0.912
Odd	Class 1	232	Blue	0.815	0.901	0.873	0.987
	Class 3	698	Green	0.967	0.974	0.897	0.945

A nice table generator for LaTeX (and more!) ➔ https://www.tablesgenerator.com

Don't forget to caption!

- Add meaningful captions to your tables/figures
- In LaTeX, stuff may get moved around
  - Caption + Content should be "self-contained"
  - Adding a caption makes it easier for the reader to follow
- Always address in the text the contents you add

~~Code~~

- Your Python code does not belong to the report
- The raw output of your Python code does not belong to the report
- Describe algorithms
  - Visually
  - With words
- There are some rare exceptions to adding code
  - E.g., if a contribution is purely algorithmic
  - If necessary, use pseudocode

```
QUICKSORT (A, p, r)
  if p < r
    q = PARTITION(A, p, r)
    QUICKSORT(A, p, q-1)
    QUICKSORT(A, q+1, r)
  end if
```

```
PARTITION (A, p, r)
  x = A[r]
  i = p - 1
  for j = p, …, r – 1
    if A[j] ≤ x
      i = i + 1
      swap A[i], A[j]
    end if
  end for
  swap A[i+1], A[r]
  return i + 1
```

What about LLMs?

- AI systems (e.g. ChatGPT) can of course help improving (scientific) writing
- Different courses have different policies!
- For DSL:
  - We allow using AI-aided writing
    - You are responsible of everything AI-written
  - We require disclosing this information
    - (Details will follow)
  - No support allowed for code generation